HCLSoftware

HCL DevOps Test Hub
12.0.0 Documentation
September 2026



Special notice

Before using this information and the product it supports, read the information in Notices on page mcclxxxviii.

Contents

Chapter 1. Release Notes..... 7
- What's New..... 8
- Deprecations and removals..... 50
- Defect fixes..... 52
- Known issues..... 67

Chapter 2. System Requirements..... 94
- Hardware..... 95
- Operating systems and Containers..... 95
- Host prerequisites..... 97
- Supported software..... 98

Chapter 3. Getting Started Guide 103
- Introduction to DevOps Test Hub..... 103
- Capabilities..... 107
- Installation..... 107
- User tasks..... 108
- Member access and permissions..... 108
- Compatibility and supported assets..... 109
  - Supported versions of assets and resources..... 109
  - Supported suites and tests..... 110
  - Supported virtual services..... 112
- Accessibility features..... 112

Chapter 4. Administrator Guide..... 113
- Configuration of an external database..... 113
- Installation of DevOps Test Hub..... 115
  - Installation of DevOps Test Hub on Azure Kubernetes Service..... 116
  - Installation of DevOps Test Hub on Red Hat OpenShift..... 116
  - Installation of DevOps Test Hub on Linux..... 117
  - Management of DevOps Test Hub features..... 117
- Upgrade of DevOps Test Hub..... 119
  - Upgrading of DevOps Test Hub on Azure Kubernetes Service..... 120
  - Upgrading of DevOps Test Hub on Red Hat OpenShift..... 120
  - Upgrading of DevOps Test Hub on Linux..... 120
- Migration of user data..... 120
  - Migrating user data on Red Hat OpenShift..... 123
  - Migrating user data on Ubuntu from 10.1.2 or earlier..... 124
  - Migrating user data on Ubuntu from 10.1.3 or later..... 125
- Backup and restoration of the DevOps Test Hub data..... 127
  - Backup and restoration of data on IBM® Red Hat OpenShift..... 127
  - Backup and restoration of DevOps Test Hub data on Ubuntu..... 130
  - Backup and restoration of DevOps Test Hub data on Azure Kubernetes Service..... 134
- Uninstallation of DevOps Test Hub..... 139
  - Uninstallation of DevOps Test Hub on Azure Kubernetes Service..... 139
  - Uninstallation of DevOps Test Hub on Red Hat OpenShift..... 140
  - Uninstallation of DevOps Test Hub on Linux..... 140
- Configuration of DevOps Test Hub..... 140
  - User administration..... 141
  - Management of third-party licenses..... 160
  - Certificate authority: Importing and extending lists..... 161
  - Copying third-party application Jars to Kubernetes..... 165
  - Changing the password seed..... 167
  - Server extensions..... 169
- Team space administration..... 172
  - Team space overview..... 172
  - Creating a team space..... 175
  - Tasks in a team space..... 177
  - Tasks in projects in a team space..... 180
  - HCLSoftware license portal..... 182
- Management of resources..... 190
  - Managing retention rules..... 190
  - Viewing resource usage statistics..... 193
- Configuration of an MCP server endpoint..... 194
  - Configuring an MCP server endpoint in VS Code..... 195

Chapter 5. Test Author Guide..... 198
- Edit branch overview..... 198
- Test Editor overview..... 199
  - Test navigator panel..... 200
  - Test Editor page..... 201
  - Suite Editor page..... 203
- Application management..... 204
  - Creating an application..... 204
  - Modifying application components..... 206
  - Manage OpenAPI documents with version 3 standards for a web application..... 207
- Web UI test management..... 209
  - Web UI test creation..... 210
  - Running a Web UI test..... 228
  - Viewing step-level results and updating tests..... 232
  - Stopping a test run..... 234
  - Modification of Web UI tests..... 235
  - Renaming a test asset..... 275
  - Deleting a test asset..... 275

Creating a test module from an existing set of steps....277
Viewing test assets exported from DevOps Test UI....278
Salesforce Lightning tests....280
Test suite management....292
Creating a suite....292
Suite creation with Test Genie....294
Modifying a suite....296
Running a suite sequentially....299
Running a suite concurrently....300
Running multiple suites or tests by using the wildcard pattern....302
Running suites on multiple browsers and devices....304
Accessing the console output....307
Stopping test runs within a suite....308
Rerunning failed tests in a suite....309
Deleting a suite....310
Test variables....311
Creating and assigning test variables....312
Assigning an object property to a test variable....314
Adding an Assign step....316
Assigning test variable values from a dataset by using the Run step....318
Configuring test variables in the Run step of a test or suite....320
Promoting test variables....322
Creating a dataset by using test variables....324
Editing a dataset in Test Editor....325
Multiple datasets in tests and test suites....333
Test environments....337
Creating an environment....338
Modifying an environment....340
Catalog management....342
Basic generators....342
Management of custom generators....391
Management of custom categories....398
Data management....401
Data navigator panel....402
Prerequisites for working with datasets and data definitions....403
Management of datasets....424
Management of data definition....450
Management of schemas....469
Publishing changes in data definitions, datasets, or schemas....482
Generating fabricated test data....483
Publishing all changes in an Edit branch....493

Chapter 6. Test Execution Specialist Guide....496
System modeling....496
Task flows for working with a system model....500
Adding a repository to a team space....502
Creating a system model....505
Creating components....506
Creating child components....507
Creating linkages between components....510
Deleting linkages between components....511
Modifying components....513
Deleting components....515
Publishing changes to the system model....516
Associating resources with components....517
Viewing resources that are associated with components....519
Removing resources that are associated with components....522
Publishing changes of associating or dissociating resources with components....524
Viewing the system model....526
Deleting a system model....526
Deleting a repository from a team space....527
Prerequisites to running tests....527
Test run considerations for AFT Suites or Functional tests....528
Test run considerations for API Suites or API tests....529
Test run considerations for browser-based Web UI tests....532
Test run considerations for JMeter tests....532
Test run considerations for JUnit tests....534
Test run considerations for using Machine learning capabilities....534
Test run considerations for Postman tests....536
Test run considerations for schedules....538
Test run considerations for Selenium tests....539
Test run considerations for using mobile device clouds....540
Management of agents....542
Management of performance agents....542
Management of API agents....551
Management of DevOps Test Runtime agents....557
Management of the personal agent....567
Management of remote Kubernetes clusters....572
Considerations for using a remote Kubernetes cluster....573
Registering a remote Kubernetes cluster....576
Viewing remote Kubernetes clusters....578
Adding a remote Kubernetes cluster to a project....580

Editing configurations of a remote Kubernetes cluster....582
Removing a remote Kubernetes cluster from a project....584
Unregistering a remote Kubernetes cluster from a team space....586
Management of cloud credentials....587
Adding cloud credentials of mobile device clouds....593
Adding cloud credentials of AI providers....596
Adding authorized users to a cloud credential....600
Viewing details of cloud credentials....601
Editing details of cloud credentials....605
Removing authorized users from a cloud credential....608
Deleting an existing cloud credential....609
Test run configurations....610
Configuring a Suite or test for a quick run....613
Configuring an AFT Suite run....618
Configuring an AFT Suite to run on a mobile device by using variables....632
Configuring an AFT Suite to run on a mobile device by using a cloud credential....644
Configuring an API Suite or an API test run....652
Configuring a run of a Compound Test that contains HTML tests....665
Configuring a run of a Compound Test that contains mobile tests to run by using variables....678
Configuring a run of a Compound Test that contains mobile tests to run by using a cloud credential....693
Configuring a run of a Compound Test that contains Web UI tests....704
Configuring a run of a Compound Test that contains Web UI tests to run by using variables....717
Configuring a run of a Compound Test that contains Web UI tests to run by using a cloud credential....732
Configuring a run of a Compound Test that contains performance tests....743
Configuring a run of a mobile test to run by using variables....756
Configuring a run of a mobile test to run on a mobile device by using a cloud credential....771
Configuring a run of a Performance test....782
Configuring a run of a Web UI test....795
Running server test assets....808
Running server test assets on an agent....811
Running server test assets by using a cloud credential....813
Running server suite assets....816
Running server suite assets by using a cloud credential....819
Configuring a run of an SAP GUI test....822
Configuring a run of a Web UI test to run by using variables....827
Configuring a run of a Web UI test to run by using a cloud credential....842
Configuring a run of a Windows test....854
Configuring a run of a JMeter test....867
Configuring a run of a JUnit test....876
Configuring a run of a Postman test....886
Configuring a run of a Rate Schedule or VU Schedule....896
Configuring a run of a Selenium test....909
Configuration of test runs in a GitHub Actions Workflow....919
Prerequisites to run tests in a GitHub Actions Workflow....919
Configuring a test run in a GitHub Actions Workflow....920
Management of running tests....922
Viewing the state of test assets....922
Viewing the progress of running test assets....926
Checking logs....930
Resetting the configuration settings for a test run....931
Stopping test runs....932
Canceling scheduled test runs....934
Management of virtualized services....936
Prerequisites for running HTTP virtual services....938
Viewing intercepts that are registered with a team space on DevOps Test Hub....942
Viewing virtual service resources....943
Configuring a run of a virtual service....946
Running HTTP virtual services in Kubernetes cluster without using proxies....961
Viewing running instances of virtual services....966
Viewing configurations of running instances of virtual services....971
Modifying configurations of running instances of virtual services....976
Viewing routing rules of the virtual services....978
Viewing usage statistics of virtual services....980
Stopping virtual services....982
Test results....983
Test results and reports....985
Test results data in Grafana....1000
Viewing Surefire format reports....1006
Test logs....1007
Test log queries....1022
Adding additional properties in test results....1025

Machine learning capabilities in DevOps Test Hub. 1027
Viewing Machine learning insights..... 1031
Interpretation of insights..... 1033
Viewing Machine learning analyzers..... 1037
Interpretation of parameter analysis..... 1039
Viewing reactions marked by members to Machine learning insights.....1042
Modifying Machine learning analyzer settings..... 1044
Deleting Machine learning insights.....1046
Rerunning Machine learning (ML) analysis..... 1047
Resource monitoring..... 1047
Resource monitoring capabilities..... 1048
Resource monitoring agents.....1065
Managing sources..... 1077
Collecting and viewing resource monitoring usage data..... 1085
Configuration of a change management..... 1092
Configuration of Atlassian Jira as a change management.....1093
Configuration of Azure DevOps as a change management.....1105
Configuration of DevOps Plan as a change management.....1114
Integrations with other applications.....1123
Integration plugin compatibility matrix.....1123
Integration with Azure DevOps.....1126
Integration with HCL DevOps Deploy.....1136
Integration with IBM® DevOps Deploy..... 1152
Integration with IBM® Engineering Test Management.....1166
Integration with Jenkins..... 1173
Integration with Xray test management for Jira.....1188
Access management for DevOps Test Hub.....1194
Creating an offline user token in DevOps Test Hub.....1195
Deleting an offline user token in DevOps Test Hub.....1196
Working with DevOps Test Hub REST services.....1197
Running APIs in command line..... 1197
Chapter 7. Test Manager Guide.....1199
Managing team spaces.....1199
Viewing team spaces..... 1199
Adding members to a team space.....1200
Modifying the configuration of a team space ..1202
Viewing the configuration of a team space 1204
Becoming a team space member.....1205
Managing members and their roles in a team space.....1207
Deleting a team space.....1209
Managing DevOps Test Hub projects.....1209
Working with projects in a team space.....1210
Repository considerations for a server project..... 1211
Creating a project.....1213
Adding repositories to a server project.....1214
Secrets configuration 1217
Adding users to a project..... 1217
Viewing active projects.....1218
Changing the visibility configuration of a project..... 1219
Becoming a project member.....1220
Managing access to the server projects..... 1220
Archiving or unarchiving server projects..... 1223
Deleting server projects.....1224
Project overview.....1225
Managing repositories.....1226
Adding repositories to a server project.....1227
Updating the authentication credentials of the repository..... 1230
Deleting a repository.....1231
Refreshing repositories manually..... 1231
Selecting the global branch in a project..... 1233
Creating webhooks..... 1234
Protecting test assets by using secrets.....1235
Creating a secret in a security group.....1236
Granting access to members or member roles.....1237
Managing secrets..... 1238
Notification Management..... 1240
Managing in-app notifications.....1241
Managing email notifications.....1244
Configuration of a webhook..... 1248
Chapter 8. Troubleshooting.....1273
Troubleshooting issues..... 1273
Failed to install DevOps Test Runtime on Windows 11 25H2..... 1282
Failed to load the extension in Google Chrome while recording.....1283
Chapter 9. Unable to publish test assets to a Control repository.....1285
Security Considerations.....mcclxxxvi
Notices..... mcclxxxviii
Index.....1291

Chapter 1. Release notes for DevOps Test Hub 2026.09 (12.0.0)

This document includes information about the new features introduced, features deprecated or removed, defects fixed, and known issues identified in this version of HCL DevOps Test Hub (Test Hub). You can also find the installation and upgrade instructions along with the contact information of HCL Customer Support.

Product description

Test Hub is a server that includes capabilities such as project and role-based security, Docker-based distribution and installation, and running of test cases. For more information about the server, see Overview of DevOps Test Hub.

What's new

You can find information about the features introduced, enhancements, or other changes in Test Hub. See What's New on page 8.

Deprecated features or capabilities

To find information about the features or capabilities that are deprecated in this version and are planned for removal in a future release, see Deprecations and removals on page 50.

Removed features or capabilities

To find information about the features or capabilities that are removed in this version, see Deprecations and removals on page 50.

Product download and installation

If you have purchased the licenses to use the product, you can download the product software packages from HCL® License & Download portal.

For instructions about installing the product software, see Installation of DevOps Test Hub on page 115.

Product upgrade

For instructions about upgrading the product software, see Upgrade of DevOps Test Hub on page 119.

Defect fixes and known issues

For the list of defects fixed in this version that include defects submitted by customers and defects found during testing, see Defect fixes on page 52.

For the list of known issues that are identified in this version and the previously published known issues that are still applicable, see Known issues on page 67.

Contacting HCL support

- For technical assistance, contact HCL Customer Support.
- Before you contact HCL support, you must gather the background information that you might need to describe your problem. When you describe a problem to an HCL Support specialist, be as specific as possible and include all relevant background information so that the specialist can help you solve the problem efficiently. To save time, know the answers to these questions:
  - What software versions were you running when the problem occurred?
  - Do you have logs, traces, or messages that are related to the problem?
  - Can you reproduce the problem? If so, what steps do you take to reproduce it?
  - Is there a workaround for the problem? If so, be prepared to describe the workaround.

What's New

You can find information about the features introduced, enhancements, or other changes made in HCL DevOps Test Hub (Test Hub). You can also find the features that were introduced in earlier versions of Test Hub. DevOps Test Hub is the new name of HCL OneTest™ Server from 11.0.0 onwards.

What's New in DevOps Test Hub 12.0.0

The following section lists the server features introduced in Test Hub 12.0.0.

Feature title	Description
Enhanced MCP tooling for assets and results	You can now use the newly added MCP tool to retrieve the smartshots for steps in test results. See Configuration of an MCP server endpoint on page 194.
Scenario-based test suite generation and bulk processing	You can now use Test Genie to generate scenario-based suites and AI intent for individual tests within a suite. Test Genie initially creates empty tests with intents for each scenario and then processes them in bulk to generate the tests sequentially in a suite. See Suite creation with Test Genie on page 294.
GitHub Copilot integration for test generation	You can now add and manage the GitHub Copilot AI provider on the **Cloud Credentials** page in Test Hub. You can then use Copilot to generate tests by using Test Genie. See Management of cloud credentials on page 587 and Test creation with Test Genie on page 220.
Guardrails for AI token consumption	You can now set session limits when you configure an AI provider on the **Cloud Credentials** page. During test generation, Test Hub evaluates the configured values, such as **Max tokens per session**, **Max interactions per session**, or **Max duration per session**. If any configured limit is reached, the active AI session ends, and you must start a new session to continue the test generation. See Guardrails for AI token consumption on page 597.
Stop reasons for AI-generated UI tests	You can now see stop reasons when Test Genie cannot navigate the application or cannot complete the next action while generating UI tests. The stop reason displayed in the

Feature title	Description
	generated test helps you to diagnose and resolve failures. See Test creation with Test Genie on page 220.
Enhanced AI-based test generation	Test Genie is now enhanced with support for drag-and-drop actions, drop-down selection actions, and automated verification points. Test Genie also provides better AI interaction logs, token usage metrics, and refined algorithm updates to generate higher-quality tests and suites. See Test creation with Test Genie on page 220.
Creating a test by using an existing test with a personal agent	You can now create a new test by using an existing test as the starting point. The personal agent plays back the selected test up to the last step, and then allows you to record additional steps in a live browser. See Creating a test by using existing steps of a test with the personal agent on page 226.
Support for environment selection and variable promotion	You can now select an environment from the **Environment** drop-down list in the **Test Editor** header. The selected environment applies to test authoring and test runs. The last selected environment remains as the pre-selected option unless you change it from the drop-down list. You can also promote variables to environment-level and mark the variables that contain sensitive information as secrets. The values of secrets are managed through security groups. See Test environments on page 337, Creating an environment on page 338, Modifying an environment on page 340, Test variables on page 311, Promoting test variables on page 322, Creating and assigning test variables on page 312, Running server test assets on page 808, and Protecting test assets by using secrets on page 1235.
Enhanced resource usage statistics	You now view the enhanced resource usage statistics on the **Resources** page. The page additionally displays all the projects within the team space and total size of all repositories within the project. See Viewing resource usage statistics on page 193.
Support for Salesforce application testing	You can now record and play back interactions with Salesforce Lightning applications for testing. During recording, smartshots capture the controls used in each step so that you can create additional steps, interact with other visible controls, and capture verification points. See Salesforce Lightning tests on page 280, Recording and configuring Salesforce application tests on page 288, and Running a Salesforce Lightning application test on page 291.
Support for saved filters for execution and result pages	You can now save user-defined filters created on the **Advanced** tab of the **Executions**, **Results**, and **Progress** pages. Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users. Viewing the progress of running test assets on page 926 and Creating search queries to filter test results on page 991.

Feature title	Description
Support for improved smartshot presentation and management	You can now see the test steps that use a smartshot when you select or manage smartshots. The interface also shows whether a smartshot is unused, displays the source filename as hover text, and prevents the current smartshot from being selected after it is replaced. See Application management on page 204.
Support for a standalone smartshot viewer	You can now open smartshots in a standalone viewer from the navigator. The viewer makes it easier to inspect smartshot images, their element hierarchy and usage in the tests without opening the image as a code file. See Test Editor page on page 201.
Support for inserting variables at the cursor in test editor fields	You can now insert variables in supported test editor fields to combine multiple variables in an expression without clearing the existing content. See Modification of Web UI tests on page 235, Adding a Log step on page 245, Adding a Wait step on page 246, Adding a Verify step on page 248, Adding a Select step on page 256, Adding a Type step on page 257, Adding an Open step on page 254, Adding an If step on page 262, Adding a While step on page 265, and Adding a Do-While step on page 268.
Support for result properties in the Execution dialog	You can now view and set result properties directly from the **Results** tab of the **Execution test asset** dialog. The added result properties are displayed on the **Functional Report** page, **Test Log** page, and in the **Properties** column on the **Results** page. See Configuring a run of a Web UI test on page 795 and Running server test assets on page 808.
Support for collapsible functional reports headers	You can now collapse the **Verdict** and **Details** section at the top of the functional report to make more room for test and step details. You can also control how asset paths are displayed in the details tree. See Viewing a Functional Report on page 994 and Navigating to a test, suite, or step from the functional report on page 997.
Option to record and play back API tests - Tech preview	An option is now available to record and play back API traffic. You can record API traffic based on either OpenAPI documents managed within the application or generic REST-based interactions. The **API** option in the **Create Test** dialog is available as a Tech preview feature, and it is not supported for production use.
Support for OpenAPI document management in web applications	You can now manage OpenAPI documents in web applications to organize and use API definitions as part of your test workflows. See Manage OpenAPI documents with version 3 standards for a web application on page 207.

What's New in earlier versions

You can find information about the features introduced in earlier versions of Test Hub in the following sections.

What's New in 11.0.9

The following section lists the features introduced in Test Hub 11.0.9.

Feature title	Description
AI-powered test generation with Test Genie in Test Hub	You can now use Test Genie to automatically generate test assets within Test Hub. By providing natural language instructions and connecting to a supported AI provider, you can accelerate your test creation process. While test steps are being generated, you can view AI interactions and session logs on the progress page. After the test generation is complete, you can view the generation metrics. See Test creation with Test Genie on page 220, Creating a Web UI test with Test Genie on page 222, and Modifying a Web UI test created by using Test Genie on page 224.
Support for the Anthropic AI provider - Tech preview	You can now add and manage the Anthropic AI provider on the **Cloud Credentials** page in Test Hub. You can then use Anthropic to generate tests by using Test Genie, or to populate schema generator configurations when auto-assigning generators and configurations to each field in a data definition. See Management of cloud credentials on page 587, Test creation with Test Genie on page 220, and Auto-assigning generators and AI-provided data configurations to data definitions in the Schema View on page 479.
Personal agents for local test execution and debugging	You can now download, install, and start a personal agent directly from the **Test Editor** page. A personal agent is a local agent that provides a dedicated, isolated runtime environment to run Web UI tests and suites directly on your local machine so that you can visually observe UI interactions in real time and identify the exact point of test failure. See Management of the personal agent on page 567, Installing and starting the personal agent on page 567, and Stopping or unregistering the personal agent on page 571.
MCP support for authentication	MCP clients can now dynamically register with the authorization server (Keycloak) without requiring an offline token to be passed through the authentication header. When you connect to an MCP server endpoint in Test Hub to access server-side tools directly from MCP clients, such as VS Code, you must provide your credentials to authenticate the connection. See Configuration of an MCP server endpoint on page 194 and Configuring an MCP server endpoint in VS Code on page 195.
Support for local repository creation	You can now create standalone local repositories without a remote repository reference. These local repositories operate the same way as the remote repositories and they can optionally be connected to an empty remote repository at a later point for synchronization. See Adding a repository to a team space on page 502.
Support for running server test assets and server suite assets on device clouds	You can now run server test assets and server suite assets on Android or iOS devices that are available through supported device clouds, such as BrowserStack Cloud, LambdaTest Cloud, and pCloudy Cloud. See Running server test assets by using a cloud credential on page 813 and Configuring a run of a Web UI test to run by using a cloud credential on page 842.
Customization of environments with profile overrides	You can now configure profile overrides in custom environments so that you can easily change the target profile without changing your test script. When you run a test or a suite with a profile override, the default profile, such as a specific browser or emulated device, is substituted with a different target profile, such as a cloud-native device or alternative

Feature title	Description
	browser. See Test environments on page 337, Creating an environment on page 338, and Modifying an environment on page 340.
Support for default environment selection and variable promotion	You can now reuse the last selected environment as the default in authoring and execution flows. You can also promote variables to environment-level or secret-backed values with clearer variable state and type indicators. See Test environments on page 337, Modifying an environment on page 340, Running server test assets on page 808, Creating and assigning test variables on page 312, Protecting test assets by using secrets on page 1235, and Creating a secret in a security group on page 1236.
Device emulation support on Safari for Web UI test	You can now record and play back Web UI tests on the Safari browser in the device emulation mode. You can use the Phone (X-Large) and Tablet (X-Large) options that are available for Safari on Mac. See Web UI test creation on page 210.
Support for downloading the agent from the project's Agents and Intercepts page	You can now download DevOps Test Runtime from the **Agents and Intercepts** page after navigating to your project from the initial team space. See Installing and registering DevOps Test Runtime agents on page 558.
Support for recording and playback of jQuery UI controls	You can now record and play back tests for supported jQuery UI controls. The Smartshot view includes a **Filter by jQuery** option that helps you identify supported jQuery UI controls when editing tests. See Editing a Web UI test on page 236.
Support for conditional flow control actions in Web UI tests	You can now use conditional flow control actions such as **If, While, and Do-while** directly in Web UI test steps. These actions support smartshots and use the same decision logic as verification points, so that you can perform steps based on the presence or state of a UI widget. The **Pause** step is renamed to **Wait** and now supports both **UI** and **General** modes. See Adding an If step on page 262, Adding a While step on page 265, Adding a Do-While step on page 268, and Adding a Wait step on page 246.
Support for multi-select actions in Test Navigator	You can now select multiple assets in the Test Navigator and perform actions, such as editing, moving, and deleting, on all selected assets, which makes asset management more efficient. See Test navigator panel on page 200.
Support for moving or deleting tests with dependent assets	You can now move or delete tests along with their dependent assets. Assets that are referenced exclusively by a test are processed together with the test, while shared assets remain unaffected. References to shared assets are updated automatically to prevent any impact on test run. See Test navigator panel on page 200.
Easy navigation in Test Editor to access the failed and error steps in suites	When you play back test suites in **Test Editor**, the three color-coded boxes display the counts of passed steps, failed steps, and errors. You can now view the step details of the first failed step or step with an error by clicking the respective color-coded box. See Running a suite sequentially on page 299 and Running a suite concurrently on page 300.

Feature title	Description
Support for viewing individual test results in Test Editor	You can now select and view individual test results directly within **Test Editor**. Result information is displayed in **Test Editor**, and step-level results are highlighted when the selected result matches the current version of the test. See Running a Web UI test on page 228.
Support for viewing configuration details in functional reports and test logs	You can now view configuration details directly in the functional report and test log. Configuration nodes for tests and modules display the initial variable values and dataset information to provide better visibility into the configuration used during test runs. Variable information is displayed in a more streamlined format to improve report readability and analysis. See Test logs on page 1007 and Viewing a Functional Report on page 994.
Support for generating negative test data from data definitions	You can now generate negative test data from data definitions to validate error handling and input validation scenarios. You can configure the type and amount of negative data that is generated, preview the generated data, and create datasets that combine invalid values, optionally, combined with valid data to support more effective, comprehensive testing. See Generating negative test data for a data definition on page 486.
Support for generating negative test data by overriding datasets with data definitions	You can now override a dataset with a Data Definition File (DDF) during test execution and select the type of test data generation. You can choose to generate either **standard** or **negative** test data and configure the applicable generation options before running the test. See Configuring a run of a Web UI test on page 795.

What's New in 11.0.8

The following section lists the features introduced in Test Hub 11.0.8.

Feature title	Description
Support for downloading the agent from the Agents and Intercepts page	You can now download the DevOps Test Runtime from the **Agents and Intercepts** page to install and register the system as a local agent for playing back Web UI tests. See Installing and registering DevOps Test Runtime agents on page 558.
Support for device emulation	You can now use the device mode of browsers to record and run tests or suites. Test Hub supports a set of device profiles for phones and tablets of four different sizes. Depending on the device size that you select, the device opens in Edge or Chrome browser. See Web UI test creation on page 210, Recording a Web UI test on page 211, Running server test assets on page 808, Running server suite assets on page 816, and Running server test assets on an agent on page 811.
Support for running UI tests across multiple browsers in a single run	You can now run Web UI tests across multiple browsers in a single test run. This feature allows you to configure multiple browser environments and run the test against them simultaneously, helping you validate application behavior across different browsers more efficiently. See Running suites on multiple browsers and devices on page 304.

Feature title	Description
Support to define custom environments for test runs	You can now define custom environments to manage sets of variables such as application URLs and user credentials. When you run a test by using the environment, the variables defined in the environment dynamically replace the values within your test script. See Test environments on page 337, Creating an environment on page 338, and Modifying an environment on page 340.
Support for wildcard patterns to include or exclude multiple tests or suites for a test run	You can now use wildcard patterns to include or exclude multiple tests or suites by their name for a test run. You can configure the suite to select tests or suites by adding wildcard patterns. You can use a single asterisk (*) to match items within the same folder, or double asterisks (**) to match items in multiple subdirectories. See Running multiple suites or tests by using the wildcard pattern on page 302.
Support for individual results for test run as part of a suite	You can now view results for child tests that run as part of a parent suite. Each child test generates its own result, making it easier to review failures. These results are listed in the Results tab, where you can identify failed tests and analyze their results. See Test suite management on page 292 and Test results and reports on page 985.
Support for viewing the run status of asset during publishing	You can now view the run status of assets when reviewing changes on the **Changes** page before publishing. Each executable asset displays a colored status indicator in the parent row, so that you can view the most recent run result for that specific revision of the asset. See Publishing all changes in an Edit branch on page 493.
Enhanced test repair experience	You can now repair tests by using the **Show Result** option. You can view the result details, including run status, duration, statistics, and reasons for failures. You can analyze results and verify step details without switching back to edit mode. See Viewing step-level results and updating tests on page 232.
Enhanced smartshots access to update tests	You can now access additional smartshots to update the test by using the **Replace Smartshot** option in the step details pane. The **Result Smartshots** section displays smartshots from the last test run, and the **Application Smartshots** section displays all previously captured smartshots for the application. See Viewing step-level results and updating tests on page 232.
Support for multiple dataset configurations in server assets	You can now configure multiple datasets for server test assets, such as Tests and Suites to assign dataset values to test variables during the test run. This feature supports dynamic data usage during test runs by allowing additional datasets to populate test variables, enabling subsequent steps to use unique and data-driven values. See Multiple datasets in tests and test suites on page 333.

What's New in 11.0.7

The following section lists the features introduced in Test Hub 11.0.7.

Feature title	Description
Support for running Web UI tests on Safari	You can now install the DevOps Test Runtime on the Mac operating system to register the system as an agent and then play back Web UI tests on the agent by using Safari. If the agent is stopped, any new test run attempt enters an UNCLAIMED state. When you start the agent, the test automatically runs and moves through the necessary transition states to complete the test run. See the following topics: • Management of DevOps Test Runtime agents on page 557 • Installing and registering DevOps Test Runtime agents on page 558 • Viewing agents that are registered with DevOps Test Hub on page 563 • Adding an agent to a project on page 564 • Removing an agent from a project on page 565 • Unregistering a remote agent from DevOps Test Hub on page 566 • Running server test assets on an agent on page 811
Support for the new MCP Server endpoint of Test Hub	You can now access server-side tools directly from MCP clients, such as VS Code, by connecting to the new MCP Server endpoint of Test Hub. See Configuring an MCP server endpoint in VS Code on page 195.
Support for an external database to deploy the product	Test Hub now includes support for an external PostgreSQL database for product deployment, giving you flexibility and control over your resource management. See Configuration of an external database on page 113.
Specifying a browser for the test run in Test Editor	You can now specify the browser that should be used for the test run from **Test Editor**. You can click the **Configure settings and data** step to select a browser from the list. See Running a Web UI test on page 228. See Running a Web UI test on page 228.
Notification support for concurrent asset changes	If an open asset is modified, deleted, or discarded by another user, Test Hub now notifies the other users about the changes. See Publishing all changes in an Edit branch on page 493 and Edit branch overview on page 198.
Easy navigation in Test Editor to access the failed and error steps	When you play back the Web UI test in **Test Editor**, the three color-coded boxes display the counts of passed steps, failed steps, and errors. You can now view the step details of the first failed or error step by clicking the respective color-coded box. See Running a Web UI test on page 228.
Support for downloading the result summary list	You can now navigate to **Analyze > Results**, and then click the **Download** icon to download the list displayed on the **Results** page. The result summary list is downloaded into your local file system as a Comma-Separated Values (CSV) file. See Test results and reports on page 985.
Redesigned the user interface for the Changes page	The **Author > Changes** page is now redesigned to display modified assets in a nested format, showing parent assets and their related assets. If a `.jpeg` or `.json` file is referenced by multiple tests, then such assets are listed independently. Also, when

Feature title	Description
	you discard the changes for a parent asset, the changes to the related assets are also discarded.
Support for auto-assigning generators and configurations to fields of data definition in the schema	You can now automatically assign generators and configurations to the fields of a data definition in a schema that is associated with a database. By using AI-based inference, Test Hub analyzes sample data and determines the most suitable generators and configurations for each field. When you create a schema with a database connection and export multiple tables from the database view into the schema view, the generators can be auto-assigned to all table fields. See Managing data definitions in a schema on page 475.
Support for editing datasets in the Visual view	You can now create and edit a dataset in **Test Editor** either through the catalog in the **Visual** view or through the written code in the **Code** view. However, for encrypted datasets, switching to the **Code** view is disabled to ensure data confidentiality. See Editing a dataset in Test Editor on page 325.
Support for composite primary keys in DB-associated schemas	Test Hub now supports composite primary keys during data generation, ensuring unique key combinations and preventing duplicate rows to produce accurate datasets.

What's New in 11.0.6.1

This fix pack release focuses solely on stability improvements and defect fixes. No new features are introduced in this release.

What's New in 11.0.6

The following section lists the features introduced in Test Hub 11.0.6.

Feature title	Description
Support for defining and managing retention rules to clean up the test results	You can now create, edit, and manage retention rules to automatically clean up test result data that is outdated or no longer needed. This feature helps prevent the system from running out of resources. See Management of resources on page 190.
Support for monitoring resource usage at the team space level	Administrators can now view resource usage for projects within a team space, allowing them to make informed decisions about the projects that can be simplified. See Viewing resource usage statistics on page 193.
Support for recording Web UI tests on the Safari browser	You can now record Web UI tests by using the Safari browser on Mac operating systems. See Recording a Web UI test on page 211.
Support for copying Web UI test steps from one test to another	Now, you can copy a set of test steps and paste them into another existing test or an empty test. Additionally, Test Hub copies any variables associated with the copied steps. See Copying test steps from one test to another on page 274.

Feature title	Description
Enhanced support to identify objects	Now, you can use any of the conditions such as Contains, Doesn't contain, Starts with, Ends with, Doesn't end with, Doesn't start with, Equals, Doesn't equal, Matches regex, and Doesn't match regex to identify an object for a step. See Modification of Web UI tests on page 235.
Support for the creation of datasets from variables	You can now create datasets from a set of variables that are configured in a test or a suite in **Test Editor**. The variables are displayed as columns in the dataset. See Creating a dataset by using test variables on page 324.
Support for including additional properties in test results	You can now configure test runs to include additional properties that are available in webhook templates. These additional properties allow result events to be correlated when using systems such as HCL DevOps Velocity. See Adding additional properties in test results on page 1025.
Support for refreshing row counts and distribution ratios	You can now refresh both default row counts and default distribution ratios of the data definitions within the schema based on the latest data from the source database. See Managing data definitions in a schema on page 475.
Support for auto-assigning generators to fields of data definition in the schema	You can now automatically assign generators to fields of data definition in a schema that is associated with a database. When you create a schema with a database connection and export multiple tables from the database view into the schema view, the generators are auto-assigned to all table fields. See Managing data definitions in a schema on page 475.
Support for removing data definitions from the schema view	You can now remove one or more data definitions directly from the schema view. See Managing data definitions in a schema on page 475.
Support for viewing resource consumption in results	You can now view the total disk usage and manage storage more effectively. A new **Size** column on the **Results** page displays the total usage across all associated resources, including logs, reports, and artifacts, in human-readable units (KB, MB, GB). See Test results and reports on page 985.

What's New in 11.0.5

The following section lists the features, enhancements, or other changes made in Test Hub 11.0.5.

Feature title	Description
Enhanced test data management with schema support	You can define and manage test data by using custom schemas in Test Hub. See Management of schemas on page 469.
Support to generate and export schema-based test data	You can generate schema-based fabricated test data and export it as a `.zip` file or to a connected database in Test Hub. See Generating and exporting schema-based test data to

Feature title	Description
	a .zip file on page 490 and Generating test data by using data definitions or schema and exporting it to a database on page 491.
Support for Web UI test recording on Mac and Linux	You can now record Web UI tests in Test Hub on the Mac and Linux operating systems. Before you record a Web UI test for the first time, you must download and install DevOps Test Runtime or the recording agent. See Recording a Web UI test on page 211.
Creating a test from an existing set of steps	You can now select a set of steps that are in sequence in an existing test and create a new test with the selection. See Creating a test module from an existing set of steps on page 277
Support to generate Web UI tests by using designs with Test Genie - Tech preview	You can now create Web UI tests by using design screens of the application with the help of Test Genie. Test Genie leverages machine learning and AI capabilities to create tests by using the designs of web applications under development. You can upload wireframe images of web applications that are available in the early stages of development to generate tests. After the web application is deployed and available to run tests, you can play back the tests. See Test creation by Test Genie - Tech Preview, Creating a Web UI test by using Test Genie, and FAQs: Test generation by Test Genie.
Updating test steps with smartshots	You can now update the steps of a test by replacing the existing smartshot with the smartshot captured from the last test run. See Viewing step-level results and updating tests on page 232.
Support to manage and create assets directly from the navigator folders	You can create new assets directly by right-clicking on the physical and logical folders in the navigator panel. See Test navigator panel on page 200 and Data navigator panel on page 402.
Renaming test assets	You can now change the name of test assets such as test suites, smartshots, and applications in your project. See Renaming a test asset on page 275.
Deleting test assets	You can now delete test assets such as tests or smartshots, that are no longer required in your project. If the selected assets are referenced in a test or test suite, then Test Hub displays a warning. See Deleting a test asset on page 275, Deleting a suite on page 310, and Test navigator panel on page 200.
Support for settings and variables step in visual editor	You can configure global settings such as default timeouts and manage variables directly from the visual editor page by using the **Configure settings and variables** step.
Support to record and play back JavaScript dialog	You can now record and play back the JavaScript pop-up boxes such as alert, confirm, or prompt dialogs that are displayed by web applications.
Support to record and play back the drag-and-drop action	You can now record and play back the drag-and-drop actions on web pages as part of the Web UI test for your applications.

Feature title	Description
Support to configure test variables in a test or suite within the Run step	You can now configure input, output, and local variables in a test or suite within the **Run** step and easily map them during the test run. See Configuring test variables in the Run step of a test or suite on page 320.
Support to assign values to test variables from a dataset by using the Assign step and Run step	You can now assign values from a dataset to test variables by using the **Assign** step and **Run** step. This feature supports dynamic data substitution during the test run, allowing subsequent steps to use unique, and data-driven values. See Adding an Assign step on page 242 and Assigning test variable values from a dataset by using the Run step on page 318.
Support to navigate from a functional report to a test step or a suite	You can now quickly navigate from a functional report directly to the related test, suite, or step. You can also locate test resources and resolve issues more efficiently. See Navigating to a test, suite, or step from the functional report on page 997.
Support to pushing a local Edit branch to a remote repository during a publish conflict	You can now push a local edit branch to a remote repository when a conflict prevents publishing. After pushing the branch, you can create a pull request and manually resolve conflicts in Git before merging the changes into the target branch. See Publishing all changes in an Edit branch on page 493.

What's New in 11.0.4

The following section lists the features, enhancements, or other changes made in Test Hub 11.0.4.

Feature title	Description
DevOps Test Runtime for recording Web UI tests	Test Hub now provides DevOps Test Runtime or the recording agent to record Web UI tests. You can download, install, and update the agent before recording tests. See Recording a Web UI test on page 211.
Navigating test assets in Test Editor	You can view virtual folders at different levels, apply filters by asset type, and link editor tabs to their respective locations in the test navigator panel. See Test navigator panel on page 200.
Support to create and modify web applications in Test Editor	You can create and modify web applications to record or create a Web UI test. See Creating an application on page 204 and Modifying application components on page 206.
Support for creating and managing Web UI Tests	You can now create and manage Web UI tests. You can record tests in the supported browsers such as Google Chrome, Microsoft Edge, or Mozilla Firefox by using the recording agent. You can select an application that is configured in Test Hub and perform user actions to record test steps. After the recording is complete, the test steps are displayed in **Test Editor**. Alternatively, you can create tests by selecting the empty test option and adding steps through the **Visual** or **Code** view. See Web UI test creation on page 210,

Feature title	Description
	Recording a Web UI test on page 211, Creating a Web UI test by using the empty test option on page 217.
Support for play back of Web UI tests	You can now play back the Web UI tests that you created in Test Hub in the supported browsers such as Google Chrome, Microsoft Edge, or Mozilla Firefox. **Test Editor** displays the progress of the test run and you can view the information such as error messages, logs, and the test verdict in the **Console** output view. After the playback is complete, you can view the step-level results and reports. See Running a Web UI test on page 228, Accessing the console output on page 230, and Viewing step-level results and updating tests on page 232.
Support for modification of Web UI tests	You can now use the following options to update Web UI tests that you have recorded by using **Test Editor**: • Add user actions such as click, type, select, and key press for window controls and other supported keys • Add conditional logic by using If, While, and Do-While statements • Embed a test within a test • Add a test step that aborts the test run • Add a test step to pause the test for a specific duration • Add verification points to verify an objectÃƒÂ¢Ã¢â€šÂ¬Ã¢â€žÂ¢s property • Define the timeout for a test step • Disable a step See Modification of Web UI tests on page 235.
Smartshots for updating tests	You can now add and update user actions, and test steps by using smartshots in **Test Editor**. See Adding user actions in a test from smartshots on page 239.
Support for variable creation and assignment in Web UI tests	You can now create variables and assign values to them. Additionally, you can assign the value of a variable to a test element or assign the value of a UI object property to a test variable. See Test variables on page 311.
Support for creating and managing the Test suites	You can now create and manage a suite and also combine the multiple tests and suites into a single workflow and organize them into a suite. You can run the test suites in the suite either sequentially or in parallel. You can also run the failed tests and suites. See Test suite management on page 292.
Viewing functional reports for Web UI tests and suites	A new **Functional Report** is now available for the Web UI tests and suites that you create and run on Test Hub. The report contains the test results details such as the summary of the test run, test environment details, and screenshots. See Viewing a Functional Report on page 994.

What's New in 11.0.3

The following section lists the features, enhancements, or other changes made in Test Hub 11.0.3.

Feature title	Description
Rollout of the new license and download portal	Test Hub licenses and downloads from version 11.0.3 onwards are now available on the My HCLSoftware (MHS) portal. MHS is a cloud-based web application designed to help you manage your software entitlements and licenses effectively and this portal replaces HCL® License & Download portal (FlexNet Operations Cloud).
Support for Functional Report (Experimental)	A new **Functional Report** is available now for Functional tests, API tests, Performance tests, Compound tests, AFT suites, and API suites that you run on Test Hub. The report contains a summary of the execution, a test environment, a detailed view of all tests, and test steps. See Viewing Functional Report (Experimental) on page 994.
Enhanced the execution of SAP GUI tests on DevOps Test Hub	You can now run SAP GUI tests that were created in DevOps Test UI on Test Hub by using a remote performance agent that is installed with the SAP GUI runtime. See Configuring a run of an SAP GUI test on page 822.
Revamped Account Settings options in the User Account Menu	The Account Settings option now opens the **General Settings** page with the following three panels: • User Profile • Third-party License Agreements • Notifications The **User Profile** panel displays user information and includes a button to open Keycloak in a separate window. The **Notifications** panel is redesigned, which features collapsible sections and checkboxes instead of dropdowns. Additionally, the **Notification Settings** option is removed from the **User Account** menu.
Streamlined team space management and navigation	The **Manage** tab is now the primary option for managing licensing, webhooks, and configuration within a team space. The new layout is redesigned to match the layout of Projects page, which makes it easier to access team space contents and switch between team space and projects.
Improved user interface on system model page of the team space	The search and publish options are moved to the **Content** panel, and the system model page has been restyled to match the design of Project pages.
Improved user interface on infrastructure and	The infrastructure and virtualization pages are redesigned to match the layout of the Project pages.

Feature title	Description
Virtualization pages of the team space	
Enhanced user interface on the Data Navigator Panel of the Data page	The Data Navigator panel now has the option view virtual folders at different levels, apply filters by asset type, and link editor tabs to their locations in the navigator.
Enhanced the process for connecting Docker performance agent to Test Hub	You can connect Docker performance agent to Test Hub by generating an offline token for authentication and provide the necessary connection information during the test run. See Connecting a Docker performance agent to Test Hub on page 543.
Enhanced the process of running APIs outside the Swagger UI with offline tokens and bearer access	You can now run APIs outside of the Swagger UI by generating an offline token and adding a bearer access token with each API call. See Running APIs in command line on page 1197.

What's New in 11.0.2

The following section lists the features, enhancements, or other changes made in Test Hub 11.0.2.

Feature title	Description
Enhanced Machine learning (ML) insights on Performance tests	A new parameter called **Performance Agent Health** is now added in Performance tests or Schedules to improve machine learning (ML) insights for performance testing. See Machine learning capabilities in DevOps Test Hub on page 1027.
Support to run SAP tests on DevOps Test Hub	You can now run the SAP tests that were created in DevOps Test UI on DevOps Test Hub. For more information, refer to Task flow for running SAP tests on DevOps Test Hub.
Enhancements to the Integration user interface	The Integration feature now includes an enhanced user interface. Previously, the **Change Management** and **Test Management** were presented in separate tabs. Now, they are integrated into a single improved user interface that offers seamless management of both Test and Change integrations.
Enhancements to the Webhooks page	The Webhooks page now features an enhanced user interface. The appearance of the **Templates** and **Webhooks** tabs is improved. Further, the **Edit channel type** option is relabeled as **Channel Type**, and the options **Show all levels** and **Hide other levels** are replaced with a default-selected **Show all levels** checkbox.

Feature title	Description
Enhanced the user interface on the Configuration page	The **Configuration** page is updated and now features a more refined user interface. Previously, the project configuration details and repositories were presented in separate tabs. Now, they are combined into one streamlined user interface. You can now seamlessly manage configuration details and repositories.
Enhanced user interface of the Edit branch dialog	The **Edit branch** dialog is updated and now features a more refined user interface. Previously, if your edit branch was trailing behind the source branch, there were neither alerts nor any options for you to pull the changes. Now, with the improved user interface, a warning message is displayed if your chosen edit branch is behind the source branch. Additionally, you can easily pull the changes from the source branch to your edit branch by using the new checkbox.

What's New in 11.0.1

The following section lists the features, enhancements, or other changes made in Test Hub 11.0.1.

Feature title	Description
Adding an external database connection and importing data to create test assets	You can now add an external database connection to a project in Test Hub, and then create a database query to import data from the database. You can also create test assets such as datasets and Data definitions by using the database query.
Enhanced the UI with the new Security page	You can now use the Security page to create and manage API secrets and dataset classifications. You can also add the credentials of external databases to data classifications. You can now provide Read access to the classifications at the role level, while Write access is restricted. The Access Control List UI is enhanced so that you grant both Read and Write access to members with classification privileges. See Protecting test assets by using secrets on page 1235, Managing an encrypted dataset on page 447.
Installing and configuring the DevOps Test Hub Results Data plugin by using a proxy connection	You can now install and configure the DevOps Test Hub Results Data plugin by using a proxy connection in Grafana instead of using additional helm parameters during the installation of Test Hub. See Installing and configuring the DevOps Test Hub Results Data plugin on page 1000.

Feature title	Description
Viewing resource monitoring usage data on Grafana	You can now view the resource monitoring usage data by creating queries on Grafana, and retrieve this data that is collected for a certain period during a test run in Test Hub. See Viewing resource monitoring usage data on Grafana on page 1089.

What's New in 11.0.0

The following section lists the features, enhancements, or other changes made in Test Hub 11.0.0.

Feature title	Description
Renaming of OneTest products	The following OneTest products have new names starting from 11.0.0: • HCL OneTest™ Server is now HCL DevOps Test Hub. • HCL OneTest™ UI is now HCL DevOps Test UI. • HCL OneTest™ API is now HCL DevOps Test Integrations and APIs. • HCL OneTest™ Performance is now HCL DevOps Test Performance. • HCL® Quality Server is now HCL DevOps Test Virtualization Control Panel. • HCL® OneTest™ Embedded is now HCL DevOps Test Embedded. • HCL OneTest™ Virtualization is now HCL DevOps Test Virtualization. • HCL OneTest™ Studio is now HCL DevOps Test Workbench.
Enhancements to the data fabrication feature	The data fabrication feature is enhanced and you can now perform the following tasks: • Create an empty Data definition, or import the header row fields from a CSV file or a dataset. You can then add the basic generators or modify the generators to suit your test data fabrication requirements. • Create folders in your project repository and select them as locations to save your data definitions. • Use the Save All option to save all files that are opened for editing or have unsaved changes. • Use the right-click options to close any or all files that are opened for editing or have unsaved changes. • Use the distribution parameter, which is enhanced to automatically allocate and rebalance the distribution among the existing and new configurations that are added subsequently. • Preview the sample test data for each configuration of a generator.
Changes to the Licensing environment variables	With the renaming of HCL OneTest Products to HCL DevOps Test Products in 11.0.0, the HCL Licensing environment variables that contained *ONETEST* are changed to be used without *ONETEST*. For example, the *HCL_ONETEST_LICENSING_URL* environment variable

Feature title	Description
	now must be called *HCL_LICENSING_URL*. Note that the old environment variable names are still supported.
Support to view configuration details of a Virtual Service instance	You can now view the configuration details of a Virtual Service that you configured during a run from the Instances page. The details such as the datasets selected for overrides, variables, JVM arguments, environment variables, and additional parameters are displayed, if they are configured.
Enhancements to editing datasets	The editing capabilities of datasets are further enhanced in Test Hub 11.0.0. You can now perform different actions to edit the dataset from the UI by using the Editing panel, the context menu options, and the insert options available in the Dataset panel.
New options to edit and save Data definitions	You can now use the right-click options for Data definitions from the Data Navigator to open the Data definition for editing or saving it as a different Data definition.

What's New in 10.5.4

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.5.4.

Feature title	Description
Generation of test data enhanced through redesign	The test data generation that was previously possible by using HCL® OneTest™ Data is enhanced through the redesign of the user interface and the supported architecture and is now available in a new avatar. You can use the in-built data generators or create customized data generators to suit your test data requirements. You can preview the test data in the UI before you generate the test data. You can also download the generated test data. You can view the history of the test data generated, and you can regenerate the test data with or without altering any of the parameters used. See Data fabrication.
Support for Machine learning insights on performance tests	HCL OneTest™ Server is enhanced to provide Machine learning (ML) insights on performance tests that match certain criteria when you run tests from projects in a team space. HCL OneTest™ Server uses ML algorithms to analyze the following types of parameters in performance tests:

Feature title	Description
	• **Response Time Lock-Step Pattern** • **Response Time Standard Deviation Pattern** • **Throughput Drop Pattern** The findings from the analysis are presented as a recommendation for each of the parameters on the **Infrastructure > Analyzers** page in the team space. You can improve ML recommendations by altering the default **Threshold** value that specifies the level of confidence required in the insights. You can also improve the accuracy of the recommendations by running multiple tests with differing loads repeatedly. See Machine learning capabilities in DevOps Test Hub on page 1027.
Enhanced the dataset editor and managing capabilities	The CSV Editor that is used to edit datasets is enhanced and now opens within the HCL OneTest™ Server UI, which previously opened in a separate browser tab. Further, the pages and the methods to create, open, or view datasets are changed in their presentation in the UI. See Management of datasets on page 424.
Enhanced dataset classification	You can now set the dataset classification (encrypted password) for a dataset irrespective of its location in branches in a repository. The dataset classification that you set is applied to all branches and you need not set a password for each branch that contains the dataset. If you want to use different passwords, then you can create a different dataset by cloning the dataset that you want. See Managing an encrypted dataset on page 447.
Support for in-built Chrome and Edge browsers	HCL OneTest™ Server now provides support for running browser-based tests by using the in-built Chrome and Edge browsers after you accept the license terms for usage. If you do not accept the license terms at the time of signing in or later, tests that use these browsers fail to run successfully. See Management of third-party licenses on page 160.
Support to run AFT Suite on a mobile device by using variables	You can now run an AFT Suite, which contains mobile or Web UI tests on a device, emulator, or simulator that is connected to an agent of a device cloud. You must enter the details of the agent or device cloud as variables while you configure a run of the AFT Suite. See Configuring an AFT Suite to run on a mobile device by using variables on page 632.

Feature title	Description
Support to override the project in a mobile device cloud	You can now select or create a project to override the project that is configured for a cloud credential when you want to run a mobile test, Web UI test, Compound test, or an AFT Suite. The tests run on mobile devices that are connected to a mobile device cloud. See Configuring an AFT Suite to run on a mobile device by using a cloud credential on page 644, Configuring a run of a Compound Test that contains mobile tests to run by using a cloud credential on page 693, or Configuring a run of a Compound Test that contains Web UI tests to run by using a cloud credential on page 732.
Support to add reports and logs to Jira issues	After you configure Jira as a change management system on HCL OneTest™ Server, you can now add the test reports or test log to the Jira issue that you create for a test result. You can then view the attached reports or log from your project in Jira. See Creating Jira issues for the test results on page 1096.
Support to view test log statistics in Grafana	You can now view the overall statistics and time-series statistics in Grafana by using queries for the test logs that are generated in HCL OneTest™ Server. See Viewing test results data in Grafana on page 1003.
Enhanced the display of events in Web UI or Functional test logs	The display of the events in the Web UI or Functional test log is enhanced. Previously, the events were displayed under the **UIScript** node and placed under the **Test** node, which did not represent the correct hierarchy. Now, the events are placed under the **Test** node and the **UIScript** node is no longer displayed, thus displaying the correct hierarchy of the events in the test log.
Requirement of Java 17 for Resource Monitoring agents	The Resource Monitoring Java Agent now requires Java 17 JRE or JDK to be installed on the computer that is used to collect the Resource Monitoring data from HCL OneTest™ Server.

What's New in 10.5.3

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.5.3.

Feature title	Description
Support for Revenera 2022.12	With Revenera 2018.02 (LLS 1.0) reaching EOS, you can now use Revenera 2022.12 (LLS 3.0) as a local license server to host licenses for HCL OneTest™ Server.

Feature title	Description
Support to run mobile tests on device clouds	You can now configure a connection from HCL OneTest™ Server to a mobile device cloud by using the credentials of your cloud subscription account. You can then add team space members as authorized users of the cloud credential. The authorized users can run mobile tests on the mobile devices that are connected to a mobile device cloud. See Management of cloud credentials on page 587.
Enhancement to the webhooks feature	The webhooks feature is enhanced to enable you to configure the behavior of notifications that were not sent because the webhook was disabled. You can decide to discard or send the queued notifications when you re-enable the webhook. Webhook templates defined at the team space are now also displayed at the project level. See Configuration of a webhook on page 1248.
Support to download test reports	You can now download the reports and logs of the tests from the **Results** page. Reports can be saved to a folder on your computer as a compressed file or PDF, which depends on the format of the generated report. See Test results and reports on page 985.
Support to override input tags of virtual services	You can now specify values for stub input tags by using advanced configuration parameters when you run a virtual service. See Additional configuration parameters for virtual services on page 955.
Support to add an Azure repository	You can now add an Azure repository that contains test resources to your project in HCL OneTest™ Server, and then run the resources from the server. See Repository considerations for a server project on page 1211.
Removed support of Prometheus for virtual services	Virtual service usage in HCL OneTest™ Server no longer utilizes Prometheus. If you upgrade from a previous version of HCL OneTest™ Server to 10.5.3, any virtual service usage that was captured before 10.5.3 will not be visible in the team space or project virtualization usage pages.

What's New in 10.5.2

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.5.2.

Feature title	Description
Running Selenium tests on HCL OneTest™ Server	You can now run Selenium tests that are within a Maven project from HCL OneTest™ Server. You must commit the Maven project as a `pom.xml` to a Git repository. After you add the Git repository to your project on HCL OneTest™ Server, you can configure a run of the Selenium test from the **Execution** page. See Configuring a run of a Selenium test on page 909.
Running virtual services on API agents from HCL OneTest™ Server	You can now configure an HCL OneTest™ API Agent to register with a team space in HCL OneTest™ Server by using the server hostname, team space name, and your offline token. When you start the virtual services, you can select the HCL OneTest™ API agent as the location and not select the Kubernetes cluster. See Management of API agents on page 551.
Support for IBM® Rational® ClearQuest® as a change management system	You can now create Rational® ClearQuest® work items for the tests that ran on HCL OneTest™ Server, after you configure Rational® ClearQuest® as a change management system on HCL OneTest™ Server.
Configuring external webhooks on HCL OneTest™ Server	You can configure the URL of third-party applications such as Teams or Slack, as a webhook on HCL OneTest™ Server when you want to receive push notifications about events that occur on the server. You can create templates and select specific events for which you want to receive notification messages in the defined template for those events. See Configuration of a webhook on page 1248.

What's New in 10.5.1

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.5.1.

Feature title	Description
Support to export test results to a Jira Xray project	When you use Atlassian Jira as a change management system on HCL OneTest™ Server and you use Xray as the test management application in Jira, you can now integrate Xray with HCL OneTest™ Server. With this integration, you can export the test results of the tests that run on HCL OneTest™ Server to an issue in your Jira Xray project. You can then view and use the test results in your project in Jira. See Integration with Xray test management for Jira on page 1188.

Feature title	Description
Copying of the generated keys individually	When you configured Atlassian Jira as a change management system on HCL OneTest™ Server, you were able to copy the public and consumer keys in a single action. However, it was difficult to differentiate the keys when you pasted them in the individual fields on the Jira server configuration page. You can now copy the public and consumer keys separately and use them to configure the Jira server.
Support to create defects on tests run by other project members	Previously, you were able to create defects only on tests that you ran on HCL OneTest™ Server. You can now create defects in the change management system configured in your project on tests that are run by other project members too.
Enhanced publishing of association changes in the system model	Previously, you were able to publish the changes made to the component associations with test resources in the system model from the **Author > Changes** page. You can now review and publish the association or dissociation changes of test resources with components in a system model from the **System Model** page. You can use the  icon on the **System Model** page to publish the association changes.

What's New in 10.5.0

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.5.0.

Feature title	Description
Support to install Jaeger as part of HCL OneTest™ Server is removed	The support to install Jaeger during the installation of HCL OneTest™ Server is removed.
Support for HCL® Compass as a change management system	You can now create HCL® Compass records for the tests that you run on HCL OneTest™ Server, after you configure the HCL® Compass server as a change management system on HCL OneTest™ Server.
Running tests on remote Kubernetes clusters	You can now run the supported test resources on remote Kubernetes clusters that you register with a team space on HCL OneTest™ Server and add the Kubernetes cluster to your project in the team space. You can select the remote cluster as a location to run the tests when you configure a run. You can edit the configurations of a cluster, remove a cluster from a project, or unregister the cluster from a team space.

Feature title	Description
	See Management of remote Kubernetes clusters on page 572.
Support to run tests or resources as a GitHub Action	You can now run tests or resources that are contained in your project on HCL OneTest™ Server as a GitHub Action from a Git repository. See Configuration of test runs in a GitHub Actions Workflow on page 919.
Creating an *Edit* branch to contain resources before committing to the repository branch	When you want to create or edit datasets, test assets, virtual service resources, or resources that are associated with components in a system model, you must now create an *Edit* branch, which is a temporary branch that will contain these until the time you commit or publish these to a branch in the project repository or the team space repository.
Support to schedule recurring test runs	You can now schedule recurring test runs on HCL OneTest™ Server.
Panning icon removed from the System Model page	The Pan icon that existed previously to pan the view in the **System Model** page is now removed. You can now pan the view by clicking the area and moving the area in the direction you want with the left-mouse button pressed.
Enhancements to the HCL OneTest™ Studio extension	When you run tests from Azure DevOps Pipeline by using the HCL OneTest™ Studio extension, you can now store the JUnit report that is generated for the test in the XML format. See Running DevOps Test Hub tests in an Azure DevOps Pipeline on page 1128.
Enhancements to the Jenkin plugin	You can now view the test results by using the Test Results Analyzer feature in Jenkins if you have installed the Test Results Analyzer plugin in Jenkins. See Running tests from Jenkins on page 1187.

What's New in 10.2.3

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.2.3.

Feature title	Description
Changes to accessing the software from Harbor	You can now access the HCL OneTest™ Server Helm charts from the Harbor repository without providing Okta credentials.

Feature title	Description
Support to view or hide license details	As a system administrator or team space owner, you can now view or hide the details of the **License Server** on the **License Configuration** page.
Configuring Azure DevOps as a change management system	Previously, you might have added Atlassian Jira as a change management system on HCL OneTest™ Server, if you used Jira to create and monitor issues. You can now add Azure DevOps as a change management system on HCL OneTest™ Server, if you use Azure DevOps to create and monitor defects or bugs, issues, or other work items. After you configure Azure DevOps on HCL OneTest™ Server, you can create work items for the test runs from HCL OneTest™ Server and view the created work items on Azure DevOps by clicking the links within HCL OneTest™ Server.
Viewing the location where virtual services ran	On the **Progress** and **Results** pages, you can now view the location where the virtual service instance ran based on the location that you selected when you started a virtual service instance. The location displayed is the name of the remote Kubernetes cluster or the *Default Cluster* if the virtual service ran on HCL OneTest™ Server.
Enhancement to publishing of datasets from HCL OneTest™ Server	HCL OneTest™ Server now provides the capability to publish a dataset subsequent to publications from other users when the changes made to the datasets do not conflict with the changes made by the other users.
Viewing or editing load test profile details for Schedules	You can now view the load test profile details for the Schedules that you added in HCL OneTest™ Performance in the **Description** column on the **Author > Tests** page. You can also edit the load test profile details if you do not want to run the Schedules with the configured profile.
Adding notes to a test result	You can now add notes to a test result and group these notes into a discussion in the **Details** panel on the **Results** page. You can also perform a search for specific content in the notes.
Enhanced the option to export queries from HCL OneTest™ Server	Previously, you were able to export the queries that you created from HCL OneTest™ Server as a curl command. You can now use the **Export** option to export the queries to Shell, Curl, or PowerShell scripts. You can now select the script, the output format, and the number of results that are returned by the query.
Enhanced the viewing of test logs	Previously, the test log displayed all the activities that were captured for a test run and viewing the essential information was tedious. You can now select a view that filters the

Feature title	Description
	activities that you might be interested in as a tester. You can however, view all the activities that are captured for a test run by switching off the filter. The **Test Log** page now displays the view correctly when you return to the page after viewing another page on the server UI. You can also use an option on the UI to view the test log list.
Viewing attachments in the test log	You can now view the attachments such as images, documents, PDFs, scripts, json or xml files, that are captured by the test log. You can view the attachments that are supported for display in a web browser. You can also copy or download the attachments to your computer.
Viewing the test run duration and time from test logs	After you run a test, you can view the calendar time and the duration for each of the test steps from the test log.
Enhancements to integration plugins	While you configure test runs from the integrated product, you can now use additional parameters such as: Datasets, Labels, Secret Collection, Start Date, and Variables in the following integration plugins: • Azure DevOps • HCL® Launch • Jenkins
Support to run virtual services by using server plugins	Previously, you were able to run test assets that were in your project on HCL OneTest™ Server from the integrated applications by using the server plugins. You can now run or start instances of virtual service resources that are in your project on HCL OneTest™ Server from integrated applications such as Azure DevOps, Jenkins, and HCL® Launch.
Support to run additional test assets from Jenkins	You can now run the following types of test assets from the Jenkins server by using the HCL OneTest™ Server Jenkins plugin: • API tests • Functional tests • HCL AppScan CodeSweep • JMeter tests • JUnit tests • Performance tests • Postman resources • Virtual services

Feature title	Description
Viewing test run results on the Azure test plans dashboard	When you use the HCL OneTest™ Studio Azure extension in Azure pipelines to run tests on HCL OneTest™ Server, you can view the test results on the **Jobs** page. If you want to view the results of the test runs on the test plans dashboard, you must set up test plans with test cases for your project in Azure. You must also provide details about the Azure server, your personal access token, the Azure project, the test case, and the test plan while you configure the test run for the test results to display on the test plans dashboard.

What's New in 10.2.2

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.2.2.

Feature title	Description
Support to install HCL OneTest™ Server on OpenShift is removed	OpenShift is no longer supported as a platform on which you can install HCL OneTest™ Server 10.2.2.
Authoring tests on HCL OneTest™ Server	You can now use the test authoring capability of HCL OneTest™ Server that utilizes a containerized version of the HCL OneTest™ Studio desktop authoring products to author tests and virtual services. You can also modify test assets or virtual service resources that you create by using the containerized products or the test assets or resources in the project repositories that you created in the desktop products and committed to the repositories.
Support to add multiple repositories simultaneously	Previously, you were able to add individual Git repositories one at a time to your project on HCL OneTest™ Server. Now, you can add multiple repositories simultaneously that are contained in the same repository domain. See Adding repositories to a server project on page 1214.
Support to configure and run individual test assets	Previously,you were able to configure and run Suites, Compound Tests, or Schedules on HCL OneTest™ Server. Now, you can configure and run individual test assets such as an API test, a Functional test, or Performance test. See Test run configurations on page 610.

Feature title	Description
Support to run instances of virtual service resources on a remote Kubernetes cluster	You can now configure and run instances of virtual service resources on a remote Kubernetes cluster. You can select the remote Kubernetes cluster as the location to run the virtual service resources after you set up and add the remote cluster to your project. See Configuring a run of a virtual service on page 946.
Support to run test assets on remote agents or from a remote Kubernetes cluster	You can now select remote agents or a remote Kubernetes cluster as a location to run the test assets. You can select the remote agent as the location to run the test asset after you set up and add the remote agent or the Kubernetes cluster to your project.
Enhanced Jenkins plugins to run tests from a Jenkins pipeline project	The HCL OneTest™ Server Jenkins plugins are enhanced so that you can now run tests as a job in a Jenkins Pipeline project.
Configuring additional Jira fields	Jira server administrator can now configure additional fields as Required fields in the project on the Jira server to be displayed in the Raise defect in Jira window while creating a defect from HCL OneTest™ Server. See Creating Jira issues for the test results on page 1096.
Collecting and viewing resource monitoring usage data	You can create queries to retrieve monitoring usage data that is collected for a time period during a test run. See Collecting and viewing resource monitoring usage data on page 1085, Configuring a test on page 1085, Viewing resource monitoring usage data on page 1087, and Creating queries for resource monitoring usage data on page 1091.
Viewing test logs	When a test run is complete, a test log report is generated for the tests which are configured to collect test logs data. The test log contains details about the events in the tests and their properties. See Test logs on page 1007.
Querying test log data	You can create queries to view test log data for all the test results in a project or for one particular result. See Test log queries on page 1022 and Configuring and running log queries on page 1023.
Viewing results data in Grafana	By using HCL OneTest™ Server and Grafana, you can retrieve and explore test log data that is generated in HCL OneTest™ Server.

Feature title	Description
	See Test results data in Grafana on page 1000, Installing and configuring the DevOps Test Hub Results Data plugin on page 1000, and Viewing test results data in Grafana on page 1003.
Viewing test results in Surefire format reports	After you execute JMeter, JUnit, API, and Functional tests with advanced settings in HCL OneTest™ Server, you can view the test results in a Surefire format report by using dedicated URLs. See Viewing Surefire format reports on page 1006.

What's New in 10.2.1

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.2.1.

Feature title	Description
Integration with IBM Engineering Test Management	You can now integrate HCL OneTest™ Server with IBM® Engineering Test Management to run the test assets in a project in your team space from Engineering Test Management. After the tests run, you can view the results on the **Results** page. See Integration with IBM Engineering Test Management on page 1166.
Overriding results database	When you want to override the Results Database that is configured in the API Suite project, you can now provide the URL, user name, password, and the database driver as program arguments in the **Advanced** settings in the **Execute test asset** dialog box. You can optionally use an encrypted password that is generated by using the **EncryptPassword** application that is in the installation directory of HCL OneTest™ API.
Running a scan of an application code	You can use HCL OneTest™ Server to scan you application code, similar to using the AppScan CodeSweep plugin in Visual Studio. You must create a `.codesweep` file that contains information about the assets in the repository and the folders with the application code that must be scanned. You can optionally, specify the rules to ignore, folders with code to be excluded from the scan, or the parameters for a verdict of the scan in the file. After you commit the assets to a remote repository, you can add the repository to your project in a team space, and then run a scan of the assets on HCL OneTest™ Server. See Configuration of a scan of an application code.
Running tests on remote Kubernetes clusters	

Feature title	Description
	You can now run the supported test resources on remote Kubernetes clusters that you register with a team space on HCL OneTest™ Server and add the Kubernetes cluster to your project in the team space. You can select the remote cluster as a location to run the tests when you configure a run. You can edit the configurations of a cluster, remove a cluster from a project, or unregister the cluster from a team space. See Management of remote Kubernetes clusters on page 572.

What's New in 10.2.0

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.2.0.

Feature title	Description
Installing server software on Azure Kubernetes Service	You can now install HCL OneTest™ Server on Azure Kubernetes Service that has a Kubernetes environment to run functional, integration, and performance tests. See Installation of DevOps Test Hub on Azure Kubernetes Service on page 116.
Creating and maintaining multiple team spaces	Previously, HCL OneTest™ Server supported only one logical partition as the initial team space. Now, you can create and maintain multiple team spaces in HCL OneTest™ Server. The initial team space is available for all licensed users of HCL OneTest™ Server before you create any other team space. You can perform the following tasks in team spaces: • Configure licenses or licensed users for each team space. • View all team spaces created on HCL OneTest™ Server and the team spaces that you can access from the **Team Space Dashboard** page. • Register Dockers, agents, intercepts, or Resource Monitoring agents with a team space before you use them in your projects of that team space. • Add team space members with the following roles: ◦ *Team Space Owner* ◦ *Project Creator* ◦ *Architect* ◦ *Member* • Isolate the projects from the users who are not members of the team space. See Team space overview on page 172.

Feature title	Description
Configure a repository with any default branch in a team space	Now, you can configure any branch in a repository as the default branch in a team space.
Associating test suites to the components of the System Model	You can now associate test suites to the components of the System Model in addition to the virtual resources. After you associate the test suite, you can then click the test suite to view them on the **Execution** page. See Associating resources with components on page 517.
Starting or stopping virtual services associated with components	After you associate virtual services with components in the **System Model** page, you can then navigate to the **Resources** or **Instances** page to either start an instance or stop a running instance. See Viewing resources that are associated with components on page 519.
Viewing and filtering test assets by their associated components on the Execution page	The components in the **System Model** page that are associated with test assets can now be viewed in the **Components** column on the **Execution** page. You can also use the component as a parameter in a filter rule to obtain test assets associated with a specific component. You can return to the **System Model** page by clicking the component displayed for the test asset in the **Components** column on the **Execution** page.
Viewing and stopping virtual service instances associated with components	After you start an instance of the virtual services that are associated with components in the **System Model** page, you can then navigate to the **Instances** page to either view the running instance or stop a running instance. You can also view the components that are associated with running instances of virtual services from the **Instances** page. See Viewing running instances of virtual services on page 966.
Introducing the Concurrent Virtual Services licenses	Now, you need a separate Concurrent Virtual Services license to run an instance of the virtual services on HCL OneTest™ Server. See DevOps Test Hub licensing information on page 183.
Viewing running instances associated with components	After you have associated virtual services with components in the System Model and started instances, you can view the instances that are associated with the components from the **Instances** page. You can also filter the view to display running instances by using components as a criteria in the filters.

Feature title	Description
Hiding inactive instances from the virtual service Instances page view	You can now hide the virtual service instances that are stopped or those that failed, from the display on the virtual service **Instances** page via a single button. See Viewing running instances of virtual services on page 966.
Stopping multiple running instances of virtual services	You can now stop multiple or all running instances of virtual services from the **Instances** page. See Stopping virtual services on page 982.
Navigating to the Progress page to view the state of test asset runs	You can now navigate to the **Progress** page from the **Execution** page to view the runs that were started, scheduled, or completed by clicking the **Show in the Progress page** icon that is enabled for the specific test asset. See Viewing the state of test assets on page 922.
Stopping or Canceling multiple test runs	You can now stop multiple or all running test assets from the **Progress** page. You can also cancel multiple or all scheduled runs of test assets from the Progress page. See Stopping test runs on page 932 or Canceling scheduled test runs on page 934.
Defining resource monitoring sources for a team space	As a team space owner, you can define resource monitoring sources in the team space so that they can be shared between multiple projects in the team space. During a performance schedule execution with resource monitoring labels, the sources defined in a team space are collected with the sources that are defined in a project whenever they match the provided labels. As a user of a team space, you can use the resource monitoring sources that are defined in the team space.
Agents listed for selection based on capabilities	When you install static agents and add the agents to your project on HCL OneTest™ Server, the agents based on the capabilities that match with the agent specified in the test assets are listed for an override. The agents with the best matching capabilities are listed on the top followed by other agents with mismatched capabilities.
Adding categories and properties to user-defined capabilities	The user-defined capabilities feature is enhanced. Now, you can group user-defined capabilities under categories that you create. You can also add name-value pairs as properties for the capabilities. See Working with the capabilities of a performance agent on page 548.

Feature title	Description
Establishing relationship among schemas based on referential integrity	Previously, you were able to only view the relationships among JDBC-supported schemas on the Schema Canvas page. You can now establish parent-child relationships among multiple non-JDBC schemas on the Schema Canvas page.
Generating DDL statements for multiple schemas	You can now generate the Data Definition Language (DDL) statements for single or multiple non-JDBC schemas. You can use the DDL statements to create a table in your database. Each job of generating DDL statements for the selected schemas generates a SQL file.
Generating the test data by using multiple schemas	Previously, you were able to generate the test data for only multiple JDBC-supported schemas. You can now generate the test data for multiple non-JDBC schemas by maintaining referential integrity. You can also modify the relationships among non-JDBC schemas on the **Schema Canvas** page. You can view the relationship among the selected schemas on the **Schema Canvas** page.

What's New in 10.1.3

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.1.3.

Feature title	Description
Introducing Team Spaces on HCL OneTest™ Server	HCL OneTest™ Server is now provisioned to display an initial Team Space, which is a logical partition of HCL OneTest™ Server. The operations that you managed at the server level can now be performed at the Team Space level. You can now configure licenses or an SMTP server, create projects, register the remote agents, dockers, or intercepts within a Team Space. See Team space overview on page 172.
System modeling	You can create a system model to view the logical representation of any application under test in a Team Space. The system modeling is a Tech Preview feature. You can configure a Team Space repository, which is different from the project repository, to save the system model. You can create components that represent different types of assets and associate the components by defining the relationship between the components. You can also create child components for components. You can use these concepts to build a system model and view the model. You can then publish the system model to the Team Space repository.

Feature title	Description
	You can also associate the virtual services in a project repository with components of the system model. From the associated component, you can view and run the virtual services. See System modeling on page 496.
Working with virtual services	You can now view the virtual service resources that are in your project repositories from the **Resources** page and then configure a run of the virtual service. Working with virtual services is a Tech Preview feature. After you start a virtual service, the running instances can be viewed from both the **Resources** page and **Instances** page. You can perform the following tasks on virtual services: • View information about the details, behavior, usage, routing rules, and diagnostic information of the virtual service instance. • Modify the behavior or the logging level of a running instance. See Management of virtualized services on page 936.
Stubs are excluded from Execution and Progress pages	Starting from 10.1.3, stubs are not displayed on the **Execution** page to start them nor are they displayed on the **Progress** page after they are started. Stubs must be started from the **Virtualization > Resources** page and stubs that are running can be viewed from the **Virtualization > Instances** page. See Management of virtualized services on page 936.
Running JUnit tests on HCL OneTest™ Server	You can now run JUnit tests that are within a Maven project from HCL OneTest™ Server. You must commit the Maven project as a `pom.xml` to a repository. After you add the repository to your project on HCL OneTest™ Server, you can configure a run of the JUnit tests from the **Execution** page. After the tests are run, you can view the detailed reports of the tests. See Configuring a run of a JUnit test on page 876.
Agents are selected automatically for an override	When you install static agents V10.1.2 and add the agents to your project on HCL OneTest™ Server, HCL OneTest™ Server determines and then displays the capabilities of the agents. Further, HCL OneTest™ Server compares the capabilities of the agents with the required capabilities specified for the agent in the test assets and automatically selects the most suitable agent for an override. You can either proceed with running the test on the agent

Feature title	Description
	selected or select an agent from the agents displayed that have the same capabilities. See Adding a performance agent to a project on page 546.
Working with capabilities that you define for static agents	When you install static agents V10.1.3 and register them with HCL OneTest™ Server, you can add, view, edit, or delete the capabilities of static agents. See Working with the capabilities of a performance agent on page 548.
Refreshing of the migrated project repositories	After you migrate a project from a previous version to a later version of HCL OneTest™ Server, the repositories that are configured for the project are refreshed only after the project owner logs in to HCL OneTest™ Server.
Generating the test data by using multiple schemas	Previously, when you used the JDBC connection to generate the test data by using HCL® OneTest™ Data, then you were able to generate the test data for only one schema. You can now generate the test data for multiple schemas simultaneously by maintaining referential integrity. Multiple schemas that you use to generate the test data now belong to a single JDBC connection. You can also view the relationship among the selected multiple schemas on the Schema integrity page.
Execution of properties based on priority	When you set multiple properties for any item type, then HCL® OneTest™ Data uses an order of priority while executing the properties.
Establishing a secure and trusted integration between applications and HCL® OneTest™ Data	You can now establish a secure and trusted integration between applications and HCL® OneTest™ Data by placing the SSL certificate in the trust store.

What's New in 10.1.2

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.1.2.

Feature title	Description
Documentation	You can now quickly add your comments to provide your feedback about the documentation for HCL OneTest™ Server by posting to a product forum. To add your comments as feedback, you must first register at HCL Support. Your login information along with your comments are governed by the privacy policy of HCL. By commenting, you accept the terms of use of HCL.

Feature title	Description
Installation	You can now install HCL OneTest™ Server on Ubuntu (using k3s) platform. See Installation of DevOps Test Hub on Ubuntu.
Enhanced license configuration settings	If you are an administrator, the **Configure license** option enables you to change the configuration values such as the URL and ID of the License Server. An administrator can also add, remove, or replace named users and enable or disable the automatic self-acquisition of the named user licenses.
Refreshing repositories	The repositories are refreshed automatically at regular intervals to ensure that each repository is in synchronization with the remote repository. You can also trigger the refresh activity manually for a repository to fetch the changes instantly. See Refreshing repositories manually on page 1231.
Creating webhooks	You can now create a webhook for a repository in Git server. For each push event in the Git server, the webhook notifies HCL OneTest™ Server to trigger the refresh activity to pull the latest changes to the server repository. See Creating webhooks on page 1234.
Running Postman test collections	You can now run tests that you create in Postman on HCL OneTest™ Server. You must export the collections and environments to files that you commit to a repository. You must add the repository to your project on HCL OneTest™ Server. Before you run the Postman tests, you must ensure to enable the Postman microservice. You can configure a run of the entire collection or any of its subfolders from the **Execution** page. While you configure the run, you can provide additional command options as you did in the Newman CLI. After the tests are run, you can view the detailed reports of the tests. See Configuring a run of a Postman test on page 886.
Running JMeter tests on HCL OneTest™ Server	You can now run tests the JMeter tests that you create in Apache JMeter on HCL OneTest™ Server. Before you run the JMeter tests, you must enable the JMeter microservice on HCL OneTest™ Server. You must package the JMeter tests along with other associated assets such as properties files, certificates, library jars, and dependencies as a project. The project must contain the project file (`jmeterRoot.jprj`). The tests and the other assets must be contained in subdirectories within the project directory. You must commit the JMeter project to a repository. After you add the repository to your project on HCL OneTest™ Server, you can configure a run of the JMeter tests from the **Execution** page.

Feature title	Description
	After the tests are run, you can view the detailed reports of the tests. See Configuring a run of a JMeter test on page 867.
Viewing capabilities of registered agents	Previously, you were able to view the details about the agents. You can now view and ascertain the system and application capabilities of all agents that are registered with HCL OneTest™ Server. With the information presented, you can make informed decisions about selecting an agent for running tests from HCL OneTest™ Server. See Adding a performance agent to a project on page 546.
Managing notifications	You can now choose whether to receive in-app, email, or both notifications for each notification channel. By default in-app notifications are enabled. The notifications are about tasks that need completion from users or for information purposes. See Managing email notifications on page 1244.
Configuring notifications	As an administrator, you can now set up an SMTP email server to send email notifications. The notifications are about tasks that need completion from the users or for information purposes. See Configuring an SMTP server to manage email notifications on page 1246.
Support for integration with Azure	You can now use the HCL OneTest™ Studio extension to establish a trusted and secure connection between Azure DevOps Server and HCL OneTest™ Server to run the HCL OneTest™ Server tests from the Azure DevOps Server. See Integration with Azure DevOps on page 1126.
Enhanced the integration with HCL® Launch	You can now use the HCL OneTest™ ServerLaunch plugin to establish a trusted and secure connection between the HCL® Launch server and HCL OneTest™ Server to run the HCL OneTest™ Server tests from the HCL® Launch server.
Support for integration with Jenkins	You can now use the HCL OneTest™ Server Jenkins plugin to establish a trusted and secure connection between the Jenkins server and HCL OneTest™ Server to run the HCL OneTest™ Server tests from the Jenkins server. See Integration with Jenkins on page 1173.
Support for integration with Jira	HCL OneTest™ Server is now integrated with Jira software, as a result, you can create a defect in Jira from the HCL OneTest™ Server UI. See Configuration of Atlassian Jira as a change management on page 1093.
Importing a CSV file with a separator character	You can now import a CSV file that has a separator character into the dataset and configure the required separator values in the CSV Editor.

Feature title	Description
	See Editing a dataset on page 433.
HCL® OneTest™ Data integration with HCL® Launch	You can now integrate HCL® OneTest™ Data with HCL® Launch by using the HCL® OneTest™ DataLaunch plugin to generate the test data on the HCL® Launch server.
Setting regular expressions as a basic property	Previously, the **Regular Expression** field was available only under **Advanced properties**. You can now also find this field as a basic property of an item type in the **Properties** dialog box.
Managing HCL® OneTest™ Data project	You can now archive or delete the project, if you no longer require the data design environment that you created in your HCL® OneTest™ Data project. You can also unarchive any of your archived projects.

What's New in 10.1.1

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.1.1.

Feature title	Description
Reviewing system requirements	You can now find system requirements information about hardware and software compatibility in this documentation. For more information, see System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94.
Upgrading the server software	You can now upgrade to the latest version of HCL OneTest™ Server on the Red Hat OpenShift platform by using the helm upgrade command from 10.1.0. See Upgrade of DevOps Test Hub on page 119.
Support for running Compound Tests that contain mobile tests	You can now configure a run for a Compound Test that contains mobile tests and then run the Compound Test on HCL OneTest™ Server. See Configuring a run of a Compound Test that contains mobile tests to run by using variables on page 678.
Overriding Resource Monitoring settings in a performance schedule	Now in the Execution dialog, the Resource Monitoring tab displays the settings related to the Resource Monitoring sources and labels as they were set in the performance schedule. This information is helpful to decide whether you want to override them with the Resource Monitoring sources that are available in your Server project, or not. See Controlling resource monitoring sources in a schedule on page 1079.

Feature title	Description
HCL OneTest™ Server integration with HCL® Launch	You can now integrate HCL OneTest™ Server with HCL® Launch by using the HCL OneTest™ ServerLaunch plugin to run tests on the HCL® Launch server.
Service virtualization: Tech Preview **Virtualizing a service in a namespace of HCL OneTest™ Server**	Previously, you specified the short name of a service that you wanted to virtualize as the host for the HTTP transport and specified the namespace in the recording settings for an Istio proxy. For example, to record the reviews service that had an FQDN of *reviews.default.svc.cluster.local* the host for the HTTP transport to be specified was *reviews*, and the namespace to be specified for the Istio Proxy was *default*. However, Istio routing in HCL OneTest™ Server only uses the host name and port to generate the routing rule and for stubbing. Therefore, the host name must contain the namespace so that HCL OneTest™ Server can correctly identify the service that is being stubbed. Now, you can enable HCL OneTest™ Server to use the namespaces that you want by providing the namespaces at the time of the installation of the server software.
Managing test data generated jobs and API history	HCL® OneTest™ Data now periodically deletes all the generated test data and the jobs of test data generation with API history to clear the memory. After changing the default configuration settings, you can retain the test data generated jobs and the API history of the jobs for a specific period.
Generating test data for MongoDB	You can now generate the test data for the MongoDB by using HCL® OneTest™ Data. HCL® OneTest™ Data generates the test data, and then directly writes the generated test data in MongoDB.
Generating test data for Excel	You can now generate the test data for an Excel file by using HCL® OneTest™ Data. You can then download the test data on your computer.
Generating test data by using the pairwise data generation method	You can now generate an optimum test data that covers all possible combinations of the values of the item types for each pair of item types in a schema by using the pairwise data generation method.
Copying artifacts of a schema to another schema	You can now copy artifacts of a schema to a new or an existing schema.

Feature title	Description
Performing operations by using REST APIs in HCL® OneTest™ Data	You can now perform operations on the schemas in HCL® OneTest™ Data by using a set of REST APIs from any external application.
Saving a copied dataset	You can now save a copy of a dataset in a different folder or make a copy with a different name, by using the **Save As** option. See Management of datasets on page 424.
Displaying icons to view who is editing datasets	In HCL OneTest™ Server, you can now see all members who are editing a particular dataset. See Editing a dataset on page 433.

What's New in 10.1.0

The following section lists the features, enhancements, or other changes made in HCL OneTest™ Server 10.1.0.

Feature title	Description
Installation	HCL OneTest™ Server can only be installed on the following platforms: • Red Hat OpenShift 4.2+ • Ubuntu (using MicroK8s) See Installation of DevOps Test Hub on page 115.
Server Licensing	You can purchase named user licenses for HCL OneTest™ Server and install the licenses on the FlexNet Operations server. For more information, see HCLSoftware license portal on page 182.
Backup and restore of user data	You can back up and restore the user data in HCL OneTest™ Server V10.1.0. You can also migrate the backed-up user data from the previous versions of HCL OneTest™ Server to HCL OneTest™ Server V10.1.0. For more information, see Backup and restoration of the DevOps Test Hub data on page 127.

Feature title	Description
Running AFT Suites on remote agents	You can now select an agent added to your project on HCL OneTest™ Server to override the agent configured in the AFT XML file in the test asset.
Selecting test resources	You can now select either the current or previous versions of the test resources that you want to run while you are configuring the test run.
Resetting test run configurations	You can now reset the settings you configured for a test run when you are either initiating or repeating the test run. For more information, see Resetting the configuration settings for a test run on page 931.
Support for running an HTML test in a Compound Test on HCL OneTest™ Server	You can now, run an HTML test as a compound test on HCL OneTest™ Server and choose to override the configured browser while configuring the test. For more information, see Configuring a run of a Compound Test that contains HTML tests on page 665.
Navigating from a result to the associated Jaeger trace	Now, test logs are delivered in a Jaeger format by default for executed compound tests and schedules. You can still set a Program Argument to re-enable the traditional test log format when you run the tests. For more information, see Test results and reports on page 985.
Relaunching an execution from the Results view	From the Results view, you can re-execute a test with the same parameters as the ones that you previously set for the previous test run. You can also re-execute multiple tests at a time with the same parameters as the initial ones. For more information, see Re-executing tests from results on page 998.
Controlling Resource Monitoring sources	Now you can use label matching to control what Resource Monitoring sources are collected during the execution of a Performance schedule. For more information, see Controlling resource monitoring sources in a schedule on page 1079.
Functional Test reports	A Unified Report is generated as a result of a functional test execution. For more information, see Test results and reports on page 985.

Feature title	Description
Querying metrics	Now the Resource Monitoring service can query metrics from a target monitored by a running Prometheus server with default or customized queries. For more information, see Monitoring metrics collected by a Prometheus server on page 1054.
Scrapping metrics	The Resource Monitoring service can scrap metrics exposed by an OpenMetrics exporter through metric counters. For more information, see Monitoring metrics exposed by an OpenMetrics exporter on page 1058.
Selecting branches	In earlier versions, branch selection was configured on the Repositories page and applied globally to all users. Now, each user is presented with a Branch field that lists all the branches of the repositories added to your project. The list is available in the Execution, Datasets and Overview pages HCL OneTest™ Server. For more information, see Selecting the global branch in a project on page 1233.
Creating secret collections	Members with the Tester role assigned in a project, can now create secrets collections. For more information, see Protecting test assets by using secrets on page 1235.
Project repository	You can now add a repository with multiple folders to a single project. With this improvement, each project is now restricted to a single copy of a given repository.
HCL OneTest™ Server integration with Azure DevOps integration with Azure DevOps	You can now integrate HCL OneTest™ Server with Azure DevOps by using the HCL OneTest™ Studio extension to run tests as tasks in Azure DevOps pipelines. For more information, see Integration with Azure DevOps on page 1126.
Running stubs	You can now run stubs that you create in HCL OneTest™ API, from the **Execution** page on HCL OneTest™ Server. You can run IBM® WebSphere® MQ stubs and HTTP stubs on HCL OneTest™ Server. For more information, see Configuring a run of a virtual service on page 946.
Running stubs for Istio services	After you create stubs for Istio in HCL OneTest™ API that virtualize services in a Kubernetes cluster, you can run those stubs on HCL OneTest™ Server. The stubs can virtualize services that use the Istio service mesh for the following types of requests:

Feature title	Description
	• Requests received by services in the Istio service mesh. • Requests sent from namespaces in the Istio service mesh to external services that are not in the Istio service mesh.
Stub usage statistics	After you run stubs on HCL OneTest™ Server, you can view the stub usage statistics from the **Stub Usage** page. For more information, see Viewing usage statistics of virtual services on page 980.
Routing rules	You can view the details of the intercepts, proxies, or stubs with routing rules defined that are connected to HCL OneTest™ Server, from the **Routing Rules** page. For more information, see Viewing routing rules of the virtual services on page 978.
Using Excel Data Sources	Excel Data Sources defined in HCL OneTest™ API projects can now be used as Datasets in your HCL OneTest™ Server projects. Note that Excel Data Sources that define a specific worksheet to use are ignored.
HCL® OneTest™ Data	HCL® OneTest™ Data is a test data generation tool. You can use HCL® OneTest™ Data to generate the sample test data to perform tests. This tool is now available as a component of HCL OneTest™ Server.
Authoring datasets	Now, you can create, edit, or delete a dataset in HCL OneTest™ Server. You can also publish the created dataset to Git repository so that other members of the project can use your dataset in their test asset run. For more information, see Management of datasets on page 424.

Deprecations and removals

You can find the features or capabilities announced as deprecated and the features or capabilities that are removed in HCL DevOps Test Hub (Test Hub). DevOps Test Hub is the new name of HCL OneTest™ Server from 11.0.0 onwards.

When features or capabilities are announced as deprecated, they continue to be supported although such features or capabilities are no longer enhanced and might be removed in a subsequent release. If you use a deprecated feature, you must migrate to the alternate feature, if applicable, before the deprecated feature is removed.

The removed features will not be available in the product version in which they are removed.

Note: You can contact the HCL Client Support team if you have any concerns.

You can find information about the following deprecations or removals:

- Product features or capabilities on page 51
- Dependent software on page 52

Product features or capabilities

You can find information about the features or capabilities of Test Hub that are announced as deprecated in an earlier version, and the version in which they are removed.

Note: The features removed in this version are listed first, followed by the features deprecated. The features removed or deprecated in earlier versions are listed later in the table.

Feature or capability	Feature deprecated in version	Alternate feature, if applicable	Feature removed in version
Support to generate Web UI tests by using design screens of the application with Test Genie, which was a tech preview feature.		None	11.0.7
Scanning of the application code by using HCL® AppScan CodeSweep	11.0.6	You can use the HCL® AppScan CodeSweep plugin from the Visual Studio Marketplace.	12.0.0
Running the tests on remote docker hosts	11.0.3	You can run the tests on the remote kubernetes cluster. See Management of remote Kubernetes clusters on page 572.	11.0.7
The test authoring capability of HCL OneTest™ Server that utilizes a containerized version of the HCL OneTest™ Studio desktop authoring products to author tests and virtual services.	10.5.0	None	11.0.0
Support to run Istio based virtual services	10.5.0		11.0.0
Use of Jaeger to display test logs.	10.2.3	The capability provided by Jaeger to provide the test log traces is replaced with the use of the built-in results service to provide test logs.	11.0.0

Feature or capability	Feature deprecated in version	Alternate feature, if applicable	Feature removed in version
Support of Prometheus for recording usage data of virtual services		The capability provided by Prometheus is replaced with the use of the built-in logging capability for virtual services.	10.5.3

Dependent software

You can find information about the dependent software that were announced as deprecated in an earlier version and the version in which they were removed. Dependent software can include the operating systems, license servers, applications, or software required for using Test Hub.

Note: The features removed in this version are listed first, followed by the features deprecated. The features removed or deprecated in earlier versions are listed later in the table.

Dependent software	Deprecated in version	Alternate OS or software, if applicable	Support removed in version
HCL Local License Server 2.0 (also known as 2021.05)	Not applicable	HCL Local License Server 3.0 (also known as 2022.12)	10.5.3
HCL Local License Server 3.0 (also known as 2022.12)	Not applicable	HCL Local License Server 2024.04	11.0.2

Important Advisory

Microsoft Azure will retire its Basic SKU IP addresses in the Virtual Network IP Services soon. So, you must upgrade the Basic SKU IP addresses to Standard SKU IP addresses. For more information, refer to Upgrading a basic public IP address to Standard SKU - Guidance.

Defect fixes

You can find the defects that are fixed in this version of HCL DevOps Test Hub (Test Hub). You can also find the defects that were fixed in the earlier versions of Test Hub. DevOps Test Hub is the new name of HCL OneTest™ Server from 11.0.0 onwards.

You can find the following information:

- Defects fixed in DevOps Test Hub 2026.09 (12.0.0) on page 53
- Defects fixed in earlier versions on page 53

Defects fixed in DevOps Test Hub 2026.09 (12.0.0)

There are no announcements on defect fixes in Test Hub in this release.

Defects fixed in earlier versions

You can find information about the defects fixed in each of the following versions:

Defects fixed in 11.0.9

The defects fixed in Test Hub 11.0.9 are as follows:

ID	Description
THUB00011294	Previously, if a project was deleted while a virtual service in that project was still running, the virtual service might continue running. This problem is fixed.

Defects fixed in 11.0.8

There are no announcements on defect fixes in Test Hub in this release.

Defects fixed in 11.0.7

The defects fixed in Test Hub 11.0.7 are as follows:

ID	Description
TP-100149	Previously, when you tried to create a data asset in Test Hub on the OpenShift platform, the asset creation might have failed if the configured repository size exceeded 2 GB with folders that contained thousands of assets. This problem is fixed.
TP-101491	Previously, when running a suite in parallel on the server with multiple scripts, tests that iterated the same dataset did not behave correctly because all tests shared a single dataset cursor. Only the first test iterated the data, while subsequent tests did not iterate any data. This problem is fixed. Shared datasets are now used only for variable assignment in the configuration block and in **Assign** steps.

Defects fixed in 11.0.6.1

The defects fixed in Test Hub 2025.10 (11.0.6.1) are as follows:

ID	Description
TP-100421	Previously, the webhooks that were in the **Enabled** state in Test Hub repeatedly sent notifications every 30 seconds for the selected events. This problem is fixed.
TP-100380	Previously, when you selected the Safari browser on a Mac system for recording a Web UI test, occasionally the recording failed. This failure occurred when the application selected for recording had a configured URL. This problem is fixed.

Defects fixed in 11.0.6

There are no announcements on defect fixes in Test Hub in this release.

Defects fixed in 11.0.5

The defects fixed in Test Hub 11.0.5 are as follows:

ID	Description
TP-97070	Previously, the projects listed under the **Projects** menu did not display full project names, and therefore, it was difficult to distinguish between projects with similar names. This problem is fixed. You can now view full project names by hovering over them, making it easier for you to identify and select the correct project.
TP-96290	Previously, when you deleted test results in Test Hub, the related test log attachments remained in the PostgreSQL database. As a result, the `pg_largeobject` table continued to consume a large amount of disk space, even after deletion. This problem is fixed. Now, after the test results are deleted, the associated test logs are removed from the database, helping to free up storage.

Defects fixed in 11.0.4

The defects fixed in Test Hub 11.0.4 are as follows:

ID	Description
TP-94914	Previously, when multiple test runs were stopped or completed simultaneously, concurrent API requests accessed the result APIs, which resulted in frequent restart of the result pods. This problem is fixed.

Defects fixed in 11.0.3

The defects fixed in Test Hub 11.0.3 are as follows:

ID	Description
TP-93601	Previously, when you used Test Hub on IBM Red Hat OpenShift, you might have got an error message, "Failed to create data definition", during the creation of a data definition and dataset. This problem is fixed.

Defects fixed in 11.0.2

The defects fixed in Test Hub 11.0.2 are as follows:

ID	Description
TP-92219	Previously, when you ran a JMeter test and if this test modified the default truststore of the JVM, the test completed successfully, but did not show any relevant log details. This problem is fixed. You can now view the relevant log details when you run a JMeter test.
TP-92142	Previously, as a project owner, you might have experienced issues when you tried to approve access requests for your project. This problem is fixed.
TP-92873	Previously, when you ran an API automation test, the offline token may have expired in the middle of the test run. This problem is fixed.

Defects fixed in 11.0.1

The defects fixed in Test Hub 11.0.1 are as follows:

ID	Description
TP-91596	Previously, when you ran a Web UI test with a custom code to use JDBC libraries, Test Hub displayed java.lang.UnsupportedClassVersionError. This problem is fixed.

Defects fixed in 11.0.0

The defects fixed in Test Hub 11.0.0 are as follows:

ID	Description
TP-89626	Previously, the terminated Execution pods incorrectly displayed the status as an error when the API test or API Suite returned the result as *Fail*. This problem is fixed.
TP-88262	Previously, when you ran Web UI tests that had a large number of test steps, the unified report failed to be exported as a PDF. This problem is solved.
TP-89553	Previously, the unified report was displayed incorrectly when you ran a Compound Test that contained mobile tests, which ran on either the devices, emulators, or simulators that were connected to an agent computer or the devices that were connected to a mobile device cloud. This problem is fixed.

Defects fixed in 10.5.4

The defects fixed in HCL OneTest™ Server 10.5.4 are as follows:

ID	Description
TP-89167	Previously, when you exported a statistic report for a test result, the downloaded report displayed the Agent Health report as the default report. This problem is fixed. The downloaded report now displays the UI Statistical report as the default report.
TP-86087	Previously, when you started a large number of test assets simultaneously, you were required to determine the capacity of the system, size of projects, and adjust the rate accordingly. This problem is fixed.

Defects fixed in 10.5.3

The defects fixed in HCL OneTest™ Server 10.5.3 are as follows:

ID	Description
TP-88119	Previously, if you set the Helm option **gateway.allowOrigin** during installation of HCL OneTest™ Server, it was mandatory for you to include the server URL in the string. This problem is fixed. Now, you need to only include strings for additional servers and you do not have to include the server URL string in the **gateway.allowOrigin** option.
TP-88111	Previously, the default configuration in Keycloak incorrectly assigned all users of HCL OneTest™ Server the privileges of a server administrator. This problem is fixed.
TP-88079	Previously, when members of a team space entered the URL of a page in a project, they were incorrectly provided the access to view the page even though they were not a member of that project. This problem is fixed.
TP-86646	Previously, when filtering the virtual service instances that are listed on the Instances page by activity, HCL OneTest™ Server incorrectly filtered by overall activity rather than recent activity. This problem is fixed.
TP-86105	Previously, the project overview page did not display the statistics of the test results that were published to HCL OneTest™ Server from any of the desktop products. This problem is fixed. Now, if a project has no configured repositories, the project overview page displays the statistics of the test results published to HCL OneTest™ Server from any of the desktop products.

Defects fixed in 10.5.2

The defects fixed in HCL OneTest™ Server 10.5.2 are as follows:

ID	Description
TP-86049	Previously, when you entered non-alphanumeric characters in search boxes on any page on HCL OneTest™ Server, the search resulted in errors.

ID	Description
	This problem is fixed.

Defects fixed in 10.5.1

The defects fixed in HCL OneTest™ Server 10.5.1 are as follows:

ID	
TP-84738	When you associated Suites and tests with components in the system model in HCL OneTest™ Server 10.5.0, the associated components were not displayed under the **Components** column on the **Execution** and **Results** pages. This problem is fixed.
TP-84688	Previously, when you deleted multiple associations to test resources that were in different projects, HCL OneTest™ Server failed to delete all the associations in the system model. This problem is fixed.
TP-84675	Previously, when you added a static agent to multiple projects in HCL OneTest™ Server, the agent was not added to all the projects and failed after a few projects were added. This problem is fixed.
TP-84492	Previously in Keycloak, when using the LDAP user provided certificates were not imported into the Keycloak truststore that resulted in errors when authenticating. This problem is fixed.

Defects fixed in 10.5.0

The defects fixed in HCL OneTest™ Server 10.5.0 are as follows:

ID	Description
TP-84022	Previously, when you ran a Web UI test with an associated dataset, the test failed to read all the rows and stopped after completing a few iterations. This problem is fixed.

ID	Description
TP-83686	A **Sign-Up** link is no longer offered on the login page by default to improve security posture. It can be enable during install or afterward from Keycloak by enabling **Login > User Registration**.
TP-82448	Previously, when you ran a performance test that contained integration tests on HCL OneTest™ Server, the hierarchy of test iteration in the test log was incorrect. Therefore, the integration test steps in the test log were displayed under the first activity that was captured for the performance test. This problem is fixed.
TP-76020	Previously in Keycloak, when using the LDAP user provided certificates were not imported into the Keycloak truststore that resulted in errors when authenticating. This problem is fixed.

Defects fixed in 10.2.3

The defects fixed in HCL OneTest™ Server 10.2.3 are as follows:

ID	Description
TP-81758	Previously, if you specified a secrets collection while configuring a run of a virtual service, the specified collection was ignored and an attempt was made to use a secrets collection with the same name as the environment. This problem is fixed.
TP-81035	Previously, the default configuration of HCL OneTest™ Server had access to requests from any domain. Therefore, HCL OneTest™ Server was exposed to cross-origin resource sharing (CORS) attacks. This problem is fixed.
TP-81017	Previously, if you attempted to edit an API Suite that was in the git branch of your server project, the containerized API editing session was not launched because of non-omitted spaces in the path of the asset. This problem is fixed.

ID	Description
TP-80637	Previously, while you installed HCL OneTest™ Server on Azure Kubernetes Service (AKS) if the postgresql-init job did not complete early, it might have remained in a Pending state due to insufficient CPU resources. This problem is fixed.
TP-80565	Previously, when you deployed HCL OneTest™ Server on Azure Kubernetes Service (AKS), and if you did not set up the Domain Name Server (DNS) resolution of the client, you were unable to open a new editor session. This problem is fixed.
TP-80321	Previously, when you ran a performance or functional integration test on HCL OneTest™ Server that had the API test type as a RAML definition, the test failed to run. This problem is fixed.

Defects fixed in 10.2.2

The defects fixed in HCL OneTest™ Server 10.2.2 are as follows:

ID	Description
TP-79765	Previously, if two instances of a virtual service resource were run simultaneously with different environments and later stopped, the virtual services displayed their status as *STOPPED* but the virtual services might be left running in the container. This problem is fixed.
TP-79322	Previously, when a virtual service on the server was running and the server execution service restarted, it resulted in the loss of the routing rules for the virtual service. This problem is fixed.
TP-79272	Previously, when you started an HTTP virtual service by using the REST API and supplied an environment name where the name differed from the desktop project value in its letter-casing, for example, *env* compared to *Env*, the virtual service might have started but the routing rules were not created. This problem is fixed.

ID	Description
TP-79218	Previously, if you attempted to use the dataset override when you started an instance of a virtual services resource, you might have noticed that the dataset override did not take place. This problem is fixed.
TP-79185	When you run two virtual services concurrently from HCL OneTest™ Server, where the virtual services come from the same desktop project but are run with different environments, traffic might fail to be routed to one of the virtual service and HTTP 502 errors might be encountered. This problem is fixed.
TP-78842	Previously, when you ran a test on a remote Kubernetes cluster from a HCL OneTest™ Server installation that was configured with a non-default ingress secret name, the remote execution pod failed to start. This problem is fixed.
TP-78841	Previously, when you scheduled a test to run on a remote agent associated with your project in your team space, the test run failed to run and an error message was displayed. This problem is fixed.
TP-78219	Previously, when you updated the name of the project in Jira, and if you have used that project name on HCL OneTest™ Server, you were unable to configure the updated project name in the **CHANGE MANAGEMENT SYSTEM** tab of HCL OneTest™ Server. This problem is fixed.
TP-78109	Previously, when you stopped a large number of virtual service instances, you might have seen a 504 error although the virtual service instances were stopped successfully. This problem is fixed.

Defects fixed in 10.2.1

The defects fixed in HCL OneTest™ Server 10.2.1 are as follows:

ID	Description
TP-77049	Previously, when you ran the `reconcile-secrets.sh` script to update the password seed, the script displayed an error message.

ID	Description
	This problem is fixed.
TP-76098	Previously, during the deployment of HCL OneTest™ Server V10.2.0 on Azure Kubernetes Service (AKS) V1.20 or later, the deployment was failing with an error message. This problem is fixed.
TP-74534	Previously, when you tried to publish a created dataset after the migration from an earlier version to 10.1.3 or 10.2.0, you were unable to publish the dataset to a Git repository. This problem is fixed.
TP-73109	Previously the command-line interface was displaying warnings about Ingress deprecation during the installation of HCL OneTest™ Server 10.1.3 or 10.2.0 on Lightweight Kubernetes (k3s). This problem is fixed.

Defects fixed in 10.2.0

The defects fixed in HCL OneTest™ Server 10.2.0 are as follows:

ID	Description
TP-71950	Previously, when the server was under high load, in some cases, the results microservice restarted indefinitely. This issue occurred when the Kubernetes supervisor stopped the results microservice during a database locking process. This problem is fixed.
TP-71416	Previously, when you attempted to run an API Suite on HCL OneTest™ Server that is installed on OpenShift, and when the API project contains a vast set of resources, the **Execute** button in the **Execute test asset** dialog box was enabled after a delay or at times not enabled at all. This problem is fixed.
TP-70940	Previously, during the installation of HCL OneTest™ Server, checks were not made to verify that the supplied ingress domain was valid. This problem is fixed. Now, when you provide the details for the ingress domain, checks are performed for the following constraints:

ID	Description
	• The domain name must not exceed 253 characters in length. • Each element of the domain must not exceed 63 characters. An additional check is made to ensure that the pattern, derived from the ingress domain, used to create hostnames dynamically, generates hostnames that conform to the same constraints.You can disable the additional check by specifying **--set preflight.validateHostPattern=false**. If the ingress domain name does conform to the constraints, some of the service virtualization features might not work correctly.

Defects fixed in 10.1.3

The defects fixed in HCL OneTest™ Server 10.1.3 are as follows:

ID	Description
TP-72278	Previously, when you attempted to run an API Suite on HCL OneTest™ Server that is installed on OpenShift, and when the API project contained a vast set of resources, the **Execute** button in the **Execute test asset** dialog box was enabled after a delay. This problem is fixed.
TP-58061	Previously, when you shut down HCL OneTest™ Server, the Suites, Compound Tests, or Schedules that were running did not stop. This problem is fixed.

Defects fixed in 10.1.2

The defects fixed in HCL OneTest™ Server 10.1.2 are as follows:

ID	Description
TP-70414	Previously, when you installed HCL OneTest™ Server V10.1.1 on an OpenShift platform where OpenShift Container Storage (OCS) was not available (For example, the IBM Cloud) then test asset runs failed. This problem is fixed.
TP-69452	Previously, when you started stubs on HCL OneTest™ Server, occasionally the stubs remained in transition and did not run. This problem is fixed.

ID	Description
TP-67330	In HCL OneTest™ Server V10.1.2, Kafka is replaced with RabbitMQ. Now you do not see a restart of the Kafka pod and some of the other microservices when the server is in an idle state.
TP-67306	Previously, from the **Execution** tab, you could not use the keyboard to search and select a branch. This problem is fixed.
TP-66731	Previously, you might have experienced errors when you published large reports, either from desktop products or when you executed a test on the server. This problem is fixed. You no longer have to set the options recommended as a workaround to publish large reports.
TP-62379	If you (as a user) requested to join an orphaned project, you saw a 404 not found error. You now see the following message: `You can't request to join a project that has no owners.`
TP-56453	After about 20 minutes of an inactive browser session for HCL OneTest™ Server, if you attempted to use the existing browser session, you had to authenticate again by providing your username and password. This problem is fixed.

Defects fixed in 10.1.1

The defects fixed in HCL OneTest™ Server 10.1.1 are as follows:

ID	Description
TP-68740	This version addresses a problem with datapool encrypted values being logged as clear text to Jaeger log.
TP-67208	Previously, when you used the **Undo** or **Redo** options for a change in a dataset by using the CSV editor, the **Save** or **Discard** options were grayed out, thus you could not save or discard the changes. This problem is fixed.

ID	Description
TP-67032	Previously, before you upgraded to a new version, you had to cancel any scheduled execution. This problem is fixed.
TP-60737	Previously, GIT repositories that contained projects at the root level and were not under a subdirectory were not supported and did not work with HCL OneTest™ Server. This problem is fixed.

Defects fixed in 10.1.0

The defects fixed in HCL OneTest™ Server 10.1.0 are as follows:

ID	Description
TP-66237	Previously, HCL OneTest™ Server projects were sorted by date. Now, server projects are sorted alphabetically.
TP-63775	Previously, in HCL OneTest™ Server, it was not clear that classifications could only be made for encrypted datasets. This problem is fixed.
TP-63156	Previously, Docker images that were created with no tag were assigned the *latest* tag. As the docker ps command did not show the *latest* tag anymore since the Docker 19.03.05 release, the Docker images were not detected and were not correctly monitored. This problem is fixed.
TP-63149	Previously, intermediate *<none>*:*<none>* Docker images were represented as is and could not be selected individually in Resource Monitoring counters. This problem is fixed.
TP-62643	If you backed up a project with both API suites and stubs on HCL OneTest™ Server 10.0.0, when you restored that backup on HCL OneTest™ Server 10.0.2, the stubs were missing. This problem is fixed.

ID	Description
TP-62571	If you wanted to add multiple repositories to a project in HCL OneTest™ Server you could not add another repository until the last repository completed its synchronization. This problem is fixed.
TP-62417	Previously, a user who was no longer a member of a project might be displayed in the **Secrets** tab. If a user was removed from a project and then added back to that project, the user would have the same access to secrets as they had before being removed. This problem is fixed.
TP-62338	Previously, an IBM WebSphere MQ stub that was configured to use more than one transport did not appear on the **Execution** page of the HCL OneTest™ Server. This problem is fixed.
TP-61897	Previously, if you restored a backup, it failed due to existing execution containers. This problem is fixed.
TP-61786	Previously, when you entered a URL for a test result directly into a browser or you clicked a link to a test result in the Jenkins console output from the Jenkins HCL OneTest™ Server plugin, and you were not logged in to HCL OneTest™ Server, you saw the following message: `Request cannot be processed: Unauthorized` page. This problem is fixed.
TP-61179	Previously, executing a test on a non-English system failed. This problem is fixed.
TP-61153	Previously, the project search field was case-sensitive. For example, if your project was named *MYPROJECT* and you searched for *myproject*, HCL OneTest™ Server did not show any results. This problem is fixed.
TP-58125	Previously, HCL OneTest™ Server did not release system resources after they had exited. This problem is fixed.

ID	Description
TP-57543	Previously, if the server hostname was changed, it was necessary to run `./prepare.sh` and log in to the Keycloak Admin Console to change the **testserver** client URLs. Otherwise the message `Invalid parameter: redirect_uri` was displayed on the **Login** page. This problem is fixed.

Known issues

You can find the known issues that are identified, if any, in this version of HCL DevOps Test Hub (Test Hub). You can also find the known issues identified in the earlier versions of Test Hub. DevOps Test Hub is the new name of HCL OneTest™ Server from 11.0.0 onwards.

You can find the following information:

- Known issues in DevOps Test Hub 2026.09 (12.0.0) on page 67
- Known issues from earlier versions on page 68

Known issues in DevOps Test Hub 2026.09 (12.0.0)

The known issues identified in Test Hub 2026.09 (12.0.0) are as follows:

ID	Description	Identified in version	Applicable in and until version
THUB00011915	If you configured any device from a device cloud as the **Override** option through an environment, the test fails to run on the device cloud when you run a test using that environment. To work around this problem, you must initiate the test run from the **Execution** page, and then select the device profile from the **Browser** drop-down in the **ENVIRONMENT** tab of the **Execute test asset** dialog.	12.0.0	
THUB00012741	When you run a web service-based VU schedules concurrently in Test Hub, the test run might remain in progress and appear to be hung. The CPU usage of the execution process might also be unusually high. The same test might run successfully multiple times, but intermittently hang. To work around this problem, you must manually stop the hung execution on the **Progress** page, and then rerun the test.	12.0.0	

Known issues from earlier versions

The known issues identified in the earlier versions of Test Hub that are still applicable are as follows:

ID	Description	Identified in version	Applicable in and until version
THUB00009285	When applications use PrimeReact select components, the recorder might capture the parent element instead of the actual select option element. As a result, the playback might fail when interacting with the recorded step. To work around this problem, you must update the recorded step to target the actual child element instead of the parent element.	11.0.8	
TP-101658	Only the user who initiated the test run can create work items in Azure or DevOps Plan, and issues in Jira from the **Results** page.	11.0.7	
TP-100421	The webhooks configured in Test Hub might repeatedly send notifications every 30 seconds for the selected events. To work around this problem, you must disable the webhook by selecting the **Disabled** option in the **Webhooks** tab.	11.0.6	11.0.6
TP-100149	When you try to create a data asset in Test Hub on the OpenShift platform, the asset creation might fail if the configured repository size exceeds 2 GB with folders containing thousands of assets.	11.0.6	11.0.6
TP-97948	When you run a test suite or a test with the Run step that contains tests from the other Git repository, the test run fails. To work around this problem, you must add only the assets from the same Git repository to a test or suite.	11.0.5	
TP-92483	When you attempt to verify the agent during a test recording, the "Get an app to open this 'devops-test' link" pop-up window is displayed. This problem occurs when you installed the DevOps Test runtime agent at least once, and you selected the "Always allow https://localhost:9000 to open links of this type in the associated app" checkbox before you uninstalled the agent. To work around this problem, you must clear the browser cookies and verify the agent installation again.		

ID	Description	Identified in version	Applicable in and until version
TP-93914	When multiple test runs stop or finish simultaneously, numerous requests access the result APIs concurrently. This simultaneous access can cause the results pods to restart frequently, which overloads the system and leads to instability, data inconsistencies, and service interruptions. To work around this problem, minimize the number of tests that might finish simultaneously by halting them in groups of five or fewer. This approach helps you to prevent overloading the result APIs with concurrent requests. If the issue persists, contact your admin to reconfigure the server by disabling the Analyze Results feature to reduce the system load. **Note:** To disable the Analyze Results feature, include the following Helm option in your current Helm upgrade command: `--set-string results.extraEnv.RESULTS_ANALYZER_ENABLED="false"`	11.0.3	11.0.3
TP-92219	The test log for a JMeter test might be unavailable if the default trust store of the JVM is altered by the test. To work around this problem, you must perform the following steps when you run each JMeter test: 1. Select the JMeter test, and then click **Execute**. The **Execute test asset dialog** is displayed. 2. Click **Advanced**. 3. Enter `CISTERNA_CONFIG_APPEND` in the first column, and `&trustStorePath=/opt/java/openjdk/lib/security/cacerts` in the second column in the **Additional Configuration Parameters** field. 4. Click **Add**.	11.0.1	11.0.1

ID	Description	Identified in version	Applicable in and until version
	5. Click **Execute**. A message is displayed that confirms the initiation of the test execution.		
TP-90652	You cannot run a Windows test on agent computers that are selected as the location to run the test when you execute the test in Test Hub 11.0.0.	11.0.0	
TP-89553	The unified report is displayed incorrectly when you run a Compound Test that contains mobile tests to run on either the devices, emulators, or simulators that are connected to an agent computer or the devices that are connected to a mobile device cloud. To enable the unified report to display correctly, you must provide the program argument **-history testlog** in the **Advanced** configuration when you configure a run of such Compound Tests.	10.5.4	10.5.4
TP-75715	HCL OneTest™ Server does not support the sorting of the test results on the **Results** page based on the person who started the test run when you click the **Started by** column header.	10.5.4	
TP-88328	The data generation does not generate test data when JavasSript rules are used to create item types. Also, the data generation feature does not validate the Regular expressions used to define item values in a schema.	10.5.3	
TP-88027	You cannot perform a direct upgrade of HCL OneTest™ Server 10.5.3 from versions 10.5.1 or earlier.	10.5.2	10.5.3
TP-88111	The default configuration in Keycloak incorrectly assigns all users of HCL OneTest™ Server the privileges of a server administrator.	10.5.0	10.5.2

ID	Description	Identified in version	Applicable in and until version
TP-88079	Members of a team space are able to view the details of a project when they enter the URL of a page in a project in spite of not being a member of that project.	10.5.2	10.5.2
TP-86722	HCL OneTest™ Server incorrectly creates duplicate records of the test runs in the integrated Xray-Jira project.	10.5.2	10.5.2
TP-86633	You cannot perform a direct upgrade of HCL OneTest™ Server 10.5.2 from versions 10.2.2 or earlier.	10.5.2	10.5.2
TP-86087	When programmatically starting a large number of test assets at once, users should be mindful of the capacity of the system, size of projects and adjust the rate accordingly.	10.5.2	10.5.3
TP-85868	Virtualization usage graphs at the team space or project level in HCL OneTest™ Server 10.5.2 are not displayed for virtual services that ran on an earlier version of the server.	10.5.2	10.5.2
TP-81529	When you run and complete a test by using the HCL OneTest™ Server UCD plugin, the UCD log displays the state of the test with a lag in real time. For example, the state is displayed as "In transition" when the test is in the "Running" state.	10.5.2	
TP-61446	When you unregister a remote docker agent, running and schedule tests are not canceled. To workaround this problem, stop any running tests and cancel any scheduled future executions for that agent before unregistering the remote docker agent.	10.5.2	
TP-85354	When you perform bulk update of test assets by using the test authoring feature on HCL OneTest™ Server, you encounter timeout errors when you attempt to commit or publish the changed assets.	10.5.1	

ID	Description	Identified in version	Applicable in and until version
	To work around this problem, you can commit or publish the assets in smaller batches.		
TP-78142	You cannot migrate the data about the IBM Engineering Test Management that is configured as a quality management solution on HCL OneTest™ Server 10.5.0 to HCL OneTest™ Server 10.5.1.	10.2.1	10.5.0
TP-65557	When you created projects in HCL OneTest™ Server 10.5.0 or earlier by using LDAP/AD, but did not import the LDAP/AD users into the migrated projects in 10.5.1, and deleted the users in the project, the project cannot be opened or assigned to other users.	10.1.0	
TP-84473	When you configure IBM® Engineering Test Management and Atlassian Jira on HCL OneTest™ Server 10.2.3, the information about Engineering Test Management and Jira is not visible after you migrate HCL OneTest™ Server from 10.2.3 to 10.5.0. To work around this problem, you must enter the details of Engineering Test Management and Jira again on HCL OneTest™ Server 10.5.0.	10.5.0	
TP-84027	When you run a schedule that has a custom code to write data into a dataset from a non-Edit branch of HCL OneTest™ Server, the **Changes** page does not list the updated dataset to publish the changes made to the dataset. To work around this problem, you must run the schedule from an Edit branch.	10.5.0	10.5.0
TP-83997	When you run a compound test that contains an API test on HCL OneTest™ Server and the API test uses a dataset or a different API test is selected at the time of test configuration, the unified report for the compound test does not provide the correct or complete information about the API test.	10.5.0	

ID	Description	Identified in version	Applicable in and until version
TP-80168	When you create a schema, the screen reader incorrectly calls the 'Expand' action as 'Maximize' and 'Collapse' action as 'Minimize' in the 'Dictionary', 'Structure', and 'Properties' panels.	10.5.0	10.5.3
TP-80166	The icons for 'Item' type and 'Group' type objects in a schema fail to meet the AA and AAA contrast (between foreground and background) requirements.	10.5.0	10.5.3
TP-78225	You cannot migrate edited test assets that were created in earlier versions to HCL OneTest™ Server 10.5.	10.2.1	10.5.0
TP-78170	The generation of test data with a large number of records into an spreadsheet in the `.xls` format sometimes fails.	10.5.0	10.5.3
TP-77323	HCL® OneTest™ Data fails to generate data in the XSD format when you provide specific seed values.	10.5.0	10.5.3
TP-75894	The 'Root' element of a schema in Data Fabrication cannot be accessed through keyboard actions.	10.5.0	10.5.3
TP-61311	When you archive a project that has a running test, that test is not stopped. The test completes execution even though it was suppose to stop.	10.5.0	
TP-82448	When you run a performance test that contains integration tests on HCL OneTest™ Server, the test log captures and displays all integration test steps under the first activity that was captured for the performance test.	10.2.3	10.2.3
TP-81800	There is a time delay for the changes that you make to your personal details provided at the time of signing up on HCL OneTest™ Server to be reflected in the UI.	10.2.3	

ID	Description	Identified in version	Applicable in and until version
TP-81017	The editing session for an API Suite is not launched when you attempt to edit the asset that was committed to remote repository with non-omitted spaces in its path.	10.2.2	10.2.2
TP-80637	While you install HCL OneTest™ Server on Azure Kubernetes Service (AKS), if the postgresql-init job does not run early, then the postgresql-init job remains in a Pending state due to insufficient CPU resources. To work around this problem, you must delete the pods controlled by deployments that make the required resources temporarily available.	10.2.2	10.2.2
TP-80565	When you deploy HCL OneTest™ Server on Azure Kubernetes Service (AKS), and if you do not set up the Domain Name Server (DNS) resolution of the client, then you cannot open a new editor session in HCL OneTest™ Server. The browser displays an error message that the IP address of the editor hosts is not found. To work around this problem, you must map the hostname details of the editor that are displayed in the error message with the server IP address in the hosts file. You can locate the hosts file in the following location on Windows systems: `C:\Windows\System32\drivers\etc`	10.2.2	10.2.2
TP-80471	HCL OneTest™ Server supports any Git repository accessible over the HTTP or SSH protocols. When you add or reconfigure a repository that is associated with a project, HCL OneTest™ Server verifies the configuration by attempting to connect and issuing a request to the given URI of the repository. You must ensure that only trusted URIs are provided as part of the configuration.	10.2.2	10.2.2
TP-80470	The default configuration of the HCL OneTest™ Server provides access to requests that are coming from any domain. The request can be integrated into other web	10.2.2	10.2.2

ID	Description	Identified in version	Applicable in and until version
	servers, and exposes HCL OneTest™ Server to Cross-origin resource sharing (CORS) attacks. To work around this problem, you must overwrite the configuration properties value during the installation of the HCL OneTest™ Server to restrict CORS to a specific allowlist by adding the following helm argument:--set-string 'gateway.extraEnv.JAVA_TOOL_OPTIONS=-Dgateway.cors.allowed-origin-patterns=*.example.com,*mydomain.net'Where example.com and mydomain.net are the domains that you want to provide access to CORS. **Note:** You can provide multiple domain names separated by comma (,).		
TP-78678	You cannot deploy HCL OneTest™ Server 10.2.1 on Azure Kubernetes Service (AKS) if you have installed the latest version of the NGINX Ingress Controller. To work around this problem, you must install the NGINX Ingress Controller by using the helm chart 3.26.0.	10.2.1	10.2.1
TP-78225	If you edited test assets in your project on HCL OneTest™ Server 10.2 and then migrated to HCL OneTest™ Server 10.2.1. After migration if there were changes added to the remote repository related to the edited test assets then the values displayed in the **Last Changed By** field might be incorrect.	10.2.1	10.2.3
TP-78219	If you update the name of the project in Jira that you used to configure on HCL OneTest™ Server, you cannot configure the updated project name in the CHANGE MANAGEMENT SYSTEM tab of HCL OneTest™ Server.	10.2.1	10.2.1
TP-78146	If you update or delete a Project area that you used to configure Engineering Test Management on HCL OneTest™ Server, then you cannot perform the following tasks:	10.2.1	10.5.0

ID	Description	Identified in version	Applicable in and until version
	• Delete your project from HCL OneTest™ Server. • Delete the configured Engineering Test Management from the HCL OneTest™ Server project.		
TP-78028	The Engineering Test Management displays the status as **Connected** on HCL OneTest™ Server even when Engineering Test Management is unavailable for any reason.	10.2.1	
TP-76628	In the Azure environment, when you generate the test data for a schema with a regular expression property set for an item type, then the job to generate the test data fails. To work around this problem, you must use a sample file that contains values for the restriction instead of setting the regular expression property for any item type.	10.2.0	10.5.3
TP-76496	When you restart the pods, the dataset service might fail to restart due to a lock on the dataset tables. To work around this problem, you must run the following commands before you restart the Kubernetes cluster: 1. Run the following command to get the password for the dataset service: kubectl get secret (my-server}-datasets -o yaml -n {namespace} 2. Run the following command to remove the lock from the dataset table: kubectl exec (my-server}-postgresql-0 -n {namespace} -i – sh -c 'PGPASSWORD=<password> psql -U datasets -d datasets -c "DROP TABLE databasechangeloglock"' 3. Run the following command to restart the Kubernetes cluster: sudo systemctl start k3s	10.2.0	10.2.0

ID	Description	Identified in version	Applicable in and until version
	Note: You must substitute the following variables with the actual value in the commands: ◦ *{my-server}* with the release name of the server that you provided during the installation of the server software. ◦ *{namespace}*with the name of the namespace that you created during the installation of Test Hub. ◦ <password> with the value that you obtained from Step 1.		
TP-76105	TLS1.1 and TLS1 are not supported on HCL OneTest™ Server. When you try to run those test assets, the test run fails with an error message about an unsupported security algorithm.	10.2.0	10.2.0
TP-74890	After you upgrade HCL OneTest™ Server from V10.1.2 to V10.1.3, then you might fail to generate test data. To work around this problem, you must perform the following steps: 1. Get the persistence volume claims (pvc) with namespace: kubectl get pvc -n <my-ots> 2. Describe the pvc to get the path of the volume: kubectl describe pv <volume> 3. Provide access permissions to the volume claim: sudo chmod 777 -R <path> 4. Change the ownership of the HIP volume: sudo chown -R root:root <path> 5. Describe the pvc to get the path of the volume: kubectl describe pv <volume number> 6. Provide access permissions to the volume claim: sudo chmod 777 -R <path> 7. Change the ownership of the data app volume: sudo chown -R root:root <path>	10.1.3	10.5.3

ID	Description	Identified in version	Applicable in and until version
	8. Run the following command: kubectl get pods -n <test-system> 9. Restart the pods by using the following command: kubectl delete pods <my-ots-hip-rest> my-ots-data-app-0 my-ots-hip-server-redis-master-0 my-ots-hip-server-0 -n <test-system> 10. Run the following commands to view the pods: kubectl get pods -n <test-system> watch kubectl get pods -n <test-system>		
TP-74534	Before you migrate user data from an earlier version to the latest version, you must publish any created datasets to the Git repository. After the migration, when you try to publish a created dataset it might fail. If it does fail, your only option is to recreate the dataset from scratch.	10.1.3	10.2.0
TP-74329	When you upgrade HCL OneTest™ Server from V10.1.1 to V10.1.3, then you might observe issues in the generation of test data on the migrated projects. To work around this problem, you must export all your HCL® OneTest™ Data projects from V10.1.1 to your computer, and then import the projects to V10.1.3.	10.1.3	10.1.3
TP-74116	After you import any JSON schema into the HCL® OneTest™ Data project, and if you modify the basic properties of the item types, then you might face issues to generate test data.	10.1.3	10.5.3
TP-73109	When you install HCL OneTest™ Server on Lightweight Kubernetes (k3s) and warnings about Ingress deprecation are displayed, you can ignore these warnings.	10.1.3	10.2.0
TP-71416	When you attempt to run an API Suite on HCL OneTest™ Server that is installed on OpenShift, and when the API project contains a vast set of resources, you might see that the **Execute** button in the **Execute test asset** dialog box is enabled after a delay or at times is not enabled at all.	10.1.3	10.1.3

ID	Description	Identified in version	Applicable in and until version
TP-72106	When you expand a dataset from the **Datasets** page, you cannot navigate to all the cells in the dataset by using the keyboard keys.	10.1.2	10.5.3
TP-71956	When you have HCL OneTest™ Server on Ubuntu and you run performance test assets such as VU schedule or Rate schedule on a static agent, then the test runs fail with an error message. To work around this problem, you must add the following parameter to the helm install command during the installation of HCL OneTest™ Server: --set networkPolicy.enabled=false If you have already installed HCL OneTest™ Server, you can modify the helm parameter value by using the following commands: helm upgrade {{main}} ./ ibm-rtas-prod -n test-system \ --reuse-values --set networkPolicy.enabled=false Where *{{main}}* is the name of the release that you provided during the installation of HCL OneTest™ Server.	10.1.2	10.1.2
TP-71950	When the server is under high load, the results microservice might be restarted indefinitely. This happens when the server was killed while holding a database lock. To work around the problem, perform the following steps: **Note:** In the following commands, <myrtas> refers to the release name of your choice and <namespace> refers to the name of the namespace that you created during the server installation process. 1. Increase the value of failureThreshold in the following command to start the service: kubectl patch sts -n <namespace> <myrtas>-results --patch	10.1.2	10.1.3

ID	Description	Identified in version	Applicable in and until version
	'{"spec": {"template": {"spec": {"containers": {"name": "results","startupProbe":{"failureThreshold":120}}}}}'}} 2. Get the database password by running the following command: kubectl get secret -n <namespace> <myrtas>-results -o jsonpath="{.data.postgresql-password}" | base64 --decode 3. Delete the troublesome database lock by running the following commands: kubectl exec -n <namespace> <myrtas>-results-postgresql-0 -i – sh -c 'PGPASSWORD=<PASSWORD> psql -U results -d results -c "DROP TABLE DATABASECHANGELOGLOCK"'kubectl exec -n <namespace> <myrtas>-results-postgresql-0 -i – sh -c 'PGPASSWORD=<PASSWORD> psql -U results -d analytics -c "DROP TABLE DATABASECHANGELOGLOCK"'kubectl exec -n <namespace> <myrtas>-results-postgresql-0 -i – sh -c 'PGPASSWORD=<PASSWORD> psql -U results -d upload -c "DROP TABLE DATABASECHANGELOGLOCK"' 4. Restart the pod by running the following command: kubectl delete pod -n <namespace> <myrtas>-results-0		
TP-71416	When you attempt to run an API Suite on HCL OneTest™ Server that is installed on OpenShift, and when the API project contains a vast set of resources, you might see that the **Execute** button in the **Execute test asset** dialog box is enabled after a delay.	10.1.2	10.1.3
TP-71030	After you install the product, if pods restart with the following log message: `ACCESS_REFUSED - Login was refused using the authentication mechanism PLAIN` For details see the broker logfile. You must uninstall the product, delete the pvc data-{{main}}-rabbitmq-0, and then install the product.	10.1.2	10.1.2

ID	Description	Identified in version	Applicable in and until version
TP-71004	Stubs are a tech-preview feature, as such some functionality might not work in all environments.	10.1.2	10.2.3
TP-69141	When you install V10.1.2 of the product, you must decrypt and encrypt each dataset by using the CSV Editor to apply the strongest encryption on each dataset that you used in the earlier version of the product.	10.1.2	10.5.3
TP-67572	If a dataset is created on the server using a path within an integration tester project and the created CSV file is subsequently referenced in a file data source, then the server presents two datasets relating to the same CSV file.	10.1.2	10.5.3
TP-70414	When you install HCL OneTest™ Server V10.1.1 on an OpenShift platform where OpenShift Container Storage (OCS) is not available (For example, the IBM Cloud) then test asset runs fail. This is because the default storage class does not support ReadWriteMany (RWX). To work around this problem, you must specify the following helm parameter during the server install: `--set execution.userlibs.persistence.storageClass=ibmc-file-bronze`	10.1.1	10.1.1
TP-69650	When you run the ubuntu-init.sh script to set up the Kubernetes environment and if the snap command is unavailable, then the command-line interface incorrectly displays the following error message for snapd: `./ubuntu-init.sh: line 232: ehco: command not found` The intended message is: `Error: snapd is not installed` Please ensure that snapd and snap are installed correctly, for example: sudo apt-get install snapd	10.1.1	
TP-69424	When you back up user data of V10.1.0 and restore it on V10.1.1 in the Ubuntu environment, one of the pods does not start. This problem occurs occasionally.	10.1.1	

ID	Description	Identified in version	Applicable in and until version
	For example, if the result pod does not start, to work around this problem, perform the following steps: **Note:** In the following commands, <*myrtas*> refers to the release name of your choice and <*namespace*> refers to the name of the namespace that you created during the server installation process. 1. Increase the value of initialDelaySeconds in the following command to start the service: kubectl patch sts -n <namespace> <myrtas>-results --patch '{"spec": {"template": {"spec": {"containers": [{"name": "results","readinessProbe":{"initialDelaySeconds":180 }}]}}}}' 2. Get the database password by running the following command: kubectl get secret -n <namespace> <myrtas>-results -o jsonpath="{.data.postgresql-password}" | base64 --decode 3. Delete the troublesome database lock by running the following command: kubectl exec -n <namespace> <myrtas>-results-postgresql-0 -i -- sh -c 'PGPASSWORD=<password> psql -U results -d results -c "DROP TABLE DATABASECHANGELOGLOCK"' 4. Restart the pod by running the following command: kubectl delete pod -n <namespace> <myrtas>-results-0		
TP-66941	You cannot run tests on the remote static agents connected to HCL OneTest™ Server that is installed in an OpenShift environment.	10.1.1	10.1.1

ID	Description	Identified in version	Applicable in and until version
TP-66834	During the installation, when you migrate user data from the previous version to V10.1.0, the datasets database does not restore properly. As a result, Data classification information disappears from HCL OneTest™ Server. You must classify the encrypted dataset again by creating a new classification.	10.1.1	10.1.1
TP-66731	You might experience errors when you publish large reports, either from desktop products or when executing a test on the server. To circumvent this problem, it is recommended that you first try to reduce your reports size by reducing the logging or details level. If the detail level cannot be reduced, you can increase the server limits to support larger reports. To do so, you must rerun the helm command that was performed when you installed Test Hub and add the following arguments: `--set-string ingress.annotations.nginx\\.ingress\\.kubernetes\.io/proxy-body-size=2g --set extraEnv.SPRING_SERVLET_MULTIPART_MAX_REQUEST_SIZE=2GB --set resources.limits.memory=4Gi` This example enables the server to accept up to 2 GB reports. This comes at the expense of a higher memory requirement. It is usually recommended to set the **resources.limits.memory** to *2 GB* over the reports size limit.	10.1.1	
TP-67215	When you configure HCL OneTest™ Server test on the UCD server and you enter a value for any of the options available for **Show Hidden Properties** in the **Edit Properties** for Run HCL OneTest™ Server test window, the values for the options do not work as designed during the test run. To work around this problem, you must not set any values for any options for **Show Hidden Properties**.	10.1.0	

ID	Description	Identified in version	Applicable in and until version
TP-66941	You cannot run tests on the remote static agents connected to HCL OneTest™ Server that is installed in an OpenShift environment.	10.1.0	
TP-64528	When you ran either a Rate or a VU Schedule on HCL OneTest™ Server and selected a disconnected agent or multiple agents to run on, the run failed and displayed an Error 500 but the execution job stayed in the *In Transition* state.	10.1.0	
TP-61494	If you unarchive a project that was archived while a test was running, the UI incorrectly shows that the test is still running after the unarchive. To workaround this problem, make sure there are no tests running when you archive a project.	10.1.0	
TP-59943	If you configure Keycloak to use an LDAP directory service, you must import and synchronize LDAP users to Keycloak. This is a known issue with Keycloak: https://issues.redhat.com/browse/KEYCLOAK-13822 To workaround the problem in Keycloak, on the Add user federation provider page make sure the following settings are set as shown: Import Users = ON, Sync Registrations = ON, and Cache Policy = DEFAULT	10.1.0	
TP-63794	When you back up the server, large amounts of additional disk space can be used. To reduce the disk space used, you should first remove the execution container volumes.	10.0.2.1	
TP-63156	Docker images that are created with no tag are assigned the "latest" tag. As the "docker ps" command does not show the "latest" tag anymore since the Docker 19.03.05 release, the Docker images are not detected and not correctly monitored.	10.0.2.1	

ID	Description	Identified in version	Applicable in and until version
TP-63149	Intermediate <none>:<none> Docker images are represented as is and cannot be selected individually in Resource Monitoring counters.	10.0.2.1	
TP-62729	When upgrading from a previous release to V10.0.2, users might see a 500 error when accessing datasets. This problem can be resolved by performing the following step: Add the following environment variable to automation/docker-compose.yml in the HCL OneTest™ Server installation directory. The line must go under the "datasets-app: environment:" section (located around line 128). Then reboot the server. SPRING_JPA_HIBERNATE_DDL-AUTO: create **Note:** Adding this environment variable will cause saved datasets options (for example, current row) to be reset to the default upon server startup. Other migrated data will be unaffected. The environment variable must be removed once the issue is resolved.	10.0.2	10.0.2
TP-62643	If you back up a project with both API suites and stubs on HCL OneTest™ Server V10.0.0, when you restore that backup on HCL OneTest™ Server V10.0.2, the stubs are missing. To workaround this problem, you can create a new project or you can move the existing GIT repository folder to a sub folder of the current repository and then move it back to the root directory. For example, a project contains a Git repository called **Calculator.git** in the root (/) folder. When you edit the root folder, you can see several other Git repository folders:`/GitNewProjectMyDemoCalculator` You select another folder, for example Calculator, and save. The Git repository called **Calculator.git** is now in the `Calculator` folder. You edit the Calculator folder, select	10.0.2	

ID	Description	Identified in version	Applicable in and until version
	the root folder, and save. The Git repository is now in the root (/) folder again.		
TP-62571	If you want to add multiple repositories to a HCL OneTest™ Server project, you cannot add another repository until the last repository completes its synchronization. To workaround this problem, wait for the last repository to synchronize, then add another repository to your project.	10.0.2	
TP-62417	A user who is no longer a member of a project might be displayed in the Secrets tab. If a user is removed from a project and then added back to that project, the user will have the same access to secrets as they had before being removed.	10.0.2	
TP-62379	If you (as a user) request to join an orphaned project, a 404 not found error results. To workaround this problem, ask your administrator to become the project owner, then request to join that project again.	10.0.2	
TP-62338	An IBM WebSphere MQ stub that is configured to use more than one transport does not appear on the Execution page of the HCL OneTest™ Server.	10.0.2	
TP-61897	If you restore a backup, it fails due to existing execution containers.	10.0.2	
TP-61786	When you enter a URL for a test result directly into a browser or you click a link to a test result in the Jenkins console output from the Jenkins HCL OneTest™ Server plugin, and you are not logged in to HCL OneTest™ Server, you see the following message: `Request cannot be processed: Unauthorized page.`	10.0.2	

ID	Description	Identified in version	Applicable in and until version
	To workaround this problem, log in to HCL OneTest™ Server, and then enter the URL or click the test result link to see the test result in a browser.		
TP-61494	If you unarchive a project that was archived while a test was running, the UI incorrectly shows that the test is still running after the unarchive. To workaround this problem, make sure there are no tests running when you archive a project.	10.0.2	
TP-61446	When you unregister a remote docker agent, running and schedule tests are not canceled. To workaround this problem, stop any running tests and cancel any scheduled future executions for that agent before unregistering the remote docker agent.	10.0.2	
TP-61311	When you archive a project that has a running test, that test is not stopped. The test completes execution even though it was suppose to stop.	10.0.2	
TP-61153	The project search field is case-sensitive. For example, if your project is named MYPROJECT and you search for myproject, HCL OneTest™ Server does not show any results.	10.0.2	
TP-60737	Git repositories that contain projects at the root level and not under a subdirectory are not supported and will not work currently with HCL OneTest™ Server.	10.0.2	
TP-60059	When you schedule a test run and click the **Time** field a list of times displays. If you enter a value in the **Time** field, you cannot move the cursor by using the left or right (**<-** , **->**) keyboard keys. This is a known issue with the following date picker:https://github.com/Hacker0x01/react-datepicker/issues/1834 To workaround this problem, use the cursor.	10.0.2	

ID	Description	Identified in version	Applicable in and until version
TP-60059	When you schedule a test run and click the "Time" field a list of times displays. If you enter a value in the "Time" field, you cannot move the cursor by using the left or right (<- , ->) keyboard keys. This is a known issue with the following date picker:https://github.com/Hacker0x01/react-datepicker/issues/1834 To workaround this problem, use the cursor.	10.0.1	
TP-59943	If you configure Keycloak to use an LDAP directory service, you should import and synchronize LDAP users to Keycloak. This is a known issue with Keycloak. To workaround the problem in Keycloak, on the Add user federation provider page make sure the following settings are set as shown:Import Users = ON, Sync Registrations = ON, and Cache Policy = DEFAULT	10.0.1	
TP-59774	When checking in a project without a root directory an error is thrown while at the **Execution** dialog box that does not allow you to select an environment, nor does it remember your selection of secrets. Do not check-in desktop projects to the root of the Git repository. All checked-in desktop projects must contain a parent directory within the Git repository. For example, if the desktop project contained the following assets:-`MyProject/.project-MyProject/MyTest.testsuite` Then the Git repository should contain-`MyProject/.project-MyProject/MyTest.testsuite`and not just`/.project` -> Will cause an error/MyTest.testsuite	10.0.1	
TP-59139	During the installation of HCL OneTest™ Server the following error message is shown and can be safely ignored:`139836115718592:error:2406F079:random`	10.0.1	

ID	Description	Identified in version	Applicable in and until version
	`number generator:RAND_load_file:Cannot open file:../crypto/rand/randfile.c:88:Filename=/root/.rnd`		
TP-58572	Execution results with status "Complete with Error" or "Launch failed" are displayed with inconsistent colors. In verdict statistics views (in Overview and Results pages) they are displayed in blue, whereas in the detail tables they are displayed in red.	10.0.1	
TP-58017	In case of a slow network, the cloning of a repository might fail with an error. In such a case, you can delete and add the repository again.	10.0.1	
TP-58292	If you interrupt the installation or uninstallation of the HCL OneTest™ Server software, the installation or uninstallation cannot complete. For a workaround, see Troubleshooting installation.	10.0.0	
TP-58170	Selecting a date in the Execution Results card on the Project Overview page does not update the results for that date.	10.0.0	
TP-58125	HCL OneTest™ Server does not currently remove Docker containers after they have exited. The containers will exist in a 'stopped' state but should consume only disk resources (not memory). The Docker command docker ps -a shows both running and stopped containers. This problem is normally not an issue but if the list grows large it consumes disk resources or might make it difficult to find other stopped containers. On HCL OneTest™ Server system (assuming Linux), to view stopped execution containers, use the following Docker command: docker ps --filter "status=exited" | grep 'EC_[0-9]' | awk ' {print $1}'	10.0.0	

ID	Description	Identified in version	Applicable in and until version
	To remove stopped execution containers from the system, use the following Docker command: docker ps --filter "status=exited" | grep 'EC_[0-9]' | awk '{print $1} ' | xargs docker rm		
TP-58104	When monitoring a test execution with the Statistical report, data might stop updating after about an hour. To get updated data, reload the page in your browser.	10.0.0	
TP-58061	A server shutdown will not stop currently running executions. It is recommended to not shutdown the server if there are active executions running. If the server is shut down while assets are executing the reported status of those executions might be incorrect.	10.0.0	
TP-58061	A server shutdown will not stop currently running executions. It is recommended to not shutdown the server if there are active executions running. If the server is shut down while assets are executing the reported status of those executions might be incorrect.	10.0.0	10.1.2
TP-58024	HCL OneTest™ Server Login and Account Settings pages are only available in English.	10.0.0	
TP-58017	In case of a slow network, the cloning of a repository might fail with an error. In such a case, you can delete and add the repository again.	10.0.0	
TP-58017	In case of a slow network, the cloning of a repository might fail with an error. In such a case, you can delete and add the repository again.	10.0.0	
TP-58016	If there are errors encountered during server shutdown or while uninstalling the server, they can be caused by active asset executions in progress. It is recommended to allow	10.0.0	

ID	Description	Identified in version	Applicable in and until version
	any executions in progress to complete before shutting down or uninstalling the server. If you want to force a shutdown, the following commands can be used to identify and stop active containers: Show all running and stopped containers that are part of server asset execution: docker ps -a -f 'name=EC_*' Remove containers, including running containers: docker rm -f $(docker ps -q -a -f 'name=EC_*')		
TP-57971	After adding a repository to a project, you see a GREEN status indicator. When you remove the ssh key from Github, the GREEN status changes to RED. When you update Github with a new ssh key and update the same for the repository in HCL OneTest™ Server, the status indicator does not immediately change to GREEN but it does change to GREEN after a couple of minutes.	10.0.0	
TP-57832	HCL OneTest™ Server for Windows is provided for evaluation purposes. It is intended for non-production use, for example, trials and demonstrations. It is neither supported nor recommended for production use due to insufficient stability and reliability of Docker for Windows while running Linux images at the time of release. HCL OneTest™ Server is tested on both Windows 10 and Windows Server 2016 Standard Edition and it operates correctly. However, it is suitable and supported for non-production use only. If native Linux is unavailable, you can run HCL OneTest™ Server for production use on a Linux virtual machine on their Windows host, provided that the Linux virtual machine is running a supported version of Linux.	10.0.0	
TP-57198	When working in multiple HCL OneTest™ Server tabs concurrently, a message “You are already logged in” might	10.0.0	

ID	Description	Identified in version	Applicable in and until version
	sometimes be displayed. This message occurs when your session has expired and you have since logged in again, but then attempt a navigation action in a browser tab other than the one in which you logged in. This is a known Keycloak issue. For more information, see https://issues.jboss.org/browse/KEYCLOAK-5179. To workaround this problem, close the tab and open a new one.		
TP-57198	When working in multiple HCL OneTest™ Server tabs concurrently, a message `You are already logged in` might sometimes be displayed. This message occurs when your session has expired and you have since logged in again, but then attempt a navigation action in a browser tab other than the one in which you logged in. This is a known Keycloak issue. For more information, see https://issues.jboss.org/browse/KEYCLOAK-5179. To workaround this problem, close the tab and open a new one.	10.0.0	
TP-56453	After about 20 minutes of an inactive browser session for HCL OneTest™ Server, if you attempt to use the existing browser session, you must authenticate again by providing your username and password. When you do so, the message "You took too long to log in. Try again.“ is displayed and you must enter your credentials again.	10.0.0	
HCL OneTest™ Performance Agent	With 10.0, you might expect a reduction in the number of Virtual Users (VUs) supported by HCL OneTest™ Performance Agent depending upon workload characteristics. This variation is the result of the product shipping with OpenJDK™ 8 with OpenJ9 instead of IBM JRE. OpenJDK 8 with OpenJ9 JRE exhibits a higher utilization of CPU for the workload. Up to 30% or more decrease in the supported VUs capacity may be expected for CPU-bound workloads. However, for memory-bound workloads, a much smaller, if any reduction in VUs capacity may be expected.	10.0.0	

ID	Description	Identified in version	Applicable in and until version
	If you need higher performance and have access to IBM JRE 8.0.5.35 or later that contains the latest security fixes, you can switch from OpenJDK 8 with OpenJ9 to IBM JRE. For more information, see http://www.ibm.com/support/docview.wss?uid=ibm10956413.		

Chapter 2. System Requirements for DevOps Test Hub 2026.09 (12.0.0)

This document includes information about hardware and software requirements for HCL DevOps Test Hub (Test Hub).

Contents

- Hardware on page 95
- Operating systems and Containers on page 95
    - Bit version support on page 96
    - Operating systems on page 96
    - Container Platforms on page 96
- Host prerequisites on page 97
    - Licensing on page 97
    - Runtime environment on page 97
    - Database on page 98
- Supported software on page 98
    - Development tools on page 98
    - Integrations on page 99
    - Web browsers on page 101
    - Databases on page 102
- Disclaimers on page 94

Disclaimers

This report is subject to the Terms of Use and the following disclaimers:

The information contained in this report is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied, including but not limited to the implied warranties of merchantability, non-infringement, and fitness for a particular purpose. In addition, this information is based on HCL's current product plans and strategy, which are subject to change by HCL without notice. HCL shall not be responsible for any direct, indirect, incidental, consequential, special or other damages arising out of the use of, or otherwise related to, this report or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from HCL or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of HCL software.

References in this report to HCL products, programs, or services do not imply that they will be available in all countries in which HCL operates. Product release dates and/or capabilities referenced in this presentation may change at any time at HCL's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Discrepancies found between reports and other HCL documentation sources may or may not be attributed to different publish and refresh cycles for this tool and other sources. Nothing contained in this report is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in

any specific sales, revenue growth, savings or other results. You assume sole responsibility for any results you obtain or decisions you make as a result of this report.

Notwithstanding the Terms of Use users of this site are permitted to copy and save the reports generated from this tool for such users own internal business purpose. No other use shall be permitted.

Hardware

You can find information about the hardware requirements for HCL DevOps Test Hub (Test Hub).

Hardware	**Requirement**	**Notes**
Disk space	256 GB	Additional resources are required by the cluster for the execution of test assets. Disk space requirements can grow in line with quantity of test assets and results stored.
Memory	32 GB	Additional resources are required by the cluster for the execution of test assets.
Processor	8 CPUs	Additional resources are required by the cluster for the execution of test assets.
Network	Gigabit (1000) Ethernet	
Other hardware		Operating system support is specified by the supported platform for Kubernetes and Red Hat OpenShift. Refer to their associated documentation for a list of supported operating systems.

Related information

System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94

Operating systems and Containers

You can find details about the supported operating systems and containers for HCL DevOps Test Hub (Test Hub).

Contents

- Bit version support on page 96
- Operating systems on page 96
- Container Platforms on page 96

Bit version support

Different parts of a product might run on the same operating system but support different application bitness.

For example, one part of the product might run only in 32-bit mode, whereas another might support 64-bit tolerate mode.

Bitness	Description
64-Tolerate	The product or part of the product runs as a 32-bit application in the 64-bit platforms listed as supported.
64-Exploit	The product or part of the product runs as a 64-bit application in the 64-bit platforms listed as supported.

Operating systems

Operating system	Hardware	Bitness	Notes
Ubuntu Server 22.04 LTS	x86-64	64-Exploit	Running K3s v1.36
Ubuntu Server 24.04 LTS	x86-64	64-Exploit	Running K3s v1.36
Ubuntu Server 26.04 LTS	x86-64	64-Exploit	Running K3s v1.36
RHEL 8.10	x86-64	64-Exploit	Running K3s v1.36
RHEL 10	x86-64	64-Exploit	Running K3s v1.36
RHEL 10.2	x86-64	64-Exploit	Running K3s v1.36

Container Platforms

You can find details about the supported container platforms.

Platform	Version
Azure Kubernetes Service	1.36
Red Hat OpenShift	4.20
Red Hat OpenShift	4.21

Related information

System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94

Host prerequisites

You can find the prerequisites that support the operating capabilities for HCL DevOps Test Hub (Test Hub).

Contents

- Licensing on page 97
- Runtime environment on page 97
- Database on page 98

Licensing

License Server	Version
My HCLSoftware (MHS)	Latest Cloud version
HCL Local License Server	5.0
HCL Local License Server	5.1
HCL Local License Server	5.2
HCL Local License Server	5.4

Runtime environment

Supported software	Version	Notes
Helm CLI	4.1.4 or later	Required locally for Helm.

Database

Database	Version	Note
PostgreSQL	18	To deploy Test Hub on an external database.

Related information

System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94

Supported software

You can find details about the additional software for HCL DevOps Test Hub (Test Hub) that are supported.

Contents

- Development tools on page 98
- Integrations on page 99
- Web browsers on page 101
- Databases on page 102

Development tools

Supported software	Version
GitHub	2.19
	2.21
	2.22
	3.1
	3.3
	3.4

Supported software	Version
	3.5
	3.6
	3.7
	3.8
	3.9
	3.10
	3.16
	3.12
	3.14
	3.16
	3.17
	3.19
	3.20.x

Integrations

Supported software	Version
Azure Plugins	Cloud version, latest
DevOps Test UI	12.0.0
DevOps Test Performance	12.0.0
DevOps Test Integrations and APIs	12.0.0

Supported software	Version
DevOps Test Workbench	12.0.0
DevOps Loop	3.0.0
GitHub Actions	Latest
Grafana	11.x
	13.x
HCL® Compass	2.2
HCL DevOps Deploy	8.x
HCL® Launch	7.1
	7.2.1
	7.2.3
	7.3
	7.3.2
IBM® Engineering Test Management	7.0.3
	7.1
	7.2
IBM® Rational® ClearQuest®	10.x
IBM® DevOps Deploy	8.x
IBM® UrbanCode™ Deploy	7.1

Supported software	Version
	7.2.1
	7.2.3
	7.3
	7.3.2
HCL DevOps Plan (Plan)	3.x
Jenkins	2.x
Jira	8.x
	9.x
	10.x
Jira Cloud	Cloud version, latest
Xray app for Jira	Cloud version, latest

Web browsers

Browser	Version
Apple Safari	18 or later
Google Chrome	150 or later
Microsoft Edge	149 or later
Mozilla Firefox	152 or later

Browser	Version
Mozilla Firefox ESR	153

Databases

You can connect to the following databases to define data structures and generate test data:

Databases	Version
Db2 Database (all editions)	11, 12, and 13
Oracle Database	19c, 21c, and 23ai
PostgreSQL	16, 17, and 18

Related information

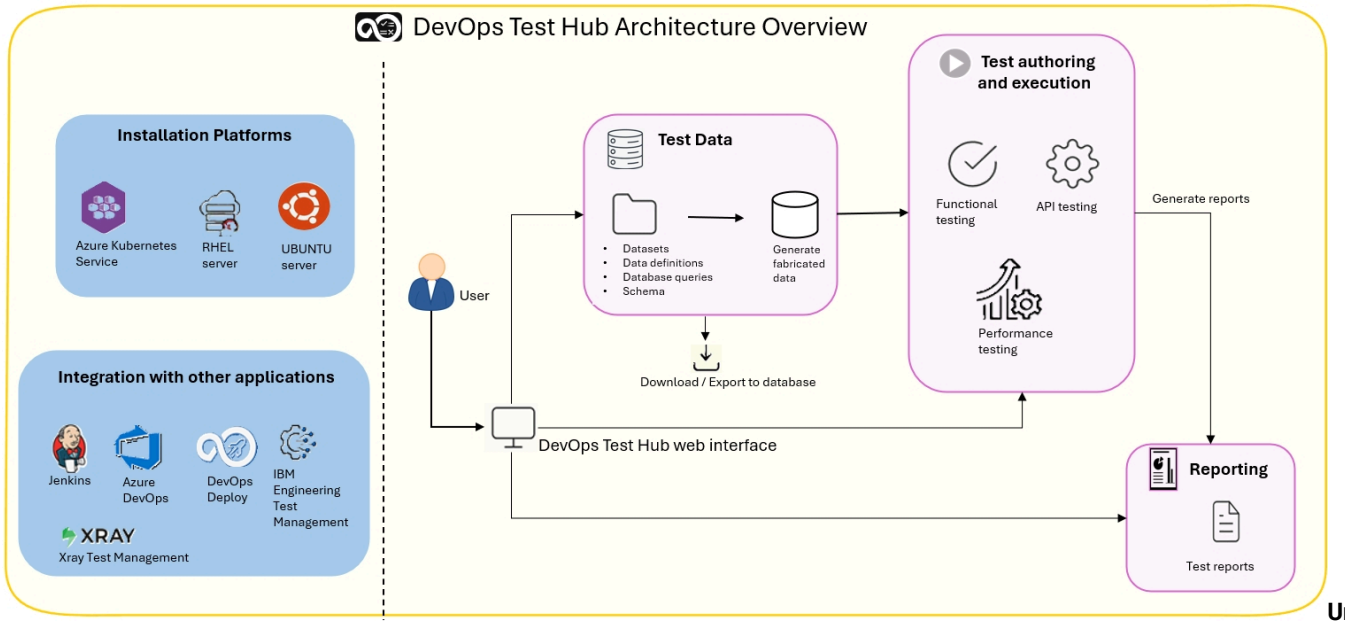
System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94

Chapter 3. Getting Started Guide

This guide provides an introduction to HCL DevOps Test Hub (Test Hub) covering key features, task flows, and setup instructions. You can get started with installation, integration, user access management, and core functionalities like test data creation, test asset management, and reporting. This guide is intended for new users.

Introduction to DevOps Test Hub

HCL DevOps Test Hub (Test Hub) is a web-based continuous testing platform that consolidates test data, environments, test runs, and reports. It is built on cloud-native technologies and Docker, and it supports API, functional, and performance tests. Test Hub offers scalability across cloud and on-premises systems with Kubernetes and IBM® Red Hat OpenShift support.

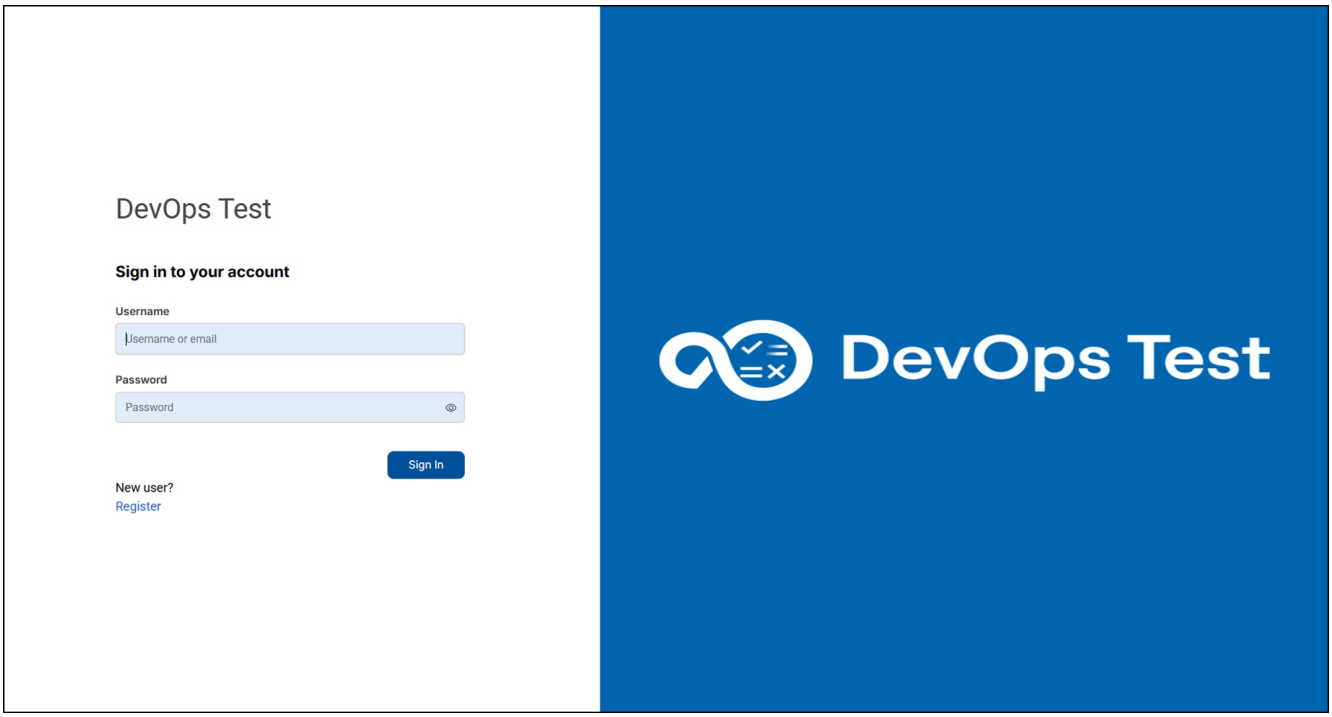


Unified test dashboard

You can use the intuitive dashboard to create, manage, and track test activities. By using the dashboard, you can efficiently manage test data and collaborate with your team on test requirements.

Login page

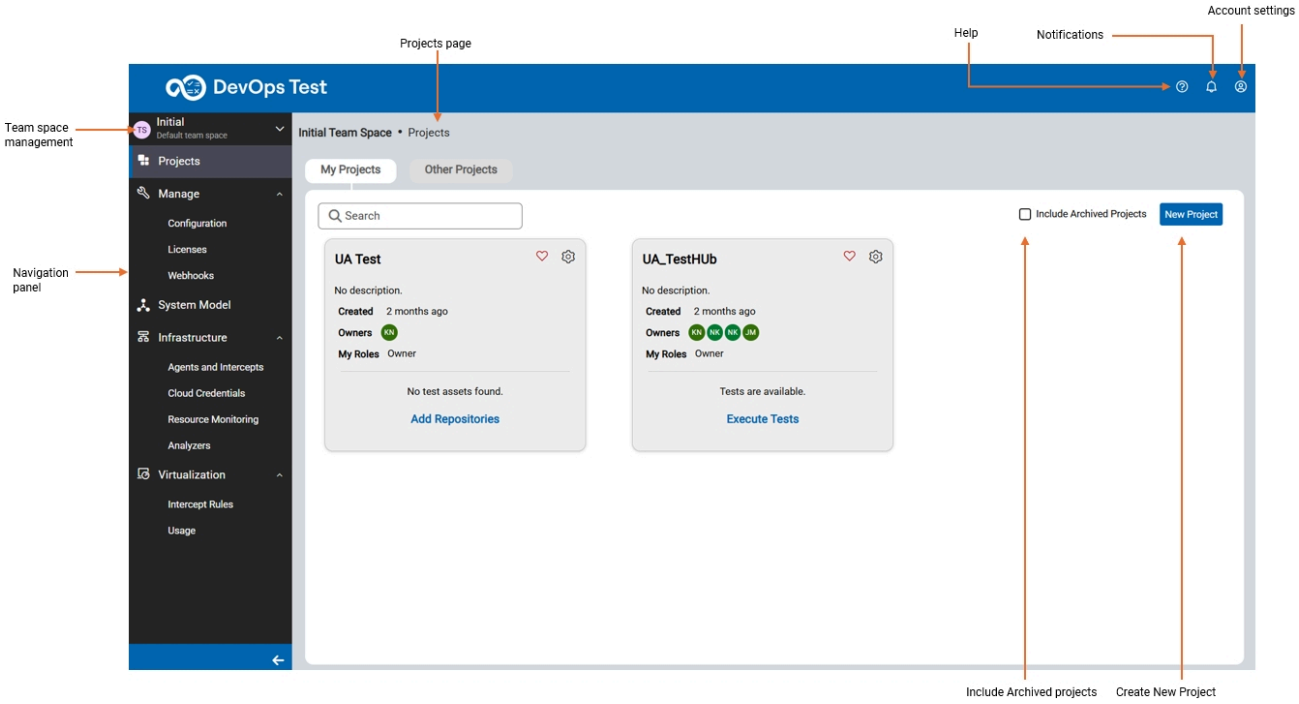
The login page is the primary access point to the platform. Only registered users can access project data and testing environments. If you are a new user, you must complete the registration process before you can log in.



Note: A session timeout prompt is displayed if you are inactive for 30 minutes. Before the session ends, a notification appears, prompting you to **continue the session** or **log out**. If no action is taken, the session is automatically terminated.

Home page

The home page lists your projects and other projects, which makes it easy for you to manage different projects within an organization. You can request access to join existing projects, create a new project, and view archived projects.



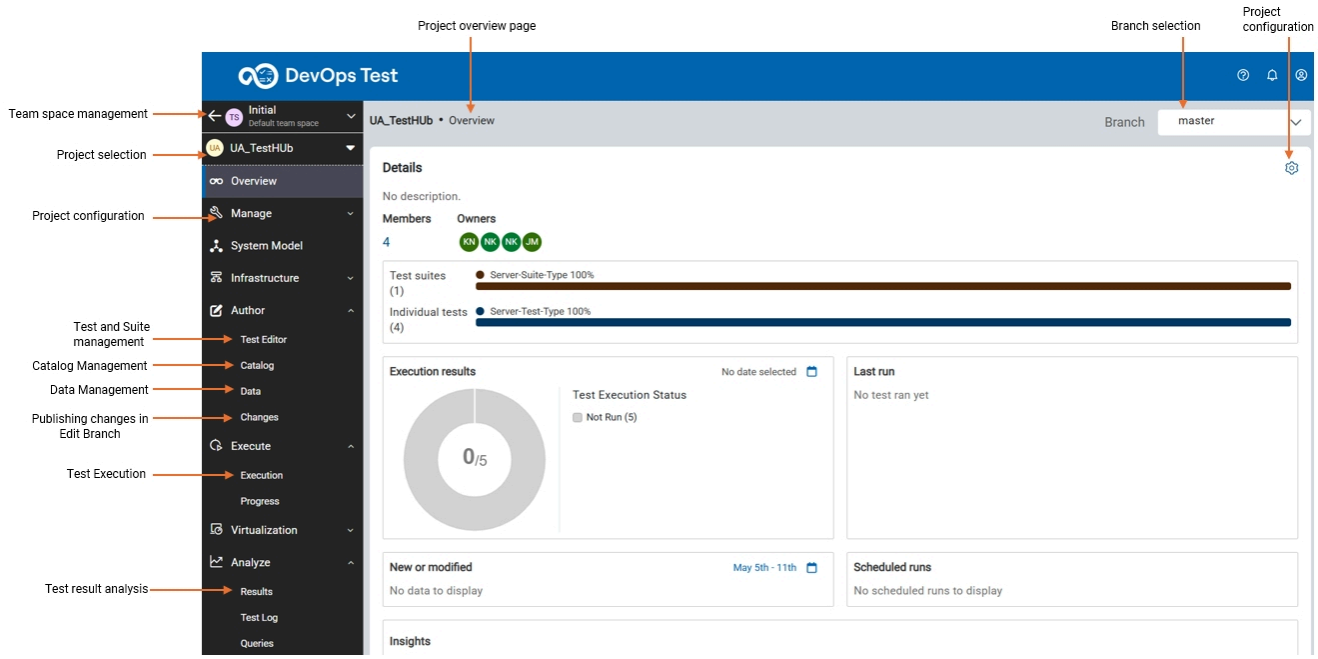
Navigation panel

The navigation panel in Test Hub provides quick access to sections such as **Projects**, **Manage**, **System Model**, **Infrastructure**, and **Virtualization**. You can easily navigate your projects, configure settings, define system models, monitor infrastructure, and manage virtualization. Each section of the dashboard is designed to support you with a specific function, making your testing workflow more efficient and intuitive.

After you select a project, the panel displays the **Author**, **Execute**, and **Analyze** sections for test creation, data management, test execution, and reporting activities.

Overview page

You can manage and monitor all testing activities in one place on the project **Overview** page.



Logout

When you want to end your session with Test Hub, you can log out by clicking the profile icon  , and selecting **Logout**.



Click here to watch the visual walkthrough.

Related information

Capabilities on page 107

Installation on page 107

User tasks on page 108

Capabilities

HCL DevOps Test Hub (Test Hub) unifies test data, test environments, execution, and reporting in a web-based platform designed for both testers and non-testers. You can efficiently manage test data, collaborate seamlessly with your team, and integrate your testing workflows with DevOps pipelines for streamlined delivery and execution.

Test data authoring and management

You can create, edit, and manage test data directly in Test Hub. With support for concurrent multi-user editing, and teams can work on test datasets at the same time without causing conflicts or overwriting each other's changes. The built-in data generator helps to generate fabricated data for different testing scenarios.

Role-based access and security

Test Huboffers robust role-based access control, where project owners can assign permissions according to defined user roles. This ensures secure test data management and controlled access to sensitive assets such as credentials.

Report analysis and resource monitoring

Test Hub includes advanced reporting capabilities by offering insights into test execution, system performance, and resource utilization. It consolidates unified test results, helping teams make data-driven decisions to improve software quality.

Connected performance testing agents

You can link existing performance testing agents to Test Hub to execute schedules and Accelerated Functional Testing (AFT) suites by using your current infrastructure for seamless integration.

Integration with DevOps tools

You can integrate Test Hub with popular DevOps tools like Jenkins, Azure DevOps, HCL DevOps Deploy, Jira, and GitHub. These integrations help you automate test execution within your CI/CD pipelines and streamline defect-tracking workflows.

Installation

You can install HCL DevOps Test Hub (Test Hub) on various platforms to manage your test data and requirements. You can integrate Test Hub with your existing infrastructure to streamline testing and collaborate more effectively with your team.

You can install Test Hub on the following platforms:

- Azure Kubernetes Service (AKS)
- Linux (by using k3s)
- IBM® Red Hat OpenShift

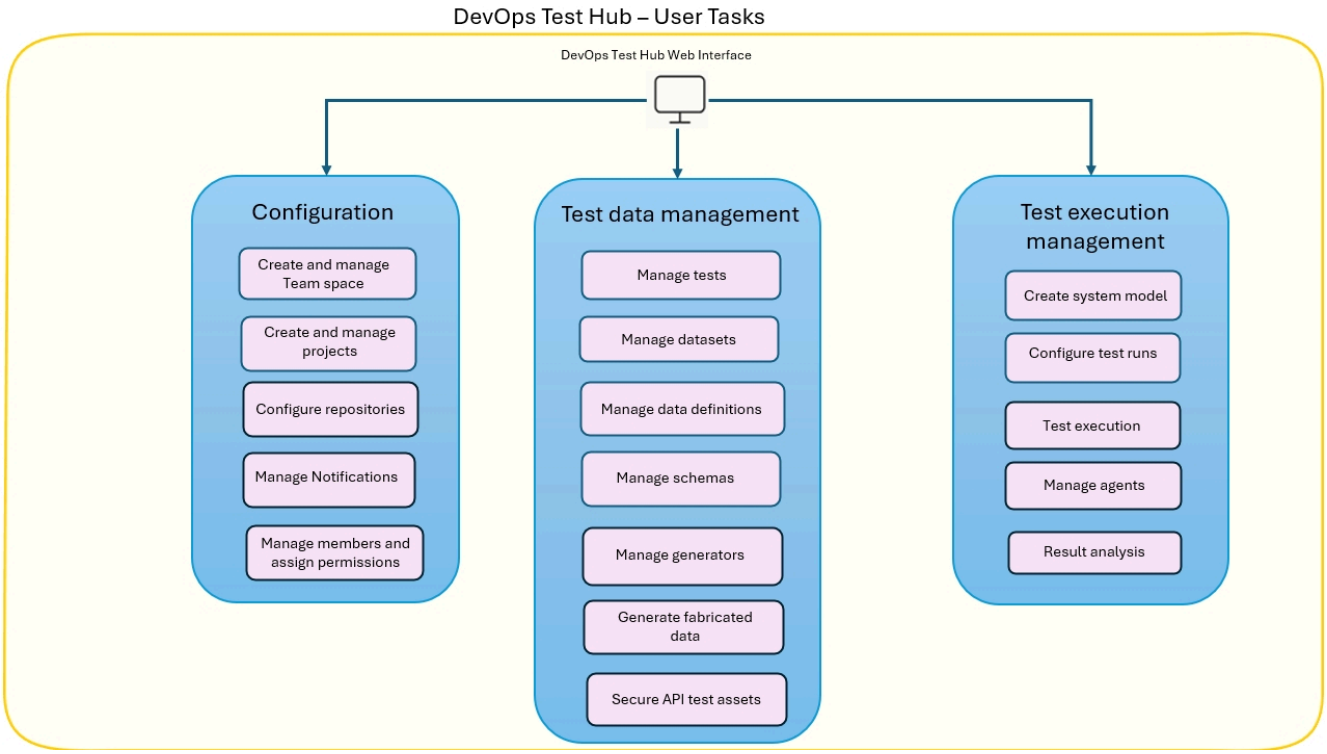
Before installing Test Hub, refer to the system requirements to ensure compatibility and a smooth setup process.

For detailed installation instructions, see the Installation of DevOps Test Hub on page 115 section in the Administrator Guide on page 113.

User tasks

Test Hub offers a range of user tasks designed to simplify collaboration and optimize testing workflows. You can efficiently create, manage, and analyze test data to ensure a smooth and effective testing process.

You can perform the following user tasks in Test Hub to support different stages of the testing lifecycle.



Member access and permissions

You can manage access in HCL DevOps Test Hub (Test Hub) by assigning roles and permissions to control who can view, edit, or execute test assets within a project. With effective access management, you can protect your project data while making collaboration easier and more secure for your entire team.

For more information on configuring your Test Hub account, see the Configuration of DevOps Test Hub on page 140 section in the Administrator Guide on page 113.

For more information on roles, access, and permissions, see Managing members and their roles in a team space on page 1207 and Managing access to the server projects on page 1220 in the Test Manager Guide on page 1199.

Compatibility and supported assets

You can find information about the environments, file formats, and asset types that are compatible with HCL DevOps Test Hub (Test Hub). You can use this information to verify if your assets and environments are supported before creating or importing tests into Test Hub.

Related information

Supported versions of assets and resources on page 109

Supported suites and tests on page 110

Supported virtual services on page 112

Supported versions of assets and resources

You can find information about the versions of assets, resources, and agents that are supported in HCL DevOps Test Hub (Test Hub). You can also find information about the versions of the desktop clients that are supported.

The following table lists the versions of assets or resources created in the desktop clients along with the agents, or proxies that are supported in Test Hub:

Resource	Supported	Not supported
Assets or resources created in HCL DevOps Test UI (Test UI)	Assets or resources created in Test UI that is of a version earlier than or the same as Test Hub.	Assets or resources created in any later version of Test UI that is later than the version of Test Hub.
Assets or resources created in HCL DevOps Test Integrations and APIs (Test Integrations and APIs)	Assets or resources created in Test Integrations and APIs that is of a version earlier than or the same as Test Hub.	Assets or resources created in any later version of Test Integrations and APIs that is later than the version of Test Hub.
Assets or resources created in HCL DevOps Test Performance (Test Performance)	Assets or resources created in Test Performance that is of a version earlier than or the same as Test Hub.	Assets or resources created in any later version of Test Performance that is later than the version of Test Hub.
HCL DevOps Test Performance Agent (Test Performance Agent)	Test Performance Agent, which is the same version as Test Hub.	Before you install the agents that are of the current version, see Management of performance agents on page 542.

Resource	Supported	Not supported
HCL DevOps Test Integrations and APIs Agent (Test Integrations and APIs Agent)	Test Integrations and APIs Agent, which is the same version as Test Hub.	Before you install the agents that are of the current version, see Management of API agents on page 551.
HTTP proxies	Installation of the proxy component from Test Virtualization Control Panel, which is the same version as Test Hub.	Before you use HTTP proxies, see Setting up intercepts on page 940.

If there is a mismatch in the versions of the assets, resources, or agents with the version of Test Hub that you are using, any of the following events can occur:

- Warnings or Errors are displayed when you add a repository to a project in your team space.
- Warnings or Errors are displayed when you open a project that you migrated from a previous version of Test Hub.
- Warnings or Errors are displayed when you add or manually refresh a repository to which you added assets or resources that were created on a desktop client with a version previous to or later than the version of Test Hub that you want to use.
- Test runs that you start or schedule on an agent fail to run.
- You cannot view the proxies that are registered with Test Hub on the **Agents and Intercepts** page.

Supported suites and tests

You can find information about the types of Test Suites and tests that you can configure for a run in a project on HCL DevOps Test Hub (Test Hub).

The following table lists the Test Suites and test types that are supported and those tests that are not supported to run on Test Hub:

Supported Suites	Supported tests	Unsupported tests
• Accelerated Functional Testing (AFT) Suites • Compound Tests that contain any of the following types of tests: ◦ Web UI tests ◦ HTML tests ◦ Mobile tests	• Individual Web UI test • Individual Windows test • Individual Mobile test	• Traditional HTML tests • Traditional non-HTML tests
• API Suites that include the following Test Suites: ◦ Test Suites that contain scenarios with references to local stubs. ◦ Test Suites that contain stubs that virtualize services run in	API tests	Suites or tests with subscribe actions that operate in the watch mode.

Supported Suites	Supported tests	Unsupported tests
any Kubernetes cluster or in the Kubernetes cluster that hosts Test Hub. • Virtual services or Stubs		
• Compound Tests that contain performance tests • Rate Schedules • VU Schedules	Individual Performance test	• A single test script of any test extension. • Tests that belong to 32-bit test extensions and SOA Quality included in VU Schedule, Rate Schedule, and Compound Test. • Citrix, SAP, or Siebel as an individual test or as part of a Compound Test.
Postman resources		
JMeter tests		
JUnit tests		
Selenium tests		

Note: You must create Compound Tests for the same type of tests. For example, you can create a Compound Test that has different mobile tests that use different mobile testing platforms.

Restriction: You cannot merge folders that contain different test projects in the same project directory. For example, you cannot create a project directory that contains a Performance test project and an API Suite project. You must create separate project directories for each test type. For example, you must create the Performance test in the project directory as `/MyPerfProject/` and the API Suite in the project directory as `/MyAPIProject`, before you commit the projects to the remote repository.

Related information

Test run configurations on page 610

Supported virtual services

You can find information about the virtual services that are supported on HCL DevOps Test Hub (Test Hub).

You can start a running instance of the virtual services from Test Hub to run on the following locations:

- A registered API agent.
- The default Kubernetes cluster of Test Hub.
- A remote Kubernetes cluster.

You can run all virtual services on registered API agents except the virtual services that use CICS transport, DB2/Z transport, or Istio.

The default Kubernetes cluster or a remote Kubernetes cluster supports running of the following types of virtual services:

- Virtual services that utilize the WebSphere® MQ transport.
- Virtual services that utilize the HTTP transport.

Related information

Management of virtualized services on page 936

Accessibility features

Accessibility features help users who have physical disabilities, such as visual and, hearing impairment, or limited mobility, to use the software products successfully.

Accessibility compliance

The product documentation is published by using Oxygen XML WebHelp Responsive. To understand the accessibility compliance status for Oxygen XML WebHelp Responsive, refer to WebHelp Responsive VPAT Accessibility Conformance Report.

Accessing UI elements

Test Hub supports navigation in the UI by using different methods such as a mouse, keyboard, or touchpad.

You can use the keyboard keys such as **Tab**, arrow keys such as **UP**, **DOWN**, **LEFT**, and **RIGHT** to navigate to the different pages in the **Navigation** pane or to the different action labels in the right pane on the UI.

Chapter 4. Administrator Guide

This guide describes how to install the HCL DevOps Test Hub (Test Hub) software.

After you install the software, you can perform administration tasks such as license configuration, user management, security, memory and disk usage management, back up and restore user data, and other tasks that a server administrator can perform. This guide is intended for administrators.

As a *Server Administrator*, you can assign the role of a *Server Administrator* to any user. For information about how to change the role, see Default user administration on page 142.

Remember: When your role is changed from a user to a *Server Administrator*, you must log out and log in again so that Test Hub can apply the change in your role.

Configuration of an external database

You can configure an external database for HCL DevOps Test Hub (Test Hub) by using a Helm chart to control and manage your resources. You can host the database in the same cluster as Test Hub or in a remote cluster. You can configure the external database either during a new installation or by upgrading an existing installation of Test Hub.

Prerequisites

Before you configure the external database, you must ensure that you have completed the following tasks:

- Installed Test Hub.

  Note: This prerequisite applies only when you upgrade an existing instance and connect to an external database.

- Installed PostgreSQL 18.
- Obtained user credentials for the external database with the following permissions:
  - Data manipulation permissions for insert, delete, update, and select.
  - Schema modification permissions to alter a table, create a table, truncate, drop, and create or modify an index.
- Ensured that the database has sufficient connections. The recommended value for *max_connections* is `143`.

External database support in Test Hub

Test Hub automatically includes and installs a local PostgreSQL server that runs in the same cluster.

Note: To use an external database, you must provide the database URL in the Helm chart.

Table 1. Helm parameters affecting the external database connection

Parameter	Description	Default value
postgresql.external	Flag to indicate whether the external database connection is used.	*false*
postgresql.hostname	Hostname or IP address of the external database server.	NA
postgresql.username	Username to access the database.	*postgres*
postgresql.existingPassword	Password to access the database.	NA
postgresql.port	Port for the database connection. **Note:** This parameter is optional. The default port is *5432*. Configure it only if your database uses a different port.	*5432*
postgresql.params.sslmode	Server certificate validation mode. **Note:** This parameter is optional. The recommended value is *verify-full*, as it provides the most secure SSL validation.	*verify-full*
global.privateCaBundleSecretName	Secret name where the self-signed certificates are stored.	NA

Example command

```
helm upgrade main --install --reuse-values . \
  --timeout 810s \
  --create-namespace \
  -f ./values-k3s.yaml \
```

```
--set global.domain=172.22.132.41.nip.io \
-n devops-system \
--set-literal passwordSeed=secretSauce \
--set signup=true \
--set hclLicensingURL=https://license.brr.com \
--set hclLicensingID=2yzfernw \
--set imageRegistry=registry.brr.com \
--set postgresql.external=true \
--set postgresql.hostname=pgdb.postgres.database.azure.com \
--set postgresql.username=judgejudy \
--set postgresql.existingPassword=Secret!0wl \
--set demo.enabled=true
```

Note: Specify `--reuse-values` to prevent `helm upgrade` from overwriting existing customizations and reverting to default values.

Methods for configuring an external database connection

You can configure the Helm chart to connect to an external database by using one of the following methods:

- **Database with a trusted certificate authority**

  you must specify the following parameters in the Helm chart:

  ```
  --set postgresql.external=true \
  --set postgresql.hostname=kmpg.postgres.database.azure.com \
  --set postgresql.username=theadmin \
  --set postgresql.existingPassword=theadminpassword
  ```

- **Database within your network by using a self-signed certificate**

  You must specify the following parameters in the Helm chart:

  ```
  --set global.privateCaBundleSecretName=my-internal-ca-bundle \
  --set postgresql.external=true \
  --set postgresql.hostname=mydbhost.corp.example.com \
  --set postgresql.username=postgres \
  --set postgresql.existingPassword=changeme
  ```

Installation of DevOps Test Hub

To get started working with HCL DevOps Test Hub (Test Hub), you must first install Test Hub.

You can install Test Hub on the following platforms:

- Azure Kubernetes Service (AKS)
- IBM® Red Hat OpenShift
- Linux Server (by using k3s)

Installation task flows

Azure Kubernetes Service

You must perform the following tasks to install Test Hub:

1. Install Azure command-line interface (Azure CLI) software and connect to your Azure subscription.
2. Install kubectl.
3. Install the Helm software.
4. Install Emissary-ingress.
5. Install Test Hub.

IBM® Red Hat OpenShift Container platform

You must perform the following tasks to install Test Hub:

1. Set up a IBM® Red Hat OpenShift Container Platform (OCP).
2. Install the OCP command-line interface (CLI) and the Helm software.
3. Install Test Hub.

Linux Server

You must perform the following tasks to install Test Hub:

1. Set up Linux Server.
2. Install the OpenSSH server and the Helm software.
3. Install Test Hub.

To learn more about the installation of Test Hub on platforms such as AKS, RHEL, or Ubuntu refer to the following topics:

Installation of DevOps Test Hub on Azure Kubernetes Service

You can find information about the tasks that you can perform to install HCL DevOps Test Hub (Test Hub) on Azure Kubernetes Service (AKS).

You must have installed Helm to run the Helm commands. For more information on installing Helm, refer to Installing Helm.

To install Test Hub on AKS, refer to the instructions in the readme file.

Installation of DevOps Test Hub on Red Hat OpenShift

You can install HCL DevOps Test Hub (Test Hub) on the IBM® Red Hat OpenShift platform.

You must have installed Helm to run the Helm commands. For more information on installing Helm, refer to Installing Helm.

To install Test Hub on IBM® Red Hat OpenShift, refer to the instructions in the readme file.

Installation of DevOps Test Hub on Linux

You can find information about the tasks that you can perform to install HCL DevOps Test Hub (Test Hub) on Red Hat Enterprise Linux (RHEL) or Ubuntu.

You must have installed Helm to run the Helm commands. For more information on installing Helm, refer to Installing Helm.

To install Test Hub on RHEL or Ubuntu, refer to the instructions in the readme file.

Management of DevOps Test Hub features

After you install HCL DevOps Test Hub (Test Hub), you can manage certain features based on your requirements.

During the installation of Test Hub, if you did not use any Helm parameters, you can still use those parameters even after installation is complete.

You can use the helm upgrade command to enable or disable server features after the installation of Test Hub is complete.

Similarly, you can use the other additional Helm parameters to enable or disable features based on your requirements. See Additional Helm parameters on page 117.

Additional Helm parameters

You can find the information about Helm parameters that you can use during the installation of HCL DevOps Test Hub (Test Hub).

As a system administrator when you install or upgrade Test Hub, you can pass values for some of the individual parameters by using `--set`. The following table lists such parameters that you can use while running the helm install or helm upgrade command to enable or disable certain features in Test Hub.

Parameters	Values	Description

keycloak.signup-
.enabled
	`true`
	
	`false`
		Use this parameter to show or hide the Sign-Up link on the Test Hub Login page.
		
		The default value of the keycloak.signup.enabled parameter is set to `false` to enhance the security of Test Hub.
		
		Note: You can use this parameter only during the installation of Test Hub.
		
		If you have administrator access to Keycloak, then you can also enable the self-registration by using the Keycloak UI after the installation of Test Hub is complete. For more information, refer to the Keycloak documentation.

Parameters	Values	Description

router.allowedOrigin	IP address or domain name of other servers.	Use this parameter to integrate call requests from other web servers to Test Hub.
		
		The default configuration of Test Hub rejects all Cross-origin resource sharing (CORS). Therefore, you must use the router.allowedOrigin parameter to restrict CORS to a specific whitelist.
		
		For example,
		`--set router.allowedOrigin=http://http.example.com:*,https://https.mydomain.net:*`
		
		Where, `http.example.com` and `https.mydomain.net` are the domains that you want to provide access to Test Hub.
		
		You can also provide the value of router.allowedOrigin as the IP address when you cannot resolve a specific Domain Name Server (DNS). If your domain can be accessible through IP address on a specific port, then you can provide the value of router.allowedOrigin as follows:
		`--set router.allowedOrigin=*://ip-address:*,*://ip-address:port_number`
		
		For example,
		`--set router.allowedOrigin="*://192.0.2.0:*"`
		`--set router.allowedOrigin="*://192.0.2.1:3000"`
		
		Note: You can provide multiple domain names or IP addresses separated by a comma (,).

networkPolicy.egress.enabled	`true`
	
	`false`
		Set this parameter value to `true` to restrict traffic from Test Hub to private IP addresses.
		
		The default configuration of Test Hub does not create any Egress policy to restrict the other endpoints that can connect with Test Hub. Therefore, you must use the networkPolicy.egress.enabled parameter to enable the Egress policy that can restrict traffic to private IP addresses.
		
		For example,
		`--set networkPolicy.egress.enabled=true`
		
		You can inspect the deployed policy by running the following command:
		`kubectl get networkpolicy -n {namespace} {{main}}-allow-egress -oyaml`

Parameters	Values	Description
		Note: You must substitute `{namespace}` with the name of the namespace and `{{main}}` with the release name that you used during the installation of Test Hub.
execution.template.resources.limits.memory	`<size_of_memory>`Gi	Use this parameter to run VU schedules that contain multiple Web UI tests or a large number of virtual users or a combination of Web UI, performance, and API tests. The default value of the execution.template.resources.limits.memory parameter is set to `3Gi`. For example, `--set execution.template.resources.limits.memory=5Gi`

Related information

Installation of DevOps Test Hub on Azure Kubernetes Service on page 116

Installation of DevOps Test Hub on Linux on page 117

Installation of DevOps Test Hub on Ubuntu

Management of DevOps Test Hub features on page 117

Upgrade of DevOps Test Hub

When you want to use the enhanced functionalities of HCL DevOps Test Hub (Test Hub), you must upgrade Test Hub to the latest version.

Before you upgrade Test Hub, you must ensure that you have completed the following tasks:

- Completed all test executions that are running on the existing version of Test Hub.
- Stopped all stub executions that are running on the existing version of Test Hub.
- Canceled any scheduled test runs that have a future date or time.
- Informed users that Test Hub is offline temporarily during the upgrade process.

For more information about specific versions of software requirements, see System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94.

The following topics provide platform specific upgrade methods::

Upgrading of DevOps Test Hub on Azure Kubernetes Service

When you want to use the latest version of Test Hub on Azure Kubernetes Service (AKS), you must upgrade HCL DevOps Test Hub (Test Hub) from a previous version.

You must have installed Helm to run the Helm commands. For more information on installing Helm, refer to Installing Helm.

To add the repository and pull the Helm chart, you must run the following commands:

```
helm pull --untar oci://hclcr.io/ot/hcl-devops --version 12.0.000
```

To upgrade Test Hub installed on Azure Kubernetes Service (AKS), refer to the instructions in the readme file.

Upgrading of DevOps Test Hub on Red Hat OpenShift

When you want to use the latest version of HCL DevOps Test Hub (Test Hub) on Red Hat OpenShift, you must upgrade Test Hub from a previous version.

You must have installed Helm to run the Helm commands. For more information on installing Helm, refer to Installing Helm.

To add the repository and pull the Helm chart, you must run the following commands:

```
helm pull --untar oci://hclcr.io/ot/hcl-devops --version 12.0.000
```

To upgrade Test Hub installed on Red Hat OpenShift, refer to the instructions in the readme file.

Upgrading of DevOps Test Hub on Linux

When you want to use the latest version of HCL DevOps Test Hub (Test Hub) on Red Hat Enterprise Linux (RHEL) or Ubuntu, you must upgrade Test Hub from a previous version.

You must have installed Helm to run the Helm commands. For more information on installing Helm, refer to Installing Helm.

To add the repository and pull the Helm chart, you must run the following commands:

```
helm pull --untar oci://hclcr.io/ot/hcl-devops --version 12.0.000
```

To upgrade Test Hub installed on Linux, refer to the instructions in the readme file.

Migration of user data

You can find information about the migration limitations and methods when you want to migrate data such as projects, test resources, configuration details of integrated applications, and so on. You can migrate data that is generated in an earlier version of HCL DevOps Test Hub (Test Hub) after you upgrade to or install the latest version, and then use that data in the latest version.

Limitations for user data migration

Before you can decide on the migration method for the version of Test Hub, you must review the limitations that apply to the migrated data.

Table 2. Migration limitations

Migrating from	Migrating to	Limitation	The workaround, if available
10.5.4 or earlier	11.0.0	You cannot migrate the Jira integration or its associated Xray integration to Test Hub 11.0.0 or later, as the method to authenticate the Jira connection is changed in 11.0.0 or later. **Note:** The links to the Jira or Xray project that are created in earlier versions will be available and will work after Jira and Xray are reconfigured in Test Hub 11.0.0 or later.	You must reconfigure Jira as a change management system on Test Hub 11.0.0 or later. You must also reconfigure the Xray integration, if configured.
10.5.0	10.5.1	You cannot migrate the IBM® Engineering Test Management data that is generated when it is configured as a quality management solution on Test Hub 10.5.0 to Test Hub 10.5.1.	You must reconfigure IBM® Engineering Test Management as a quality management solution in the latest version of Test Hub.
10.5.0 or earlier	10.5.1	If you configured Jira as a change management system on Test Hub, and subsequently changed the name of the project in Jira, then you cannot then configure the changed project name in the **CHANGE MANAGEMENT SYSTEM** tab of Test Hub 10.5.1.	You must reconfigure Jira as a change management system in the latest version of Test Hub.
10.5.0 or earlier	10.5.1	If you configured IBM® Engineering Test Management as a quality management solution on Test Hub 10.5.0, and subsequently updated or deleted a Project area in Engineering Test Management, then you cannot perform the following tasks:	You must reconfigure IBM® Engineering Test Management as a quality management solution in the latest version of Test Hub.

Table 2. Migration limitations (continued)

Migrating from	Migrating to	Limitation	The workaround, if available
		• Delete your project from Test Hub. • Delete the configured Engineering Test Management from the Test Hub project.	
10.2.3	10.5.0	If you configured IBM® Engineering Test Management on Test Hub 10.2.3, the information about Engineering Test Management is not visible after you migrate Test Hub from 10.2.3 to 10.5.0 or later.	You must reconfigure IBM® Engineering Test Management as a quality management solution in the latest version of Test Hub.
10.2.3	10.5.0	If you configured Jira on Test Hub 10.2.3, the information about Jira is not visible after you migrate Test Hub from 10.2.3 to 10.5.0 or later.	You must reconfigure Jira as a change management system in the latest version of Test Hub.
10.2.3 or earlier	10.5.0	You cannot migrate edited test assets that were created in Test Hub 10.2.3 or earlier, to Test Hub 10.5.0 or later.	You might have to open and edit the test assets in the latest version of Test Hub.
10.2.0	10.2.1 or later	If you edited test assets in your project on Test Hub 10.2.0 and migrated them, then the values displayed in the **Last Changed By** field might be incorrect for the edited test assets.	

Migration methods

The data that is generated in Test Hub can be migrated and used in a later version of Test Hub with except for the limitations detailed in Table 2: Migration limitations on page 121.

You can refer to the following table to know the details of the migration when you want to upgrade to the latest Test Hub version:

Table 3. Migration method used during an upgrade

OS / Platform	Server version	Upgrade method	Migration method
Ubuntu	10.1.3 or later	Upgrade by new install	When you upgrade Test Hub by uninstalling the existing version, and installing the required version of Test Hub, then you must perform the migration task after the installation of Test Hub is complete. For more information about the migration process, see Migrating user data on Ubuntu from 10.1.3 or later on page 125 or Migrating user data on Ubuntu from 10.1.2 or earlier on page 124.
	10.1.2 or earlier		
Azure Kubernetes Service	10.2.3 or later	In-place upgrade	When you upgrade Test Hub by using the helm upgrade command, the migration of user data takes place during the upgrade process.

Migrating user data on Red Hat OpenShift

When you want to use the data, which was in your earlier version, in a new version of Test Hub, you can migrate those user data.

Before you begin

You must have upgraded Test Hub to a new version.

About this task

You must replace {main} with the name of the release that you provided during the installation of Test Hub in all the instances in this procedure.

Remember: The `devops-system` is the name of the namespace that you created during the installation of Test Hub. If you created a namespace by using a different value, then you must use that value in place of `devops-system` in all the instances in this procedure.

1. Run the following script from the `ibm-rtas-prod/files` directory to create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes:

   ```
   migrate.sh create-pvcs -n devops-system {{main}}
   ```

2. Run the following script from the `ibm-rtas-prod/files` directory to merge the data into Test Hub:

   ```
   migrate.sh merge-dbs -n devops-system {{main}}
   ```

3. Run the following command to remove the resources that were created during the migration process:

   ```
   migrate.sh delete-temp-resources -n devops-system {{main}}
   ```

4. Run the following script from the `/files` directory to verify and test the installed instance:

```
/files/helm-test-diag.sh {main} -n devops-system
```

Results

You have migrated the user data from the earlier version to the new version.

What to do next

You can perform the following tasks:

- Configure licenses for team spaces. See Configuring licenses for team spaces on page 184.
- Configure Test Hub. See Configuration of DevOps Test Hub on page 140.

Migrating user data on Ubuntu from 10.1.2 or earlier

When you want to use the data, which was in your earlier version, in a new version of Test Hub, you can migrate those user data.

Before you begin

You must have upgraded Test Hub to a new version.

About this task

The following table lists the variables that you must replace with the actual value in the command.

Variables	Description
{main}	The name of the release that you provided during the installation of Test Hub.
{backup-release}	The Helm release name of Test Hub where you store the backup data.
{target-release}	The Helm release name of Test Hub where you want to restore the data.



Remember: The `devops-system` is the name of the namespace that you created during the installation of Test Hub. If you created a namespace by using a different value, then you must use that value in place of `devops-system` in all the instances in this procedure.

1. Run the following script from the `hcl-onetest-server/files` directory to create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes:

```
migrate.sh create-pvcs -n devops-system {{main}}
```

2. Run the following script from the `hcl-onetest-base` directory to back up the data:

```
sudo backup.sh create-pvc-links -v ~/migration-pvc-links
```

3. Run the following command to stop the cluster and Test Hub:

```
k3s-killall.sh
```

4. Run the following script from the `hcl-onetest-base` directory to restore the backed-up data:

```
sudo backup.sh restore -v ~/migration-pvc-links --release {backup-release}:{target-release}
 {backup-file-name}
```

5. Run the following command to restart Kubernetes and to start Test Hub:

```
sudo systemctl start k3s
```

6. Run the following script from the `hcl-onetest-server/files` directory to merge the data into the server:

```
migrate.sh merge-dbs -n devops-system {{main}}
```

7. Run the following command to remove the resources that were created during the migration process:

```
migrate.sh delete-temp-resources -n devops-system {{main}}
```

8. Run the following script from the `/files` directory to verify and test the installed instance:

```
/files/helm-test-diag.sh {main} -n devops-system
```

Results

You have migrated the user data from the earlier version to the new version.

What to do next

You can perform the following tasks:

- Configure licenses for team spaces. See Configuring licenses for team spaces on page 184.
- Configure Test Hub. See Configuration of DevOps Test Hub on page 140.

Migrating user data on Ubuntu from 10.1.3 or later

When you want to use the data, which was in your earlier version, in a new version of Test Hub, you can migrate those user data..

Before you begin

You must have upgraded Test Hub to a new version.

About this task

The following table lists the variables that you must replace with the actual value in the command.

Variables	Description
{main}	The name of the release that you provided during the installation of Test Hub.
{backup-release}	The Helm release name of Test Hub where you store the backup data.
{target-release}	The Helm release name of Test Hub where you want to restore the data.

The `devops-system` is the name of the namespace that is created during the setting up of the k3s environment.

1. Run the following script from the `hcl-onetest-base` directory to create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes:

   ```
   sudo backup.sh create-pvc-links
   ```

2. Stop the Kubernetes cluster by running the k3s-killall.sh command.
3. Restore the backed-up user data by running the following commands:

   ```
   sudo ./backup.sh restore [options] {backup-file-name}
   ```

   You can use the following parameters along with the restore command:

   - --namespace or -n: Use this parameter to restore a specific namespace from the backup file. If you do not mention the namespace, then volumes from all the namespaces in the backup file are restored. You can map one namespace to another namespace by using a colon (:). The syntax is:

     ```
     --namespace <name of the namespace> [:<target-namespace>]
     ```

   - --release or -r: Use this parameter if the Helm release name of the server to which the backup is being restored is different than the Helm release name of the server where the backup was taken. The syntax is:

     ```
     --release <backup-release>:<target-release>
     ```

   - --volumes or -v: Use this parameter to specify the directory path of the volumes. The syntax is:

     ```
     --volumes <path-of-the-directory>
     ```

   - -k or --confirm: Use this parameter to skip the confirmation step.

4. Restart the Kubernetes cluster by running the systemctl start k3s command to start Test Hub.
5. Run the following script from the `hcl-onetest-server/files` directory to create all the missing databases:

   ```
   migrate.sh create-missing-dbs {{main}}
   ```

   Note: You can expect the output of the command to contain several errors on objects that already exist. They do not indicate a problem with the execution of the script.

6. Run the following script from the `/files` directory to verify and test the installed instance:

   ```
   /files/helm-test-diag.sh {main} -n devops-system
   ```

Results

You have migrated the user data from the earlier version to the new version.

What to do next

You can perform the following tasks:

- Configure licenses for team spaces. See Configuring licenses for team spaces on page 184.
- Configure Test Hub. See Configuration of DevOps Test Hub on page 140.

Backup and restoration of the DevOps Test Hub data

You must back up HCL DevOps Test Hub (Test Hub) data before you uninstall the current version of Test Hub or upgrade to the latest version. You must back up the data to avoid data loss or inaccessibility of data.

A backup is the process of creating a copy of the data on your system and storing it elsewhere, generally on secondary storage. You can then use that copy to recover if your original data is lost or becomes inaccessible.

A restore is a process of copying the backed-up data from the secondary storage and restoring it to the original location. You can restore the backed-up data when your original data is lost or becomes inaccessible.

You must back up and restore the data when you perform the following tasks:

- Move the existing environment to a new system.
- Change the name of the release or namespace that you used during the installation of Test Hub.
- Minimize the downtime of Test Hub during disaster recovery.
- Upgrade to the latest version of Test Hub.

To learn more about backup and restoration of Test Hub data on platforms such as Azure Kubernetes Service (AKS) and Ubuntu, see the following topics:

- Backup and restoration of DevOps Test Hub data on Azure Kubernetes Service on page 134
- Backup and restoration of DevOps Test Hub data on Ubuntu on page 130

Backup and restoration of data on IBM® Red Hat OpenShift

To secure the data in HCL DevOps Test Hub (Test Hub) that is installed on the IBM® Red Hat OpenShift platform, you can back up the data. At any point in time, after you install Test Hub, you can restore the backed-up data.

Preparing IBM® Red Hat OpenShift cluster to backup and restore DevOps Test Hub data

You must prepare your Red Hat OpenShift cluster before you back up or restore the data of HCL DevOps Test Hub (Test Hub).

Before you begin

You must have installed Red Hat OpenShift CLI. For more information, refer to the Getting started with the CLI section in the Red Hat OpenShift documentation.

About this task

Velero is one of the tools that are available to back up and restore the data of Test Hub that is installed on the Red Hat OpenShift platform. You must prepare your cluster before you back up or restore the data by using Velero. You can also use the other tools to back up and restore the data. If you use a different tool, then you must include the Persistent Volumes in the cluster.

1. Log in to the cluster by using oc login.
2. Install and configure Velero with the Restic Integration.
3. Set the name of the namespace in which Test Hub is installed by running the following command:

```
NS=<namespace>
```

4. Update stateful sets to apply the annotations required by Velero to back up the PVs used by the pods by running the following commands:

```
for sts in $(oc get sts -n "$NS" -o name); do \
  if ! oc -n "$NS" patch --type=json "$sts" -p \
    '[{"op":"add","path":"/spec/template/metadata/annotations/backup.velero.io~1backup-volumes",
 "value":"data"}]' 2>/dev/null; then
      oc -n "$NS" patch --type=json "$sts" -p \
       '[{"op":"add","path":"/spec/template/metadata/annotations",
 "value":{"backup.velero.io/backup-volumes":"data"}}]'
  fi \
done
```

5. Change the security restriction of the pods by running the following commands:

```
for sts in $(oc get sts -n "$NS" -oname); do \
  if oc get -n "$NS" "$sts" -ojsonpath='{.spec.template.spec.securityContext}' | grep -q
 "runAsNonRoot:true"; then \
    echo $sts
    oc patch -n "$NS" "$sts" --type json \
      -p '[{"op":"replace", "path":"/spec/template/spec/securityContext", "value": {"runAsNonRoot":
 false}}]'
  fi \
done
```

Results

You have prepared your Red Hat OpenShift cluster to back up the server data.

What to do next

You can back up the server data. See Backing up DevOps Test Hub data on Red Hat OpenShift on page 128.

Related information

Velero Documentation

Backing up DevOps Test Hub data on Red Hat OpenShift

When you want to upgrade to a new version of Test Hub or to move your existing environments to new systems, you must back up the data.

Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall Test Hub.
- Installed Test Hub and all of the Test Hub pods are in `Running` state.

- Communicated to the users that Test Hub might be unavailable for some time until the process is complete.
- Prepared your cluster to back up and restore data by using Velero. See Preparing IBM Red Hat OpenShift cluster to backup and restore DevOps Test Hub data on page 127.

1. Log in to the cluster by using oc login.
2. Start the backup process by running the following command:

```
velero backup create <backup_file_name> --include-namespaces=<name_of the_namespace>
```

Note: You must replace `<backup_file_name>` and `<name_of the_namespace>` with the actual value in the command.

Results

You have backed up the data of Test Hub.

Related information

Velero Documentation

Restoring DevOps Test Hub data on IBM® Red Hat OpenShift

You can restore the data that is backed up in Test Hub at any point in time after you install latest version.

Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall Test Hub.
- Communicated to the users that Test Hub might be unavailable for some time until the process is complete.
- Backed up the data from Test Hub.

1. Log in to the cluster by using oc login.
2. Start the restore process by entering the following command:.

```
velero restore create --from-backup=<backup-name> --restore-volumes
```

Results

You have restored the data of Test Hub.

Related information

Velero Documentation

Backup and restoration of DevOps Test Hub data on Ubuntu

To secure the data of HCL DevOps Test Hub (Test Hub) that is installed on the Ubuntu server, you can back up the data. At any point in time, after you install Test Hub, you can restore the backed-up data.

Backing up DevOps Test Hub data on Ubuntu

When you want to upgrade to a new version of Test Hub or to move your existing environments to new systems, you must back up the data.

Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall Test Hub.
- Been granted with the *sudo* access.
- Installed Test Hub and all of the Test Hub pods are in `Running` state.
- Communicated to the users that Test Hub might be unavailable for some time until the process is complete.

1. Log in to the Ubuntu server and open a terminal.
2. Run the following command to extract the base:

   ```
   helm pull --untar hclsoftware/hcl-onetest-base --version 13.1054.0
   ```

3. Change to the `hcl-onetest-base` directory.

   This directory contains the `backup.sh` script which is required to complete the backup operation.

4. Create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes by running the following command:

   ```
   sudo ./backup.sh create-pvc-links
   ```

   You can use the following optional parameter along with the **create-pvc-links** command:

   --volumes or -v: Use this parameter to specify the directory path of the Volumes.

   Remember:

- If you did not provide the volume parameter, by default, the command creates a sub-directory in the same directory where you have the backup script.
- You must ensure that the directory you create must be empty or not exist to avoid script failure.
- If you specify a directory by using the -v parameter, then you must use the same parameter value when you restore the data.

5. Run the following command to stop the cluster and Test Hub:

```
k3s-killall.sh
```

6. Run the following command to create a backup of the existing user data:

```
sudo ./backup.sh create [options] <backup-file-name>
```

After you run this command, a backup of the local Persistent Volumes is created. The backup is created as tar archives that is compressed by using gzip (.tar.gz). The **create** command archives the Persistent Volumes into the *<backup-file-name>*.

You can use the following parameters along with the **create** command:

- --namespace or -n: Use this parameter to back up the Persistent Volumes in the specified namespace. If you do not mention the namespace, then all the Volumes from all the namespaces are included in the backup. The syntax is:

```
-namespace <name of the namespace>
```

- --volumes or -v: Use this parameter to specify the directory path of the Volumes. The syntax is:

```
-volumes <path-of-the-directory>
```

Remember:

- If you did not provide the volume parameter, by default, the command creates a sub-directory in the same directory where you have the backup script.
- If you specify a directory by using the -v parameter, then you must use the same parameter value when you restore the data.

For example,

```
sudo ./backup.sh create --namespace devops-system my-backup.tar.gz
```

This command creates a backup file named *my-backup.tar.gz* that contains all of the Persistent Volumes associated with pods available in the *devops-system* namespace.

7. Run the following command to restart the cluster and Test Hub:

```
sudo systemctl start k3s
```

Results

You have backed up the data of Test Hub.

Restoring DevOps Test Hub data on Ubuntu

You can restore the data that is backed up in Test Hub at any point in time after you install latest version.

Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall Test Hub.
- Been granted with the *sudo* access.
- Communicated to the users that Test Hub might be unavailable for some time until the process is complete.
- Backed up the data from Test Hub.

About this task

The following procedure is for the restoration of the server data from the current release. If you are upgrading Test Hub, then you can restore the data from the earlier version to the latest version during the installation of Test Hub. For more information, see the **Related information** section.

1. Log in to the Ubuntu server and open a terminal.
2. Run the following command to extract the base:

   ```
   helm pull --untar hclsoftware/hcl-onetest-base --version 13.1054.0
   ```

3. Change to the `hcl-onetest-base` directory.
   This directory contains the `backup.sh` script which is required to complete the restore operation.
4. Create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes by running the following command if you have not done before:

   ```
   sudo ./backup.sh create-pvc-links
   ```

   You can use the following optional parameter along with the **create-pvc-links** command:

   --volumes or -v: Use this parameter to specify the directory path of the Volumes.

   For example, `backup.sh create-pvc-links -v hcl-onetest-base/my-pvc-links` creates a directory called `my-pvc-links` that stores the metadata required to restore your user data.

    **Remember:**

- If you did not provide the volume parameter, by default, the command creates a sub-directory in the same directory where you have the backup script.
- You must ensure that the directory you create must be empty or not exist to avoid script failure.

5. Run the following command to stop the cluster and Test Hub:

```
k3s-killall.sh
```

6. Run the following command to restore the backed-up user data:

```
sudo ./backup.sh restore [options] <backup-file-name>
```

The **restore** command overwrites the existing Persistent Volumes with data from the *<backup-file-name>*.

You can use the following parameters along with the restore command:

- --namespace or -n: Use this parameter to restore a specific namespace from the backup file. If you do not mention the namespace, then volumes from all the namespaces in the backup file are restored. You can map one namespace to another namespace by using a colon (:). The syntax is:

```
--namespace <name of the namespace> [:<target-namespace>]
```

- --release or -r: Use this parameter if the Helm release name of the server to which the backup is being restored is different than the Helm release name of the server where the backup was taken. The syntax is:

```
--release <backup-release>:<target-release>
```

- --volumes or -v: Use this parameter to specify the directory path of the volumes. The syntax is:

```
--volumes <path-of-the-directory>
```

- -k or --confirm: Use this parameter to skip the confirmation step.

For example,

```
sudo ./backup.sh restore --namespace test-system:devops-system -r rel-1:rel-2 -k my-backup.tar.gz
```

This command restores the volumes that are backed up from the *rel-1* release in the *test-system* namespace to the *rel-2* release in the *devops-system* namespace and skips the confirmation step.

7. Run the following command to restart the cluster and Test Hub:

```
sudo systemctl start k3s
```

Note:

- The cluster and Test Hub might take some time to restart and until that time, you cannot access the Test Hub URL.
- You can run the `kubectl get pods -A` command to verify that the Kubernetes environment is working. After a while, the status of the pods must be `Running` state.

Results

You have restored the data of Test Hub.

Related information

Upgrading of DevOps Test Hub on Linux on page 120

Backup and restoration of DevOps Test Hub data on Azure Kubernetes Service

To secure the data in HCL DevOps Test Hub (Test Hub) that is installed on Azure Kubernetes Service (AKS), you can back up the data. At any point in time, after you install Test Hub, you can restore the backed-up data.

Attention: The instructions provided are an example of how an AKS cluster can be configured to use Velero for backing up and restoring the Test Hub data. The procedures to backup and restore the server data might be not valid in all the scenarios.

You must refer to the Velero and Azure Kubernetes Service documentation to identify how you can configure Velero in your AKS cluster based on your existing AKS setup to backup and restore the Test Hub data.

- Velero Documentation
- AKS Documentation

Preparing the Azure Kubernetes Service cluster to back up the DevOps Test Hub data

When you want to back up the HCL DevOps Test Hub (Test Hub) data, you must prepare your Azure Kubernetes Service (AKS) cluster to store the backed-up data.

Before you begin

You must have completed the following tasks:

- Ensure that you have an Azure subscription along with an **Owner** or **Contributor** and **User-Access Administrator** role on the Azure subscription.

  You can create a subscription from Microsoft Azure if you do not have a valid subscription.
- Downloaded the Velero 1.14.x from the Velero repository.

CAUTION: The instructions provided in this procedure are an example of how an AKS cluster can be configured to use Velero 1.14.x Microsoft Azure plugin for backing up and restoring the Test Hub data. The procedures to backup and restore the server data might be not valid in all the scenarios.

- Copied the Velero executable file into one of the directories in the **PATH** environment variable.
- Installed Azure Command-Line Interface (CLI). For more information, refer to the Azure CLI documentation.

About this task

Velero is one of the tools that is available to back up and restore the server data. You must prepare your AKS cluster where you installed Test Hub before you back up the data by using Velero.

1. Run the following command to log in to AKS interactively:

   ```
   az login
   ```

   Result

   The command-line interface opens a browser and displays the Log-in page of AKS.
2. Sign in with your AKS account credentials.
3. Run the following command to set your active subscription:

   ```
   az account set --subscription <subscription_name>
   ```

   You must replace `<subscription_name>` with the name of your active Azure subscription.
4. Run the following command to set the location of your AKS cluster:

   ```
   az configure --defaults location=<location_name>
   ```

   For example, az configure –defaults location=eastus2

   Tip: You can run the following command to get the list of all the available locations:

   ```
   az account list-locations --query "[].{DisplayName:displayName, Name:name}" -o table
   ```
5. Run the following commands to deploy the velero backup template for taking the backup of files:

   ```
   RESOURCE_GROUP=devops
   INSTANCE=devops0
   IDENTITY_NAME=velero
   BLOB_CONTAINER=velero
   AZURE_BACKUP_RESOURCE_GROUP=Velero_Backups
   AZURE_STORAGE_ACCOUNT_ID="velero$(date -u +%s)"
   MANAGED_CLUSTER_RESOURCE_GROUP=$(az aks show -g $RESOURCE_GROUP -n $INSTANCE -o tsv --query
    nodeResourceGroup)
   cd hcl-devops/azure/    #Navigate to the directory where you have the installation files
   az group create -n $AZURE_BACKUP_RESOURCE_GROUP --location  <location_name>
   az deployment group create -g $MANAGED_CLUSTER_RESOURCE_GROUP --template-file azure/backup.bicep
    --parameters backupResourceGroupName=$AZURE_BACKUP_RESOURCE_GROUP identityName=$IDENTITY_NAME
    storageAccountId=$AZURE_STORAGE_ACCOUNT_ID blobContainerName=$BLOB_CONTAINER
   ```

Note: You must replace `<location_name>` with the name of your location of the AKS cluster.

6. Run the following commands to create service account and cluster role binding:

```
AZURE_SUBSCRIPTION_ID=$(az account list --query '[?isDefault].id' -o tsv)
AZURE_TENANT_ID=$(az account show -s $AZURE_SUBSCRIPTION_ID --query tenantId -o tsv)

IDENTITY_CLIENT_ID="$(az identity show -g $MANAGED_CLUSTER_RESOURCE_GROUP -n $IDENTITY_NAME
 --subscription $AZURE_SUBSCRIPTION_ID --query clientId -o tsv)"

kubectl create namespace velero

cat <<EOF | kubectl apply -n velero -f -
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    azure.workload.identity/client-id: $IDENTITY_CLIENT_ID
  name: velero
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: velero-cluster-admin
subjects:
- kind: ServiceAccount
  name: velero
  namespace: velero
roleRef:
  kind: ClusterRole
  name: cluster-admin
  apiGroup: rbac.authorization.k8s.io
EOF
```

7. Run the following commands to establish federated identity credential between the identity and the service account issuer & subject:

```
SERVICE_ACCOUNT_ISSUER=$(az aks show --resource-group $RESOURCE_GROUP --name $INSTANCE --query
 oidcIssuerProfile.issuerUrl -o tsv)

az identity federated-credential create \
  --name kubernetes-federated-credential \
  --identity-name "$IDENTITY_NAME" \
  --resource-group "$MANAGED_CLUSTER_RESOURCE_GROUP" \
  --issuer "$SERVICE_ACCOUNT_ISSUER" \
  --subject system:serviceaccount:velero:velero
```

8. Run the following command to create the velero credentials file that contains all the relevant environment variables:

```
cat << EOF  > ./credentials-velero
AZURE_SUBSCRIPTION_ID=$AZURE_SUBSCRIPTION_ID
AZURE_RESOURCE_GROUP=$MANAGED_CLUSTER_RESOURCE_GROUP
AZURE_CLOUD_NAME=AzurePublicCloud
EOF
```

9. Run the following command to install velero:

```
velero install \
    --provider azure \
    --plugins=velero/velero-plugin-for-microsoft-azure:v1.10.1 \
    --service-account-name velero \
    --pod-labels azure.workload.identity/use=true \
    --bucket $BLOB_CONTAINER \
    --secret-file ./credentials-velero \
    --backup-location-config
 useAAD="true",storageAccountURI="https://
$AZURE_STORAGE_ACCOUNT_ID.blob.core.windows.net",resourceGroup=$AZURE_BACKUP_RESOURCE_GROUP,storageAccou
nt=$AZURE_STORAGE_ACCOUNT_ID,subscriptionId=$AZURE_SUBSCRIPTION_ID \
    --snapshot-location-config
 apiTimeout=5m,resourceGroup=$AZURE_BACKUP_RESOURCE_GROUP,subscriptionId=$AZURE_SUBSCRIPTION_ID \
    --wait
```

10. To verify that the velero installation is complete, run the following command and verify that the pods are running.

```
kubectl get pod -n velero
```

Results

You have prepared your cluster to back up the server data.

What to do next

You can back up the server data. See Backing up the DevOps Test Hub data on Azure Kubernetes Service on page 137

Related information

Velero Documentation

Velero plugin for Microsoft Azure documentation

AKS Documentation

Backing up the DevOps Test Hub data on Azure Kubernetes Service

You might want to back up the HCL DevOps Test Hub (Test Hub) data to protect against data corruption or malicious attacks. You might also want to back up the data when you want to upgrade to a new version of Test Hub.

Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall Test Hub.
- Installed Test Hub and all of the Test Hub pods are in `Running` state.
- Communicated to the users that Test Hub might be unavailable for some time until the process is complete.
- Prepared your cluster to store the backed-up data by using Velero. See Preparing the Azure Kubernetes Service cluster to back up the DevOps Test Hub data on page 134.

About this task

If you plan to back up the data immediately after you prepared your Azure Kubernetes Service (AKS) cluster, then you must go to Step 3 on page 138 to start the backup process.

1. Run the following command to log in to AKS interactively:

   ```
   az login
   ```

   Result

   The command-line interface opens a browser and displays the Log-in page of AKS.
2. Sign in with your AKS account credentials.
3. Run the following command to start the backup process of the Test Hub data:

   ```
   BACKUP_NAME=<Your_backup_name>
   ```

   For example: BACKUP_NAME=devops0.backup.20241002

   ```
   NAMESPACE=<Your_Test_Hub_Namespace>
   ```

   For example: NAMESPACE=devops-system

   ```
   velero backup create $BACKUP_NAME --include-namespaces $NAMESPACE --snapshot-move-data
    --include-resources pvc,pv
   ```

   Note: Backups with velero can now be created including targeted namespaces and only persistent volumes:

   Tip: You can run the velero backup describe $BACKUP_NAME command to verify the progress of the backup operation.

Results

You have backed up the data of Test Hub on AKS.

What to do next

You can restore the server data. See Restoring the DevOps Test Hub data on Azure Kubernetes Service on page 138.

Related information

Velero Documentation

Restoring the DevOps Test Hub data on Azure Kubernetes Service

You can restore the HCL DevOps Test Hub (Test Hub) data that is backed up in Test Hub when required.

Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall Test Hub.
- Communicated to the users that Test Hub might be unavailable for some time until the process is complete.
- Backed up the data from Test Hub.

About this task

If you plan to restore the server data immediately after you backed up the data from Test Hub that you installed on Azure Kubernetes Service (AKS), then you must go to Step 3 on page 139 to start the restoration process.

1. Run the following command to log in to AKS interactively:

   ```
   az login
   ```

   Result

   The command-line interface opens a browser and displays the Log-in page of AKS.
2. Sign in with your AKS account credentials.
3. Run the following command to start the restore process of the Test Hub data:

   ```
   RESTORE_NAME=<Your_Restore_Name>
   ```

   For example: RESTORE_NAME=devops0.restore.20241002

   ```
   velero restore create $RESTORE_NAME --from-backup $BACKUP_NAME
   ```

   Tip: You can run the velero restore describe $RESTORE_NAME --details command to verify the progress of the restore operation.

Results

You have restored the data of Test Hub on AKS.

Related information

Velero Documentation

Uninstallation of DevOps Test Hub

When you no longer require HCL DevOps Test Hub (Test Hub), you can uninstall Test Hub. You can uninstall Test Hub from the platform on which you installed Test Hub.

Uninstallation of DevOps Test Hub on Azure Kubernetes Service

When you want to install a new version of HCL DevOps Test Hub (Test Hub) or you want to reinstall Test Hub when an ongoing installation fails, you can uninstall Test Hub and its components from the Azure Kubernetes Service (AKS) cluster.

Before starting the uninstallation, you must have closed Test Hub, any open web browsers, and all other applications that are enabled by Test Hub.

Also, you must have installed Helm to run the Helm commands. For more information on installing Helm, refer to Installing Helm.

To uninstall Test Hub on Azure Kubernetes Service (AKS), refer to the instructions in the readme file.

Uninstallation of DevOps Test Hub on Red Hat OpenShift

When you want to install a new version of HCL DevOps Test Hub (Test Hub) or you want to reinstall Test Hub when an ongoing installation fails, you can uninstall Test Hub and its components from the Red Hat OpenShift platform.

Before starting the uninstallation, you must have closed Test Hub, any open web browsers, and all other applications that are enabled by Test Hub.

To uninstall Test Hub on RHEL or Ubuntu, refer to the instructions in the readme file.

Uninstallation of DevOps Test Hub on Linux

When you want to install a new version of HCL DevOps Test Hub (Test Hub) or you want to reinstall Test Hub when an ongoing installation fails, you can uninstall Test Hub and its components from the Red Hat Enterprise Linux (RHEL) or Ubuntu platform.

Before starting the uninstallation, you must have closed Test Hub, any open web browsers, and all other applications that are enabled by Test Hub.

To uninstall Test Hub on RHEL or Ubuntu, refer to the instructions in the readme file.

Configuration of DevOps Test Hub

You can find information about the tasks that you must perform after you installed HCL DevOps Test Hub (Test Hub) to configure Test Hub.

The following table describes the tasks that you can perform as an administrator after you installed Test Hub:

Require-ment	Tasks	More information
Required	Review the default user management to know how Test Hub manages the users and their authentication.	User administration on page 141
Required	Import Certificate Authority (CA) into a browser to prevent certificate errors when you access the Test Hub UI.	Certificate authority: Importing and extending lists on page 161
Optional	Copy the third-party application Jars to Kubernetes to run API Suites that uses a transport.	Copying third-party application Jars to Kubernetes on page 165

Require-ment	Tasks	More information
Optional	Configure multi-factor authentication (MFA) for log-ging into Test Hub.	
Optional	Change the password seed to provide enhanced se-curity to Test Hub.	Changing the password seed on page 167
Optional	Disable the server extensions that are enabled dur-ing the installation of Test Hub, if you do not want the server extensions to run on the cluster to save server resources.	Disabling server extensions on page 170

User administration

After you install HCL DevOps Test Hub (Test Hub), you must consider how the software manages users and authentication.

You can review the default user management provided by the server and decide what additional controls you might want to add. If you manage users and authentication through an existing LDAP/AD server, you can review how to use that server to manage Test Hub users.

Enabling the Sign up link on the Login page of DevOps Test Hub

You can enable the **Sign up** link when it is not visible on the **Login** page of HCL DevOps Test Hub (Test Hub) by using the Keycloak UI.

Before you begin

You must have completed the following tasks:

- Installed Test Hub.
- Assigned the role of a Keycloak Administrator.
- Copied the URL of the Keycloak UI, username, and password on completion of the installation of server software.

About this task

The **Sign up** link on the **Login** page of Test Hub is not visible when you install Test Hub by using the default values. The **Sign up** link is hidden to provide the enhanced security to Test Hub. You can enable the **Sign up** link on the **Login** page when you want to provide self-registration of users in Test Hub.

You can enable the **Sign up** link on the **Login** page in two ways:

- You can use the keycloak.signup.enabled parameter during the installation of Test Hub.

  For more information about installing Test Hub by using the keycloak.signup.enabled parameter, see the **Related information** section.

- You can use the Keycloak UI after the installation is complete.

  For more information on Keycloak UI, see the **Related information** section.

The following procedure describes steps to be performed to enable the **Sign up** link by using Keycloak UI.

1. Log in to the Keycloak Admin Console.
2. Click **Realm Settings** from the left navigation pane.
3. Click the **Login** tab.
4. Set **User Registration** to **ON**.
5. Click **Save** to apply the changes.

Results

You have enabled the **Sign up** link on the **Login** page of Test Hub. The **Sign up** link is now visible on the **Login** page of Test Hub.

What to do next

You can click the **Sign up** link to create an account in Test Hub.

Related information

Installation of DevOps Test Hub on Azure Kubernetes Service on page 116

Installation of DevOps Test Hub on Ubuntu

Keycloak

Default user administration

You installed HCL DevOps Test Hub (Test Hub) and you want to know about how Test Hub manages user access.

Test Hub uses Keycloak (https://www.keycloak.org/) to manage and authenticate users.

If you manage and authenticate users by using an LDAP and Active Directory server, you can configure Keycloak to connect to that server. For more information, see LDAP user administration on page 144.

Keycloak uses the concept of a realm to manage and authenticate users. When you install Test Hub, a realm called *platform* is created for you in Keycloak. All server users belong to this realm and when they log in to Test Hub, they log into that realm.

Important: You must ensure that you have set the realm as *platform* before you proceed with any configurations in Keycloak for Test Hub.

As an administrator, it is important to consider the following points about the server administration:

- By default, there is no administrator for Test Hub.

  Such an administrator is required for accessing additional functions, which includes claiming ownership of projects and unarchiving them. But you can assign administrative privileges to any user. You must do this by adding the admin role to the user in Keycloak.

- You must sign up a user that you want to be the administrator. You must go to the Login page at `https://<fully-qualified-dns-name>:443` and sign up.

  Note: Do not use that admin user to perform non-administration tasks. Instead, sign up another user.

- After you sign up the user that you want to be the administrator for Test Hub, you must log in to the Keycloak Admin Console at `https://<fully-qualified-dns-name>:443/auth/admin/master/console/#/platform` to make that user the server administrator.

  The default username for the Keycloak administrator is `keycloak`. The password is randomly generated when the software is installed. You can see the password by using the following kubectl command:

  ```
  kubectl get secret -n devops-system {main}-keycloak -o jsonpath="{.data.password}" | base64 --decode;
   echo
  ```

  Note: You must substitute `{main}` with the release name that you used during the installation of Test Hub.

  After you log in to the Keycloak Admin Console, from the **Users** page, you can search and select the user that you want to make an administrator. From the **Groups** tab, you can join the user to the **Admins** group.

  For more information about assigning user roles, see Assigning permissions using roles and groups in the Keycloak documentation.

Now that you are the server administrator, it is important to consider the following points about the default user management and authentication:

- Minimum password length defaults to 8 characters
- Email verification of new users is turned off
- The Forgot Password feature is turned on by default but no instructions are sent to the user to reset their password
- Forgotten user passwords are changed by you if you do not enable Keycloak to send instructions to reset a password

You can review the following sections about changing the default authentication controls.

Email settings

By default, the *platform* realm sets the Forgot Password switch on. However, as an administrator, you must enable Keycloak to send an email to the user with instructions to reset their password. If you want to verify an email, you must also enable Keycloak to send an email to the user to verify their email address.

You must provide SMTP server settings for Keycloak to send an email. After you log in to the Keycloak Admin Console, see Email Settings in the Keycloak documentation.

Then, to set up the email verification, see Forgot Password in the Keycloak documentation.

Password policy

By default, the platform realm has a password policy where the minimum length of a password is 8. As an administrator, you can update password policies in Keycloak.

After you log in to the Keycloak Admin Console, see Password Policies in the Keycloak documentation.

User password

If you did not enable Keycloak to send instructions to a user about how to reset a password, you must use the Keycloak Admin Console to change their password for them.

After you log in to the Keycloak Admin Console, see User Credentials in the Keycloak documentation.

User deletion

When a user is inactive or no longer needs to access the server, you can delete that user.

After you log in to the Keycloak Admin Console, refer to Deleting Users in the Keycloak documentation.

User sessions

The default session timeout that is set in Keycloak is 30 minutes.

As an administrator, you can configure a longer session timeout in Keycloak. Refer to Managing user sessions in the Keycloak documentation.

LDAP user administration

After you installed HCL DevOps Test Hub (Test Hub), you can use a Lightweight Directory Access Protocol (LDAP) and Active Directory (AD) HTTP server to manage users and authentication. Some of the following best practices, can help you to use an LDAP and AD server to manage user access to Test Hub.

Test Hub uses Keycloak (https://www.keycloak.org/) to manage and authenticate users.

Existing user databases hold user credentials. Keycloak federates these existing external user databases through the concept of storage providers. By default, Keycloak supports an LDAP and Active Directory storage provider. By adding a storage provider, you can map LDAP user attributes into Keycloak. You can also configure more mappings.

Before you configure Keycloak to use an existing LDAP/AD provider, you might consider the following best practices:

- Set up your LDAP/AD provider as a read-only repository so that Test Hub cannot change it.
- Add and remove users in LDAP/AD and not the Keycloak local user database.
- Import and synchronize your LDAP/AD users to your Keycloak local database.
- Map a login style name, for example, `user1@server.com`, by using the `UserPrincipalName` attribute in LDAP/AD to a `username` in Keycloak. If you want the full name of the user as your login style, use the `cn` attribute in LDAP/AD.



Note: The LDAP/AD username attribute must match the LDAP/AD provider username attribute (**Username LDAP attribute**) in Keycloak for the LDAP/AD provider to connect with Keycloak.

The following sections use these best practices to guide you to set up Keycloak to connect to your LDAP/AD HTTP server.

LDAP provider selection in Keycloak

You can use the Keycloak Admin Console to add an LDAP/AD provider.

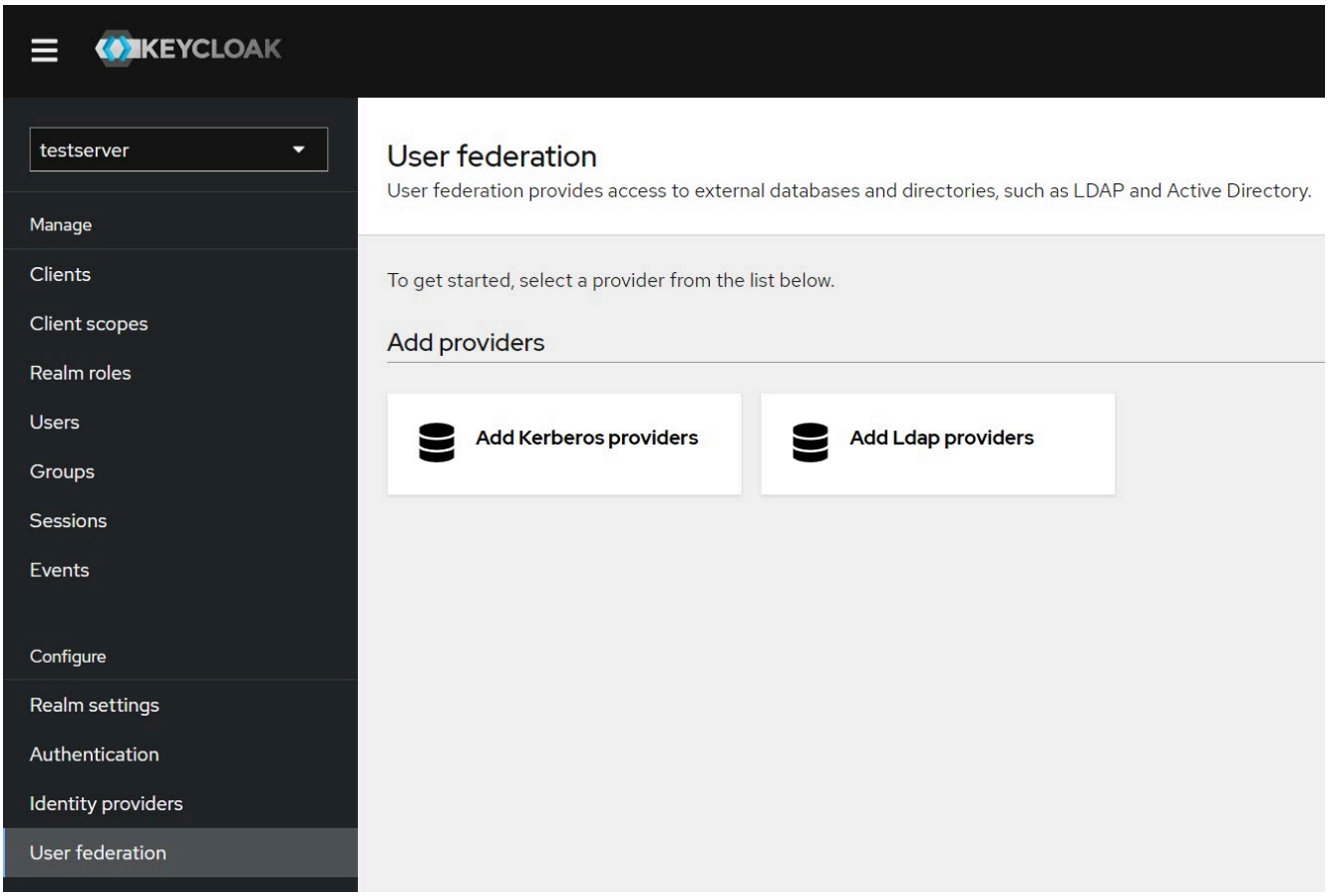
You can log in to the Keycloak Admin Console at `https://<fully-qualified-dns-nam>auth/admin/master/console/#/platform`. The default username for the Keycloak administrator is `keycloak`. The password is randomly generated when you installed the Test Hub software. You can see the password by using the following kubectl command:

```
kubectl get secret -n devops-system {{main}}-keycloak-postgresql -o jsonpath="{.data.password}" | base64
 --decode; echo
```



Note: You must substitute `{main}` with the release name that you used during the installation of Test Hub.

After you log in, you can go to the User Federation page to add your provider.



You must select the **Add Ldap providers** option and complete your LDAP/AD connection parameters.

Required settings for a successful connection to your LDAP/AD provider

In the **Add Ldap provider** page, you must complete the fields or select the required values from the list in the field to ensure a successful HTTP connection to your LDAP/AD provider.

Select or enter details in the fields that are displayed in the following sections on the **Add Ldap provider** page:

Section	Field	Action
General options	Console Display Name	Enter a name for the LDAP server that you can identify when it is displayed in the Keycloak Admin Console. For example, you can enter the name as LDAP.
	Vendor	Select the vendor that is providing the LDAP service from the list of providers. For example, you can select **Active Directory** from the list of vendors.
Connection and authentication settings	Connection URL	Enter the LDAP/AD URL so that Keycloak can connect to your LDAP/AD user database. For example, `ldap://<hostname>.<domain>`.

Section	Field	Action
	Enable StartTLS	Set to *On*, if you want to use a secure connection.
	Test connection	Click to test the connection with the LDAP server.
	Bind type	Retain the default value set as *simple*.
	Bind DN	Enter the user ID of the LDAP/AD user database administrator. Make sure that you test the authentication and confirm that the authentication is successful.
	Bind credentials	Enter the password of the user ID of the LDAP/AD user database administrator.
	Test authentication	Click to test the authentication with the LDAP server database.
LDAP searching and updating	Edit mode	Select the *READ_ONLY* option from the list. This setting means that your LDAP/AD user database is read-only. No user data defined through the mapping of attributes in Keycloak such as the `username` is written back from the Keycloak user database to your LDAP/AD user database. This read-only setting also means that a new user cannot sign up from Test Hub as no user data can be written to your LDAP/AD user database.
	Users DN	Enter the directory where the LDAP users are listed, for example, `cn=Users,dc=Devops,dc=COM`.
	Username LDAP attribute	Retain the default value `cn` for the `username`, which is a first name, last name, or you can use `userPrincipalName`, which is username@domain. You might want a login that matches your company style. For example, you might prefer joetester@mycompany.com instead of Joe Tester. If you do make this change the `username` that you specify now must match the `username` in the Mapper. Keycloak makes this change for you.
	User object classes	Enter the User Object Classes as *User* because LDAP user records are found based on a user object class.
	User LDAP filter	Enter the filter so that a subset of all your LDAP users are imported as users by Keycloak. For example, you can set up a *Devops* user group for your LDAP/AD user database such that only those users are imported to Keycloak. `(memberOf=cn=Testers,dc=Devops,dc=COM)`

Mappers

Keycloak uses mappers to map the user attributes defined in the Keycloak user model such as **username** and **email** to the corresponding user attributes in the LDAP/AD user database. By default, when you saved your settings and created your LDAP/AD provider, the following mappers were created.

The username attribute that you specified in the **Username LDAP attribute** must match the **username** attribute defined in the Keycloak mapper for the LDAP/AD user database to connect with Keycloak.

Because you changed the **Username LDAP attribute** from the default value `cn` to `userPrincipalName`, Keycloak made the same change in the mapper called **username** to match.

User synchronization

You must import all users from your LDAP/AD user database by using the option to **Synchronize all users**. Users are imported based on your saved settings when you set up your LDAP/AD provider.

A successful import is followed by a success message with the number of users imported.

You can view all the LDAP/AD database users that were imported and authenticated from the **Users** page in the Keycloak Admin Console.

Users are listed with Username, Email, Last Name, and First Name. The value of the other attributes is fetched from the LDAP/AD user database by using mappers.

Important: If you get a message of a failed import for an LDAP/AD user, then the failed import might be due to any one of the following reasons:

- The **User logon name** field under the **Account** tab of that LDAP/AD user is blank.
- The value of the **Custom User LDAP Filter** field to import only a subset of all LDAP users does not match with the LDAP set of users.

You can check the log file of the Keycloak pod to understand the reason for the failed import.

You can perform the following steps to view the log file of the Keycloak pod:

1. Run the following command to view the list of all pods:

   ```
   kubectl get pods -n devops-system
   ```

2. Identify the Keycloak pod `ssocloak-0`.
3. Run the following command to view the log of the Keycloak pod:

```
kubectl logs {{main}}-ssocloak-0 -n devops-system
```

Note: Where, {{main}} is the name of the release that is provided during the installation of the server software.

The log file of the Keycloak pod is displayed.

A sample of a log file is as follows:

```
12:49:00,535 ERROR [org.keycloak.storage.ldap.LDAPStorageProviderFactory] (default task-40)
Failed during import user from LDAP: org.keycloak.models.ModelException: User returned from LDAP has
 null username!
Check configuration of your LDAP mappings.
Mapped username LDAP attribute: userPrincipalName, user DN: CN=DefaultAccount,CN=Users,DC=DEVOPS,DC=COM,
attributes from LDAP: {whenChanged=[20190527143130.0Z], whenCreated=[20190527143130.0Z],
 cn=[DefaultAccount],
userAccountControl=[66082], pwdLastSet=[0]}
```

Updating user profile

You can view and update your profile details such as language preferences, username, email, first name, and last name.

About this task

As an account owner, you must ensure that your profile information is accurate and up-to-date. You can change your profile details except for the username.

Note: You can change your account details only if HCL DevOps Test Hub (Test Hub) is not configured to use an LDAP and AD server. See LDAP user administration on page 144.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Account** icon , and then click **Account Settings**.
   Result
   The **General Settings** page of your account is displayed.
3. Click **Update Profile** in the **User Profile** section.
   Result
   The **Personal Info** page is displayed.
4. Update your profile details, and then click **Save**.

Results

You have updated your profile details.

Related information

Default user administration on page 142

Changing account password on page 150

Changing account password

You can regularly change your account password to enhance the security of your DevOps Test Hub account.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Account** icon  , and then click **Account Settings**.
   Result
   The **General Settings** page of your account is displayed.
3. Click **Update Profile** in the **User Profile** section.
   Result
   The **Personal Info** page is displayed.
4. Click **Account Security > Signing in** in the navigation pane.
   Result
   The **Signing in** page is displayed.
5. Click **Update** to change your password.
   Result
   A new web page to re-authenticate is displayed.
6. Complete the authentication by entering your username and password in the **Username** and **Password** fields.
7. Upon successful authentication, type the new password in the **Update Password** dialog.
8. Type your password again to confirm your new password, and then click **Submit**.

Results

You have changed your account password.

Related information

Default user administration on page 142

Updating user profile on page 149

Configuration of multi-factor authentication in Keycloak

As a *Server Administrator*, you can configure multi-factor authentication that is provided by Keycloak so that the authenticity of the users is validated when they sign in or log into Test Hub.

Test Hub uses Keycloak for user administration and user authentication is one of its features.

Keycloak mandates users create their login credentials at the time of signing up with Test Hub. To enhance the security of Test Hub and to prevent unauthorized users from logging in when user credentials are compromised, the *Server Administrator* can use the multi-factor authentication that is provided by Keycloak.

Multi-factor authentication

The multi-factor authentication (MFA) feature that is provided by Keycloak includes the following methods to authenticate the users when they sign in or log into Test Hub:

- Use of a password that is created by the user when the user creates an account in Test Hub is the default layer of authentication.
- A one-time password (OTP) that is generated by any of the OTP generators that the users install on their mobile device and register their mobile device with Keycloak. The OTP method is an optional layer of authentication that needs to be configured and enabled in Keycloak by the *Server Administrator*.

As a *Server Administrator*, you can configure the OTP policy as a MFA method in Keycloak. See Setting up a one-time password policy as MFA in Keycloak on page 151.

After you set up the OTP policy in Keycloak, you must enable or activate the OTP policy so that the MFA UI is displayed for users at the time of logging into Test Hub. See Enabling the one-time password policy as the default MFA action in Keycloak on page 154.

At times, if users report that their credentials are compromised or have lost their mobile device, the *Server Administrator* can reset the OTP credential of the user that is saved in Keycloak. Resetting the OTP credential enables the users to register their mobile devices again to use the OTP authentication. See Resetting the one-time password policy for a user in Keycloak on page 155.

Setting up a one-time password policy as MFA in Keycloak

As a *Server Administrator*, you can set up a one-time password (OTP) authentication policy in Keycloak as part of the multi-factor authentication (MFA) to validate the authenticity of the users when they sign in or log into Test Hub.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Server Administrator* of Test Hub. See Default user administration on page 142.
- Installed Test Hub. See Installation of DevOps Test Hub on page 115.
- Configured the licenses for Test Hub. See HCLSoftware license portal on page 182.

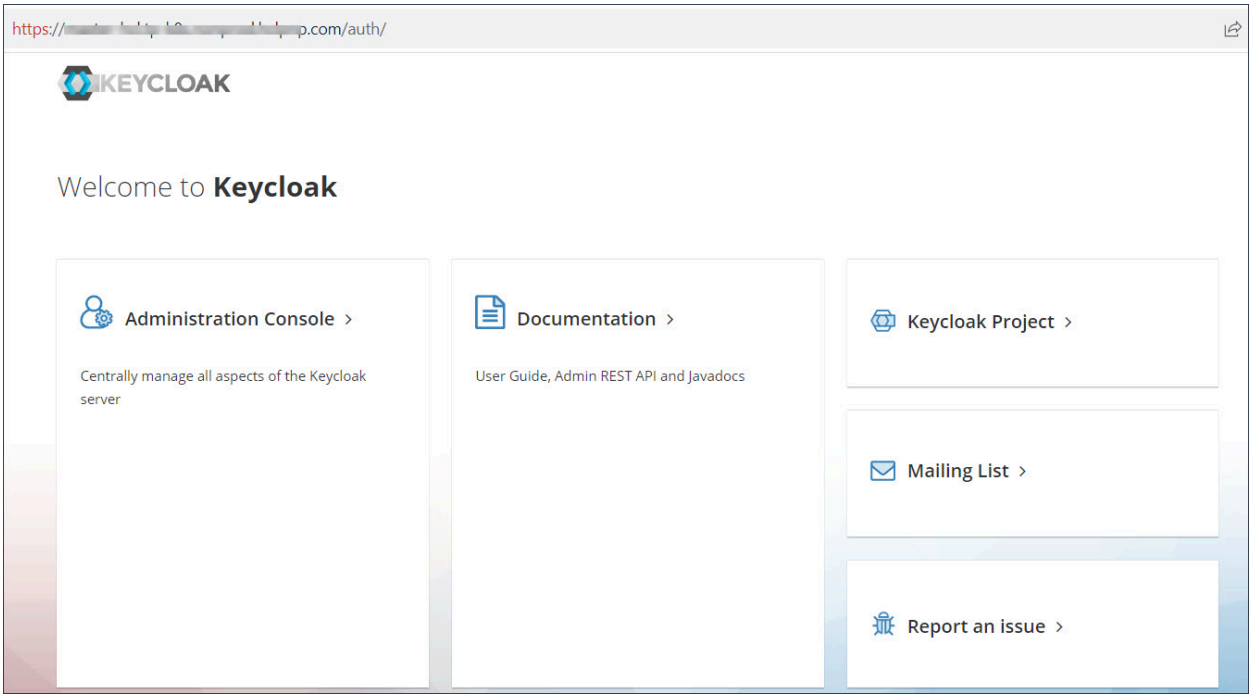
About this task

Keycloak provides an OTP policy that can be enabled and configured in Keycloak for the installed instance of Test Hub. You can refer to the Keycloak documentation about the OTP policies.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open Keycloak by entering `/auth` in the **Address bar** after the URL of your server instance, and then press the **Enter** key.

   For example, if the URL of the server instance is *https://<myserver.mycompany.com>*, add `/auth` after the URL as in *https://<myserver.mycompany.com>/auth*

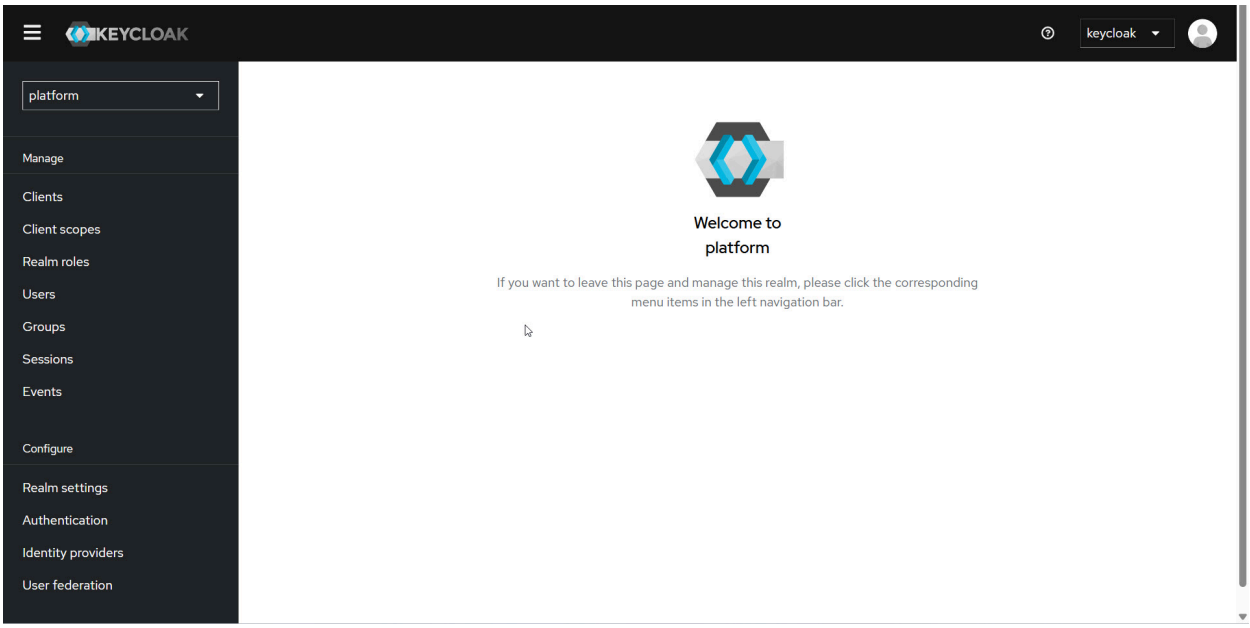
   The Keycloak dashboard is displayed.



3. Click **Administration Console**.
4. Log into the Keycloak administration console by using the default username as *keycloak* and the password generated for you by Keycloak, if prompted.

   The Keycloak console is displayed.
5. Change the realm to *platform*, if it is not already in this realm.

   The *platform* realm dashboard is displayed.

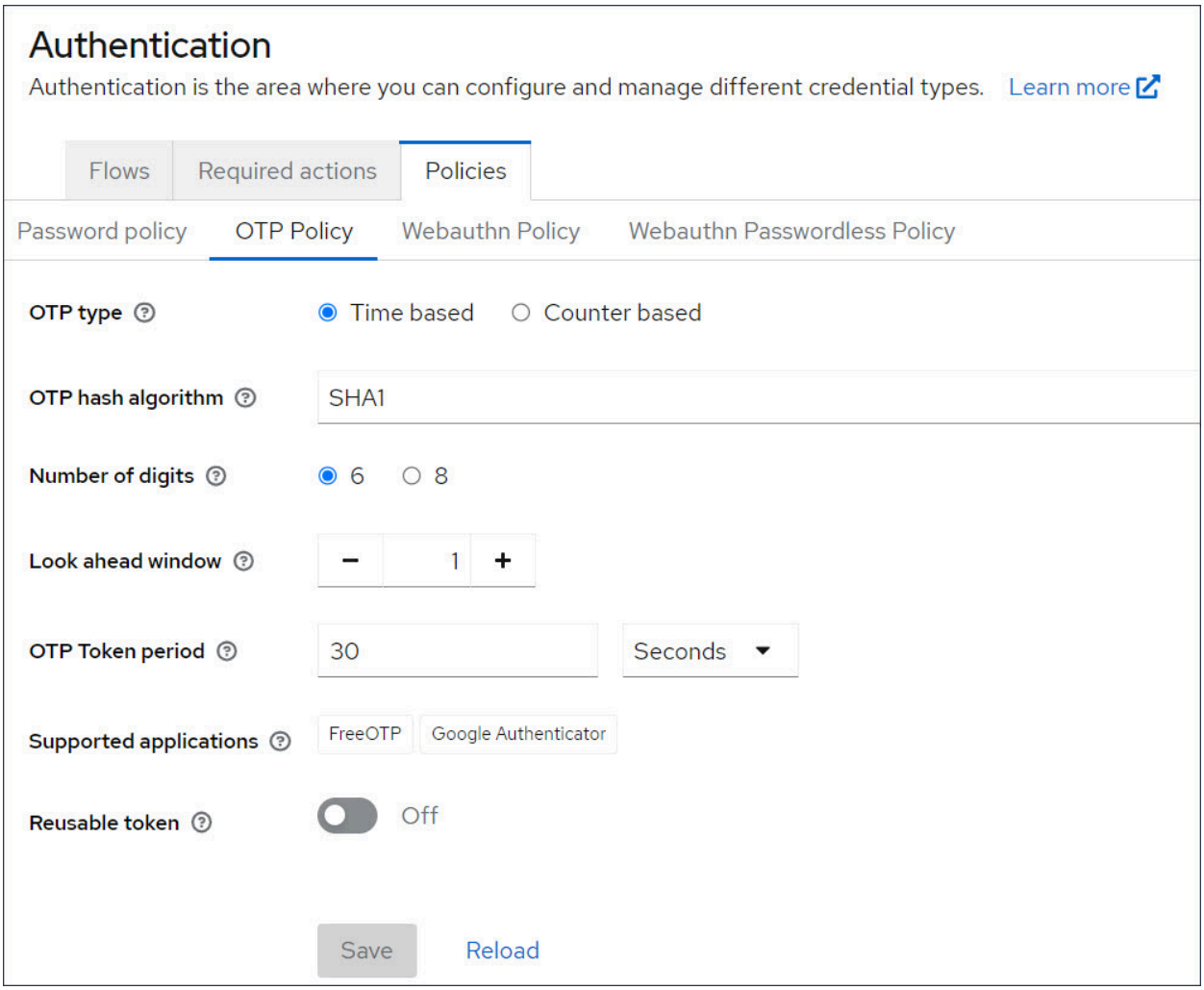


6. Click **Authentication** in the navigation pane.

   The **Authentication** page is displayed.

7. Click the **Policies** tab, and then click the **OTP Policy** tab.

   The OTP Policy configuration settings are displayed.



8. Read the description provided for each of the fields by clicking and decide the values that you want to set for each of the fields, which are different from the default values listed.

   You can save the changed configuration or use the default configuration without making any changes.

Results

You configured the settings or retained the default settings for an OTP policy in Keycloak.

What to do next

You must enable the OTP policy in Keycloak as the default option for authentication. See Enabling the one-time password policy as the default MFA action in Keycloak on page 154.

Enabling the one-time password policy as the default MFA action in Keycloak

After a *Server Administrator* sets up a one-time password (OTP) authentication policy in Keycloak, the *Server Administrator* must enable the policy so that Keycloak can deploy the OTP policy as the default multi-factor authentication (MFA) action.

Before you begin

You must have completed the following tasks:

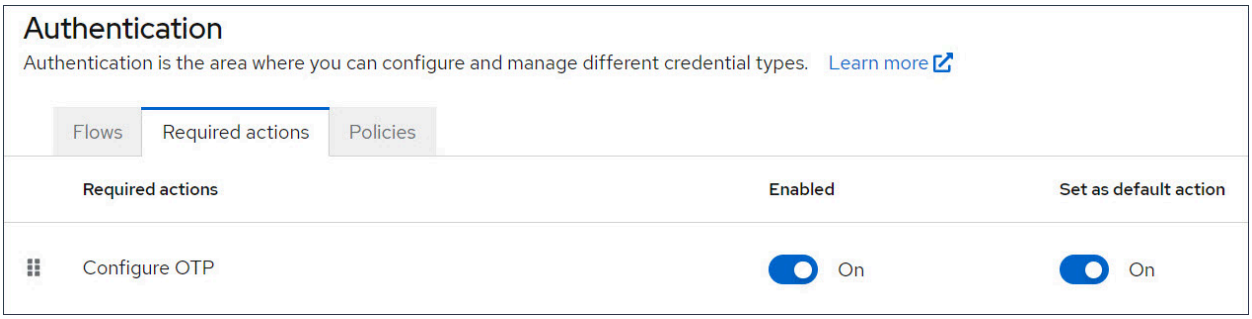
- Ensured that you are assigned a role as a *Server Administrator* of Test Hub. See Default user administration on page 142.
- Configured a one-time password policy for MFA in Keycloak. See Setting up a one-time password policy as MFA in Keycloak on page 151.
- Logged into the *platform* realm in Keycloak.

About this task

The usage of an OTP policy for authentication is enabled but not set as the default action in Keycloak. As a *Server Administrator*, you must set the OTP policy as the default MFA action to authenticate users accessing the server instance of Test Hub.

1. Click **Authentication** in the navigation pane.

   The **Authentication** page is displayed.
2. Click the **Required actions** tab.
3. Click the toggle under **Set as default action** column for the **Configure OTP** item to change from *Off* to *On*.



Results

You set the OTP policy as the default MFA action.

Resetting the one-time password policy for a user in Keycloak

If multi-factor authentication is enabled for Test Hub, and a user is not able to log in because the mobile device that generates an OTP is lost, the user must request the *Server Administrator* to reset their credentials. The *Server Administrator* must reset the credentials of the user so that the user can register the mobile device again.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Server Administrator* of Test Hub. See Default user administration on page 142.
- Configured a one-time password policy as MFA in Keycloak. See Setting up a one-time password policy as MFA in Keycloak on page 151.
- Enabled a one-time password policy as the default MFA action in Keycloak. See Enabling the one-time password policy as the default MFA action in Keycloak on page 154.
- Received requests from users to enable registration of their new mobile devices for authentication.
- Logged into the *platform* realm in Keycloak.

About this task

When as a *Server Administrator*, you configure and enable a one-time password (OTP) authentication policy in Keycloak, all users who attempt to log into Test Hub must provide their password and an OTP. Users must install the OTP generators on their mobile devices and register their mobile devices with Keycloak.

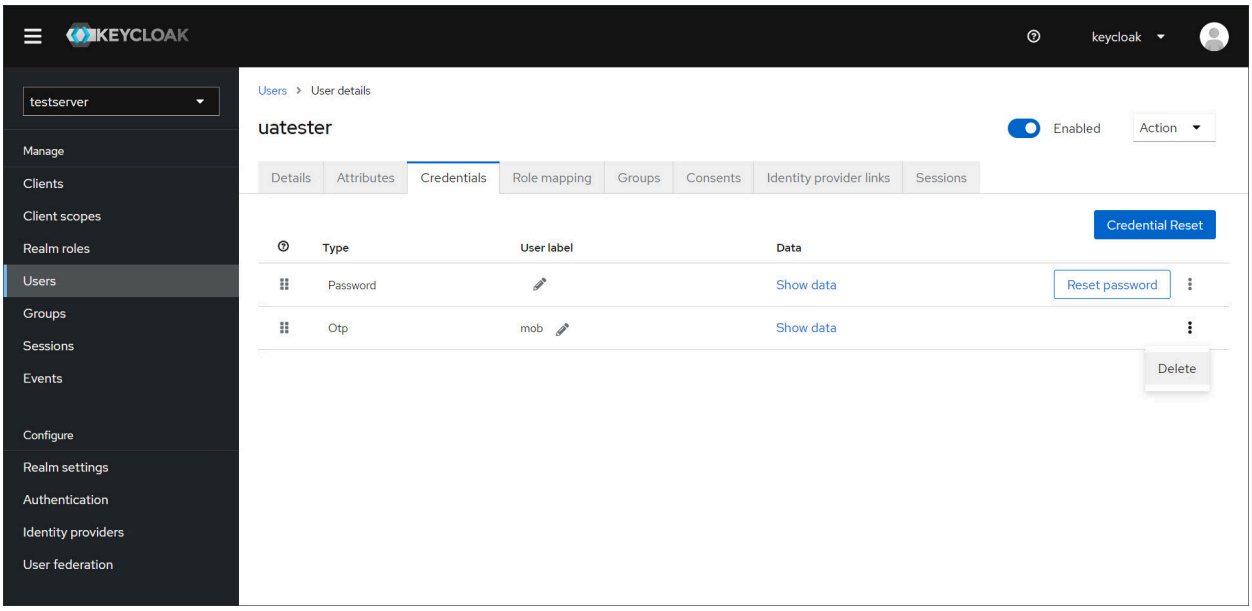
If a user loses the mobile device that has the OTP generators installed, then that user cannot provide an OTP to log into Test Hub.

You must then delete the stored credentials of the user for the OTP authentication in Keycloak and re-enable the OTP action for the user. The user must register the new device again for OTP authentication.

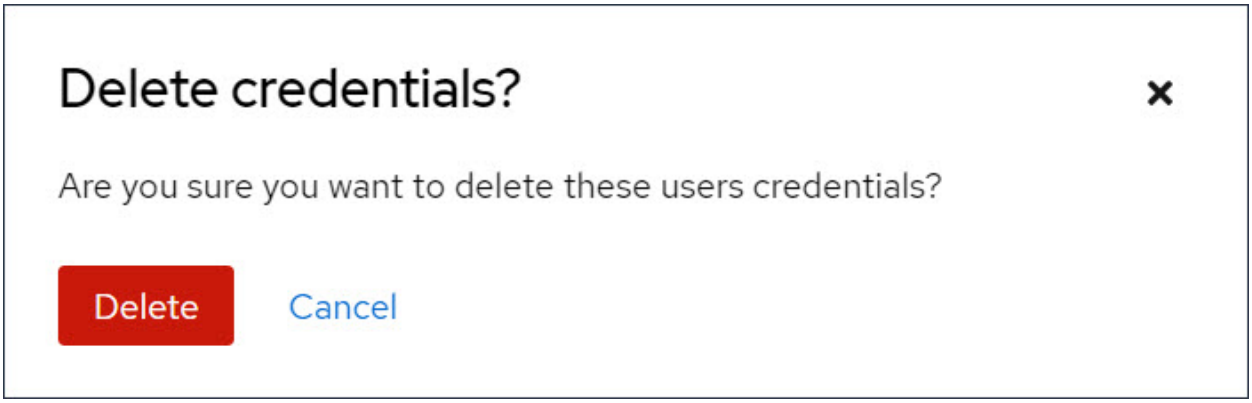
1. Click **Users** in the navigation pane.

   The **Users** page is displayed.
2. Search for the user by entering the username of the user in the **Search** field.
3. Click the username that is displayed as a result of the search.

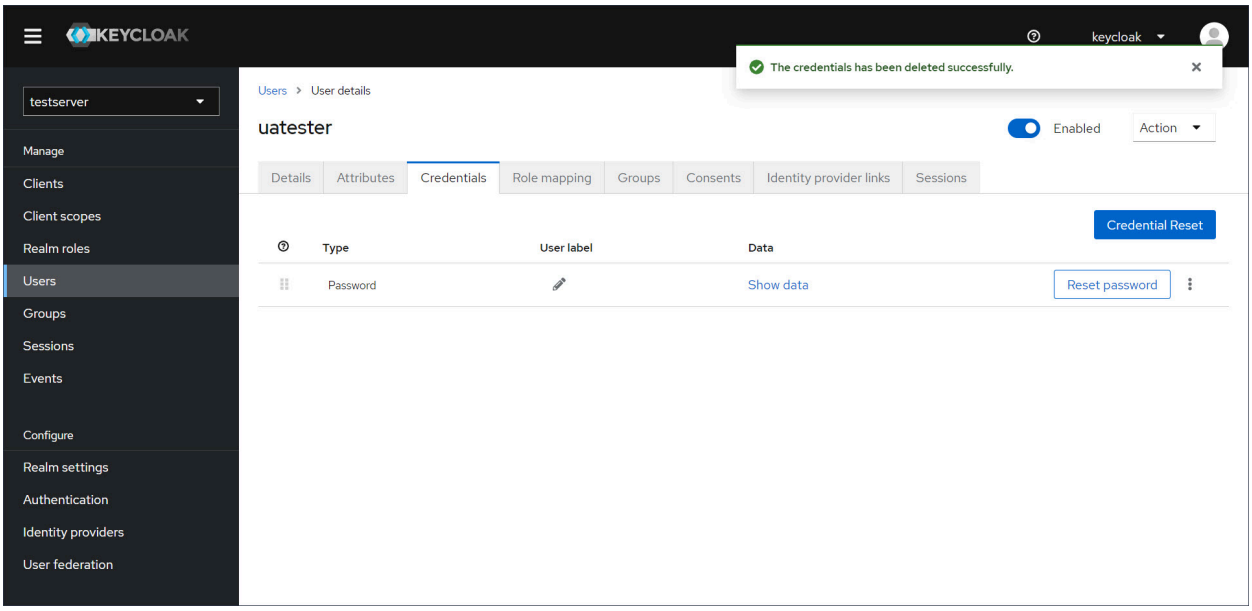
   The **User details** page is displayed.
4. Click the **Credentials** tab.
5. Click the menu icon  in the row of **Otp**, and then click **Delete**.



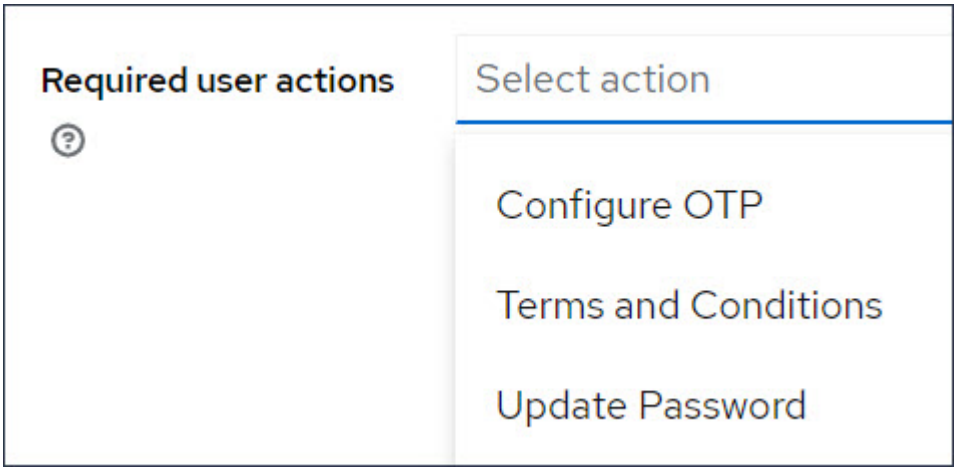
6. Click **Delete** in the **Delete credentials?** dialog.



The saved details of the OTP credential are removed for the user.

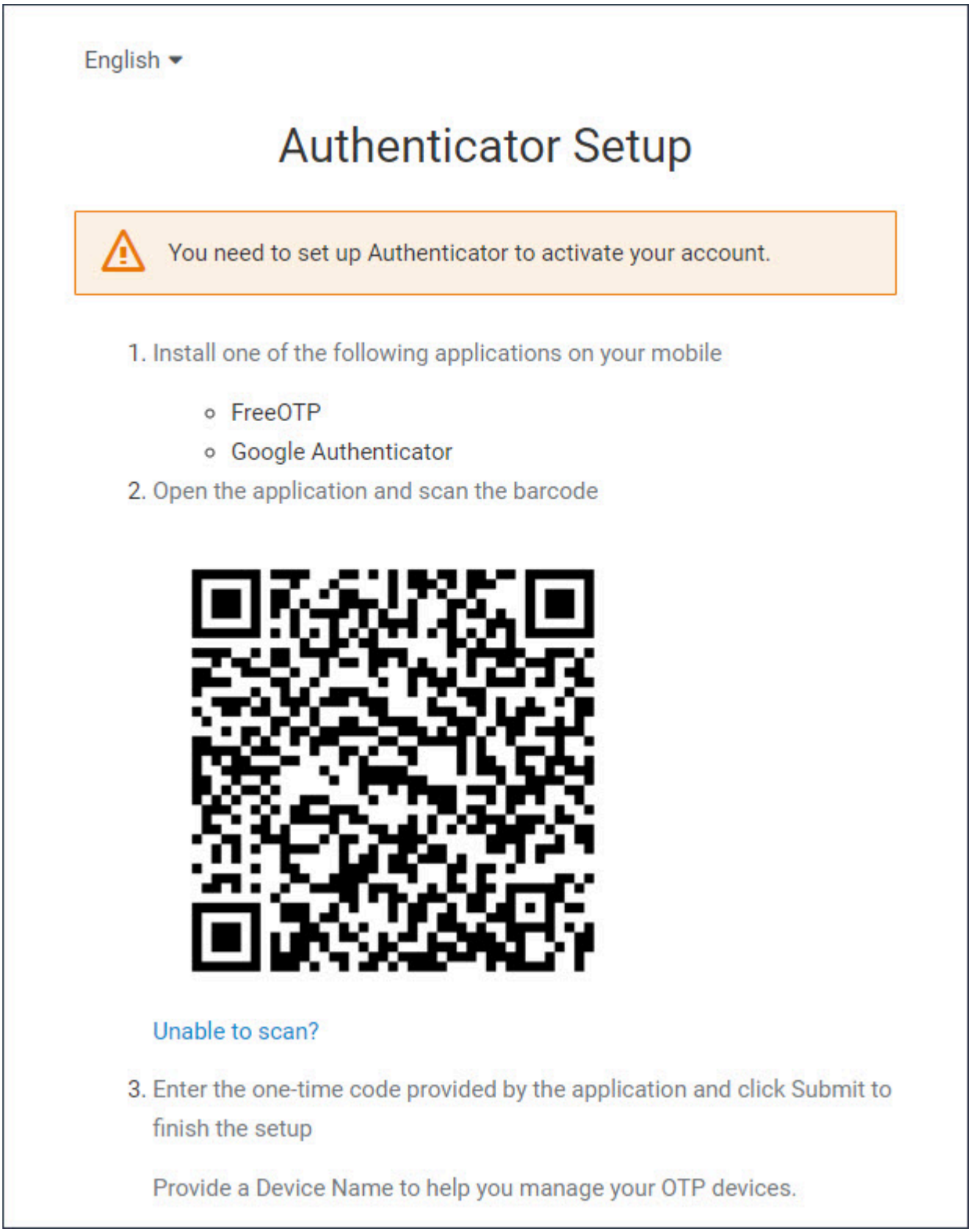


7. Click the **Details** tab on the **User** page.
8. Select the **Configure OTP** option from the **Required user actions** list.



9. Click **Save**.

When the user attempts to sign in or log into Test Hub, the **Authenticator Setup** dialog is displayed, and the user can register their new device for the OTP authentication with Keycloak.



Results

You reset the OTP policy for a user.

Management of third-party licenses

When you want to use built-in web browsers such as Google Chrome or Microsoft Edge, you must first accept the third-party license agreement of these browsers. You can manage third-party license agreements on the **General Settings** page in HCL DevOps Test Hub (Test Hub).

You can find the following information:

- Accepting third-party licenses on page 160
- Removing acceptance of third-party licenses on page 160

Accepting third-party licenses

You can accept any third-party license agreements on the **General Settings** page in HCL DevOps Test Hub (Test Hub).

About this task

Test Hub prompts you to accept the third-party license agreements when you sign in or log in.. If you do not accept the third-party license agreements, Test Hub continues to prompt whenever you log in.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Account** icon  , and then click **Account Settings**.
   Result
   The **General Settings** page of your account is displayed.
3. Click the **Terms of Service** links in the **Third-Party Agreements** section to review the license agreements and terms of use for Google Chrome and Microsoft Edge.
4. Perform any of the following actions:
   - Select the checkbox for the browser that you want to use to accept the license agreement.
   - Select both **Google Chrome** and **Microsoft Edge** checkboxes, if you want to use both the browsers.
5. Click **Save**.

Results

You accepted the license agreements for either Google Chrome or Microsoft Edge browser or both the browsers.

Removing acceptance of third-party licenses

You can remove the acceptance of third-party licenses when it is no longer required.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.

2. Click the **User** icon  , and then click **Account Settings**.

   Result

   The **General Settings** page of your account is displayed.

   The **Third-Party Agreements** section displays checkboxes for **Google Chrome** and **Microsoft Edge** options.

3. Perform any of the following actions:
   - Clear the checkbox for the browser that you do not want to use.
   - Clear both the **Google Chrome** and **Microsoft Edge** checkboxes, if you do not want to use both the browsers.
4. Click **Save**.

Results

You removed your acceptance of the license agreements for either Google Chrome or Microsoft Edge browser or both the browsers.

Certificate authority: Importing and extending lists

When you want browsers and other applications to access HCL DevOps Test Hub (Test Hub), you must enable browsers to trust the certificate authority (CA) of Test Hub. For some applications that integrate with Test Hub, you must also extend the CA lists that are trusted by the applications.

Before you import CA and extend the CA list, you must save the CA assigned to Test Hub. See Saving the certificate authority on page 161.

You can find the following tasks to import the certificate authority in the browser that you want to use:

- Importing the certificate authority into the Google Chrome browser on page 162
- Importing the certificate authority into the Microsoft Edge browser on page 163
- Importing the certificate authority into the Mozilla Firefox browser on page 163

You can find the following tasks when you want to extend the CA lists:

- Extending the trusted certificate authority list on Linux on page 164
- Extending the trusted certificate authority list on Windows on page 164

Saving the certificate authority

You can save the certificate authority (CA) when you want to import the CA or extend the CA lists that are trusted by the applications.

Before you begin

You must have completed the following tasks:

- Installed HCL DevOps Test Hub (Test Hub) on a Kubernetes or OpenShift environment.
- Ensured that you have permissions to view the secrets that are created during installation of Test Hub on a Kubernetes or OpenShift environment.

1. Access the Kubernetes or OpenShift cluster where you installed Test Hub.
2. Run the following command to save the CA in `ca-master.crt` file.

```
kubectl get secret -n devops-system ingress -o "jsonpath={.data['ca\.crt']}" | base64 --decode >
 ca-master.crt
```

Remember: `devops-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of the `devops-system`.

Results

You have successfully saved the CA.

What to do next

You can perform the following tasks:

- Import the CA into the Google Chrome browser. See Importing the certificate authority into the Google Chrome browser on page 162.
- Import the CA into the Microsoft Edge browser. See Importing the certificate authority into the Microsoft Edge browser on page 163.
- Import the CA into the Mozilla Firefox browser. See Importing the certificate authority into the Mozilla Firefox browser on page 163.
- Extend the trusted CA list on Windows. See Extending the trusted certificate authority list on Windows on page 164.
- Extend the trusted CA list on Linux. See Extending the trusted certificate authority list on Linux on page 164.

Importing the certificate authority into the Google Chrome browser

You must import the certificate authority (CA) into the Google Chrome browser to prevent certificate errors when accessing the HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- You must have saved the CA assigned to Test Hub. See Saving the certificate authority on page 161.
- Installed the browser that you want to use to access Test Hub.

1. Open the Google Chrome browser.
2. Navigate to **Customize and control Google Chrome > Settings**.
3. Click **Security** from the **Privacy and security** pane.
4. Click **Manage certificates**, and then **Import**.
5. Click **Next** in the **Certificate Import Wizard** window.

6. Click **Browse** and select the CA that you want to import, and then click **Next**.
7. Select the **Place all certificates in the following store** option to store the CA securely.
8. Click **Browse**, and then select **Trust Root Certification Authorities** as certificate store.
9. Click **OK**, and then **Finish**.

Results

You have imported the CA into the Google Chrome browser.

Importing the certificate authority into the Microsoft Edge browser

You must import the certificate authority (CA) into the Microsoft Edge browser to prevent certificate errors when accessing the HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- You must have saved the CA assigned to Test Hub. See Saving the certificate authority on page 161.
- Installed the browser that you want to use to access Test Hub.

1. Type `certmgr.msc` in the **Start** menu **Search** filed, and then press **Enter**.
2. Expand **Trusted Root Certification Authorities**, and then select **Certificate**.
3. Right-click on the empty space, and then select **All tasks > Import**.
4. Click **Next** in the **Certificate Import Wizard** window.
5. Click **Browse** and select the CA that you want to import, and then click **Next**.
6. Select the **Place all certificates in the following store** option to store the CA securely.
7. Click **Browse**, and then select **Trust Root Certification Authorities** as certificate store.
8. Click **Next**, and then **Finish** to import the certificate.

Results

You have imported the CA into the Microsoft Edge browser.

Importing the certificate authority into the Mozilla Firefox browser

You must import the certificate authority (CA) into the Mozilla Firefox browser to prevent certificate errors when accessing the HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- You must have saved the CA assigned to Test Hub. See Saving the certificate authority on page 161.
- Installed the browser that you want to use to access Test Hub.

1. Open the Mozilla Firefox browser.
2. Enter `“about:preferences#privacy”` in the address bar, and then press **Enter**.

3. Locate and click the **View Certificates** option.
4. Select **Authorities** tab, and then click **Import**.
5. Select the CA that you want to import, and then click **Open**.
6. Select **Trust this CA to identify websites** in the **Downloading Certificate** window.
7. Click **OK** to import the CA into the browser, and then close the window.
8. Restart Firefox.

Results

You have imported the CA into the Mozilla Firefox browser.

Extending the trusted certificate authority list on Linux

HCL DevOps Test Hub (Test Hub) trusts the certificates signed by a recognized certificate authority (CA). When you have any additional CAs, you must extend the trusted certificates list by using an environment variable on Linux.

Before you begin

You must have saved the CA assigned to Test Hub. See Saving the certificate authority on page 161.

1. Open the terminal.
2. Run the following command to create an environment variable on the Linux operating system:

   ```
   export NODE_EXTRA_CA_CERTS= {path of the downloaded CA with the file extension}
   ```

   For example,

   ```
   export NODE_EXTRA_CA_CERTS=/opt/mycert/ca-master.crt
   ```

Results

You have extended the trusted CA certificates list on Linux.

Extending the trusted certificate authority list on Windows

HCL DevOps Test Hub (Test Hub) trusts the certificates signed by a recognized certificate authority (CA). When you have any additional CAs, you must extend the trusted certificates list by using an environment variable on Windows.

Before you begin

You must have saved the CA assigned to Test Hub. See Saving the certificate authority on page 161.

1. Right-click the Windows icon  from the Windows taskbar, and then click **System**.
2. Click **Advanced system settings** from the **Related setting** section.
3. Click the **Advanced** tab, and then click **Environment Variables**.
4. Click **New** from the **System variables** section.
5. Enter `NODE_EXTRA_CA_CERTS` in the **Variable name** field.
6. Enter the path of the downloaded CA with the file extension in the **Variable value** field.

For example, if you stored the `ca-master.crt` CA in the `C:\Users\Desktop\cert` location, then enter the **Variable value** field as `C:\Users\Desktop\cert\ca-master.crt`

7. Click **OK**.

Results

You have extended the trusted CA certificates list on Windows.

Copying third-party application Jars to Kubernetes

You can run API Suites in a project on HCL DevOps Test Hub (Test Hub). If the API Suite uses a transport and the transport requires third-party application `Jar` files for a successful run, you must ensure that the third-party application `Jar` files are available at the test run time. To achieve this, you must copy the third-party application `Jar` files to the computer where Test Hub is installed on Kubernetes.

Before you begin

You must have server administrator or cluster administrator privileges.

You must have completed the following tasks:

- Identified the third-party application `Jar` files that are required and copied the files. See Test run considerations for API Suites or API tests on page 529.
- Copied the third-party application `Jar` files from HCL DevOps Test Integrations and APIs (Test Integrations and APIs) to the directory or folder in Kubernetes from where you can run the kubectl commands.

About this task

You can copy the third-party application `Jar` files to the folder that is specific for the application under the `/data/<application_name>` folder. You need not extract the files.

You can perform this task any time after you have installed Test Hub and you plan to run an API Suite that meets any of the following conditions:

- The API Suite uses transports and the transports require third-party application `Jar` files to be available at the test run time.
- The API Suite in the project has a results database configured.

You can copy the JDK version `Jar` files that you have used to create the JUnit tests.

You can get help on the kubectl commands by running the command: kubectl cp --help from the `/kube` directory. For more information, refer to the kubectl documentation.

1. Use the following table to find the name of the folder that corresponds to the specific third-party application for the transport used in the API Suite.

   Table 4. Name of the folder for the application

Application	Name of the folder to use
Camel	Camel
CentraSite	CentraSite
CICS	CICS
Coherence	Coherence
Database	JDBC
IMS	IMS
Integra	Integra
JMS	JMS
SAP RFC	SAP
Software AG Universal Messaging	SoftwareAGUM
TIBCO EMS	TIBCO
TIBCO Rendezvous	
TIBCO SmartSockets	
WebSphere Application Server Service Integration Bus (SiBus)	WAS
WebLogic	WebLogicJMX
Software AG webMethods	webMethods

2. Open the command prompt from the `/kube` directory and copy the `Jar` files to the folder that corresponds to the application by using the following command:

   ```
   kubectl cp <compressed_files> devops-system/{{main}}-userlibs-0:/data/<application_name>/
   ```

   For example, if you are copying the Database `Jar` files, then the name of the folder is *JDBC* and the command to use is:

   ```
   kubectl cp mysql-connector-java.jar devops-system/{{main}}-userlibs-0:/data/JDBC/
   ```

   Note: You must substitute `{main}` with the release name that you used during the installation of Test Hub.

3. Verify whether the `Jar` files are copied by running the following command:

```
kubectl exec {{main}} -userlibs-0 -n devops-system -- ls /data/<folder_created>
```

For example, if you copied the Database `Jar` files, `mysql-connector-java.jar` to the folder *JDBC*, then the command to verify is:

```
kubectl exec {{main}} -userlibs-0 -n devops-system -- ls /data/JDBC
```

The following information is displayed for the `JDBC` folder:

```
mysql-connector-java.jar
```

Note: You must substitute `{main}` with the release name that you used during the installation of Test Hub.

Results

You have successfully copied the third-party application `Jar` files to the folder in Kubernetes.

What to do next

You can configure the API Suite run. See Configuring an API Suite or an API test run on page 652.

Changing the password seed

As an administrator, you can change the password seed that is used when you install HCL DevOps Test Hub (Test Hub) to provide enhanced security to Test Hub.

Before you begin

You must have completed the following tasks:

- Installed Test Hub. See Installation of DevOps Test Hub on page 115.
- Installed the JSON command-line tool, jq, and ensured that the jq is in your environment PATH. For information about jq, refer to jq documentation.
- Installed the Curl command line tool. For more information refer to curl documentation.

About this task

When you install Test Hub, you supply a password seed when you run the `helm install` command. This password seed is used to generate several Kubernetes secrets. Kubernetes Secrets can contain the following information:

- The authentication credentials for micro-services.
- An encryption key for the user-created secrets.

When you change the password seed for Test Hub, you must consider the following scenarios:

- Test Hub cannot communicate until you reconcile the passwords which are in Kubernetes Secrets by using the old and a new password seed.
- Users cannot read secret collections or other secrets that they have created in Test Hub until you re-encrypt them using a new password seed.

Important: You must provide an offline token and old password seed that you used during the installation of Test Hub to re-encrypt user secrets.

1. Run the following command to change the password seed for Test Hub:

```
helm upgrade {{main}} ./hcl-onetest-server -n devops-system \
--reuse-values \
--set global.hclOneTestPasswordAutoGenSeed= {my-new-super-secret}
```

Notes: You must substitute the value of the following variables with the actual value in the command:

   - `{{main}}` with the release name that you used during the installation of Test Hub.
   - `{my-new-super-secret}` with a new value of your choice as the password seed.
   - You must run the following `helm upgrade` command from the same directory where the `helm install` command was run during the installation of Test Hub. Because the upgrade is dependent on the helm charts and .yaml file values used during the run time of the `helm install` command.

2. Run the following script to generate new server secrets from the updated password seed and to save them to the persistent storage:

```
./hcl-onetest-server/files/reconcile-secrets.sh -n devops-system {{main}}
```

3. Run the following command to restart all the pods:

```
kubectl delete pods -n devops-system \
  -lapp.kubernetes.io/instance={{main}} \
  -lapp.kubernetes.io/managed-by=Helm
```

4. Run the following commands to re-encrypt the user-created secrets by providing the old password seed:

```
export ACCESS_TOKEN=$(curl -k -X POST {SERVER_URL}/rest/tokens/ \
-H "Content-Type: application/x-www-form-urlencoded" \
-H "accept: application/json" \
-d "refresh_token={OFFLINE_TOKEN}" | jq -r '.access_token')

curl -k -X POST {SERVER_URL}/rest/secrets/re-encrypt/ \
     -H "Authorization: Bearer $ACCESS_TOKEN" \
     -H "Content-Type: application/json" \
     -d "{\"type\":\"helm\",\"password_auto_gen_seed\":\"{OLD_SEED}\"}"
```

Note: You must substitute the value of the following variables with the actual value in the following commands:

- `{SERVER_URL}` with the URL of your Test Hub UI.
- `{OFFLINE_TOKEN}` with the offline token that belongs to a user with the administrator role.
- `{OLD_SEED}` with the previous password seed that you used during the installation of Test Hub.

5. Run the following command to display the log file of the gateway pod:

```
kubectl logs {{main}} -gateway-abcdefghij-abcde -n devops-system
```

Note: You must substitute the value of the following variables with the actual value in the command:

- `{{main}}` with the release name that you used during the installation of Test Hub.
- `abcdefghij-abcde` with an identifier that is assigned to the name of the gateway pod.

  You can run the `kubectl get pods -n devops-system` command to obtain the identifier that is assigned to the gateway pod.

Result

The following message is displayed when re-encryption is completed:

```
reEncrypt complete. StringyReEncryptor [total=100, fixed=100, broken=0, noop=0]
```

Results

You have successfully changed the password seed for Test Hub.

Server extensions

You can run specific tests in HCL DevOps Test Hub (Test Hub) by enabling the server extensions.

The server extensions are enabled at the time of installation of Test Hub. When the server extensions are enabled, you can perform the following tasks on Test Hub:

- Add repositories that contain the test assets for the JMeter tests, JUnit tests or Postman collections.
- Configure a test run for the following test types:
  - **JMeter**
  - **JUnit**
  - **Postman**

To learn more about enabling and disabling the server extensions, refer to the following topics:

Disabling server extensions

When you do not want the server extensions in HCL DevOps Test Hub (Test Hub) to run on the cluster to save server resources, you can disable server extensions.

Before you begin

You must ensure that the server extensions are enabled at the time of server installation by running the following command:

```
kubectl get pods -n devops-system
```

The status of the server extensions that are enabled is displayed as `Running`.

About this task

You must substitute `{{main}}` with the release name that you provided during the installation of Test Hub in the following procedure.

The `devops-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of `devops-system` in all the instances in this procedure.

1. Log in to the platform that you used to install Test Hub.

   For example, you can log in to the Ubuntu server using an SSH session if you installed Test Hub on Ubuntu.

2. Run the following commands that depend on the server extension you want to disable:

To disable the server extension	Run the command

JMeter	`helm upgrade {{main}} -n devops-system ./hcl-onetest-server \`
	`--reuse-values \`
	`--set global.hclOneTestJMeterEnabled=false`

JUnit	`helm upgrade {{main}} -n devops-system ./hcl-onetest-server \`
	`--reuse-values \`
	`--set global.hclOneTestJUnitEnabled=false`

Postman	`helm upgrade {{main}} -n devops-system ./hcl-onetest-server \`
	`--reuse-values \`
	`--set global.hclOneTestPostmanEnabled=false`

All server extensions	`helm upgrade {{main}} -n devops-system ./hcl-onetest-server \`
	`--reuse-values \`
	`--set global.hclOneTestPostmanEnabled=false \`
	`--set global.hclOneTestJMeterEnabled=false \`
	`--set global.hclOneTestJUnitEnabled=false`

3. Run the following command to verify the status of the extensions:

```
kubectl get pods -n devops-system
```

Note: The pod for the server extension that you disabled is not displayed.

Results

You have disabled the required server extensions on Test Hub.

Enabling server extensions

If you find that the server extension is disabled in HCL DevOps Test Hub (Test Hub) and you want to run the tests supported by the server extension, then you must enable them.

Before you begin

You must ensure that the server extensions are enabled at the time of server installation by running the following command:

```
kubectl get pods -n devops-system
```

The status of the server extensions that are enabled is displayed as `Running`. If the server extension you want to use is not displayed as running, then you must enable the server extension.

About this task

You must substitute `{{main}}` with the release name that you provided during the installation of Test Hub in the following procedure.

The `devops-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of `devops-system` in all the instances in this procedure.

1. Log in to the platform that you used to install Test Hub.

   For example, you can log in to the Ubuntu server using an SSH session if you installed Test Hub on Ubuntu.

2. Run the following commands that depend on the server extension you want to enable:

To enable the server extension	Run the command

JMeter	`helm upgrade {{main}} -n devops-system ./hcl-onetest-server \`
	`--reuse-values \`
	`--set global.hclOneTestJMeterEnabled=true`

JUnit	`helm upgrade {{main}} -n devops-system ./hcl-onetest-server \`
	`--reuse-values \`
	`--set global.hclOneTestJUnitEnabled=true`

Postman	`helm upgrade {{main}} -n devops-system ./hcl-onetest-server \`
	`--reuse-values \`
	`--set global.hclOneTestPostmanEnabled=true`

To enable the server extension	Run the command

All server extensions	`helm upgrade {{main}} -n devops-system ./hcl-onetest-server \`
	`--reuse-values \`
	`--set global.hclOneTestPostmanEnabled=true \`
	`--set global.hclOneTestJMeterEnabled=true \`
	`--set global.hclOneTestJUnitEnabled=true`

3. Run the following command to verify a specific extension or all the extensions are enabled:

   ```
   kubectl get pods -n devops-system
   ```

   Result

   The status of the specific extension or all the extensions in the pod is displayed as `Running`.

Results

You have enabled the required server extensions on Test Hub.

Related information

Configuring a run of a JMeter test on page 867

Configuring a run of a JUnit test on page 876

Configuring a run of a Selenium test on page 909

Configuring a run of a Postman test on page 886

Team space administration

You can find information about team spaces including the initial team space on HCL DevOps Test Hub (Test Hub) along with the tasks that you can perform to administer a team space.

You can find the following information about team spaces:

- Team space overview on page 172
- Creating a team space on page 175
- Tasks in a team space on page 177
- Tasks in projects in a team space on page 180
- Management of licenses in DevOps Test Hub on page 182
- Installation of HCLSoftware Local License Server

Team space overview

Team spaces are logical partitions of HCL DevOps Test Hub (Test Hub). Team space is an exclusive, separate, and secure space on Test Hub that enables you to share a single installation among multiple teams.

You can find the following information about team spaces:

- Initial team space on page 173
- Team spaces on page 173
- Migration from earlier releases of Test Hub on page 175

Initial team space

After you install Test Hub and log in for the first time, you enter the **Initial team Space**.

The initial team space is the default logical partition that is created on an instance of Test Hub.

As a server administrator, you must consider the following points:

- After you install Test Hub and log in for the first time, the **Projects** page of the initial team space is displayed.
- You must configure the license server for the initial team space.
- Add or remove users in the team space and assign them different roles in a team space.
- You can assign the role of a *Team Space Owner* to any other member in the initial team space.
- Change the type of the initial team space from the default type of *permissive*.

  The following table describes the type that you can assign to a team space:

Type	Description
Permissive	Specifies that any licensed user of Test Hub can access a team space and get assigned the default roles of a *Project Owner* and *Member* in the team space.
Restrictive	Specifies that all licensed users of Test Hub must request for access to a team space and must be provided access and assigned roles by the *Team Space Owner*. The **Restrictive** type of team space can be either **Public** or **Private**. A **Public** team space is visible in the team spaces dashboard, while a **Private** team space is not visible in the team spaces dashboard.

- You can evaluate the requirement and create team spaces other than the initial team space.

As a licensed user, you must consider the following points:

- When you log in for the first time immediately after the installation of Test Hub, the **Projects** page of the initial team space is displayed.
- After you log in to the initial team space, you are assigned the default roles of *Project Creator* and *Member* of the initial team space.
- On subsequent logins, the **My Projects** page in the initial team space is displayed as the default page.

Team spaces

You can either work with the initial team space or you can create multiple team spaces on Test Hub. The team space facilitates users of the same team space to work together.

Apart from being a container for projects, a team space facilitates the registration of Resource Monitoring agents, designing the system model, and viewing the registered agents, or intercepts.

After you create a team space as a **server administrator**, you must consider the following actions:

Note: You are assigned the default role of a *Team Space Owner* for team spaces that you create.

- Add users in the team space and assign them different roles in a team space.
- Assign the role of a *Team Space Owner* to other team space members, if you want other members to own the team space.

  Note: As a *Team Space Owner*, you can remove any member of your team space from the assigned role. The member who is removed from a team space, can send a request and become a member of that team space after the request is accepted.

- If the *Team Space Owner* is no longer a member of a team space, then you as the server administrator can access that team space with the default role of a *Team Space Owner*. You can then assign the role of a *Team Space Owner* to any other member.
- Delete the initial team space or any other team space that you created, if required.

  Note: If you delete the initial team space, the users of the initial team space are redirected to the **Team Space Dashboard** page.

You as a server administrator or the *Team Space Owner* must perform the following tasks in a team space before other members can start using the team space:

- Configure the license server.
- Configure the team space type as either *Restrictive* or *Permissive*.
- Configure the SMTP server and add members who can receive notifications about the operations performed in the team space.

As a member of a team space, you must consider the following points:

- When you log in for the first time immediately after the installation of Test Hub, the **Projects** page of the team space is displayed.
- After you log in to the team space, you are assigned the default roles of a *Project Creator*, and *Member* of the team space.
- You can request to become a member of any other team space. After your request is approved, you are assigned the default role of a *Member* of that team space.
- You can become a member of either a **Permissive** or **Restrictive** type of team space.
- On subsequent logins, the **Team Space Dashboard** page is displayed as the default page.

Migration from earlier releases of Test Hub

The initial team space is the only logical partition in Test Hub 10.1.3. If you migrate from an earlier release of Test Hub to 10.2 or later, then all the assets and resources that you created and associated with the initial team space are migrated to associate with the initial team space in Test Hub 10.2 or later.

Note:

The results of migration are as follows:

- All projects are contained in the initial team space.
- If you created a repository in a team space, then you can find the repository in the initial team space in 10.2.
- All agents, and intercepts that you registered with the server are then registered with the initial team space and you can view from the **Agents and Intercepts** page of the initial team space.
- If you configured a resource monitoring agent in any project, then the details of the configured agent are available in the project.
- You can find the details of SMTP configuration on the server. Only you as a server administrator can view the details of SMTP configuration.
- The license server that is configured for the server for earlier release of Test Hub is then configured for the initial team space.
- All users are assigned the role of a *Project Creator* and *Architect* of the initial team space.
- All users with server administrator roles are assigned the default role of a *Team Space Owner* of the initial team space.

Notes:

- All users of earlier releases are migrated to Test Hub 10.2 or later. You can view the list of all migrated users in the application only when they log into the application at least once.
- After you migrate the server data from a previous version to a newer version, the **Project Owner** must log in to Test Hub so that other members of the project can access the test assets in that project.

Related information

Team space administration on page 172

Default user administration on page 142

Creating a team space

To manage projects in the logical containers of HCL DevOps Test Hub (Test Hub), you must create team spaces.

Before you begin

You must have been assigned the role of a server administrator of Test Hub.

About this task

As a server administrator, you can create multiple team spaces on Test Hub.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane, and then click **Create Team Space**.
   Result
   The **New Team Space** dialog is displayed.
3. Provide a name to the team space.
   Result
   The **URL alias** field is automatically filled.

   Note: If you modify the name of a team space, the **URL alias** is also modified automatically.

4. **Optional:** Modify the **URL alias** for the team space, if required.

   Note: You must enter the **URL alias** that starts with a letter.

5. Select an option to set the permissions to access the team space. You can select any one of the following options:
   - **Permissive** – When a team space is set as **Permissive**, the new user gets the permissions automatically on sending the request to become a member of that team space.
   - **Restrictive** – When a team space is set as **Restrictive**, the *Team Space Owner* provides permission to the new user who becomes a member of that team space.

     You can set the visibility mode of the **Restrictive** type team space as **Public** or **Private**.
     - The **Public** team space is visible to all users in the **Team Space Dashboard**.
     - The **Private** team space is visible only to the members of that team space in the **Team Space Dashboard**.

     Note: If you select the **Restrictive** type team space, then the default setting of the team space becomes **Public**.

6. Provide a brief description of the team space.
7. Select the owner of the team space from the list of users who accessed the application.

   The selected user is assigned the role of a *Team Space Owner*.

   Note: You can assign the *Team Space Owner* role to only one user when you create a team space.

8. Click **Create**.

Result

The dashboard of the team space is displayed.

Results

You have created a team space.

If you are assigned the role of a *Team Space Owner*, then you can view the name of the team space that you created in the navigation pane.

If you assigned any other licensed user with the role of a *Team Space Owner*, then you can view the team space under the **Other Team Spaces** panel.

What to do next

You must configure the license for your team space. See Configuring licenses for team spaces on page 184.

Tasks in a team space

You can find information about the tasks that you can perform as a *Member*, *Project Owner*, *Architect*, or *Team Space Owner* in a team space on HCL DevOps Test Hub (Test Hub).

The roles assigned in a team space for users are as follows:

Persona	**Initial team space**		**Any team space**	
	Role	**Notes**	**Role**	**Notes**
Server administrator	*Team Space Owner*	Default role assigned	*Team Space Owner*	Default role assigned
Licensed user	*Team Space Owner*	Role assigned by Server administrator	*Team Space Owner*	Role assigned by Server administrator
	Architect	Role assigned by *Team Space Owner*	*Architect*	Role assigned by *Team Space Owner*
	Project Creator	Default role assigned	*Project Creator*	• Default role assigned in a *Permissive* team space. • Role assigned by *Team Space Owner* in a *Restrictive* team space.

Persona	Initial team space		Any team space	
	Role	**Notes**	**Role**	**Notes**
	Member	Default role assigned	*Member*	Default role assigned in either a *Permissive* or *Restrictive* team space.

The following table shows the tasks that can be performed by licensed users with their assigned roles in a team space:

Tasks	Team Space Owner	Architect	Project Creator	Member
Configuring licenses	✔			
Configuring a Simple Mail Transfer Protocol (SMTP) server to send email notifications.	✔			
Changing the license server configuration	✔			
Adding members in a team space	✔			
Assigning different roles to other team space members	✔			
Modifying configuration details of a team space	✔			
Deleting a team space	✔			
Renaming a team space	✔			
Adding or removing the team space repository from a team space	✔	✔		
Designing a system model	✔	✔		
Creating projects in a team space	✔		✔	
Adding agents	✔	✔	✔	✔
Adding, modifying, or removing resource monitoring sources	✔			
Connecting resource monitoring agents	✔			
Adding resource monitoring labels	✔			
Working with projects in a team space	✔	✔	✔	✔

The following table provides the links to the information about the tasks that can be performed by users with different roles in a team space:

Tasks	Roles	More information
Configuring a team space license server	*Team Space Owner*	Configuring licenses for team spaces on page 184
Configuring the Named Users	*Team Space Owner*	Adding users to the Named Users list on page 187
Modifying the configuration of a team space	*Team Space Owner*	Modifying the configuration of a team space on page 1202
Viewing the configuration of a team space	All members of a team space	Viewing the configuration of a team space on page 1204
Reassigning the role of team space members	*Team Space Owner*	Managing members and their roles in a team space on page 1207
Becoming a team space member	• *Team Space Owner* • All users can send a request	Becoming a team space member on page 1205
Viewing team spaces	All members of a team space **Notes:** • The **Permissive** and public **Restrictive** team spaces are visible to all the users. • The private **Restrictive** team spaces are only visible to its members.	Viewing team spaces on page 1199
Deleting a team space	*Team Space Owner*	Deleting a team space on page 1209
Adding a project in a team space	*Project Creator*	Creating a project on page 1213
Viewing existing projects	All members of a team space	Viewing active projects on page 1218
Viewing details about the users or license server	All members of a team space	Viewing license details of the team space on page 188
Installing remote agents and configuring them to register with Test Hub	All members of a team space	Management of agents on page 542

Tasks	Roles	More information
Viewing remote agents, or intercepts that are registered with Test Hub in the initial team space	All members of a team space	• Viewing performance agents that are registered with DevOps Test Hub on page 545 • Viewing API agents that are registered with DevOps Test Hub on page 552 • Viewing intercepts that are registered with a team space on DevOps Test Hub on page 942
Adding resource monitoring sources and agents	*Team Space Owner*	Resource monitoring on page 1047
Adding resource monitoring labels to the sources	*Team Space Owner*	Controlling resource monitoring sources in a schedule on page 1079
Viewing the performance metrics	All members of a team space	Viewing resource monitoring usage data on page 1087
Creating a model of the system under test	*Architect*	Creating a system model on page 505
Adding a repository to a team space	• *Team Space Owner* • *Architect*	Adding a repository to a team space on page 502
Viewing an existing system model	All members of a team space	Viewing the system model on page 526

Related information

Team space administration on page 172

Tasks in projects in a team space

You can find information about the roles that you can assign in a project for members of a team space and the tasks that you can perform in projects in a team space.

Members of a team space with the role of a *Project Creator* can create projects in the team space and become the *Owner* of a project. The owners of projects can add *Members* of the team space to a project and assign them the following roles:

- *Owner*
- *Tester*
- *Viewer*

For the list of actions that members assigned any of the roles can perform, see Managing access to the server projects on page 1220.

All projects are created in a team space. When you open a project, the options that are displayed in the navigation pane are specific to the projects within a team space.

When you are working on a project in a team space and you want to return to the options available to work in the team space, you can click the **Back** icon  in the navigation pane.

Note: You must be assigned the role of an *Owner*, a *Tester*, or a *Viewer* of a project.

The following table lists the tasks that members of a team space can perform while working with projects in a team space:

Tasks	**More information...**
Viewing projects in a team space.	Viewing active projects on page 1218
Adding a project.	Creating a project on page 1213
Adding a repository that contains test assets and resources that you want to run on the server.	Adding repositories to a server project on page 1214
Adding secrets to a security group.	Secrets configuration on page 1217
Encrypting the data set resources.	Managing an encrypted dataset on page 447
Configuring a change management system.	Configuration of a change management on page 1092
Adding members to a project and configuring their roles.	Adding users to a project on page 1217
Selecting the global branch in a project to view the test assets and resources.	Selecting the global branch in a project on page 1233
Adding agents to your project, if you are using remote agents as a location to run certain tests.	Adding a performance agent to a project on page 546
Reading the considerations that you must take into account before you configure a test run.	Prerequisites to running tests on page 527
Reading the considerations that you must take into account before you configure a run or start a virtual service.	Prerequisites for running HTTP virtual services on page 938
Configuring a run of the test assets.	Test run configurations on page 610
Configuring a run of the virtual service resource.	Configuring a run of a virtual service on page 946

Tasks	More information...
Viewing the progress of a test run.	Viewing the progress of running test assets on page 926
Managing a running test.	Management of running tests on page 922
Viewing the results of a test run after the run is completed.	Test results on page 983
Reviewing the test results and creating defects for test runs from the **Results** page.	Creating Jira issues for the test results on page 1096

HCLSoftware license portal

Licensing for HCL DevOps Test Hub (Test Hub) is administered through My HCLSoftware (MHS). You can manage your software entitlements and licenses by using MHS .

When a software order is placed and acknowledged, a software entitlement is created. You must then follow the instructions in the Software Order Acknowledgment document that you receive to activate your entitlement, create devices, and download the software from the portal.

The license portal provides both software distribution and management of your software entitlements that are purchased from HCLSoftware. The portal provides control and flexibility on how to consume your licenses. An organization identifies one of its resources as a License Manager (also called Tech or Portal Admin) who is familiar with the language of licenses. For more information refer to, My HCLSoftware Licensing documentation.

Management of licenses in DevOps Test Hub

As a *Team space Owner* or *Server Administrator*, you can configure and manage licenses for your team spaces in HCL DevOps Test Hub (Test Hub). As a member of any team space, you can only view the number of licenses that are configured for the team space. If you are not a member of any team space, you cannot view or perform any operations in any of the team spaces on Test Hub.

You can get all the information of configured license server for different team spaces from various *Team Space Owners*. You can navigate to the Test Hub Swagger UI by accessing the following URL:

`https://myorg.com/test/#/Initial/admin/docs`

Where `myorg.com` is the hostname of Test Hub.

In the Swagger UI, you can select the **Gateway** category from the drop down, open the **Licensing management** rest APIs, and then click the **GET /rest/licensing/inventory** API. To generate the report that contains all the information of license server, you can click the **Try it out** option, enter the required details, and then click **Execute**.

The license inventory report contains information such as details about the configuration of team space licensing, a list of *Team Space Owners* and their email addresses, information about licenses that are mapped to the license server, and the current connectivity status of the license server.

Depending on your privileges, you can perform the following tasks in Test Hub:

Privileges	Tasks	More information
Team space Owner OR *Server Administrator*	Read about licensing on Test Hub.	DevOps Test Hub licensing information on page 183
	Configure licenses.	Configuring licenses for team spaces on page 184
	Configure named users.	Adding users to the Named Users list on page 187
	Remove or replace named users.	Removing named users on page 188
Licensed user	Perform tasks or operations in a team space.	Tasks in a team space on page 177
	Viewing details about licenses.	Viewing license details of the team space on page 188

DevOps Test Hub licensing information

You can find information about the licensing on Test Hub.

To use the features offered by Test Hub, you must purchase the following licenses:

- Named user
- Concurrent Virtual Service

Named user licenses

As a *Team space Owner*, you can configure the License Server and add users to the **Named Users** list for the team spaces.

The **Named user** licenses are automatically self-acquired on a first-come-first-serve basis. The licenses are always available for the users who are added to the **Named Users** list. Also, those users can perform the project tasks in their team space even though the **Disable automatic self-acquisition of named user licenses** checkbox is cleared in the **License Configuration** page. However, the other users receive an error notification about the unavailability of the license when they try to perform the project tasks.

Remember: A single **Named user** license is used from the total number of **Named user** licenses when you perform most of the operations in Test Hub.

Important: You can perform only the following operations in team space without using a license:

- Configure the License Server
- Modify the user roles in a team space
- Delete the team space

Remember: The **Named user** license is returned to the license pool only under any of the following conditions:

- When the **Named user** is no longer a member of any team space.
- When the **Named user** is removed as a user of Test Hub.

Important: The **Named user** license can take a few minutes to return to the license pool.

Concurrent Virtual Service licenses

Test Hub requires an additional license to use the service virtualization feature. You must have sufficient **Concurrent Virtual Service** licenses to run instances of virtual services that are contained in API Suites on Test Hub.

A single **Concurrent Virtual Service** is used from the total number of **Concurrent Virtual Service** licenses when any of the following events occur:

- An instance of a virtual service is started by running a virtual service resource. A license is consumed for every running instance of the virtual service.
- An API suite is run that has a scenario that references a local virtual service and the scenario begins

The **Concurrent Virtual Service** license is returned to the license pool when the virtual service instance stops.

For example, consider you decided to have three team spaces including the initial team space, then you must configure the licenses required for each team space on the License Server based on your requirement. You must then use the URL and ID of those License Servers to configure the licenses for the team spaces.

Related information

Configuring licenses for team spaces on page 184

Configuring licenses for team spaces

As a *Team space Owner*, you can configure the License Server for the team spaces on the **License Configuration** page so that members of the team space can perform various operations such as creating a project, running tests, and so on.

Before you begin

You must have completed the following tasks:

- Configured the licenses that you purchased.
- Been assigned the role of a *Team space Owner* of the team space.
- Optional: Created a team space. See Creating a team space on page 175.

About this task

You must have a valid license to work with the team spaces that you create on Test Hub. You must provide a separate URL and ID of the License Server for the team spaces when you configure the License Server. If you provided the URL and ID of the License Server that is already configured for any other team space, then the configuration of the license for the team space fails with an error message. If you delete the team space, then the URL and ID of the License Server used for that team space is released. You can then use that for another team space.

Note: When you upgrade Test Hub from the previous version to the latest version, then the license configuration of the previous release applies to the initial team space. You can verify the information from the **License Configuration** page of the initial team space.

1. Log in to Test Hub.
2. Click the **Switch Team Space** icon from the left navigation pane.
3. Open the team space for which you want to configure the license from the **Switch to** option.

   Note: The **Switch to** option is displayed only if you have two or more team spaces.

4. Click **Manage > License** in the navigation pane.
   Result
   The **License Configuration** page is displayed.
5. Enter the values in the **License Server URL** and **License Server ID** fields to configure the licenses for the team space.

   Note: The **License Server ID** is also known as **deployment ID** in My HCLSoftware (MHS).

   Result
   You can view the following messages based on the value that you enter in the **License Server URL** and **License Server ID** fields:

Values	Messages
No value	Enter team space license server URL.

Values	Messages
Incorrect format of the License Server URL or License Server ID	This is not a valid format of the License Server URL or License Server ID.
Correct value	“License server available” If the URL and ID has been used by another team space, shows a message: “License server already used License server is already used by another team space.”

Note: The **License Statistics** displays the Total, Available, and In Use counts for each type of license.

6. Perform the following actions to view or hide the details of the License Server:

Note: The default status of the **License Server URL** and **License Server ID** fields are hidden.

Fields	Icons	Action	Results
◦ **License Server URL**		Click	Test Hub shows the URL and ID of the License Server that you have configured.
◦ **License Server ID**		Click	Test Hub hides the URL and ID of the License Server that you have configured.

7. **Optional:** Click **Recheck** to verify the current status of the License Server.

   Based on the availability of License Server, you can view the following messages:

   - License Server is available
   - License Server is unreachable

8. **Optional:** Select the **Disable automatic self-acquisition of named user licenses** option if you do not want to self-acquire **Named users** license on a first-come-first-serve basis for your team space.
   The following table describes the functionality of the **Disable automatic self-acquisition of named user licenses** option:

Option	When	Description
Disable automatic self-acquisition	Clea	Any member of a team space can get a license when there are sufficient licenses are available for the team space.

Option	When	Description
of named user licenses	red ☐	
	Selected ☑	Only such users who are added to the **Named Users** list get a license to perform the tasks in their team space. If other users try to perform any project operations in any of the team spaces, they cannot get a license even if sufficient licenses are available.

Results

You have configured the license for the team space.

What to do next

You can perform any of the following tasks:

- Add users to the **Named Users** list. See Adding users to the Named Users list on page 187.
- Add members to your team space. See Adding members to a team space on page 1200.

Adding users to the Named Users list

As a *Team space Owner* in HCL DevOps Test Hub (Test Hub), you can add users to the **Named Users** list so that the licenses are always available for those users.

Before you begin

You must have completed the following tasks:

- Been assigned the role of a *Team space Owner* of the team space.
- Optional: Created a team space. See Creating a team space on page 175.
- Configured the license for the team space. See Configuring licenses for team spaces on page 184

.

1. Log in to Test Hub.
2. Click the **Switch Team Space** icon  from the left navigation pane.
3. Open the team space for which you want to add users from the **Switch to** option.

   Note: The **Switch to** option is displayed only if you have multiple team spaces.

4. Click **Manage > License** in the navigation pane.
   Result

The **License Configuration** page is displayed.

5. Click **Add Users** in the **Named Users** section.
6. Enter the name or the email ID of the user in the **Search** field of the **Add named users** dialog, and then select the name from the list.
7. Click **Add User** to make selected users as **Named users**.

Results

You have added users to the **Named Users** list.

Removing named users

When you want to remove a user from the list of named users, you as a *Team space Owner* in HCL DevOps Test Hub (Test Hub), can perform these changes from the **License Configuration** page.

Before you begin

You must have completed the following tasks:

- Assigned the role of a *Team space Owner* for the team space that you want to modify the users from the **Named Users** list.
- Added users to the list of named users for the team space on Test Hub.

1. Log in to Test Hub.
2. Click the **Switch Team Space** icon  from the left navigation pane.
3. Open the team space for which you want to remove the users from the **Named users** list from the **Switch to** option.

   Note: The **Switch to** option is displayed only if you have multiple team spaces.

4. Click **Manage > License** in the navigation pane.
5. Click the **Delete** icon  in the **Named Users** section for the users that you want to remove from the **Named Users** list.

Results

You have removed a user from the **Named Users** list.

Viewing license details of the team space

You can view license details of the team space to know information such as number of licenses that are configured, licensed users or details of the License Server on the **License Configuration** page.

Before you begin

The *Team space Owner* must have configured the licenses for the team spaces on HCL DevOps Test Hub (Test Hub).

About this task

You as a licensed user can view only the number of licenses configured the details of the licensed users on the **License Configuration** page. However, if you are a *Team space Owner* or *Server Administrator*, you can also view the URL and ID of the License Server that is configured for the team space.

1. Log in to Test Hub.
2. Perform one of the following actions described in the following table to view the license details of the team space:

Roles	Action
Team space Owner or *Server Administrator*	Perform Steps 3 on page 189 to 6 on page 189.
Any other users	Perform Steps 3 on page 189 to 5 on page 189.

3. Click the **Switch Team Space** icon from the left navigation pane.
4. Open the team space for which you want to view the license details from the **Switch to** option.

   Note: The **Switch to** option is displayed only if you have multiple team spaces.

5. Click **Manage > License** from the navigation pane.
   Result

   The **License Configuration** page is displayed with the following license information:

   - The number of **Named user** licenses that are currently in use.
   - The number of **Named user** licenses that are configured on the License Server.
   - The number of **Concurrent Virtual Service** licenses that are configured on the License Server.
   - The number of **Workbench** licenses that are configured on the License Server.
   - The status of License Server.
   - Whether the self-acquisition of named user licenses feature is enabled or disabled.
   - The name and email ID of the users that are configured as named users by the *Team space Owner*.

   Note: The **License Statistics** displays the Total, Available, and In Use counts for each type of license.

6. Perform the following actions to view or hide the details of the License Server:

Fields	Icons	Action	Results
◦ **License Server URL**		Click	Test Hub shows the URL and ID of the License Server that you have configured.
◦ **License Server ID**		Click	Test Hub hides the URL and ID of the License Server that you have configured.

Results

You have viewed the details about the licenses for the team space.

Management of resources

You can find information about managing resources on HCL DevOps Test Hub (Test Hub) along with the tasks that you can perform to define retention rules and monitor resource usage for test results.

By defining the retention rules, you can prevent the system from running out of resources and ensure only the required data is maintained.

Managing retention rules

You can create, edit, and delete retention rules to automatically manage and clean up outdated or no longer needed test result data. This ensures that outdated or unnecessary results are managed and storage is optimized.

Before you begin

You must have been assigned the Team Space Owner role in Test Hub.

About this task

Retention rules help teams manage storage effectively, especially in large-scale testing environments with frequent or long-running test runs. The retention rule provides an automated way to identify and reclaim resources associated with test results. You can choose to perform the following tasks:

- Delete the entire result, removing it from the system completely
- Delete only the test log and report data, reducing disk usage while keeping result metadata

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Manage > Resources**.
   The **Resources** page is displayed.
3. Click the **Rules** tab.

   The **Rules** page displays the following retention rules for the team space:

- **Delete Stale Branches**: Deletes test results created on Git branches that no longer exist and are older than 3 days. For example, results from branch edits or Git cleanup.
- **Delete Old Stub Results**: Deletes results of type APISTUB that are older than 30 days.

Note: You can disable these default rules if required.

In the **Rules** tab, you can perform the actions that are listed in the following table:

Task	Action
Enabling or Disabling default retention rule	Perform the following steps: a. Click the **Enabled** button to enable or disable the rules. The **Enable Retention Rule** dialog is displayed. b. Click **Ok** to confirm your action.
Adding retention rule	Perform the following steps: a. Click **Add Rule**. The **New Retention Rule** dialog is displayed. b. Enter a name in the **Rule Name** field. c. Select the conditions from the drop-down list. The condition types are: ▪ **Result age is older than (days)** ▪ **Branch status is** ▪ **Result verdict is one of** ▪ **Result size is greater than (MBs)** ▪ **Result labels contains one of** ▪ **Result labels does NOT contain any of** ▪ **The project is one of** ▪ **The test type is one of** ▪ **Max results per unique test** ▪ **Max results per unique test and verdict** **Note:** You can add multiple conditions to define a retention rule. For example, you can select the condition **The project is one of** and choose a project. You can add another condition **Result verdict is one of** and select the value **Pass**.

Task	Action
	The **Max results per unique test** and **Max results per unique test and verdict** conditions are not displayed if you have selected any of the following conditions: ▪ **Result age is older than (days)** ▪ **Result verdict is one of** ▪ **Result size is greater than (MBs)** The estimated impact is displayed at the bottom of the window, showing how much data will be freed. d. Select one of the following options: ▪ **Entire Result**: Removes all traces of the test result. ▪ **Just report/testlog**: Retains metadata such as verdict and stop reason. e. Click **Create** to add a rule.
Editing retention rule	Perform the following steps: a. Click the **Edit** icon next to the rule that you want to modify. The **Edit Rule** dialog is displayed. b. Modify the name in the **Rule Name** field. c. Select the conditions from the drop-down list. The conditions types are: ▪ **Result age is older than (days)** ▪ **Branch status is** ▪ **Result verdict is one of** ▪ **Result size is greater than (MBs)** ▪ **Result labels contains one of** ▪ **Result labels does NOT contain any of** ▪ **The project is one of** ▪ **The test type is one of** ▪ **Max results per unique test** ▪ **Max results per unique test and verdict** **Note:** You can add multiple conditions to define a retention rule. For example, you can select the condition **The project is one of** and

Task	Action
	choose a project. You can add another condition **Result verdict is one of** and select the value **Pass**. The **Max results per unique test** and **Max results per unique test and verdict** conditions are not displayed if you have selected any of the following conditions: ▪ **Result age is older than (days)** ▪ **Result verdict is one of** ▪ **Result size is greater than (MBs)** The estimated impact is displayed at the bottom of the window, showing how much data will be freed. d. Select **Entire Result** to remove all the traces of test result or select **Just report/testlog** to retain the metadata (For example: verdict, stop reason). e. Click **Update** to apply the changes.
Running retention rule	Perform the following steps: a. Click the **Execute retention rule now** icon to run the rule immediately to identify and delete results matching its conditions. The **Confirm Recycle** dialog is displayed. b. Click **Yes** to confirm the action
Deleting retention rule	Perform the following steps: a. Click the **Delete Rule** icon to remove the rule from the team space entirely. The **Delete Rule** dialog is displayed. b. Click **Delete** to confirm the action.

Results

You have successfully created, updated, executed, or deleted retention rules.

Viewing resource usage statistics

You can view statistics about the resource usage to monitor and optimize the retention of results.

Before you begin

You must have been assigned the Team Space Owner role in Test Hub.

About this task

When the server runs low on resources, it is difficult to identify which projects are consuming the most resources. This information is accessible to admins at the team space level to make informed decisions about the projects that can be pruned.

1. Log in to Test Hub.

   The **Projects** page of the initial team space is displayed.

2. Click **Manage > Resources**.

   The **Resources** page is displayed.

3. Click the **Statistics** tab.

   The **Statistics** table is displayed with the following details.

Statistics details	Description
Project	The name of the project. All the projects within the team space are listed.
Archive status	Shows whether the project is currently active or archived.
Users	The number of users associated with the project.
Results	The size of the results stored for the project in KB, MB, or GB.
Last Result	The date and time of the most recent test result.
Repositories	Total size of all repositories within the project in KB, MB, or GB.
Last Repository Modification	The date and time when the project repository was last modified.

   You can sort the table by any column by clicking the column headers so that you can quickly identify projects that consume the most resources or have the most recent activity.

Results

You have viewed detailed resource usage statistics for projects in the team space.

Configuration of an MCP server endpoint

HCL DevOps Test Hub (Test Hub) provides a Model Context Protocol (MCP) server endpoint. The configuration of the MCP endpoint enables AI assistants within Visual Studio Code (VS Code) to securely integrate with your testing environment, read the test data, and perform specific actions that are supported by the tools.

When you configure the MCP endpoint in VS Code, the supported AI extension functions as an MCP client that communicates with the MCP server. This connection allows the AI assistant to list projects and test assets, retrieve test result data and summaries, and run tests directly within the development environment.

You can use the following endpoint to connect to the MCP server:

```
https://<server_hostname>/test/mcp
```

MCP tools available in Test Hub

The following table lists the MCP tools that are available in Test Hub for running tests and retrieving test results.

Tools	Description	Input	Output
testhub__get_projects	Retrieves a list of projects available in Test Hub.	None	List of available projects
testhub__list_tests	Retrieves a list of tests in Test Hub for a specific project and branch.	projectId, branch	List of tests for the specified project and branch
testhub__get_results	Retrieves test result summaries for multiple test runs in Test Hub.	projectId, branch	Test result summaries for multiple test runs
testhub__get_result_by_id	Retrieves the test run result and detailed result data.	projectId, resultId	Test run result and detailed data
testhub__run_test	Runs a test in a specific project.	projectId, assetId, branch	Details of the test run, including execution ID
get_results_screenshots	Retrieves the smartshots associated with the failed steps of the failed tests.	projectId, resultId, context	Smartshots of the failed steps

Configuring an MCP server endpoint in VS Code

You can configure an MCP server endpoint that enables an MCP-compliant client, such as VS Code, to access specific server-side tools. The endpoint offers several tools that provide access to teamspace, project, and test asset details, along with the ability to run tests and retrieve test results.

Before you begin

You must have completed the following tasks:

- Obtained the hostname or IP address of Test Hub.
- Logged in to VS Code.

About this task

While configuring the endpoint, you will be prompted to authenticate the connection by entering the credentials. And when you run tests using the tool, you must provide an offline token for authentication.

1. Press `Ctrl+Shift+P` to open the command palette in VS Code.
2. Type `MCP add` and select the **MCP: Add Server...** option from the list.
3. Select the **HTTP (HTTP or Server-Send-Events)** option.
4. Paste the URL of the Test Hub MCP server in the **Enter Server URL** field in the following format:

   `https://<server_hostname>/test/mcp`

   For example, `https://testhubserver.com/test/mcp`
5. Enter a name to identify the server, or accept the default when prompted to enter the server ID in the **Enter Server ID** field.

   For example, `myTestHub_mcp`.
6. Select one of the following workspace options when prompted:
   - **Global**: Ensures the tools are available from all your workspaces.
   - **Workspace**: Ensures the tools are available from only the current workspace.

   An `mcp.json` configuration file is created.
7. Open the `mcp.json` file from the explorer.
8. Perform one of the following steps to connect to the server:
   - Click **Start** that is displayed above the server name in the `.JSON` file.
   - Click **Extensions**, right-click the server name, and then select **Start Server**.

   VS Code displays a notification stating that the MCP server requires authentication.
9. Click **Allow**.

   A second notification is displayed asking whether you want to open an external website.
10. Click **Open**.

    Your default browser opens and displays the Test Hub login page.
11. Sign in with your Test Hub credentials.

    A consent screen is displayed requesting access to Test Hub MCP tools.
12. Click **Yes** to approve and complete the connection.

    After successful authentication, VS Code displays the number of available MCP tools.



Note: The MCP server times out after 30 minutes of inactivity. If no tool calls occur during this period, your session will expire and you are prompted to re-authenticate.

Results

You have configured an MCP server endpoint in VS Code.

What to do next

In the VS Code chat window, you can type a prompt that requires data from the server. For example, you can enter `Get the list of projects` or `List the latest failed test runs`. If you enter a prompt such as `Execute the test <test_name>` for running the test, the tool retrieves the test assets details from the server and then displays a prompt to enter the offline token

to process the request. You must generate a token, copy it and paste the token in the VS Code chat window for the test run to complete.

Chapter 5. Test Author Guide

This guide describes how to create and edit test assets in HCL DevOps Test Hub (Test Hub), and then publish test assets to the remote Git repository that you have configured in your project. This guide is intended for testers and test managers.

For an overview of the test creation and management by using Test Editor, click here to watch the video.

Edit branch overview

When you want to create or modify a test resource in HCL DevOps Test Hub (Test Hub), you must use an Edit branch. An Edit branch is a temporary branch created from the selected branch in the repository configured within the project, where you can save the created or edited resource.

You must be a *Team Space Owner*, *Project Owner*, or *Member* with a *Tester* role to create or edit a test resource.

When you want to create or edit a test resource, you must first select the branch in the project repository into which you want to add the created resource or the branch that contains the resource you want to edit.

Note: If you create an Edit branch from a read-only repository, a warning is displayed that publishing will fail.

When you create the Edit branch, any project member can use your Edit branch while the branch is available. When you provide a name for the Edit branch, you must adhere to the rules of Git branch naming conventions. For more information, refer to the Git documentation. You can view all the Edit branches from the **Branch** drop-down list when other project members create Edit branches.

Note: If there are too many simultaneous requests for Edit branch creation in a project, Git might take some time to complete the branch creation.

To differentiate the remote branch and Edit branch from the **Branch** drop-down list, an edit icon and the initials of the member who created the Edit branch are shown for the Edit branch. After you create the Edit branch, you can also view the Edit branch by accessing the remote Git repository in a browser.

If you want to save the created or edited resource to the branch in the project repository, publish the changes from the Edit branch to the selected branch. When the project repository is replicated automatically to the remote repository, the created or edited resources are then available in the remote repository in the branch they were published. The Edit branch that you used is automatically deleted after publishing, and it disappears from the **Branch** drop-down list and from the remote Git repository when you perform any of the following actions:

- Publish changes to the remote Git repository
- Discard the changes made in all test assets

Note: After the Edit branch disappears, the **Branch** field is set to the previously selected branch name.

If publishing the changes in edit branch fails due to conflict with remote branch, you can push the edit branch to remote repository by using **Push** option in the **Publish** dialog. Then create a pull request to merge your changes into the target branch by resolving the conflicts manually in Git.

When multiple users use the same edit branch, the modifications, renaming, moving, or deletions made by a user are notified to the other users. The notifications are displayed in the following scenarios:

- A user modifies the test asset. In this scenario, the notification also provides the following options:
  - **Reload**: Refreshes the page with the updated details.
  - **Compare**: Displays the **Compare <asset_name>** pop-up with the last saved version and the current version for comparison. The changes are highlighted.
- A user moves or renames a test asset.
- A user deletes a test asset.
- A user discards the changes on the **Changes** page.

Test Editor overview

You can use the **Test Editor** tab to create and manage test assets such as applications, Web UI tests, test suites, and environments in HCL DevOps Test Hub (Test Hub).

You can navigate to the **Test Editor** page and **Suite Editor** page from the **Author > Test Editor** tab in Test Hub.

When you work on individual tests by creating or configuring test assets, the **Test Editor** page streamlines the entire testing process. You can efficiently organize applications, tests, suites, smartshots, environments that you created or modified in your project by using the test navigator panel in **Test Editor**.

Related information

Application management on page 204

Web UI test management on page 209

Test suite management on page 292

Test Editor page on page 201

Suite Editor page on page 203

Test navigator panel

The **Test Editor** page contains a central hub called the test navigator panel. You can use the test navigator panel to manage and organize test assets in multiple folders of your project.

You can right-click a test asset in the test navigator panel and select an option from the context menu to perform the following actions:

Actions	Description
Search	You can search for a test asset in a project by entering the name of the test asset in the **Search** field.
New	You can create an application, test, suite, or an environment in a project. See Creating an application on page 204, Recording a Web UI test on page 211, Creating a suite on page 292, and Creating an environment on page 338.
Edit	You can modify application components, Web UI tests, suites, or environments when you want to run a test asset with different values. See Modifying application components on page 206, Editing a Web UI test on page 236, Modifying a suite on page 296, and Modifying an environment on page 340.
Rename	You can change the name of an application, test, suite, environment, or smartshot within a project.
Move	You can change the location of a test, suite, environment, or smartshot and move it under a different application within a project. When moving a single asset, you can choose to include dependencies that are referenced only by the selected asset. If a dependency asset is shared by another asset, you cannot select that dependent asset for moving.
Delete	You can delete an application, test, suite, environment, or smartshot when it is no longer required in a project. When deleting a single asset, you can choose to include dependencies that are referenced only by the selected asset. If a dependency asset is shared by another asset, you cannot select that dependent asset for deletion. See Deleting a test asset on page 275, Deleting a suite on page 310.
Compare	You can view and compare changes made in the edit branch with unpublished changes in the Web UI test, suite or environment. You can select the **Compare** option, which appears when you right-click the test asset.
Save As	You can create a copy of an existing application, test, suite, or an environment with a new name in a project. You can select the **Save As** option, which appears when you right-click the test asset.
Collapse All	You can minimize all expanded test assets in the project view to simplify navigation and focus on top-level items.
Expand All	You can expand all sections or folders in the project view to quickly view the all the test assets.

You can select multiple test assets in the Test Navigator panel by pressing `Ctrl` and clicking individual assets, or by pressing `Shift` and clicking to select a range of assets.

Note: If you configured multiple repositories, the test navigator panel displays the test assets in all the configured repositories. However, you cannot add suites, tests, or modules from one repository to suites or tests of the other repository.

You can use the following options in the test navigator toolbar when you manage and organize tests, applications, and suites across multiple folders of your project.

Icon	Description
	You can use the **Virtual Folder** icon to view folders at different levels. The available options are as follows: • **Root Folders**: Shows virtual folders at the top, followed by the folder structure of the repository. • **Embedded Folders**: Shows the folder structure of the repository first, with files organized into virtual folders. • **None**: Excludes virtual folders entirely, and presents only the original folder structure of the repository.
	You can use the **Link with editor** icon to connect Test Editor tabs to their corresponding locations in the test navigator panel. By using this icon, you can easily navigate and quickly access specific test assets within your project.
	You can apply a filter to view folders by their asset type. You can select one or more of the following asset types and apply the filter: • Datasets • Tests • Environments • Others • Smartshots • Suites
	You can use the **Save All** icon to save all the changes made to multiple test assets or folders within your project.

Test Editor page

The **Test Editor** page consists of two sections: a left pane and a right pane. You can use the left pane to view, edit, and organize the test. You can use the right pane to manage and analyze the test within the project.

You can use the following options in the left pane of the **Test Editor** page:

Options	Actions
View	In the default setup, the test is displayed in the **Visual** view. To view the test in **Code** view, you must click the **Code** option. The view that is selected will be active till you close the test.
Find Step	Enter the name of the test in the **Find Step** to search the test assets in the current test.
	Select the  icon to begin the test run.
Undo 	Select the **Undo**  icon to undo the recent changes.
Redo 	Select the **Redo**  icon to redo the recent changes.
Save 	Select the **Save**  icon to save all the changes made to the test.
Publish 	Select the **Publish**  icon to publish the test.

You can use the following tabs in the right pane of the **Test Editor** page:

Tabs	Actions
Catalog	Select the **Catalog** tab to add additional steps to the tests. See Adding user actions in a test from smartshots on page 239, New test steps from the General catalog on page 242, and New steps for UI-related actions and user actions in a test from the UI catalog on page 252.
Results	Select the **Results** tab to view the test results. See Running a Web UI test on page 228.
Usage	Select the **Usage** tab to see where the current test is referenced in other tests. You can click the  icon to open the referenced tests. If there are no references to the current test, then the tab displays that the current test has no dependants.
Details	Select the **Details** tab to view the Git details such as **Asset ID**, **Repo**, **Path**, **Branch**, and **Commit**.

Viewing a Smartshot

You can select a **SmartShot** (`.jpeg`) file from the **Test navigator** panel to open it in the **Test Editor** page.

The **SmartShot** is displayed in the Test Editor with the **Elements** tab to display the elements hierarchy of the Smartshot. You can select elements from the hierarchy and view the corresponding details on the SmartShot, consistent with the existing implementation.

Related information

Web UI test management on page 209

Suite Editor page

The **Suite Editor** page consists of two sections: a left pane and a right pane. You can use the left pane to view, edit, and organize the suite. You can use the right pane to manage and analyze the suite within the project.

You can use the following options in the left pane of the **Suite Editor** page:

Options	Actions
View	In the default setup, the suite is displayed in the **Visual** view. To view the suite in **Code** view, click the **Code** option. The view that is selected will be active till you close the test.
Find Step	Enter the name of the suites and tests in the **Find Step** to search suites and tests in the current suite.
	Select the  icon to begin the test run.
Undo 	Select the **Undo**  icon to undo the recent changes.
Redo 	Select the **Redo**  icon to redo the recent changes.
Save 	Select the **Save**  icon to save all the changes made to the suite.
Publish 	Select the **Publish**  icon to publish the suite.

You can use the following tabs in the right pane of the **Suite Editor** page:

Tabs	Actions
Results	Select the **Results** tab to view the test results. See Running a suite sequentially on page 299 and Running a suite concurrently on page 300.
Usage	Select the **Usage** tab to see where the current suite is referenced in other suites. You can click the  icon to open the referenced suites. If there are no references to the current suite, then the tab displays that the current suite has no dependants.
Details	Select the **Details** tab to view the Git details such as **Asset ID**, **Repo**, **Path**, **Branch**, and **Commit**.

Related information

Test suite management on page 292

Application management

When you want to create and manage Web UI tests in HCL DevOps Test Hub (Test Hub), you must first create an application within the project.

By creating an application in the project, you can streamline test management by ensuring that the tests are run on the correct application. The application organizes the process by linking each test asset to a specific application. Thereby improving the efficiency of test run.

Components of an application

The components are the key elements in the application that you might want to test. These components include the following details:

- **URL**: The web address where the application is hosted.
- **Description**: A brief explanation or details about the application.
- **Image**: A graphic representation or a logo of the application.

When you open an application, the following tabs are available:

- **Details**: The application URL, description, and dependencies are displayed in this tab. You can modify these details as required.
- **Smartshots**: The smartshots captured for the application are displayed in this tab. Smartshots are organized into two collapsible sections:
  - **Used Smartshots**: Displays smartshots that are referenced by one or more tests. The label displays the test that references the smartshot.
  - **Unused**: Displays smartshots that are not referenced by any test. Use the **Select all** checkbox to select all unused smartshots for bulk deletion. You can also hold the **Ctrl** key and select multiple smartshots to delete them in bulk.
- **APIs**: The APIs associated with the application are displayed in this tab.

Managing applications primarily consists of two key activities:

- Creating an application on page 204.
- Modifying application components on page 206.

Creating an application

You can create an application within your project in HCL DevOps Test Hub (Test Hub) and select it to create or record a Web UI test for that specific application.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

When you create an application within your project, you can initiate the test creation process, capture interactions and behaviors on the Web UI, and automate testing workflows.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Navigate to **Author > Test Editor**.
   The **Test Editor** page and test navigator panel are displayed.
4. Perform any of the following actions:
   - Click the **Action Menu** icon  and select **Application**.
   - Right-click the project repository in the test navigator panel and select **New Application**.
   - Click **Create Application** on the **Test Editor** page, which is displayed when no applications are created in your project.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 6 on page 206.
5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

     Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:

a. Select **Create new edit branch**.
b. Enter a name for the Edit branch in the **Branch Name** field.
c. Click **Save**.

6. Type a unique name in the **Name** field of the **Create Application** dialog.
7. Click **Create**.
   The application is successfully created and it opens in the **Test Editor** for further configuration.
8. Type the web address in the **URL** field.
   Test Hub detects and displays the logo of the website in the **Image** field.
9. Type a brief summary or relevant details about the application in the **Description** field.
10. Click the **Save** icon to save the details.

Results

You have created an application.

What to do next

You can perform the following tasks:

- Record a Web UI test. See Recording a Web UI test on page 211.
- Create a Web UI test. See Creating a Web UI test by using the empty test option on page 217.
- Modify application components. See Modifying application components on page 206.

Modifying application components

You can modify the existing application components within your project in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Ensured that your project contains an application.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Navigate to **Author > Test Editor**.

Result

The **Test Editor** page and test navigator panel are displayed.

4. Select the **Application** in the test navigator panel and click **Edit**.
   Alternatively, right-click the application in the test navigator panel and click **Edit**.
   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 6 on page 207.
5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
6. You can modify the following details of an application in the **Test Editor** page:
   - **URL:** Update the web address if the location of the application has changed.
   - **Description:** Modify the summary of the application or details to reflect any new information or changes to the application.
   - **Image:** Change the logo or graphical representation of the application by clicking the **Edit Icon** in the **Image** field.
7. Click the **Save** icon to save the details.

Results

You have modified the application components.

Manage OpenAPI documents with version 3 standards for a web application

You can upload and manage OpenAPI documents that follow version 3 standards for a web application. OpenAPI documents define the APIs that are available for API testing.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Ensured that your project contains an application.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Navigate to **Author > Test Editor**.
   Result
   The **Test Editor** page and test navigator panel are displayed.
4. Select the **Application** in the test navigator panel and click **Edit**.
   Alternatively, right-click the application in the test navigator panel and click **Edit**.
   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 6 on page 208.
5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
6. Click the **APIs** tab.
7. Click **Add APIs**.
   The **Upload** dialog is displayed.
8. Click **Select files** and select one or more OpenAPI documents in JSON or YAML format.

   The selected documents are displayed in the **Upload** dialog box.

Test Hub validates each document. The document must contain the required OpenAPI version and title information. An error indicator and a description of the validation error are displayed for documents that do not meet the validation requirements.

9. Click **Upload**.

   The uploaded OpenAPI documents are displayed in the **APIs** tab with information such as the API title and description.

   An APIs virtual folder is created in the left navigator, and the uploaded documents are displayed in the folder.

10. Select the document that you want to delete and click the **Delete** icon to delete an OpenAPI document.

    The **Delete** dialog is displayed.

11. Select the dependencies that you want to delete along with the OpenAPI document.
12. Click **Delete**.

    The OpenAPI document and the selected dependencies are deleted. The document is removed from the **APIs** tab, and from the virtual folder in the left navigator.

Results

You have uploaded and managed OpenAPI documents for the web application.

Note: External URLs are not currently supported for OpenAPI schemas. Upload and manage OpenAPI documents within the project to support change management and prevent unexpected breakages.

Web UI test management

You can test your web applications by recording Web UI tests in HCL DevOps Test Hub (Test Hub). Test Hub supports industry-standard browsers such as Google Chrome, Microsoft Edge, and Mozilla Firefox to record and play back Web UI tests. After the playback, you can evaluate the test results.

You can perform the following tasks to create and then manage your test assets in Test Hub:

- Recording a Web UI test on page 211
- Creating a Web UI test by using the empty test option on page 217
- Running a Web UI test on page 228
- Viewing step-level results and updating tests on page 232
- Editing a Web UI test on page 236
- Adding user actions in a test from smartshots on page 239
- New test steps from the General catalog on page 242
- New steps for UI-related actions and user actions in a test from the UI catalog on page 252
- Adding conditional logic and redefining the flow in tests from the Flow catalog on page 260
- Creating and assigning test variables on page 312

- Assigning an object property to a test variable on page 314
- Disabling test steps on page 271
- Configuring timeouts for tests and test steps on page 272
- Deleting a test asset on page 275

Web UI test creation

You can create Web UI tests by using the recording option or the empty test option in the HCL DevOps Test Hub (Test Hub). Test Hub automatically enables web browsers and provides DevOps Test Runtime or the recording agent to configure the test environment before you start recording tests for web applications.

You can create tests for the web applications that you have already configured in **Test Editor**. When you create a test, a `.dtx.yaml` file is created and listed in the test navigator panel. Test Hub captures mouse and keyboard actions that the user performs to interact with the web pages when you record a test. Test Hub supports the following HTML5 controls on the web pages:

- Text box
- Password field
- Text area
- Button
- Checkbox
- Drop-down list
- Date selector
- Radio button
- Slider
- JavaScript popup boxes such as alert, confirm, or prompt dialogs
- Drag-and-drop boxes
- File upload field

Test Hub also supports the use of multiple tabs and windows to record a test.

Application URL management

If you have configured the application with a URL, then Test Hub opens the browser with the configured URL when you record a test for that application. Otherwise, you must enter or paste the URL in the browser when you record a test. When the application contains a URL, Test Hub creates a variable App_<app_name>_URL to store the value of the URL. Otherwise, if you manually enter or paste the URL, then the URL is embedded in the **Open** step.

When you record a test for an application that requires credentials to be supplied in an authorization dialog, before running the test, you must edit the test to modify the URL by adding the credentials as follows:

https://username:password@url

For example, https://test_user:Mypassword!23@sample.com

Also, you must convert any special characters in the username and password by using the URL encoding format.

For example, you can convert Mypassword!23 to Mypassword%2123.

Drag-and-drop boxes

The **Drag** action that is recorded contains two values: From and To. The **From** tab in the **Object Identified By** section displays the identifier of the object that is being dragged, and the **To** tab displays the identifier of the new location for the dragged element.

File upload scenario

You can validate the file upload functionality for a web application by recording and playing back the file upload function. When you record a file upload function, a **Type** step is added with a variable to save the file path and file name. The uploaded file is listed under `blob\Others` in the test navigator panel.

Support for device emulation

You can use the device mode of browsers to record tests, and run tests or suites. Test Hub supports the following device profiles:

Device profile	Default browser
Phone (Small)	Edge
Phone (Medium)	Chrome
Phone (Large)	Edge
Phone (X-Large)	Safari (only on a Mac system)
Tablet (Small)	Edge
Tablet (Medium)	Chrome
Tablet (Large)	Edge
Tablet (X-Large)	Safari (only on a Mac system)

Recording a Web UI test

You can record screens and user actions on the web page of an application to create a Web UI test in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198.
- Ensured that your project contains an application.

About this task

When you choose to create a Web UI test by using the recording option, the installation status of the recording agent is displayed. Initially, when you are recording a test for the first time, the status is displayed as 'Not installed' with the **Download Installer** link to install the recorder agent. The `devops-test-runtime.msi` file is downloaded. You must then double-click the .msi file to install the agent. After the installation is complete, you must click **Verify** to continue with the recording process.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Perform one of the following options:
   - Click **Create asset > Create Test** on the **Test Editor** page.
   - Click the **Action menu** icon  in the test navigator panel to open the menu and click the **Test** option.
   - Right-click the application in the test navigator panel, select **New > Test**.

   Alternatively, you can create a test by clicking the **Create Test** icon  on the application **Details** page if there is an edit branch active.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 6 on page 212.
5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

     Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
6. **Optional:** Click **Open** in the browser notification that might be displayed for you to confirm the opening of DevOps Test Runtime.
   The browser notification is displayed only when the recording agent is already installed.
7. Perform the following steps in the **Details** tab of the **Create Test** dialog:

a. Select an application for which you want to create a test from the **Application** drop-down list.

   The application folder is selected as the default location for the test.

b. **Optional**: Select a folder if there are subfolders in **Location** to save the test assets that you are creating.

c. Enter a name for the test in **Asset Name**.

8. Click **Next**.
   The **Source** tab is displayed.
9. Select the **By recording** option to record a test in the **Select source** drop-down list.
   Test Hub checks whether the recording agent is installed on your computer.
10. Perform one of the following steps depending on the installation status of the agent:

If...	**Then...**
The agent is not installed and the screen displays the following options: ◦ Windows (64-bit .msi) ◦ Mac OS (64-bit .zip) ◦ Debian/Ubuntu (64-bit .deb) ◦ RHEL/Fedora/OpenSUSE (64-bit .rpm)	Perform the following steps to install the agent on Windows: a. Click **Download** to download the recording agent installer. The `devops-test-runtime.msi` file is downloaded. b. Double-click the `.msi` file to install the agent. The installation window with the progress bar is displayed. c. Click **Proceed** to accept the download of the proprietary certificate if prompted. d. Click **Verify** to confirm the agent installation. A browser notification might get displayed for you to confirm the opening of the agent. You must click **Open**. The additional options for recording are displayed. Go to step 11 on page 216.
	Perform the following steps to install the agent on Mac: a. Click **Download** to download the recording agent installer. The `devops-test-runtime.zip` file is downloaded. b. Extract the `.zip` file: unzip devops-test-runtime.zip c. Quarantine the installer: xattr -dr com.apple.quarantine ./* d. Run the installer: ./install.sh

If...	Then...
	e. Click **Verify** to confirm the agent installation. The additional options for recording are displayed. Go to step 11 on page 216.
	Perform the following steps to install the agent on Ubuntu: a. Click **Download** to download the recording agent installer. The `devops-test-runtime_12.0.0_amd64.deb` file is downloaded. b. Navigate to the directory where the installer is downloaded: cd <download-directory> c. Run the installer: sudo apt install ./devops-test-runtime_12.0.0_amd64.deb d. Click **Verify** to confirm the agent installation. The additional options for recording are displayed. Go to step 11 on page 216.
	Perform the following steps to install the agent on Linux: a. Click **Download** to download the recording agent installer. The `devops-test-runtime-12.0.0-1.x86_64.rpm` file is downloaded. b. Navigate to the directory where the installer is downloaded: cd <download-directory> c. Run the following command: INGRESS_DOMAIN=<domain_name> dnf localinstall ./devops-test-runtime-12.0.0-1.x86_64.rpm d. Click **Verify** to confirm the agent installation. The additional options for recording are displayed. Go to step 11 on page 216.
The agent is installed, but not updated, and the screen displays the following options: ◦ Windows (64-bit .msi) ◦ Mac OS (64-bit .zip)	Perform the following steps to update the agent on Windows: a. Click **Upgrade**. The installation screen with the progress bar is displayed. The latest DevOps Test Runtime is installed and configured.

If...	Then...
◦ Debian/Ubuntu (64-bit .deb) ◦ RHEL/Fedora/OpenSUSE (64-bit .rpm)	b. Click **Proceed** to accept the download of the proprietary certificate if prompted. The additional options for recording are displayed. Go to step 11 on page 216.
	Perform the following steps to install the agent on Mac: a. Click **Download** to download the recording agent installer. b. Uninstall the existing agent: rm -rf ~/Applications/devops-test-runtime.app c. Extract the downloaded `.zip` file: unzip devops-test-runtime.zip d. Quarantine the installer: xattr -dr com.apple.quarantine ./* e. Run the installer: ./install.sh f. Click **Verify** to confirm the agent installation. The additional options for recording are displayed. Go to step 11 on page 216.
	Perform the following steps to install the agent on Ubuntu: a. Click **Download** to download the recording agent installer. b. Uninstall the existing agent: sudo apt remove devops-test-runtime c. Navigate to the directory where the installer is downloaded: cd <download-directory> d. Run the following command: sudo apt install ./devops-test-runtime_12.0.0_amd64.deb e. Click **Verify** to confirm the agent installation. The additional options for recording are displayed. Go to step 11 on page 216.
	Perform the following steps to install the agent on Linux: a. Click **Download** to download the recording agent installer. b. Uninstall the existing agent:

If...	Then...
	sudo yum remove devops-test-runtime c. Navigate to the directory where the installer is downloaded: cd <download-directory> d. Run the installer: sudo INGRESS_DOMAIN=<domain_name> dnf localinstall ./devops-test-runtime-12.0.0-1.x86_64.rpm e. Click **Verify** to confirm the agent installation. The additional options for recording are displayed. Go to step 11 on page 216.
The agent is installed.	Go to step 11 on page 216.

11. Select a browser from the **Browser** drop-down list.
    The following browsers are listed:

Operating system	Browsers
Windows, RHEL, or Ubuntu	Chrome, Edge, and Firefox
Mac OS	Chrome, Edge, Firefox, and Safari

Also, the following device profiles are listed:

Device profile	Default browser
Phone (Small)	Edge
Phone (Medium)	Chrome
Phone (Large)	Edge
Phone (X-Large)	Safari (only on a Mac system)
Tablet (Small)	Edge
Tablet (Medium)	Chrome
Tablet (Large)	Edge
Tablet (X-Large)	Safari (only on a Mac system)



Note: If you selected the Edge or Firefox browser on a Mac system, you might see the following error: `The recorder encountered a problem: No compatible WebDriver was found or the browser is already running.`

To resolve this issue, you must check whether the `~/.cache/selenium` folder is password protected, and then change the access permission to make it a regular folder.

12. Select **UI** from the **Interactions** drop-down list.
13. Click **Record**.
    A new browser window is displayed, and the application that you selected for recording opens. If you selected Chrome, Edge, or Firefox, the browser opens in a temporary profile with no cache or cookies. For Safari, when the browser opens, you must manually clear the cache and cookies as required.

    Note: If you do not want to continue with the application that is open, you can type or paste the required URL to record the test. Test Hub inserts the **Open** step to capture the new URL that you used for recording the test.

14. Click the UI controls and perform user actions in the application to record test steps.
    You must allow some delay of a few seconds between two user actions to ensure that the UI elements are recognized and registered properly.
15. Close the browser to stop the recording.
    The '`The asset is now ready for your review`' message is displayed in the **Create Asset** dialog.
16. Click **Done**.
    The newly recorded test is displayed in **Test Editor**. The test is displayed in the **Visual** view, which is the default view. You can switch to the **Code** view to view all the step details.

Results

You have recorded a Web UI test.

What to do next

You can perform the following tasks:

- Run the test and view the results. See Running a Web UI test on page 228.
- Edit the test in Test Editor. See Editing a Web UI test on page 236.
- Create variable data for the test. See Test variables on page 311.

Related information

Failed to install DevOps Test Runtime on Windows 11 25H2 on page 1282

Failed to load the extension in Google Chrome while recording on page 1283

Creating a Web UI test by using the empty test option

You can create a Web UI test by selecting the empty test option and then adding the steps and elements either through the catalog in the Visual view or through the written code in the Code view.

Before you begin

You must have completed the following tasks:

- Ensured that you are a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198.
- Ensured that your project contains an application.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Perform one of the following options:
   - Click **Create asset > Create Test** on the **Test Editor** page.
   - Click the **Action menu** icon  in the test navigator panel to open the menu and click the **Test** option.
   - Right-click the application in the test navigator panel and select **New > Test**.

   Alternatively, you can create a test by clicking the **Create Test**  icon on the application **Details** page if there is an edit branch active.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 6 on page 218.
5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

     Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
6. **Optional:** Click **Open** in the browser notification that might be displayed for you to confirm the opening of DevOps Test Runtime.
   The browser notification is displayed only when the recording agent is already installed.
7. Perform the following steps in the **Details** tab of the **Create Test** dialog:

a. Select an application for which you want to create a test from the **Application** drop-down list.

   The application folder is selected as the default location for the test.

b. **Optional**: Select a folder if there are subfolders in **Location** to save the test assets that you are creating.

c. Enter a name for the test in **Asset Name**.

8. Click **Next**.
   The **Source** tab is displayed.
9. Select the **Create an empty test** option in the **Select source** drop-down list.
10. Click **Done**.
    An empty test is displayed in **Test Editor**. The test is displayed in the **Visual** view on the **Test Editor** page. You can also switch to the **Code** view.
11. Choose one of the following options:
    - Add your code to build the test in the **Code** view. Go to step 19 on page 219.
    - Add steps in the **Visual** view. Go to step 12 on page 219.



Note: You can also copy the steps from an existing test and paste them in the empty test by right-clicking the **Configure settings and data** step.

12. Drag and drop an element from one of the following catalogs to the **Test Editor** page:
    - **General**
    - **UI**
    - **Flow**

    A test step is added.
13. Click the step to add the details.
    The step details pane is displayed and the Smartshot pane does not display any image.
14. Drag and drop a `.jpeg` file to the Smartshots pane from the **shot > Smartshots** folder under the selected application.
15. Select an option from the **UI object** drop-down list.
16. Select a user action in the **Action** drop-down list or enter a value in the **Value** field depending on the type of step you have added from the catalog.
17. Click the **click here to add identifier** link and then in the **Object Identified By** section, specify the condition and value that identifies the UI object.

    Alternatively, you can drag-drop a property that identifies the object from the **Properties** list.
18. **Optional:** Click the **Click here to add relative** to link to define the location and name of the related object.
19. Click the **Save** icon  to save the modifications.

Results

You have created a Web UI test by using the empty test option.

What to do next

You can perform the following tasks:

- Run the test and view the results. See Running a Web UI test on page 228.
- Edit the test in Test Editor. See Editing a Web UI test on page 236.
- Create variable data for the test. See Test variables on page 311.

Related information

New test steps from the General catalog on page 242

New steps for UI-related actions and user actions in a test from the UI catalog on page 252

Adding conditional logic and redefining the flow in tests from the Flow catalog on page 260

Test creation with Test Genie

Test Genie is an AI-powered capability in HCL DevOps Test Hub (Test Hub) that automatically translates your natural language instructions into usable test scripts for web applications.

Traditionally, creating automated tests requires manually recording interactions against a live application or writing a test code. Test Genie accelerates this process so that you can define test steps by using plain text prompts. You provide a descriptive prompt, such as adding specific items to a shopping cart and proceeding to checkout, and the AI generates the corresponding test assets.

To improve the accuracy of the generated test steps, Test Genie uses the application context. When you initiate a generation request, the system passes your natural language steps along with the application URL and any application descriptions to the AI model. This context helps the AI understand the target interface and correctly map your instructions to the underlying web elements.

Supported AI providers

Disclaimer:

This release supports Anthropic in HCL DevOps Test Hub (Test Hub) as a Tech Preview. The Tech Preview is intended for you to view the capability of Anthropic for test generation offered by HCL Test Hub, and to provide your feedback to the product team. You are permitted to use the information only for evaluation purposes and not for use in a production environment. HCL provides the information without obligation of support and "as is" without warranty of any kind.

To use Test Genie, you must configure at least one AI provider in the **Infrastructure > Cloud Credentials** page of Test Hub. Test Genie supports the following AI providers:

- Anthropic - Tech Preview
- OpenAI

- GitHub Copilot

  Restriction: Only one test generation session through GitHub Copilot can be active at a time across the entire Test Hub server. Any additional Copilot requests are queued and wait until the current session completes.

For optimal results, you can use OpenAI with the `GPT-5.4-mini` model.

Capabilities and limitations

Test Genie streamlines the initial creation of test scripts, but it operates within specific boundaries regarding test validation and test modification. The following points describe the scope of Test Genie:

- **Generative scope:** Test Genie generates the navigational and transactional steps of a test, such as button clicks, text entry action, select actions from drop-down controls, drag-and-drop interactions, and navigation between pages. It also provides test generation details, including the model used and token usage metrics.
- **Iterative modifications:** You can use AI prompts to modify or append steps to a test that Test Genie has already generated. To modify the AI-generated steps, you must submit a revised prompt and then generate the test again.
- **Verification points:** The AI does not automatically generate or insert verification points into the test script. You must provide prompts to add verification points at specific points if you want to validate application behavior and data.
- **Datasets:** Test Genie does not create datasets or automatically link existing dataset files to the generated test. You must configure datasets manually.
- **Error handling and messages:** Test Genie provides clear error messages for troubleshooting during execution and creation:
  - **Missing AI provider:** Displays a validation message when you do not have an AI provider (such as OpenAI or Anthropic) configured in the cloud credentials settings.
  - **Missing application URL:** Prompts when the target application URL context is omitted.
  - **Generation failures and stop reasons:** Provides informative error messages and clear stop reasons when Test Genie cannot navigate the application, cannot complete the next intended action, or when test generation fails or becomes incomplete. Test Genie can indicate failures that are caused by timeouts, invalid page structures, provider service interruptions, and provider responses such as 4xx client errors or 5xx server errors. These details help you identify whether you must refine the prompt, update the application context, modify the generated steps, or retry the request.

Session limits for AI generation

In Test Hub, you can configure session limits as **Advanced** configuration values when you add a cloud credential for an AI provider in **Cloud Credentials**. These values are saved with the selected provider credential and are applied whenever Test Genie starts a suite or test generation session by using that credential.

The three limits are **Max tokens per session**, **Max interactions per session**, and **Max duration per session (in seconds)**. During test generation, Test Hub evaluates all three limits. If any configured limit is reached, the active AI conversation session ends, and you must start a new session to continue the test generation.

These session limits are optional for OpenAI and Anthropic credentials and are not applicable for GitHub Copilot credentials. By configuring these values, you can control response depth and generation time with platform resource usage for large or complex suite-generation workloads.

Creating a Web UI test with Test Genie

You can use the AI-powered Test Genie to generate test steps for a web application based on your natural language instructions.

Before you begin

You must have completed the following tasks:

- Ensured that you are a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198.
- Ensured that your project contains an application.
- Configured at least one AI provider in the **Cloud Credentials** page. See Adding cloud credentials of AI providers on page 596.

About this task

Test Genie accelerates test creation by translating your written instructions into a test script. The AI provider uses the application URL and the details in the application description to improve the accuracy of the generated steps.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Perform one of the following options:
   - Click **Create asset > Create Test** on the **Test Editor** page.
   - Click the **Action menu** icon  in the test navigator panel to open the menu and click the **Test** option.
   - Right-click the application in the test navigator panel and select **New > Test**.

   Alternatively, you can create a test by clicking the **Create Test**  icon on the application **Details** page if there is an edit branch active.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 6 on page 222.
5. Select an existing edit branch or create a new edit branch.
6. Perform the following steps in the **Details** tab of the **Create Test** dialog:

a. Select an application for which you want to create a test from the **Application** drop-down list.

   The application folder is selected as the default location for the test.

b. **Optional**: Select a folder if there are subfolders in **Location** to save the test assets that you are creating.

c. Enter a name for the test in **Asset Name**.

7. Click **Next**.
   The **Source** tab is displayed.
8. Select the **Create an empty test** option in the **Select source** drop-down list.
9. Enter a set of natural language instructions for your test in the **Steps** text box.
   For example, for testing an e-commerce website, you can enter:

```
Add three sports category items to the cart

Remove one item during the checkout process

Log in with the credentials

Proceed to order
```

10. Click **Done**.
    An empty test is displayed in **Test Editor**.
11. Click the **Test Genie** tab.
    The instructions you entered during the test creation are displayed in the **Steps** text box under the **Test Genie** tab.
12. Review, modify, or add to your natural language instructions in the text box.
13. Click **Generate**.
    The **Generate Test** dialog is displayed.
14. Perform the following steps in the dialog:
    a. Enter or modify the web address of the application under test in the **Start URL** field as required.
    b. Select an AI provider from the configured **AI Provider** list.
15. Click **Generate**.
    The test generation process starts.
16. Perform one of the following steps to view the AI interactions:
    - Click the **AI Interactions** tab at the bottom of the screen.
    - Click the **Results** tab and the perform the following steps:

      a. Click the **Show in progress page** icon  .

         The **Progress** page is displayed. If the page contains many jobs, you must create a filter with the rule **AI Assisted Generation** to filter for AI-assisted jobs.

      b. Select **View AI interactions** from the listed actions in the **Actions** column of the AI-assisted job.

         A new tab opens with the console output. You can view the AI interactions and session logs on the progress page by refreshing the view.

      c. Switch to the test editor tab after the execution job is complete in the **Progress** page.

The **Reload** button is displayed with the following warning: `A newer version of this resource is available.`

17. Click **Reload**.
    The AI-generated test steps are displayed.

Results

You have created a Web UI test with Test Genie.

What to do next

You can perform the following steps:

- You can view the execution results, the model used, and the token usage details in the **Test Genie** tab.
- Play back the generated test. See Running a Web UI test on page 228 or Running server test assets on page 808.

Related information

Modification of Web UI tests on page 235

Viewing the progress of running test assets on page 926

Modifying a Web UI test created by using Test Genie

You can modify the natural language instructions that you provided to generate a test by using the AI-powered Test Genie. After modifying the instructions, you can regenerate the test to update it.

Before you begin

You must have completed the following tasks:

- Ensured that you are a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198.
- Ensured that you have created a Web UI test by using Test Genie. See Creating a Web UI test with Test Genie on page 222.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Double-click the `.yaml` file to open the test.
   The test steps are displayed on the **Test Editor** page.
5. Click the 

button displayed on the **Test Editor** page.

The **Edit** button appears only when no active edit branch is selected.

The **Edit branch** dialog is displayed.

If the **Edit branch** dialog is not displayed, then go to step 7 on page 225.

6. Select an existing edit branch or create a new edit branch.
7. Click the **Test Genie** tab.
   The instructions you entered during the test creation are displayed in the **Steps** text box under the **Test Genie** tab.
8. Review, modify, or add to your natural language instructions in the text box.
9. Click **Generate**.
   The **Generate Test** dialog is displayed.
10. Perform the following steps in the dialog:
    a. Enter or modify the web address of the application under test in the **Start URL** field as required.
    b. Modify the AI provider from the configured **AI Provider** list if required.
11. Perform one of the steps as required:
    - If you want to first clear the steps that are already generated, retain the selection in the **Reset script before generation** checkbox.
    - If you want to append the new steps after the last step in the generated test, clear the **Reset script before generation** checkbox.
12. Click **Generate**.
    The test generation process starts.
13. Click the **Results** tab and then click the **Show in progress page** icon  .
    The **Progress** page is displayed. If the page contains many jobs, you must create a filter with the rule **AI Assisted Generation** to filter for AI-assisted jobs.
14. Select **View AI interactions** from the listed actions in the **Actions** column of the AI-assisted job.
    A new tab opens with the console output. You can view the AI interactions and session logs on the progress page by refreshing the view.
15. Switch to the test editor tab after the execution job is complete in the **Progress** page.
    The **Reload** button is displayed with the following warning: `A newer version of this resource is available.`
16. Click **Reload**.
    The modified test steps are displayed.

Results

You have modified a test that was created by using Test Genie.

What to do next

You can perform the following steps:

- You can view the execution results, the model used, and the token usage details in the **Test Genie** tab.
- Play back the generated test. See Running a Web UI test on page 228 or Running server test assets on page 808.

Related information

Modification of Web UI tests on page 235

Viewing the progress of running test assets on page 926

Creating a test by using existing steps of a test with the personal agent

You can create a new test by adding additional steps to an existing Web UI test. With the help of a personal agent, you can play back the existing test in a live browser and then continue to record the additional steps for the test. The newly recorded actions are appended to the test after the existing steps.

Before you begin

You must have completed the following tasks:

- Ensured that you are a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198.
- Ensured that your project contains an application and at least one existing test.
- Installed and started the personal agent. See Installing and starting the personal agent on page 567.

About this task

When you create a new test by using the **Record after existing test** option, the existing test is used as the starting point. The personal agent plays back the selected test in a live browser, and then you can record the additional steps.

When you initiate the recording, depending on the scenario, one of the following error messages might be displayed:

Scenario	Error message
You closed the browser without recording any new step.	No new steps were recorded. At least one step is required.
The playback of the selected test failed.	The recorder encountered a problem: The selected test did not pass during playback.
The selected test does not contain any **Open** step to launch the browser.	The recorder encountered a problem: The selected test does not contain any browser launch steps.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Perform one of the following actions:

- Click **Create asset > Create Test** on the **Test Editor** page.
- Click the **Action menu** icon in the test navigator panel and click **Test**.
- Right-click the application in the test navigator panel and select **New > Test**.

The **Edit branch** dialog is displayed.

5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
6. Select an application for which you want to create a test from the **Application** drop-down list.
7. Enter a name for the test in **Asset Name**.
8. Click **Next**.
   The **Source** tab is displayed.
9. Select **Record after existing test** from the **Select source** drop-down list.
   The file tree selector is displayed. The list displays the existing test assets in the project.
10. Select an existing test that you want to use as the starting point.
    The status of the personal agent is displayed in the **Agent** section. If the agent is installed and available for playback and recording, then the **Record** button is displayed.
11. Select a browser from the **Browser** drop-down list and then select **UI** from the **Interactions** drop-down list.
12. Click **Record**.
    The selected test is replayed by using the personal agent in the selected browser. The progress bar shows the playback while the agent continues the playback. You must wait until the playback is complete.

    After the playback is complete, the status changes to `Interact with the browser session and close it once done`. The browser is live so that you can continue with the new steps.
13. Record the additional steps in the browser.
14. Close the browser when you finish recording.
    The progress bar displays `The asset is now ready for your review` message, and the **Done** button is enabled.
15. Click **Done**.
    The new test opens in the editor with the selected existing test listed as a run module. The newly recorded steps are added after the original steps.

Results

You have created a new test by using an existing test as the starting point and then recorded additional steps by using the personal agent.

What to do next

You can perform the following tasks:

- Run the updated test and view the results. See Running a Web UI test on page 228.
- Edit the test in Test Editor. See Editing a Web UI test on page 236.

Related information

Installing and starting the personal agent on page 567

Stopping or unregistering the personal agent on page 571

Running a Web UI test

You can run a Web UI test in HCL DevOps Test Hub (Test Hub) after you record the test, and to verify that the web application performs as expected.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Ensured that your project contains a Web UI test in the test navigator panel.

About this task

When you play back the test, the test runs in the browser that is used to record the test. The progress bar shows the progress of the test, while the three color-coded boxes display the counts of passed steps, failed steps, and errors. You can view the step details of the first failed step or error step by clicking the respective color-coded boxes. As the test runs, you can view checkmarks for the steps that pass and crossmarks for the steps that fail. After the test run is complete, you can view the overall results on the **Results** page.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Click the `.yaml` file to open the test.
   The test steps are displayed on the **Test Editor** page.

5. Perform the following steps to specify the browser that you must use for the test run:
    a. Click the **Configure settings and data** step.

    The **Configure settings and data** panel is displayed.

    b. Select one of the following options from the **Browser** drop-down list:
        - Browser
        - Device profile
        - Device from the list of devices for the configured device cloud

    The browser that is used to record the test is selected as the default browser and the browser specified here modifies the default browser.



Note: You must have accepted the license agreements for Google Chrome and Microsoft Edge browsers in the **Third-Party Agreements** section after navigating to **Account** > **Account Settings**. If the third-party agreements are not yet accepted, the list options display links so that you can navigate directly to the agreement and accept the terms.

6. Click the  icon, and then select the **Run on Default Cluster** option to begin the test run.

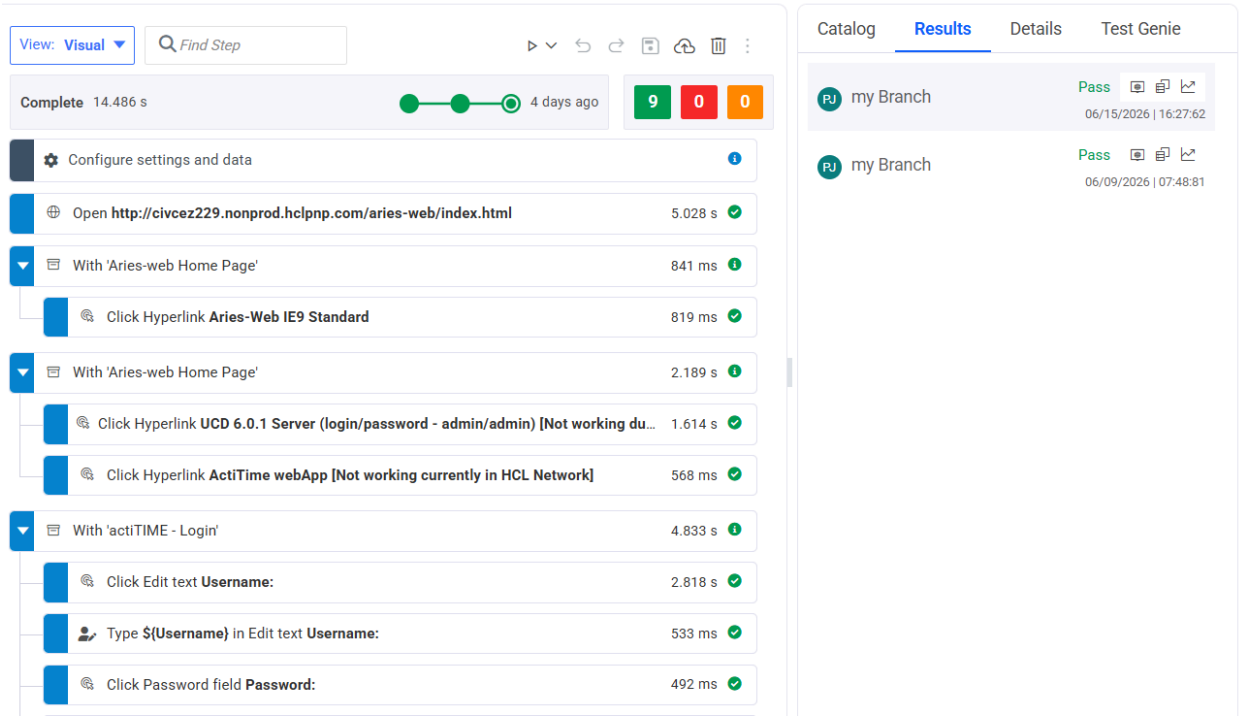
You can also select the **Run on Personal Agent** option if the personal agent is in the ready state.

You can view the following stages during the test run:

- **In transition**: The test setup is in progress.
- **Running**: The test run is active.
- **Complete**: The test run is completed. This section also displays the time taken for test run.
- During the **Running** phase, the following step result indicators are displayed with the time taken to process each step:
    - A checkmark  appears for the passed step.
    - An error mark  appears for the steps with an error.
    - A crossmark  appears for the failed step.
    - An info icon  appears for the **With** step that indicates a new screen or window.
- The **Results** tab displays the test as **In Progress** during the test run.
- After the test run is completed, the **Results** tab displays the test as **Pass** or **Fail**.

The **Result** page supports selecting individual items from the list. When a row is selected, the corresponding high-level information is displayed in the top banner of the Test Editor.

Figure 1. Running a Web UI test



7. Click to open the **Results** page, where you can view the results, reports, and logs of the completed test run.

Results

You have run a Web UI test.

What to do next

You can view the following information:

- The console output, which includes error messages and logs. See Accessing console output on page 230.
- The results, reports, and logs on the **Results** page after the test completes the run.
- The summary of the test run, which includes passed or failed status, count of passed or failed tests or test steps, and the verification points in the functional report. See Viewing a Functional Report on page 994.

Related information

Viewing the progress of running test assets on page 926

Accessing the console output

When you run tests, you can access the console output that provides information such as error messages, logs, and the test verdict. You can then use this output to troubleshoot issues by reviewing details, such as session IDs and error messages.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Ensure that your project contains a test.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Double-click the `.yaml` file to open the test.
   The test steps are displayed on the **Test Editor** page.
5. Click the **Console** tab at the bottom of the **Test Editor** page.

   Alternatively, you can click the expand icon  .

   The **Console** output view is displayed.
6. Review the console output to identify the error messages and logs.
7. Perform the following actions to manage the console output:

Action	Description
Search a keyword in the console output	Enter a keyword in the **Search console** field, and the corresponding word is highlighted in the console output.
Select all console content	Right-click the console output area and, then select the **Select All** option to select all content.
Copy the selected content	Right-click the selected content and choose **Copy** to copy the content.
Expand console output view	Click the up arrow  to expand the console output view for more details.
Restore console output view	Click the down arrow  to return the console output page to its original size.
Close console output view	Click the drop-down icon  to close the console output view.

Results

You have accessed the console output content.

What to do next

You can perform any of the following tasks:

- Decide on the actions that you might want to take based on the logs.
- Rerun the test asset.

Viewing step-level results and updating tests

You can view the step-level results of each step after running a Web UI test. You can edit and update the steps in test scripts by replacing smartshots.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a Project Owner or Tester in the project. See Managing access to the server projects on page 1220.
- Ensured that you have run the Web UI test.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.

About this task

HCL DevOps Test Hub (Test Hub) collects the UI data during the test run and smartshots that are captured during the test run can be used to update the failed step. If an object is not identified during the test run, then the step fails. The test steps are indicated in different colors to specify the status of each test step. For the failed step, you can replace the smartshot with the latest one from the last test run. Also, you can update a step by replacing the smartshot and selecting a target element.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Click Tests to view the list of tests recorded for the selected application.
5. Click the `.yaml` file to open the test.
   The test steps are displayed on the **Test Editor** page.

   Note: If you have just run the test and the edit branch is not published, then you can click the color-coded step result indicators that are displayed for each step to view the result details.

6. Click any test step for which you want to view the result details.
   The step details pane is displayed.

7. Click the **Show Result** toggle button in the upper-right corner of the step details pane to view the outcome of the step.

   The **Result** section displays the following details:

   - Step run status, such as **Completed step** or **Failure**, along with the **Duration** is displayed. The failure reason is displayed when the step fails.
   - **Start Time** and **End Time**, which you can view by clicking the **Expand view** icon  .
   - Overall step outcome, such as **PASS** or **FAIL**.

   Note: The **Show Result** option can be enabled only if the test has already been run. You can disable this option and navigate to the **Test Editor** page.

8. Click the **Fullscreen** icon  to view the details with smartshot and other configuration details in full-screen mode for better visibility.

   Note: The **Fullscreen** icon is displayed only after you click the **Expand view** icon.

9. Click the **Exit fullscreen** icon  to close the full-screen view.

   Note: The **Exit fullscreen** icon is displayed only when fullscreen mode is enabled.

10. Click the **Previous Step** icon  and the **Next Step** icon  to navigate between the previous and next test steps.
11. Perform the following actions to update the step by replacing the smartshot for the step:
    You can update both passed and failed steps.

    a. Click the **Replace Smartshot** icon  that is displayed if the edit branch is active.

       The **Replace Smartshot** pop-up window is displayed with a list of smartshots under two different sections: **Result Smartshots** and **Application Smartshots**. The **Result Smartshots** displays the smartshots from the last test run, and the **Application Smartshots** displays all the available smartshots captured earlier for the application. You can hover over an image and click the zoom icon  or  to view the enlarged image. The image is displayed with the controls to rotate right, rotate left, zoom in, and zoom out.

Note: If the test is run as part of the test suite, then the **Replace Smartshot** icon is disabled even when the edit branch is active. You can click the Open button that is displayed to open the test for editing.

b. Select a smartshot, and then click **Replace** if you want to replace the smartshot for the step.

12. Click the **Save** icon to save the modifications.

Results

You have viewed the step-level result and updated the test.

What to do next

You can perform the following tasks:

- Run the test and view the results. See Running a Web UI test on page 228.
- View the Functional Report to verify the test steps with the variable assignment. See Viewing a Functional Report on page 994.

Related information

Modification of Web UI tests on page 235

Stopping a test run

You can stop a test run when issues occur or if you decide to stop the run before completion.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Initiated a test run from the **Test Editor** page.

About this task

You can stop a running test from the **Test Editor** page if the test run is in the **In transition** or **Running** state.

1. Go to the **Test Editor** page.
2. Click the **Stop playback** icon .
   The status of the test is displayed as **Inconclusive** on the **Results** tab and **Stopped by User** on the progress bar.

Results

You have stopped a test run.

What to do next

You can re-initiate the test run from the **Test Editor** page.

Modification of Web UI tests

With **Test Editor**, you can view and customize a Web UI test that is created in Test Hub.

The **Test Editor** page displays the test scripts. The test script lists user actions and UI elements recorded as chronological sequence of events during the recording session. Actions are presented in the natural language and you can modify the test manually.

The **Test Editor** also displays the following sections:

- **Catalog** tab: Displays the catalog elements that can be dragged and dropped to an existing script as steps. The catalog elements are categorized as **General**, **UI**, and **Flow**. For more information on the usage of catalog elements, see New test steps from the General catalog on page 242, New steps for UI-related actions and user actions in a test from the UI catalog on page 252, and Adding conditional logic and redefining the flow in tests from the Flow catalog on page 260.

  Note: If you want to drag and drop a catalog element to a test when you use Chrome or Edge on a Ubuntu virtual machine, you must directly connect to the machine instead of connecting through a desktop sharing system such as VNC.

- **Results** tab: Displays the status of the test run when it is in progress, and the verdict of the previous test runs with a link to the **Results** page to view reports.
- **Variables** pane: Displays the defined variables and provides options to add or edit variables.

When you click a test step, the step details pane that is expandable is displayed with different sections as follows:

- **Smartshot**: Displays screen captures with UI objects from the web application under test. You can select a UI object, and add or modify an action related to the object. This section provides **Zoom in**, **Zoom out**, **Pan**, **Fit to available space**, and **Replace Smartshot** controls to work with images.

  The **Zoom in** icon increases the magnification of the smartshot image by 10%. The **Zoom out** icon decreases the magnification of the smartshot image by 10%. By using the **Focus on object** icon, you can move the smartshot within the available viewing area by dragging the image. While the **Focus on object** mode is enabled, smartshot interactions such as drag-and-drop and context menu actions are disabled.

  The **Full screen**

- **Details**: Displays the object type and the associated value or action depending on the step type. This section also displays values that identify the object. You can use any of the following conditions while defining the target to identify the object: Includes, Ends with, Matches regex, Starts with, Equals, Doesn't equal, Doesn't match regex, Doesn't include, Doesn't start with, Doesn't end with.

  For the XPath that identifies the object, you can also provide a relative XPath when you select **xpath** in the drop-down list of properties in the **Object Identified** section. The element for which you provided an XPath is highlighted in the **Smartshot** view.

  You can insert variables into supported input fields by clicking the **Insert Variable** icon  and selecting a variable from the list. The selected variable is inserted at the current cursor position. If you select text before selecting a variable, the selected text is replaced with the variable.

  For conditional statements, when you select a variable from the list, the variable is inserted without the `${}` notation. When you concatenate a variable with a string, the variable is inserted in the `${variable_name}` format.
- **Advanced**: Displays options to define a timeout for the step and disable the step.
- **Elements**: Displays the elements available on the web page that are captured in the smartshot.
- **Properties**: Displays the properties such as aria label, class, content, exists, href, id, label, name, placeholder, tagName, type, value, and XPath of the object. The displayed properties are the properties that are captured while recording and they are required to identify the objects during the playback.

The step details pane displays information about the currently selected step and provides options to modify the step. See Editing a Web UI test on page 236.

Editing a Web UI test

You can edit a Web UI test to add new steps, remove the existing steps, or update the step details.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.

4. Click the application for which you recorded a test in the test navigator panel.
   The assets created for the application are displayed.
5. Click Tests to view the list of tests recorded for the selected application.
6. Double-click the `.yaml` file to open the test.
   The test steps are displayed on the **Test Editor** page.

   Note:

   The top-level steps and first-level child steps are automatically expanded to provide better visibility.

7. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 9 on page 237.
8. Select an existing edit branch or create a new edit branch.
9. Select a step to view the details.
   The step details pane is displayed. If the step has a user action, then the **Smartshot**, **UI Object** type, **Action**, and **Object Identified By** details are displayed.
10. Perform any of the following actions to modify the step details as required:

Action	Description
Edit the object and action for the step	Perform the following steps: a. Hover over the UI elements in the **Smartshot** view, right-click the required UI element, and then select **Use this element as a step target** to change the object assigned for the step. The values in the **Object Identified By** field are updated based on the selected UI object. You can also select a different UI object from the list if required. b. You can use the available filter options in the **Smartshot** view to filter the displayed UI objects: i. **Filter by HTML**: Displays the identified UI objects based on HTML elements. ii. **Filter by Window**: Displays the identified UI objects based on the application window.

Action	Description
	iii. **Filter by jQuery UI**: Displays supported jQuery UI controls identified during recording. Supported controls include Button, Radio Button, Checkbox, Menu, Menu Item, Date Picker, Tabs, and Accordion. iv. **Filter by Salesforce Lighting Components**: Displays supported Salesforce Lighting Components UI controls identified during recording. c. Modify the value from the **UI Object** drop-down list if required. d. Select a user action from the **Action** drop-down list. e. **Optional**: Click the **Click here to add relative to** link, specify the relative location in the **Location** field, and specify the relative object in the **UI Object** field. f. **Optional**: Click the **Click here to add alternative identifier** link, and select a target object from the list in the **Target 2** field. Alternatively, you can drag and drop an element from the **Smartshot** view.
Open the test that is embedded in the **Run** step	Right-click the **Run** step and click **Open**. The embedded test is displayed in **Test Editor**.
Select a step or a set of steps	Right-click a step and click **Select**. Checkboxes are displayed for each step to select the required steps. When you right-click a selected step, the **Extract Steps** and **Deselect All** options are displayed.
Disable the step	Click the **Advanced** tab and select the **Disable** checkbox. Alternatively, you can right-click the step and click **Disable**. The option changes to **Enable** for the disabled step.

Action	Description
Copy the step	Right-click the step and click **Copy**.
Paste the step	Right-click a step, and then click **Paste**. The copied step is pasted after the selected step.
Delete the step	Right-click the step, and then click **Delete**.

11. Click  to close the step details pane.
12. Click the **Save** icon  to save the modifications.

Results

You have edited a Web UI test.

What to do next

You can perform the following tasks:

- Run the test and view the results. See Running a Web UI test on page 228.
- Add additional steps from the catalog. See New test steps from the General catalog on page 242, New steps for UI-related actions and user actions in a test from the UI catalog on page 252, and Adding conditional logic and redefining the flow in tests from the Flow catalog on page 260.
- Create and use variables in the test. See Creating and assigning test variables on page 312.

Adding user actions in a test from smartshots

When you want to add user actions as steps in an existing Web UI test, you can add such steps from smartshots associated with the test. You can select a graphical element in the smartshot and add a user action for that element.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

About this task

The **Test Editor** page displays a hierarchical view of the steps in a test. The step details pane displays the object properties with the associated smartshot. You can add a step as a node and add further actions as sub-steps by selecting any of the widgets in the **Smartshot** view.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 240.
2. Select an existing branch or create a new branch.
3. Drag and drop one of the following element from the **UI** catalog in the **Catalog** tab after the required step.
   - Click
   - Select
   - Type
   - Verify
   - Assign

   A test step is added.
4. Click the newly added step.
   The step details pane is displayed.
5. Select one of the panes and perform the following actions:

Name of the pane	Action
Smartshot	Hover over a UI element in the **Smartshot** view, right-click, and then select one of the following options: ◦ **Use this element as a step target** ◦ **Create user action for this element**: **Click, Select, Type, Drag & Drop, Verify, Assign** Alternatively, you can drag and drop a UI element from the **Smartshot** view to the **Object Identified By** section.
Elements	Perform the following actions: a. Expand the list of elements in **Elements**. b. Select an element, right-click, and then select one of the following options: ▪ **Use this element as a step target** ▪ **Create user action for this element**: **Click, Select, Type, Drag & Drop, Verify, Assign**
Properties	Perform the following actions:

Name of the pane	Action
	a. Click any of the UI elements in the **Smartshot** view. b. Select a property from the **Properties** list, right-click, and then select one of the following options: ▪ **Add property to step target** ▪ **Create user action with this property**: **Click, Select, Type, Drag & Drop, Verify, Assign** If the target object is already defined, then you can use the **Replace property in step target** option to replace the identifier.

If you selected the **Use this element as a step target** option, the action or condition is assigned to the selected element. Else, if you selected the **Create user action for this element** option with one of the listed actions, a step is added just before the selected step.

The step details pane is displayed with the **Smartshot**, **UI Object** type, **Action**, and **Object Identified By** details. The values displayed are dependent on the object selected.

Alternatively, you can select the **Code** checkbox, and then enter a JavaScript code that returns the target object for the selected action.

6. **Optional:** Click the **Click here to add relative to** link, specify the relative location in the **Location** field, and then specify the relative object in the **UI Object** field.
7. **Optional:** Click the **Click here to add alternative identifier** link, and select a target object from the list in the **Target 2** field.
8. Click the **Save** icon  to save the modifications.

Results

You have added user actions as steps from smartshots in a Web UI test.

What to do next

You can perform the following tasks:

- Run the test and view the results. See Running a Web UI test on page 228.
- Add additional steps from the catalog. See New test steps from the General catalog on page 242, New steps for UI-related actions and user actions in a test from the UI catalog on page 252, and Adding conditional logic and redefining the flow in tests from the Flow catalog on page 260.
- Create and use variables in the test. See Creating and assigning test variables on page 312.

New test steps from the General catalog

When you want to add a log statement, validation, variable assignment, or pause the test for a specific duration, you can add the corresponding steps from the **General** catalog.

You can add the following steps from the **General** catalog in the **Catalog** tab to update the test:

Catalog element	Description	Reference
Assign	You can assign a value to an existing variable or a new variable.	Adding an Assign step on page 242
Log	You can insert a custom log statement in a test as a step. The inserted log statement is printed in the **Console** window and the Functional report. For example, you can add a custom statement to demarcate the test steps related to order generation, user creation, or report generation in an application or you can use a log statement to display the value of a variable.	Adding a Log step on page 245
Pause	You can add Pause as a step in the recorded test for a specified duration. You can specify the pause duration in milliseconds, seconds, minutes, or hours.	Adding a Wait step on page 246
Verify	You can verify the values assigned to a variable. You can compare the values of two variables.	Adding a Verify step on page 248

Adding an Assign step

You can use the **Assign** step to assign values from datasets to test variables or store values extracted from UI elements in variables during a test run.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test or suite.
- Ensured that the required variables have been created and are included in your Web UI test or suite.
- Ensured that your project contains a dataset.

About this task

You can use the **Assign** step to assign values to variables during a test run. You can assign values from datasets to support data-driven testing or extract values from UI elements and store them in variables. The assigned values can then be used in subsequent test steps to perform flexible data-driven testing across multiple scenarios

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 3 on page 243.
2. Select an existing branch or create a new branch.
3. Drag the **Assign** statement from the **Catalog > General** pane and drop it under a test step as required.
   The **Assign** step is inserted.

   Note:

   You can also use the **Assign** step within a **while** or **do-while** step, so that the **Assign** step retrieves the values from a dataset to process each row of the dataset iteratively and assign the retrieved values to the test variables.

4. Click the **Assign** step.

   The step details pane opens with the **Details** tab which was the default selection.
5. Select the required mode from the **Mode** list:
   - **UI**
   - **General**

   A confirmation dialog is displayed, indicating that switching modes clears the current step configuration.
6. Click **Yes**.

Note:

- The default mode depends on where the **Assign** step is added. When the **Assign** step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.
- You can switch between the available modes by selecting a different option from the **Mode** list.
- If you switch modes after configuring the step, select the required mode and then click **Yes** in the confirmation dialog. The existing configuration is cleared and the step opens in the selected mode.

7. You can perform the actions that are listed in the following table to configure assign during a test run:

Mode	Action
General	Perform the following steps: a. Select an existing dataset in the **Dataset** field. b. Click **+ Add Variable** to add a variable and assign the value from dataset. c. Select an existing variable from the **Select a variable to assign** drop down. **Note:** You can also select **Create variable** from the variable list to create a new variable and assign the value from the dataset. The variable is displayed in the **Variables** section. d. Select the column in the dataset from which you want to assign the values during test run. e. Click the **Close** icon to close the run details pane.
UI	Perform the following steps: a. Hover over a UI element in the **Smartshot** view, right-click, and then select **Use this element as a step target**. Alternatively, you can drag and drop a UI element from the **Smartshot** view to the **Object identified by** section. The **UI Object** type is identified. b. Enter a property name such as Content, Id, or XPath of the object that must be used for variable assignment in **UI Property**. If you entered the property name correctly, the value of the property is displayed under **Extracted Value**. c. Select an existing variable, or Select **Add new variable**, enter a name for the variable, and then click **Add**.

8. Click the **Save** icon to save the modifications.

Results

You can configured the **Assign** step to assign values to test variables during a test run. Based on the selected mode, the Assign step assigns values from a dataset to variables or stores values extracted from UI elements in variables.

What to do next

You can run the test in **Test Editor** and view the Functional Report to verify the test steps with the variable assignment. See Running a Web UI test on page 228.

Adding a Log step

You can insert a custom **Log** statement in a test as a step. The inserted **Log** statement is printed in the **Console** window and the Functional report. For example, you can add a custom statement to demarcate the test steps related to order generation, user creation, or report generation in an application or you can use a **Log** statement to display the value of a variable.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 245.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Log** element under any test step as required from the **General** catalog in the **Catalog** tab.
4. Click the **Log** step.
   The step details pane is displayed.
5. Enter a custom message in the **Log statement** text box.
   You can insert variables into a log statement by placing the cursor at the desired location of the log statement text box and selecting the **Insert Variable** icon  . The variable is inserted at the cursor location.
6. Click the **Save** icon  to save the modifications.

Results

You have added a custom message in the test by using the **Log** step from the **General** catalog.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a Wait step

You can add **Wait** as a step in the recorded test for a specified duration. You can specify the wait time in milliseconds, seconds, minutes, or hours. Adding a **Wait** step gives the application the required time to load the UI elements and update the screen.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 246.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Wait** element under any test step as required from the **General** catalog in the **Catalog** tab.
4. Click the **Wait** step.
   The step details pane is displayed.
5. Select the required mode from the **Mode** list:
   - **UI**
   - **General**

   A confirmation dialog is displayed, indicating that switching modes clears the current step configuration.
6. Click **Yes**.

Note:

- The default mode depends on where the **Wait** step is added. When the **Wait** step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.
- You can switch between the available modes by selecting a different option from the **Mode** list.
- If you switch modes after configuring the step, select the required mode and then click **Yes** in the confirmation dialog. The existing configuration is cleared and the step opens in the selected mode.

7. Perform the following steps to assign a variable to a test element:

Mode	Action
General: You can use the **General** mode to add wait time for test steps.	Perform the following steps: a. Specify the amount of time that the step must pause in **Duration**. b. Select the unit of time for the timeout in **Unit**. c. Alternatively, you can select the **Code** checkbox to enter the time and unit in the **Duration** text box. d. **Optional**: Enter a message to indicate the reason for the pause in the **Note** text box.
UI: In the **UI** mode, the test run pauses at the Wait step until the targeted UI element is identified.	Perform the following steps: a. Select an object by dragging and dropping an element from **Smartshot** view, an element from **Elements** list, or a property-value pair from the **Properties** list. Alternatively, you can select an object from the list in **UI Object**, and then select an identifier such as Class, Content, Label, XPath, or Id of the object from the list. If you follow this method, you must ensure that the object is highlighted in the **Smartshot** view after you enter the values. You can replace the value of identifier dynamically under the **Object Identified By** section by clicking the **Insert Variable** icon and selecting a variable from the list.

Mode	Action
	Note: The **Insert Variable** option is available only in **UI** mode. b. **Optional**: Click the **Click here to add relative to** link, specify the relative location in the **Location** field, and specify the relative object in the **UI Object** field. c. **Optional**: Click the **Click here to add alternative identifier** link, and select a target object from the list in the **Target 2** field. Alternatively, you can drag and drop an element from the **Smartshot** view. d. **Optional**: Select the **Override Timeout** checkbox in the **Advanced** tab and increase the timeout value in **Duration** for the object verification.

8. Click the **Save** icon to save the modifications.

Results

You have added a **Wait** step to pause the test for a specific duration.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a Verify step

You can use the **Verify** step to validate expected conditions during Web UI test run. The Verify step checks whether the application under test behaves as expected by comparing actual runtime values with expected values defined in the test. If the verification fails, the step is marked as failed in the test run results. After you add the **Verify** step from the **General** section of the catalog, you can choose the type of verification to perform by selecting a mode in the step details pane.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.

- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 249.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Verify** element under any test step as required from the **General** catalog in the **Catalog** tab.
4. Click the **Verify** step.
   The step details pane is displayed.
5. Select the required verification mode from the **Mode** list:
   - **UI**
   - **General**

   Note: The default mode depends on where the **Verify** step is added. When the **Verify** step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.

   A confirmation dialog is displayed, indicating that switching modes clears the current step configuration.
6. Click **Yes**.

   Note:
   - The default mode depends on where the step is added. When the step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.
   - You can switch between the available modes by selecting a different option from the **Mode** list.
   - If you switch modes after configuring the step, select the required mode and then click **Yes** in the confirmation dialog. The existing configuration is cleared and the step opens in the selected mode.

7. You can perform the actions that are listed in the following table to configure verification in a **Verify** step during a test run:

Mode	Action
General: You can use **General** mode to verify variables, expressions, and business logic conditions during a test run. This mode validates runtime data instead of UI elements.	Perform the following steps:

Mode	Action
	a. In the first column, click the **Insert Variable** icon  to select a variable from the list or enter the variable name. b. In the operator field, select an option to indicate the basis of comparison of the two operands. The available operators are as follows: ▪ Equals ▪ Not Equals ▪ Greater Than ▪ Greater Than Equals ▪ Less Than ▪ Less Than Equals ▪ Strict Equals ▪ Strict Not Equals ▪ Includes ▪ Not Includes c. In the third column, click the **Insert Variable** icon  to select a variable from the list or enter the variable name in the **Value** field. Additionally, you can add, duplicate or delete rows for verification. Also, you can switch to the JS code view to create or modify the comparison statement.
UI: You can use **UI** mode to verify UI elements and their properties in the application under test during a test run. This mode ensures that UI components are displayed correctly and match expected values.	Perform the following steps: a. Select an object to be verified by dragging and dropping an element from **Smartshot** view, an element from **Elements** list, or a property-value pair from the **Properties** list. Alternatively, you can select an object to be verified from the list in **UI Object**, and then select an identifier such as Class, Content, Label, XPath, or Id of the object from the list. If you follow this method, you must ensure that the object is highlighted in the **Smartshot** view after you enter the values.

Mode	Action
	b. Click click here to add a row to provide values to be verified under **Verification Points**. c. Enter the property name and value to be verified in the newly added row. You can replace the value of verification point dynamically by clicking the **Insert Variable** icon  and selecting a variable from the list.  **Note:** If you select the text before choosing a variable, the selected text is replaced with the variable. If you do not select any text, the variable is inserted at the current cursor position. d. **Optional**: Under the **Object Identified By** section, you can replace the value of identifier dynamically by clicking the **Insert Variable** icon  and selecting a variable from the list.  **Note:** If you select the text before choosing a variable, the selected text is replaced with the variable. If you do not select any text, the variable is inserted at the current cursor position. e. **Optional**: Click **Click here to add relative to**, specify the relative location in the **Location** field, and specify the relative object in the **UI Object** field. f. **Optional**: Click **Click here to add alternative identifier**, and select a target object from the list in the **Target 2** field.

Mode	Action
	Alternatively, you can drag and drop an element from the **Smartshot** view. g. **Optional**: Select the **Override Timeout** checkbox in the **Advanced** tab and adjust the timeout value in **Duration** for the object verification.

8. Click the **Save** icon to save the modifications.

Results

You have added a **Verify** step to validate conditions in a Web UI test during a test run.

What to do next

You can run the test in **Test Editor** and view the Functional Report to verify the test steps with the variable assignment. See Running a Web UI test on page 228.

New steps for UI-related actions and user actions in a test from the UI catalog

When you want to add UI-related actions and user actions to update a test, you can add those steps from the UI catalog. UI-related actions are actions such as opening a new page and adding a new section in the test. User actions are actions such as click, select, type, and key press for user inputs.

You can add the following UI-related actions and user actions from the **UI** catalog in the **Catalog** tab to update the test:

Catalog element	Description	Reference
With	You can add a container for a new page, section, or tab on the web page.	Adding a With step on page 253
Open	You can add a step to open a URL.	Adding an Open step on page 254
Click	You can add a click action as a step.	Adding a Click step on page 255
Select	You can add a select action as a step.	Adding a Select step on page 256
Type	You can add a type action for entering the text in a text box as a step.	Adding a Type step on page 257
Press	You can add a press action for window controls or supported keys as a step. For example, you can add a step to minimize a window or press Ctrl + M.	Adding a Press step on page 259

When you insert a step after a **With** or **Open** step by using the catalog elements such as **Assign**, **Click**, **Press**, **Select**, **Type**, and **Verify** from the **UI** catalog, the step is added without a smartshot. You must drag a smartshot to the pane or click the **click here to add smartshot** link to select a smartshot from the **Add Smartshot** pop-up. You also can hover over an image

and click the zoom icon  to view the enlarged image. The image displays with the controls to rotate right, rotate left, zoom in, and zoom out.

Adding a With step

You can add a container for a new page, section, or tab on the web page by using the **With** step from the **UI** catalog.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 253.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **With** element under any test step as required from the **UI** catalog in the **Catalog** tab.
4. Click the **With** step.
   The step details pane is displayed.
5. Perform the following steps in the step details pane:
   a. Enter a description of the container in the **Note** text box.
   b. Enter a title to identify the container in the **Context** text box.
6. Click the **Save** icon  to save the modifications.

Results

You have added a **With** step from the **UI** catalog to add user actions in a Web UI test.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding an Open step

You can add a step to open a URL that is required for any test step by using the **Open** step from the **UI** catalog.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 254.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Open** element under any test step as required from the **UI** catalog in the **Catalog** tab.
4. Click the **Open** step.
   The step details pane is displayed.
5. Perform the following steps in the **Details** tab of the step details pane:
   a. Enter the URL of the page to be opened in the **URL** text box.

      You can insert variables into a **URL** text box by placing the cursor at the desired location of the URL text box and selecting the **Insert Variable** icon The variable is inserted at the cursor location.

   b. Select **Attach** to open the URL in a new tab or window.
   c. Enter a title to identify the container in the **Context** text box.
   d. Enter a description about the container in the **Note** text box.
6. Click the **Advanced** tab.
7. **Optional:** Select the **Disable** checkbox if you want to disable the step.
   The selected step is greyed out.
8. **Optional:** Perform the following steps to set the timeout value:
   a. Select the **Override Timeout** checkbox.
   b. Specify the duration of time that Test Hub must wait for the step to be completed in **Duration**.

c. Select the unit of time for the timeout in **Unit**.

Alternatively, you can select the **Code** checkbox and enter the timeout and unit in **Duration**.

9. Close the step details pane.
10. Click the **Save** icon to save the modifications.

Results

You have added an **Open** step from the **UI** catalog to add user actions in a Web UI test.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a Click step

You can add a click action as a step by using the **Click** step from the **UI** catalog.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 255.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Click** element under any test step as required from the **UI** catalog in the **Catalog** tab.
4. Click the **Click** step.
   The step details pane is displayed.
5. Perform the following steps in the step details pane:
   a. Right-click a UI object in the **Smartshot** pane and select **Use this element as step target**.

      The **UI Object** type, **Click** action, and values for **Object identified by** are displayed. Optionally, you can modify the values that identifies the object.

b. **Optional:** You can replace the value of the object with the value of a variable under the **Object Identified By** section by clicking the **Insert Variable** icon and selecting a variable from the list.

Note: If you select the text before choosing a variable, the selected text is replaced with the variable. If you do not select any text, the variable is inserted at the current cursor position.

c. **Optional**: Select the **Click here to add alternative identifier** link to select one more object that works as an alternative target if the first object was not found.

d. Close the step details pane. The test step is updated with the variable that you chose.

6. Click the **Save** icon to save the modifications.

Results

You have added a **Click** step from the **UI** catalog to add user actions in a Web UI test.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a Select step

You can add a select action as a step by using the **Select** step from the **UI** catalog.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 256.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Select** element under any test step as required from the **UI** catalog in the **Catalog** tab.

4. Click the **Select** step.
   The step details pane is displayed.
5. Perform the following steps in the step details pane:
   a. Right-click a UI object in the **Smartshot** pane and select **Use this element as step target**.

      The **UI Object** type, **Value**, and values for **Object identified by** are displayed. Optionally, you can modify the values that identifies the object by using one of the following methods:
      - Enter the values in the **Object identified by** fields.
      - Drag and drop an element from the **Elements** section or a Property-Value pair from the **Properties** section.

   b. **Optional**: Under the **Target 1** section, click **click here to add a row** to add another UI element.

      The **UI Object** type, **Value**, and values for **Object identified by** are displayed.

      You can replace the value of the object with the value of a variable under the **Object Identified By** section by clicking the **Insert Variable** icon and selecting a variable from the list.

   c. **Optional**: Click the **click here to add alternative identifier** link to select one more object that works as an alternative target if the first object was not found.
6. Click the **Save** icon to save the modifications.

Results

You have added a **Select** step from the **UI** catalog to add user actions in a Web UI test.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a Type step

You can add a type action for entering the text in a text box as a step by using the **Type** step from the **UI** catalog.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 258.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Type** element under any test step as required from the **UI** catalog in the **Catalog** tab.
4. Click the **Type** step.
   The step details pane is displayed.
5. Perform the following steps in the step details pane:
   a. Right-click a UI object such as text box that is suitable for entering text in the **Smartshot** pane and select **Use this element as step target**.

      The **UI Object** type, **Value**, and values for **Object identified by** are displayed.

      Optional: Under the **Configuration** section, enter a value to be captured as user input in the **Value** field, or click the **Insert Variable** icon  inside the field to insert a variable.

      Note: If you select the text before choosing a variable, the selected text is replaced with the variable. If you do not select any text, the variable is inserted at the current cursor position.

   b. Optionally, you can modify the values that identifies the object by using one of the following methods:
      - Enter the values in the **Object identified by** fields.
      - Drag and drop an element from the **Smartshot** view or the **Elements** section, or drag and drop a Property-Value pair from the **Properties** section.

      You can replace the value of the object with the value of a variable under the **Object Identified By** section by clicking the **Insert Variable** icon  and selecting a variable from the list.

      Note: If you select the text before choosing a variable, the selected text is replaced with the variable. If you do not select any text, the variable is inserted at the current cursor position.

   c. **Optional**: Select the **Click here to add alternative identifier** link to select one more object that works as an alternative target if the first object was not found.
6. Click the **Save** icon  to save the modifications.

Results

You have added a **Type** step from the **UI** catalog to add user actions in a Web UI test.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a Press step

You can add a press action for window controls or supported keys as a step by using the **Press** step from the **UI** catalog. For example, you can add a step to minimize a window or press `Ctrl + M`.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 259.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Press** element under any test step as required from the **UI** catalog in the **Catalog** tab.
4. Click the **Press** step.
   The step details pane is displayed.
5. Perform the following steps in the step details pane:
   a. Select an option from the **Press** list.

      The **Press** list displays the Back, Close, Enter, Forward, Keyboard, Maximize, Minimize, Refresh, Reset, Restore, and Tab options. If you selected the keyboard option, then the Modifiers and Key fields are displayed.

   b. Select one or more modifier keys such as Cmd (only for Mac), Ctrl, Shift, Alt in **Modifier**.
   c. Enter a name of the key to be used in combination with the modifier or independently in **Key**. For example, you can use Ctrl + Home or Home.

d. Select the **Native System Input** checkbox to simulate the input keys or special key actions at the screen level on the active window.

   This option is used to handle non-HTML windows such as file browse dialogs. To interact with file browse dialogs, you must keep your machine unlocked and the browser dialog must be in the foreground.

6. Click the **Save** icon  to save the modifications.

Results

You have added a **Press** step from the **UI** catalog to add user actions in a Web UI test.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding conditional logic and redefining the flow in tests from the Flow catalog

When you want to add conditional logic and redefine the flow to update a test, you can add such steps from the **Flow** catalog. You can add a test inside the test as a step and abort the test after marking it as failed. Also, you can insert the `If`, `While`, and `Do-While` conditions in the test to make a set of steps run only if a specific condition is met.

You can add conditional logic and redefine the flow by adding the following steps from the **Flow** catalog in the **Catalog** tab to update the test:

Catalog element	Description	Reference
Run	You can embed tests within a test and run them. You can add a test after any step within the test.	Adding a Run step on page 261
Fail	You can abort the test and mark the test as failed. For example, you can add this element as a resulting step after checking a condition by using the `If` statement.	Adding a Fail step on page 262
If	You can insert the `If` logic in a test to make a set of steps run only if a specific condition is met.	Adding an If step on page 262
While	You can insert the `While` logic in a test to make a set of steps run in a loop only if a specific condition is met.	Adding a While step on page 265
Do-While	You can insert the `Do-While` logic in a test to make a set of steps run in a loop only if a specific condition is met after the first iteration.	Adding a Do-While step on page 268

Adding a Run step

When you want to add a test inside the test as a step, you can use the **Run** step from the **Flow** catalog.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 261.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Run** element under any test step as required from the **Flow** catalog in the **Catalog** tab.
4. Click the **Run** step.
   The step details pane is displayed.
5. Perform the following steps in the step details pane:
   a. Select a test from the list in **Path** or drag and drop the `.yaml` file in the designated field.
   b. **Optional**: Select a dataset from the list in **Dataset** or drag and drop the `.csv` file in the designated field.
6. Click the **Save** icon  to save the modifications.
7. **Optional:** Click the expand icon  to view the embedded test steps.
   The embedded test steps are displayed in greyed-out mode indicating that you can only view the steps. You can click any step to view the step details. If you want to edit the embedded test, you must right-click the **Run** step and click **Open**. The embedded test is displayed in a new tab in **Test Editor**. When you run the test, the step result indicators are displayed for an embedded test as well.

Results

You have added a **Run** step from the **Flow** catalog to embed a test inside another test.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a Fail step

You can abort the test and mark the test as failed. For example, you can add the **Fail** element as a resulting step after checking a condition by using the `If` statement.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 262.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Fail** element under any test step as required from the **Flow** catalog in the **Catalog** tab.
4. Click the **Fail** step.
   The step details pane is displayed.
5. Enter a custom message in the **Fail statement** text box.
6. Click the **Save** icon  to save the modifications.

Results

You have added a **Fail** step from the **Flow** catalog to mark the test as failed.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding an If step

You can insert the `If` logic in a test to make a set of steps run only if a specific condition is met.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 263.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **If** element under any test step as required from the **Flow** catalog in the **Catalog** tab.
4. Click the **If** step.
   The step details pane is displayed.
5. Select the required mode from the **Mode** list:
   - **UI**
   - **General**

   Note: The default mode depends on where the **If** step is added. When the **If** step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.

   A confirmation dialog is displayed, indicating that switching modes clears the current step configuration.
6. Click **Yes**.

   Note:
   - The default mode depends on where the **If** step is added. When the **If** step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.
   - You can switch between the available modes by selecting a different option from the **Mode** list.
   - If you switch modes after configuring the step, select the required mode and then click **Yes** in the confirmation dialog. The existing configuration is cleared and the step opens in the selected mode.
7. You can perform the actions that are listed in the following table to configure an **If** step during a test run:

Mode	Action
General: Use the **General** mode to add if conditions to test steps.	Perform the following steps: a. In the first column, click the **Insert Variable** icon to select a variable from the list or enter the variable name. b. In the **Operator** field, indicate the basis of comparison of the two operands. Note that the two operands are strings. The available operators are as follows: ▪ Equals ▪ Greater Than ▪ Greater Than Equals ▪ Equals variable ▪ Less Than ▪ Less Than Equals ▪ Strict Equals c. In the third column, click the **Insert Variable** icon to select a variable from the list or enter the variable name in the **Value** field. For conditional statements, when you select a variable from the list, the variable is inserted without the `${}` notation. When you concatenate a variable with a string, the variable is inserted in the `${variable_name}` format.
UI: Add conditional if statements for the UI elements directly from the Smartshot.	Perform the following steps: a. Hover over a UI element in the **Smartshot** view, right-click, and then select **Use this element as a step target**. Alternatively, you can drag and drop a UI element from the **Smartshot** view to the **Object identified by** section. The **UI Object** type is identified. You can replace a static property value with a variable by clicking the **Insert Variable** icon . During a test run, UI elements can be targeted dynamically based on the variable value. When you select a variable from the list, the variable is inserted without the `${}` notation. When you concatenate a variable with a string, the variable is inserted in the `${variable_name}` format. b. **Optional**: Click the **Click here to add relative to** , specify the relative location in the **Location** field, and specify the relative object in the **UI Object** field.

Mode	Action
	c. **Optional**: Click **Click here to add alternative identifier**, and select a target object from the list in the **Target 2** field. Alternatively, you can drag and drop an element from the **Smartshot** view. d. **Optional**: You can select the **Code** checkbox to enter the If statement in the text box. This allows you to write the condition in the text box in the JS mode.

8. Click the **Save** icon  to save the modifications.

Results

You have configured the **If** step to evaluate a condition during a test run. The steps within the If block are run only when the specified condition is met.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a While step

You can insert the `While` logic in a test to make a set of steps run in a loop only if a specific condition is met after the first iteration. The test results display iteration details such as the iteration number, status, and test run time. The test tree displays information from the latest iteration, while the results panel displays either the last successful iteration or the first failed iteration, depending on the test run outcome.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

The **Edit branch** dialog is displayed.

If the **Edit branch** dialog is not displayed, then go to step 3 on page 266.

2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **While** element under any test step as required from the **Flow** catalog in the **Catalog** tab.
4. Click the **While** step.

   The step details pane is displayed.
5. Select the required mode from the **Mode** list:
   - **UI**
   - **General**

   Note: The default mode depends on where the **While** step is added. When the **While** step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.

   A confirmation dialog is displayed, indicating that switching modes clears the current step configuration.
6. Click **Yes**.

   Note:
   - The default mode depends on where the **If** step is added. When the **If** step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.
   - You can switch between the available modes by selecting a different option from the **Mode** list.
   - If you switch modes after configuring the step, select the required mode and then click **Yes** in the confirmation dialog. The existing configuration is cleared and the step opens in the selected mode.
7. You can perform the actions that are listed in the following table to configure a **While** step during a test run:

Mode	Action
General: Use the **General** mode to add while loop conditions to test steps.	Perform the following steps: a. In the first column, click the **Insert Variable** icon to select a variable from the list or enter the variable name. b. In the **Operator** field, indicate the basis of comparison of the two operands. Note that the two operands are strings. The available operators are as follows: ▪ Equals ▪ Greater Than ▪ Greater Than Equals ▪ Equals variable ▪ Less Than

Mode	Action
	▪ Less Than Equals ▪ Strict Equals c. In the third column, click the **Insert Variable** icon to select a variable from the list or enter the variable name in the **Value** field for the selected **UI Object**. When you select a variable from the list, the variable is inserted without the `${}` notation. When you concatenate a variable with a string, the variable is inserted in the `${variable_name}` format.
UI: Use the Smartshot to set while loop conditions for UI elements.	Perform the following steps: a. Hover over a UI element in the **Smartshot** view, right-click, and then select **Use this element as a step target**. Alternatively, you can drag and drop a UI element from the **Smartshot** view to the **Object identified by** section. The **UI Object** type is identified. You can replace a static property value with a variable by clicking the **Insert Variable** icon . During a test run, UI elements can be targeted dynamically based on the variable value. When you select a variable from the list, the variable is inserted without the `${}` notation. When you concatenate a variable with a string, the variable is inserted in the `${variable_name}` format. b. **Optional**: Click **Click here to add relative to**, specify the relative location in the **Location** field, and specify the relative object in the **UI Object** field. c. **Optional**: Click **Click here to add alternative identifier**, and select a target object from the list in the **Target 2** field. Alternatively, you can drag and drop an element from the **Smartshot** view. d. **Optional**: You can select the Code checkbox to enter the If statement in the text box. This allows you to write the condition in the text box in the JS mode.

8. Click the **Save** icon to save the modifications.

Results

You have added a **While** step that checks whether a specific condition is met in the test from the **Flow** catalog.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a Do-While step

You can insert the `Do-While` logic in a test to make a set of steps run in a loop only if a specific condition is met after the first iteration. The test results display iteration details such as the iteration number, status, and test run time. The test tree displays information from the latest iteration, while the results panel displays either the last successful iteration or the first failed iteration, depending on the test run outcome.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

About this task

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 268.
2. Select an existing edit branch or create a new edit branch.
3. Drag and drop the **Do While** element under any test step as required from the **Flow** catalog in the **Catalog** tab.
4. Click the **Do While** step.
   The step details pane is displayed.
5. Select the required mode from the **Mode** list:
   - **UI**
   - **General**

   Note: The default mode depends on where the **Do While** step is added. When the **Do While** step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.

   A confirmation dialog is displayed, indicating that switching modes clears the current step configuration.
6. Click **Yes**.

Note:

- You can switch between the available modes by selecting a different option from the **Mode** list.
- If you switch modes after configuring the step, select the required mode and then click **Yes** in the confirmation dialog. The existing configuration is cleared and the step opens in the selected mode.

7. You can perform the actions that are listed in the following table to configure a **Do While** step during a test run:

Mode	Action
General: Use the **General** mode to add do-while loop conditions to test steps.	Perform the following steps: a. In the first column, click the **Insert Variable** icon to select a variable from the list or enter the variable name. b. In the **Operator** field, indicate the basis of comparison of the two operands. Note that the two operands are strings. The available operators are as follows: ▪ Equals ▪ Greater Than ▪ Greater Than Equals ▪ Equals variable ▪ Less Than ▪ Less Than Equals ▪ Strict Equals c. In the third column, click the **Insert Variable** icon to select a variable from the list or enter the variable name in the **Value** field for the selected **UI Object**. When you select a variable from the list, the variable is inserted without the `${}` notation. When you concatenate a variable with a string, the variable is inserted in the `${variable_name}` format.
UI: Use the Smartshot to set do-while loop conditions for UI elements.	Perform the following steps: a. Hover over a UI element in the **Smartshot** view, right-click, and then select **Use this element as a step target**. Alternatively, you can drag and drop a UI element from the **Smartshot** view to the **Object identified by** section. The **UI Object** type is identified.

Mode	Action
	You can replace a static property value with a variable by clicking the **Insert Variable** icon . During a test run, UI elements can be targeted dynamically based on the variable value. When you select a variable from the list, the variable is inserted without the `${}` notation. When you concatenate a variable with a string, the variable is inserted in the `${variable_name}` format. b. **Optional**: Click **Click here to add relative to**, specify the relative location in the **Location** field, and specify the relative object in the **UI Object** field. c. **Optional**: Click **Click here to add alternative identifier**, and select a target object from the list in the **Target 2** field. Alternatively, you can drag and drop an element from the **Smartshot** view. d. You can select the **Code** checkbox to enter the do while condition in the text box in the JS mode.

8. Click the **Save** icon to save the modifications.

Results

You have configured the **Do While** step to evaluate a condition during a test run. The steps within the **Do While** block run at least once and continue to run as long as the specified condition evaluates to true.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Adding a drag-and-drop action in a test from a smartshot

When you want to manually add the drag-and-drop action in a Web UI test, you can do so by editing the recorded test and then add the drag-and-drop action by using a smartshot.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 271.
2. Select an existing branch or create a new branch.
3. Click the test step before which you want to add a drag-and-drop step.
   The step details pane is displayed.
4. Select an element that you want to drag to the new location from one of the following options:
   - The **Smartshot** view
   - The **Elements** list
   - A property-value pair from the **Properties** list
5. Right-click the element and select the **Drag & Drop** option.
   The **Details** tab displays the values in the **UI Object** and **Action** fields. Also, the **Object Identified By** section displays the **From** and **To** tabs. The **From** tab displays the identified target element.
6. Switch to the **To** tab in the **Object Identified By** section.
7. Select the destination element on which you want to drop the previously selected element from one of the following options:
   - The **Smartshot** view
   - The **Elements** list
8. Right-click the element and select the **Use this element as step target (To)** option.
   The details of the target element are populated.
9. Save the test.

Results

You manually added the drag-and-drop action in a Web UI test by using a smartshot.

What to do next

You can run the test and view the results. See Running a Web UI test on page 228.

Disabling test steps

When you want to exclude a step or a set of steps under a container from the test run, you can disable such steps in a Web UI test.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step #unique_317_Connect_42_step_next.
2. Select an existing edit branch or create a new edit branch.
3. Select a step or a container with multiple steps.
   The step details pane is displayed.
4. Click the **Advanced** tab.
5. Select the **Disable** checkbox.
   The selected step or set of steps under the selected container are greyed out.
6. Click the **Save** icon  to save the modifications.

Results

You have disabled a step or a set of steps.

What to do next

You can clear the **Disable** checkbox in the **Advanced** tab to enable the test step.

Related information

Editing a Web UI test on page 236

Configuring timeouts for tests and test steps

You can configure timeouts for tests and individual steps to define the duration of the time that the system should wait before marking an action as failed due to timeout.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

About this task

You can configure timeouts at two levels:

- Test level: You can set a global timeout for the entire test by using the **Configure settings and data** step.
- Step level: You can override the global timeout for individual steps directly within the step settings.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 273.
2. Select an existing edit branch or create a new edit branch.
3. Perform any of the following actions:

Action	Description
Configure timeout for the test	Click the **Configure settings and data** step in **Test Editor** The timeout details are displayed with **Duration** and **Unit** fields. By default, the timeout is set to 30 seconds. You can select the **Override Timeout** checkbox to modify this value.
Configure timeout for a test step	Select a step and select the **Advanced** tab. The **Timeout Details** section including the **Duration** and **Unit** fields, is displayed.

4. Select the **Override Timeout** checkbox.
5. Specify the duration of time that Test Hub must wait for the step to be completed in **Duration**.
6. Select the unit of time for the timeout in **Unit**.
7. **Optional:** Select the **Code** checkbox to enter the timeout and unit.

8. Close the step details pane.
9. Click the **Save** icon  to save the modifications.

Results

You have configured timeout settings for your test or individual test steps.

Copying test steps from one test to another

When a set of steps in a test is required in another test, you can copy them and paste them into an existing test or an empty test. You can select the required steps from a test to copy, although they are not in a sequence. If there are any variables associated with the selected step, they are also copied with the step.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 274.
2. Select an existing edit branch or create a new edit branch.
3. Right-click a step or a container and click **Select**.
   Checkboxes are displayed for each step.
4. Select the checkboxes to select any step or a container with multiple steps.
5. Right-click a selected step and click **Copy**.
6. Perform one of the following steps:
   - Open a test, right-click a step where the copied steps are required, and then select **Paste**.
   - Open an empty test, right-click the **Configure settings and data** step, and then select **Paste**.
7. Click the **Save** icon  to save the modifications.

Results

You have copied a set of steps from a Web UI test to another test.

Related information

Running a Web UI test on page 228

Renaming a test asset

You can rename test assets such as tests, suites, smartshots, and applications in your project by using the **Rename** option.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a Project Owner or Tester in the project. See Managing access to the server projects on page 1220.
- Ensured that your project contains test assets.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Right-click a test asset that you want to rename from the test navigator panel and select **Rename**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 6 on page 275.
5. Select an existing edit branch or create a new edit branch.
6. Enter a unique name in the **Rename Asset** dialog.
7. Click **Rename**.
   A success message is displayed.

Results

You have renamed a test asset in your project.

Deleting a test asset

You can delete an application, test, suite, or smartshot when it is no longer required in your project. If the test asset that is selected for deletion is referenced in another test or a test suite, or if the selected smartshot is referenced in a test, a warning is displayed.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a Project Owner or Tester in the project. See Managing access to the server projects on page 1220.
- Ensured that your project contains test assets.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Select a test or a smarshot that you want to delete from the test navigator panel.
5. Click the 

button, displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 7 on page 276.
6. Select an existing branch or create a new branch.
7. Click the **Delete** icon  .
   Alternatively, right-click a test, suite, smarshot, or an application in the test navigator panel, and then click **Delete**.
   Result
   The **Delete Asset** dialog is displayed. This dialog provides information if the selected asset is referenced by the other test asset.
8. Click the **Delete** button to delete the test asset.

Results

You have deleted a test asset from your project.

Related information

Test navigator panel on page 200

Test Editor page on page 201

Creating a test module from an existing set of steps

When a set of steps are repeated and required in multiple tests, you can create a test module for such repeatable steps. You can select the required steps that are in sequence from an existing test and extract them to create a new test module. After extraction, the selected steps are replaced by the **Run** step and the new test module is listed in the test navigator panel.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

About this task

When you extract a set of steps, the resulting asset might be one of the following types:

- A test with the application or page URL details in the **Open** step. You can play back this type of test.
- A module without the application or page URL. You must add the required URL by adding the details in the **Configure settings and data** step or **Open** step.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 277.
2. Select an existing edit branch or create a new edit branch.
3. Right-click a step or a container and click **Select**.
   Checkboxes are displayed for each step.
4. Select the checkboxes to select a set of steps or a container with multiple steps in sequence.
5. Right-click a selected step and click **Extract Steps**.
   The **Create Test from Steps** dialog is displayed. The **Extracted Content** text box displays the list of selected steps and the **Application** field displays the application for which you create the test.
6. Enter a name for the test in **Asset Name**.
7. Select an existing folder from the list in **Location** to save the test assets that you are creating.
   Alternatively, click **New Folder** to create a folder to save the test assets.
8. Click **Create**.

The selected steps are replaced by the **Run** step for the new test module and the newly created test module is listed in the test navigator panel.

Note: When you save the test under a different application by using the **Save As** option, on clicking the step, the step details pane displays the `Error loading smartshot` message for all test steps. To resolve this error, you can either manually insert a suitable smartshot from the new application by using the **Replace Smartshot** option in the step details pane, or you can play back the test and save the required smartshots directly from the test results.

Results

You have created a test module by extracting a set of steps from the existing test.

Related information

Editing a Web UI test on page 236

Running a Web UI test on page 228

Run steps on page 260

Viewing test assets exported from DevOps Test UI

When you export Web UI test projects from HCL DevOps Test UI (Test UI) to HCL DevOps Test Hub (Test Hub), Test UI extracts test assets, processes them, and generates a .yaml file that contains the step details and the associated smartshots. You can then view the test assets of the Web UI test projects that are exported from Test UI.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a Project Owner or Tester in the project. See Managing access to the server projects on page 1220.
- Exported a Web UI test project from Test UI. For more information, refer to Migrating Web UI test projects.

About this task

Test UI exports the test projects to the specified project and the selected edit branch in Test Hub. Test Hub lists the exported test projects in the test navigator panel with the respective names suffixed with UTA. The exported test projects contain sub-folders such as `tests`, `shots`, and `others`. You can also edit and run the exported Web UI tests.

If the exported test contains a dataset, then a **If** statement is added to check whether the variable has an assigned value and an **Assign** statement is added to assign the value from dataset in the test. And for compound tests with datasets, only an **Assign** statement is added to assign the dataset. Also, if the exported test contains an encrypted dataset, the encryption will be visible only on the **Data** page and not on the **Test Editor** page.

Notes:

- If two datasets in a compound test contain the same column name, then the value of the variable will be overwritten by the value from the subsequent dataset, which might result in a failure of the test run.
- If the migrated compound test with dataset mapper that is configured to run in an iterative loop, then the test run might have twice the number of iterations.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the exported test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Click the exported test project to view the list of tests.
5. Click the `.yaml` file to open the test.
   The test steps are displayed on the **Test Editor** page.
6. Click any test step for which you want to view the details.
   The step details pane is displayed with the smartshot.
7. Click the **Previous Step** icon and the **Next Step** icon to navigate between the previous and next test steps.

Results

You have viewed an exported test.

What to do next

You can perform the following tasks:

- Edit the test in Test Editor. See Editing a Web UI test on page 236.

  Notes:
  - You must compare the test steps with the original test steps in Test UI and if anything is missing, modify the imported test before you run the test.
  - The properties of UI elements might change in the application after migration of the test because of the automatic identification feature in Test UI. Therefore, you must update the object properties as per the change. Otherwise, the test might fail.
  - If there is any drag-and-drop action in the migrated test, you must verify the step details and modify the object identifiers if required.
  - There is a difference in the way the multi-windows are handled by Test UI and Test Hub in multi-window tests. As a result, the migrated test might not run properly. To resolve this, you must manually verify and correct any multi-window steps in your script after the migration.

- Create an **Assign** step to link the dataset to relevant variables if the exported test has a dataset linked. Refer to the first **Log** statement for variable mapping information.

- Create a security group if any of the exported tests have an encrypted dataset. See Creating a security group on page 404.
- Run the test and view the results. See Running a Web UI test on page 228.

Salesforce Lightning tests

You can record and play back Web UI tests for Salesforce applications that use Salesforce Lightning controls.

Salesforce Lightning controls are composite controls built from multiple underlying controls. During recording, an action on a composite control might be assigned to a child element. If playback does not work as expected, use **Element View** to select the required parent or actionable control and set it as the step target.

Smartshot capabilities

During recording, HCL DevOps Test Hub (Test Hub) captures a Smartshot for each step. For Salesforce Lightning applications, the Smartshot includes the visible controls that Test Hub recognizes. You can use the Smartshot in the test editor without recording the session again to perform the following tasks:

- **Control detection**: View the recognized controls in the Smartshot. The original target remains highlighted, and you can select another recognized control.
- **Step generation**: Select a visible control to generate an additional action, such as a click, text entry, or selection.
- **Verification points**: Select a visible control and capture a verification point for a property such as text, value, or visibility.

The following scenarios might require you to edit a test after recording:

- If a drop-down option is recorded as a `span` element, then you must select the Salesforce Lightning base combo control in **Element View** as the step target.

  Note: When you record a drop-down action, click the left side of the drop-down text value. You should avoid clicking the middle or right side of the value.

- For an SVG element, you must select the nearest parent Salesforce Lightning icon or primitive icon in the element hierarchy.
- After you record a test, if any step that is recorded as `span` fails during playback, you must select its actionable parent control as the target.
- While recording, when you perform user actions, the highlight might be on some other element. You can ignore this behaviour.

The following table lists the supported main Salesforce Lightning controls:

Control type	Control name
Base Combobox	SFL Base Combobox
Base Combobox Formatted Text	SFL Base Combobox Formatted Text

Control type	Control name
Button	SFL Button
Button Icon	SFL Button Icon
Combo Item	SFL Combo Item
Combobox (Select)	SFL Combobox (Select)
Date Picker	SFL Date Picker
Formatted Rich Text	SFL Formatted Rich Text
Icon	SFL Icon
Input	SFL Input
Input Field	SFL Input Field
Message Context Consumer	SFL Message Context Consumer
Messages	SFL Messages
Navigation Item Dropdown	SFL Navigation Item Dropdown
Navigation Menu Button	SFL Navigation Menu Button
Navigation Menu Item	SFL Navigation Menu Item
Picklist	SFL Picklist
Primitive Icon	SFL Primitive Icon
Primitive Input Sample	SFL Primitive Input Sample
Record Edit Form	SFL Record Edit Form
Record Edit Form Create	SFL Record Edit Form Create
Records Base Record Form	SFL Records Base Record Form
Records Detail Panel	SFL Records Detail Panel
Records Layout	SFL Records Layout
Records Layout Base Inputs	SFL Records Layout Base Inputs
Records Layout Block	SFL Records Layout Block
Records Layout Event Brokers	SFL Records Layout Event Brokers
Records Layout Item	SFL Records Layout Item
Records Layout Row	SFL Records Layout Row
Records Layout Section	SFL Records Layout Section

Control type	Control name
Records Modal Detail Panels	SFL Records Modal Detail Panels
Text area	SFL Text area
Time Picker	SFL Time Picker

Configuration of Salesforce test automation

You must configure HCL DevOps Test Hub (Test Hub) and Salesforce application before you record and play back Salesforce Lightning application tests.

You can configure Test Hub to record, structure, and run automated functional tests for a Salesforce application. The configuration involves:

- Defining the test scenario
- Mapping sensitive credentials to a project-level security group
- Configuring the parameters that Test Hub uses at run time.

Configuration of Test Hub for Salesforce test automation

You can configure HCL DevOps Test Hub (Test Hub) variables, environments, datasets, and runtime parameters for Salesforce test automation.

You must complete the following configuration tasks before you record or run a Salesforce Lightning test:

Configuring the initial test environment

You can configure the initial test environment by creating a security group and adding secrets for sensitive values used in Salesforce tests with HCL DevOps Test Hub (Test Hub) variables.

Before you begin

You must have completed the following tasks:

- Ensured that you have a Test Hub project.
- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Ensured that you have administrative credentials for the Salesforce sandbox.

About this task

You can use the Test Hub dashboard to select your project and configure the **Security** section, where security groups and individual secrets are stored for use by your tests. After you record or build a Salesforce test scenario, some steps enter sensitive values, such as a password or a one-time verification code. Instead of entering these values as text in the test, you can refer to them by using variable syntax, such as `${Password}` and `${Verification_Code}`.

At run time, Test Hub resolves the actual values from a configured environment file, and masks them in the console, so the same test can run for different users or environments without exposing credentials in the script.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the Salesforce application.
   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
4. Click **+ Create Security Group**, and then enter a name for the security group

   The **Create Security Group** dialog is displayed.
5. Enter a name for the security group. For example, `MySecrets`.
6. Click **Create**.
   The security group is created.
7. Click **+ Add Secret**.
8. Enter an identifier for the secret in the **Identifier** field. For example, `Password` or `Verification_Code`.
9. Click **Create**

   The secret is added to the project-level security group.

   When you record a test for a Salesforce application on the **Test Editor** page, replace the sensitive values with variables, such as `${Password}` and `${Verification_Code}`.

   Note: The variable names must match the names of the identifiers of the secret created in the project-level security group.

Results

You have configured the initial test environment by creating a security group and adding the secrets required by the test.

Mapping credentials by using an environment file

You can map the variables referenced in the test to secrets configured in the security group by using an environment file, so that HCL DevOps Test Hub (Test Hub) can resolve the secret values at run time.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a Project Owner or Tester in the project. See Managing access to the server projects on page 1220.
- Ensured that your project contains test assets.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.

About this task

You can use an environment file to map authentication values and security tokens to the variables referenced in your test. You can define authentication mapping for multiple users in a single environment file, or create separate environment files for individual users and select the required environment file when you run the test.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the Salesforce application.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Click the **Action Menu** icon  and select **Environment**, or right-click the application folder in the test navigator panel and select **My New > Environment**.
   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 6 on page 284.
5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
6. In the **Create Environment** dialog, configure the following environment details:
   a. Select a target application from the **Application** drop-down list. For example, `Mysecrets`.
   b. Enter a unique name for your environment in the **Name** field. For example, `SFEnv.dte.yaml`.
7. Click **Done**.
   The new environment file is created and displayed in **Test Editor**.
8. Perform the following steps to add specific variables in the **Variables** section:
   a. Click **+ Create Variable**.
   b. Enter the variable name in the **Name** column.

c. Enter the corresponding secret reference in the **Value** column.
For example, configure the variables that correspond to `$(Password)`and `$(Verification_Code)`.

9. Click the **Save** icon to save the environment details.
10. From the **Environment** list at the top right of the **Test Editor** page, select the required environment. For example, `SFLenv` .
The selected environment enables Test Hub to resolve the secret references from the configured security group.

Results

You have mapped the variables referenced in the test to the secrets configured in the security group by using an environment file.

Creating a dataset for credential mapping

You can create a dataset that contains the credential values for multiple users, so that HCL DevOps Test Hub (Test Hub) can resolve the appropriate values when the dataset is used with a recorded test.

Before you begin

You must have completed the following tasks.

- Read and completed the task mentioned in Configuring the initial test environment on page 282.
- Created a dataset. SeeCreating a dataset on page 425.

About this task

You can create a dataset that contains the credential values for multiple users. Test Hub uses the dataset to associate each user's credentials with the corresponding values when the dataset is used to configure a recorded test.

1. Log in to Test Hub.
The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
The **Overview** page of the project is displayed.
3. Click **Author > Data**.

The **Data** page is displayed.

4. Select the branch in the project repository from the **Branch** list.

The assets and resources in the selected branch are displayed in the data navigator panel.

5. Perform any of the following actions:
   - Click the **Action Menu** icon  and select **Dataset**.
   - Right-click the application in the data navigator panel, select **New > Dataset**.

The **Edit Branch** dialog is displayed.

If the **Edit Branch** dialog is not displayed, then go to step 7 on page 286.

6. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

     Note: A warning message is displayed if the Edit branch, which you selected, is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

   Result

   The **Create dataset** dialog is displayed.
7. Select the application, select the location, and enter a name for the dataset, such as `UserOTPAuth.csv`.
8. Click **Next**.
9. Select **Create an Empty dataset** as the source and click **Create**.
   The empty dataset is created and displayed on the **Data** page.
10. Enter the required data in the dataset. For example, add columns for **Username**, **AuthUrl**, **Password**, and **Verification_Code** and add a row for each user.

    Note: The dataset created is used later to provide credential values for the Salesforce automation users when configuring the recorded test.

11. Optional: Encrypt sensitive columns by right-clicking a column and selecting **Encrypt column data** to encrypt the column.

    The encrypted data is stored securely. Select **Decrypt column data** to view the original values when needed.

    Note: Ensure that the variable names match the corresponding dataset column names if the test configuration requires matching names.

12. Click the **Save** icon to save the dataset.

Results

You have created a dataset containing credential values for multiple users. The dataset can be used to provide credential values when configuring a recorded test.

Configuration of the Salesforce application for OTP-based authentication

You can configure Salesforce and HCL DevOps Test Hub (Test Hub) for OTP-based authentication during Salesforce test automation.

You can enable Test Hub to sign in to Salesforce automation flows that use one-time passcode (OTP) based multi-factor authentication (MFA), you must register a time-based one-time passcode (TOTP) authenticator for the automation user. You must then store the generated secret in Test Hub so that it can be resolved automatically at run time.

Note: Test Hub currently supports only OTP-based MFA for applications used in automation testing.

Registering an authenticator app for OTP-based sign-in

You must register a time-based one-time passcode (TOTP) authenticator for the Salesforce automation user, so that Test Hub can generate matching one-time passcodes during sign-in.

Before you begin

You must have completed the following tasks:

- Configured Test Hub. See Configuration of Test Hub for Salesforce test automation on page 282.
- Ensured that you have access to the Salesforce automation user's account and permission to change personal security settings.
- Ensured that you obtained the automation user and its permission set.

About this task

You can register an authenticator app for the Salesforce automation user. During registration, Salesforce displays a QR code and a corresponding secret key. The TOTP secret is used by Test Hub to generate matching verification codes at run time.

1. Log in to the Salesforce application with your credentials.
2. In Salesforce, click your user avatar in the top-right corner and select **Settings**.
   The **Settings** page is displayed.
3. In the left navigation pane, click **Advanced User Details**.
   The **Advanced User Details** page is displayed.
4. Locate **App Registration: One-Time Password Authenticator** and click **Connect**.
   The **Connect an Authenticator App** page is displayed with a QR code and a secret key.
5. Open your authenticator app, such as Microsoft Authenticator or Google Authenticator, and scan the QR code.
6. Obtain the `otpauth://` secret URI from the displayed QR code to configure OTP generation in Test Hub.
7. In your authenticator app, note the 6-digit verification code that is generated
8. Enter the 6-digit verification code in the **Verification Code** field.
9. Click **Connect**.

Results

You have registered a TOTP authenticator for the Salesforce automation user.

Creating the OTP secret in Test Hub

You can create the `otpauth://` secret URI in Test Hub to map it to the `Verification_Code` variable, so that the OTP value resolves automatically at run time.

Before you begin

You must have completed the following tasks:

- Obtained the `otpauth://` secret URI. See Registering an authenticator app for OTP-based sign-in on page 287.
- Ensured that a security group is configured for the project. See Configuring the initial test environment on page 282.

About this task

You store the `otpauth://` secret URI as a secret in the project's security group, and then reference that secret from the `Verification_Code` variable in your environment file. Test Hub uses the secret to generate the matching OTP value at run time.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the Salesforce application.
   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
4. Click **+ Add Secret**.
5. Enter an identifier for the secret in the **Identifier** field. For example, `salesforceOtpSecret`.
6. Paste the copied `otpauth://` secret URI in the **Value** field.
7. Click **Create** to save the secret.
8. Open the environment file. For example, **SFEnv.dte.yaml** in the **Test Editor** page.
9. Add a `Verification_Code` entry with the following values:
   - Set **expr** to the identifier of the secret that you created. For example, `salesforceOtpSecret`.
   - Set **type** to `secret`.

   You can configure different OTP secrets for different users in the same environment file.

Results

You have created the OTP secret in the project's security group.

Recording and configuring Salesforce application tests

You can record Salesforce Lightning application interactions from the Test Editor and configure the recorded test with credential and OTP values for different users.

Before you begin

You must have completed the following tasks:

- Configured the Salesforce application as described in Configuration of Salesforce test automation on page 282.
- Read and understood Recording a Web UI test on page 211.
- Ensured that the Salesforce application is available in the project.
- Ensured that the environment file is configured with the `Verification_Code` secret. See Creating and mapping the OTP secret in Test Hub. on page 288

About this task

You can record a Salesforce Lightning application interactions from the Test Editor and configure the recorded test to use credential and OTP values for different users. Test Hub captures the actions performed for a Salesforce scenario as Smartshots. You can map a dataset to provide credential values for multiple users.

You can reference the `Verification_Code` variable in the test step that requires an OTP.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the Salesforce application.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Perform one of the following actions to create a test:
   a. Click **Create asset > Create Test** on the **Test Editor** page.
   b. Right-click the Salesforce application and select **New > Test**.
   c. Open the **Action menu** in the test navigator panel and select **Test**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 6 on page 290.
5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

6. Perform the following steps in the **Details** tab of the **Create Test** dialog:
    a. Select an application for which you want to create a test from the **Application** drop-down list.

       The application folder is selected as the default location for the test.

    b. **Optional**: Select a folder if there are subfolders in **Location** to save the test assets that you are creating.
    c. Enter a name for the test in **Asset Name**.
7. Click **Next**.
   The **Source** tab is displayed.
8. Select the **By recording** option to record a test in the **Select source** drop-down list.
   Test Hub checks whether the recording agent is installed on your computer. For more information, refer to step 10 in the Recording Web UI test.
9. Start the recording and perform the actions for one Salesforce scenario.
   A Smartshot is captured for each recorded step.
10. When Salesforce Lightning controls overlap, open **Element View** and select the specific control required by the scenario.
    Salesforce Lightning controls can be located within approximately five pixels of one another. Select the controls that correspond to the business requirement instead of trying different controls in the Smartshot.
11. Stop the recording and save the test.
    The recorded Salesforce test is created and displayed in **Test editor**.
12. Perform the following steps to map the dataset, such as `UserOTPAuth.csv` to map the credential values for multiple users.
    a. Click **Configure setting and data** in the recorded test.

       The **Configure setting and data** dialog is displayed.

    b. In the **Data** section, click the drop-down icon and select the created dataset from the list. For example, `UserOTPAuth.csv`.
    c. Click **Close** to exit the **Configure setting and data** dialog.
13. Perform the following steps to reference the mapped OTP value in the test step where Salesforce requires the verification code.:
    a. In the recorded test, open the step *With 'Verify Your Identity | Salesforce'* that types a value into the **Verification Code** field.

       The step details pane is displayed.

    b. In the **Value** field, enter the variable reference for the mapped OTP secret.
    c. Click the **Close** icon to close the run details pane.
14. Click **Save** to save the modifications.

Results

You have recorded a Salesforce Lightning application test and configured it to use credential and OTP values for different users.

Running a Salesforce Lightning application test

You can run the recorded Salesforce Lightning application tests and add steps or verification points from Smartshots.

Before you begin

You must have completed the following tasks:

- Ensured that the Salesforce application is configured. See Configuration of Salesforce test automation on page 282.
- Ensured that a Salesforce test is recorded. SeeRecording and configuring Salesforce application tests on page 288.

About this task

You can configure runtime parameters to support dynamic data injection, and you can run a recorded test with a configured environment in either of the following ways:

- From the **Test editor** page
- From the **Execution** page.

Perform any one of the following methods to run the Salesforce automated test:

Action	Description
Run from the **Test Editor** page	Perform the following steps: a. In the **Test Editor** toolbar, select the required environment from the environment list. For example, `SFLenv`. b. Click **Run** to run the test against the selected environment. c. Optional: Select the execution cluster from the drop-down next to **Run**.
Run from the **Execution** page	Perform the following steps: a. Click **Execute**, locate and open the required test asset. b. On the **Environment** tab, select the required environment. For example, `SFLenv.dte.yaml` c. Click **Execute**.

Results

You have run the recorded Salesforce Lightning application test with the selected environment.

Test suite management

A suite is a collection of the tests and other suites. You can combine the multiple tests and suites into a single workflow and organize them into a suite.

As part of the test suite management, you can create, edit, run, and delete the suites. When you create a suite, a `.dts.yaml` file is created and listed under the **Suites** folder in the test navigator panel.

You can run the suite either sequentially or in parallel. In a sequential run, the tests and suites within the suite are run sequentially, one after the other. In a parallel run, the tests and suites within the suite are run concurrently. Setting the maximum concurrency value limits how many test instances can run at the same time. When a suite is running, you can click the Show in progress icon to monitor execution and the page displays active test suite instances up to your specified concurrency limit. For example, if a suite contains three tests and the concurrency value is set to two, the progress page will display two instances running simultaneously.

When you run a suite, each child test within the suite generates an individual result. You can open the test from the suite by right-clicking the test and selecting **Open**, and the results of the child test are displayed in the **Results** tab along with the details of the suite that includes this test.

You can rerun the failed tests and suites instead of rerunning the suite. When you run the test suite, you can access the console output to troubleshoot the issues.

Creating a suite

You can create a suite within your project in HCL DevOps Test Hub (Test Hub). You can add multiple tests and suites into a single suite for a test run.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the suite assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.

4. Perform any of the following actions:
   - Click **Create Asset > Create Suite** on the **Test Editor** page.
   - Click the **Action Menu** icon  and select **Suite**.
   - Right-click the application in the test navigator panel and select **New > Suite**.

   The **Edit Branch** dialog is displayed if there is no edit branch active. Otherwise, go to step 6 on page 293.
5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

     Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
6. Perform the following steps in the **Create Suite** dialog to create a suite:
   a. Enter a name for the suite in the **Asset Name** field.
   b. Select an existing application to save the suite asset you create.
   c. Click **Done**.
7. Perform any of the following actions to add the tests and suites to the existing suite:
   - Drag the tests and suites from the test navigator panel to **Suite Editor**.
   - Click the **Click to add** option to add the tests and suites in **Suite Editor**.
8. (Optional) To add the tests and suites to the existing suite from other application, drag the test or suite from other application on the test navigator panel to **Suite Editor**.
9. Click the **Save** icon  to save the suite.

Results

You have created a suite.

What to do next

You can perform the following tasks:

- Run the test suite and view the results. See Running a suite sequentially on page 299 and Running a suite concurrently on page 300.
- Modify the suite. See Modifying a suite on page 296.

Suite creation with Test Genie

Test Genie is an AI-powered capability that can create a scenario-based suite from requirements and design context, and then create tests for all generated intents in a single run.

Suite generation with Test Genie helps you start test authoring from a high-level feature intent instead of manually creating each test. You can provide suite context such as a summary, optional scenario, and optional UI reference images. Test Genie then creates empty tests that contain **Open** steps and AI-generated intents for each scenario.

After you review or edit the generated intents, you can generate tests for the entire suite from the suite toolbar. Test generation runs in one flow for all tests in the suite. As each test is generated, Test Genie uploads the tests incrementally to the suite.

To use Test Genie, you must configure at least one AI provider in the **Infrastructure > Cloud Credentials** page of Test Hub. Test Genie supports the following AI providers:

- Anthropic - Tech Preview
- OpenAI
- GitHub Copilot

> **Restriction:** Only one test generation session through GitHub Copilot can be active at a time across the entire Test Hub server. Any additional Copilot requests are queued and wait until the current session completes.

For optimal results, you can use OpenAI with the `GPT-5.4-mini` model.

Session limits for AI generation

In Test Hub, you can configure session limits as **Advanced** configuration values when you add a cloud credential for an AI provider in **Cloud Credentials**. These values are saved with the selected provider credential and are applied whenever Test Genie starts a suite or test generation session by using that credential.

The three limits are **Max tokens per session**, **Max interactions per session**, and **Max duration per session (in seconds)**. During test generation, Test Hub evaluates all three limits. If any configured limit is reached, the active AI conversation session ends, and you must start a new session to continue the test generation.

These session limits are optional for OpenAI and Anthropic credentials and are not applicable for GitHub Copilot credentials. By configuring these values, you can control response depth and generation time with platform resource usage for large or complex suite-generation workloads.

Related information

Test creation with Test Genie on page 220

Creating a suite with AI-generated scenarios and tests

You can create a scenario-based suite and then generate tests for all scripts in a single run by using AI-powered Test Genie. This bulk generation helps to streamline test creation for multiple scripts in a single run.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource. See Managing access to the server projects on page 1220.
- Read and understood the information in Edit branch overview on page 198.
- Ensured that your project contains an application with an application URL and application description details.
- Configured at least one AI provider in the **Cloud Credentials** page. See Adding cloud credentials of AI providers on page 596.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the target project.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and test navigator panel are displayed.
4. Perform any of the following actions:
   - Click **Create Asset > Create Suite** on the **Test Editor** page.
   - Click the **Action Menu** icon  and select **Suite**.
   - Right-click the application in the test navigator panel and select **New > Suite**.

   The **Edit Branch** dialog is displayed if there is no edit branch active. Otherwise, go to step 6 on page 293.
5. Select an existing edit branch or create a new edit branch.
6. Perform the following steps in the **Create Suite** dialog to create a suite:
   a. Enter a name for the suite in the **Asset Name** field.
   b. Select an existing application to save the suite asset you create.
   c. Click **Next**.
   d. Select **Create suite using requirements and designs** from the **Select Source** list.
      The **Context** and **AI Provider** sections are displayed to gather suite-generation inputs.

e. Enter the details of the suite in **Summary**.

f. **Optional:** Enter the test scenario details by specifying the number of scenarios to generate or describing a specific scenario in **Scenarios**.

g. **Optional:** Upload screenshots of the UI for which you are generating tests in **Reference Images**.

h. Select an AI provider from the configured **AI Provider** list.

i. Click **Done**.
The suite generation progress is displayed, and after the suite is generated, the **Reload** button is displayed in **Test Editor**.

7. Click **Reload**.
The new suite is displayed with generated empty tests that contain **Open** steps and AI-generated intents. You can click the **Show AI Intent** icon  to view the step details that are generated for test creation.
8. **Optional:** Open each empty test to edit the AI-generated intent in the **Test Genie** tab and save the changes.
9. After validating intents, click the **Generate Tests** icon  from the suite toolbar.
The **Generate Test** dialog is displayed.
10. Enter the application URL in **Start URL** if it is not displayed.
11. Select an AI provider from the configured **AI Provider** list.
12. Click **Generate**.
All tests in the suite are generated in one flow, and tests are uploaded incrementally as each test generation is completed.

Results

You have created a suite with AI-generated scenarios and generated tests for all scripts in the suite.

What to do next

You can perform the following tasks:

- Run the generated suite and view results. See Running a suite sequentially on page 299 and Running a suite concurrently on page 300.
- Revise the test intent and regenerate the individual tests by using Test Genie. See Modifying a Web UI test created by using Test Genie on page 224.

Modifying a suite

You can modify the existing suite within your project in HCL DevOps Test Hub (Test Hub) by adding, removing, or rearranging the tests and suites in a suite.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Ensured that your project contains a suite.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the suite assets.
   The **Overview** page of the project is displayed.
3. Navigate to **Author > Test Editor**.
   The test navigator panel is displayed.
4. Select the suite that you want to modify from the test navigator panel.
   Result
   The suite opens in **Suite Editor**.
5. Click the 

button, displayed on the **Suite Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the suite in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed if there is no edit branch active. Otherwise, go to step 7 on page 298.
6. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

7. Perform any of the following actions to modify the suite in **Suite Editor**:

Action	Description
Add the test or suite	Perform any of the following actions to add the test or suite to the suite: ◦ Drag the test or suite from the test navigator panel to **Suite Editor**. ◦ Click the **Click to add** option to add the test or suite in **Suite Editor**. The test or suite are added to the existing suite.
Open the test or suite	Right-click the test or suite, and then click **Open**. The test opens in **Test Editor** and the suite opens in **Suite Editor**. Alternatively, click any test step for which you want to view the result details. The step details pane is displayed. Click **Open** to open the test in the **Test Editor** for further review or editing. **Note:** If a step has not been run, results are not available for that step in the **Results** view, and the **Open** option is disabled.
Arrange the test or suite	Drag and drop the test or suite. The test or suite are arranged in the order of your preference.
Copy the test or suite	Right-click the test or suite you want to copy, and then click **Copy**. The test or suite are copied to the clipboard.
Paste the test or suite	Right-click anywhere in **Suite Editor**, and then click **Paste**. The test or suite are added to the end of the workflow.
Remove the test or suite	Right-click the test or suite you want to remove from the suite, and then click **Delete**. The test or suite are removed from the existing suite.

8. Click the **Save** icon to save the changes.

Results

You have modified the suite in your project.

What to do next

You can run the test suite and view the results. See Running a suite sequentially on page 299 and Running a suite concurrently on page 300.

Running a suite sequentially

When you want tests to run one after the other, you can run a suite sequentially that maintains the correct order and accuracy of the test results.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Ensured that a suite is available in your project.
- Added tests or other suites to the suite.

About this task

During the playback, you can run all tests or only the failed tests. The **Continue on Fail** option ensures that the suite continues to run even if a test fails.

As the test runs, the three color-coded boxes display the counts of passed steps, failed steps, and errors. You can view the step details of the first failed step or error step by clicking the respective color-coded boxes. Also, you can view checkmarks for the steps that pass and crossmarks for the steps that fail.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Click the `.dts.yaml` file in the list of Suites to open the suite.
   The tests or suites are displayed on the **Suite Editor** page.
5. Click the **Configure settings and data** step.
   The **Settings** pane is displayed.
6. Ensure that the **Run in Parallel** option is unchecked to run the test in sequence.

   Note: If **Run in Parallel** is selected, tests will execute simultaneously rather than sequentially.

7. Verify that the **Continue on Fail** option is selected as the default setting, under the **Options** tab.

Note: If you clear the **Continue on Fail** checkbox, the suite stops when the first test fails.

8. Click the icon, and then select the **Run all > Run all on Default Cluster** option.

   You can also select the **Run all on Personal Agent** option if the personal agent is in a ready state.

   You can view the following stages on the progress bar during the test run:
   - **In transition**: The test setup is in progress.
   - **Running**: The test is actively running.
   - **Complete**: The test run is completed. This section also displays the time taken for test run.
   - During the **Running** phase, you can view the following details:
     - A checkmark after each test or suite that passes, and a crossmark for any failed test or suite.
     - The **Results** tab displays the test as **In Progress** during the test run.
   - After the test run is completed, you can view the following results:
     - The **Results** tab displays the test as **Pass** or **Fail**.
9. Click to open the **Results** page, where you can view the results, reports, and logs of the completed test run.

Results

You have run a suite sequentially.

What to do next

You can view the following information:

- The details of the suites and individual test results from the **Run Details** tab.
- The console output, which includes error messages and logs. See Accessing console output on page 230.
- The test results, reports, and logs from the **Results** page after the test completes the run.
- The summary of the test run, which includes passed or failed status, count of passed or failed tests or test steps, and the verification points in the functional report. See Viewing a Functional Report on page 994.

Related information

Viewing the progress of running test assets on page 926

Running a suite concurrently

You can run a suite to run multiple tests and other suites in parallel to expedite the testing process.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Ensured that a suite is available in your project.
- Added tests or other suites to the suite.

About this task

You can run a suite concurrently to run multiple suites and tests simultaneously across several pods. You can set up the maximum number of concurrent pods, which defines how many suites or test assets can run in parallel. In the default set up, suites run their child assets sequentially on a single pod, with only one test asset running at a time.

When you select the **Run in Parallel** option, the **Continue on Fail** checkbox is disabled. Whether a test passes or fails, all test assets run independently and concurrently across the available pods.

Restriction: To run server suite assets on the Safari browser, you must use DevOps Test Runtime agents. When you use Safari, the tests in the suite run sequentially even when you selected the **Run in Parallel** option.

As the test runs, the three color-coded boxes display the counts of passed steps, failed steps, and errors. You can view the step details of the first failed step or error step by clicking the respective color-coded boxes. Also, you can view checkmarks for the steps that pass and crossmarks for the steps that fail.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Click the `.dts.yaml` file in the list of Suites to open the suite.
   The tests or suites are displayed on the **Suite Editor** page.
5. Click the **Configure settings and data** step.
   The **Settings** pane is displayed.
6. Select the **Run in parallel** checkbox to enable multiple tests to run at the same time.
   The **Continue on Fail** option gets disabled.
7. Enter the value in the **Maximum Concurrent Pods** field.

   Note:

   The **Maximum Concurrent Pods** setting defines the number of test assets from a suite that can run in parallel and the minimum value must be two. A lower value reduces server resource usage but increases run time, and a higher value speeds up the run time at the cost of higher resource consumption.

8. Click the  icon, and then select the **Run all > Run all on Default Cluster** option.

You can also select the **Run all on Personal Agent** option if the personal agent is in a ready state.

You can view the following stages on the progress bar during the test run:

- **In transition**: The test setup is in progress.
- **Running**: The test is actively running.
- **Complete**: The test run is completed. This section also displays the time taken for test run.
- During the **Running** phase, you can view the following details:
  - A checkmark  after each test or suite that passes, and a crossmark  for any failed test or suite.
  - The **Results** tab displays the test as **In Progress** during the test run.
- After the test run is completed, you can view the following results:
  - The **Results** tab displays the test as **Pass** or **Fail**.

9. Click **Analyze** to open the **Results** page, where you can view the results, reports, and logs of the completed test run, and perform the following steps:
   a. Verify that a **secondary** label appears for the secondary executions.
   b. Click on secondary and main/primary execution.
   c. Verify that the **Test Log** and **Functional Report** options are displayed only for the main/primary execution and does not display for the secondary executions.

Results

You have run a suite concurrently.

What to do next

You can view the following information:

- The details of the suites and individual test results from the **Run Details** tab.
- The console output, which includes error messages and logs. See Accessing console output on page 230.
- The test results, reports, and logs from the **Results** page after the test completes the run.
- The summary of the test run, which includes passed or failed status, count of passed or failed tests or test steps, and the verification points in the functional report. See Viewing a Functional Report on page 994.

Related information

Viewing the progress of running test assets on page 926

Management of DevOps Test Runtime agents on page 557

Running multiple suites or tests by using the wildcard pattern

When you want to enable a single suite to run multiple tests and suites based on their name, you can use a **Run** step to refer to tests and suites by using wildcard patterns. You can also use these patterns to specify the suites that must be excluded.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Test suite management on page 292 and Creating a suite on page 292.
- Ensured that there are multiple suites available in your project.

About this task

Within a **Run** step, wildcard patterns can be used to reference tests and suites. The following two wildcards are supported:

- **Single Asterisk (`*`)**: Matches zero or more characters within a single folder and it does not match the `/` character.
- **Double Asterisk (`**`)**: Matches zero or more characters including the `/` character. It can be used to match across multiple folder.

You can only match files located in the same folder as the test suite, or within its subfolders. You cannot use patterns to match files located in a different folder for a different application.

The following table lists the sample patterns:

Pattern	What it matches
*.dtx.yaml	All tests located in the same folder as the test suite.
a*.dtx.yaml	All tests starting with the letter "a" in the same folder as the suite.
**/*.dtx.yaml	All tests in subfolders under the folder containing the suite.
**/abc/*	All tests and suites located in any subfolder that is named ‘abc’ under the folder containing the suite.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Perform one of the following actions:
   - Create a new suite, open the suite, and then add a test or suite.
   - Open a suite from the list of Suites.

   The contents are displayed on the **Suite Editor** page.
5. Click a **Run** step.
   The step details pane is displayed.
6. Enter a wildcard pattern in **Path** to specify the tests or suites to be included in the test run.
7. Click **+Add Exclusion** if you want to exclude some of the suites.

A new text box is displayed.

8. Perform one of the following actions to specify the suite or test to be excluded from the test run:
   - Enter the file name.
   - Click the  button to browse and select the suite, and then click **Add**.
   - Drag a suite or test from the test navigator panel to **Suite Editor**.
   - Enter a wildcard pattern.
9. Click  to close the step details pane.
10. **Optional:** Perform the following steps to enable multiple tests to run at the same time:
    a. Click the **Configure settings and data** step, and then select the **Run in parallel** checkbox.
    b. Enter a value ≥ 2 in the **Maximum Concurrent Pods** field.

    Note: Tests that are resolved by a wildcard pattern run in parallel only if the suite contains a single run step. That is, the only step defined in the suite must be the test with the wildcard expression.

11. Click  to save the changes.
12. Click the  icon, and then select the **Run all > Run all on Default Cluster** option.

    You can also select the **Run all on Personal Agent** option if the personal agent is in a ready state.

    The **Results** tab displays the progress of the test run. After the test run is complete, the tab displays the verdict, and links to view the functional report, test log, and results page.

Results

You have run multiple suites or tests.

What to do next

You can view the following information:

- The test results, reports, and logs from the **Results** page after the test completes the run.
- The summary of the test run, which includes passed or failed status, count of passed or failed tests or test steps, and the verification points in the functional report. See Viewing a Functional Report on page 994.

Running suites on multiple browsers and devices

You can now configure a test suite to run across multiple web browsers and supported devices in one consolidated run. Instead of managing separate test runs for each browser and device, you can define all of your target browsers and devices within the configuration of the test suite. This simplifies cross-browser validation, saves time, and provides comprehensive report.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Ensured that a suite is available in your project.
- Added tests or other suites to the suite.

About this task

You can select the following browsers that are supported for cross-browser testing:

- Chrome
- Firefox
- Edge
- Safari

Also, you can select the following device profiles that are supported:

Device profile	Default browser
Phone (Small)	Edge
Phone (Medium)	Chrome
Phone (Large)	Edge
Phone (X-Large)	Chrome
Tablet (Small)	Edge
Tablet (Medium)	Chrome
Tablet (Large)	Edge
Tablet (X-Large)	Chrome

When the suite runs, all the tests in the suite run sequentially for each selected browser or device.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Double-click the `.dts.yaml` file in the list of Suites to open the suite.
   The tests or suites are displayed on the **Suite Editor** page.
5. Click the 

button, displayed on the **Suite Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

Alternatively, right-click the suite in the test navigator panel and select **Edit**.

The **Edit branch** dialog is displayed if there is no edit branch active. Otherwise, go to step .

6. Select your action to either select an existing Edit branch or create an Edit branch:
    - Perform the following steps to select an existing Edit branch:
        a. Select **Use an existing edit branch**.
        b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

        c. Click **Ok**.
    - Perform the following steps to create the Edit branch:
        a. Select **Create new edit branch**.
        b. Enter a name for the Edit branch in the **Branch Name** field.
        c. Click **Save**.
7. Click the **Configure settings and data** step 7 on page 306.
   The **Settings** pane is displayed.
8. Select the required browser and device profiles from the **Profile** drop-down list.
9. Click  to close the step details pane.
10. Click  to save the changes.
11. Click the  icon, and then select the **Run all > Run all on Default Cluster** option.

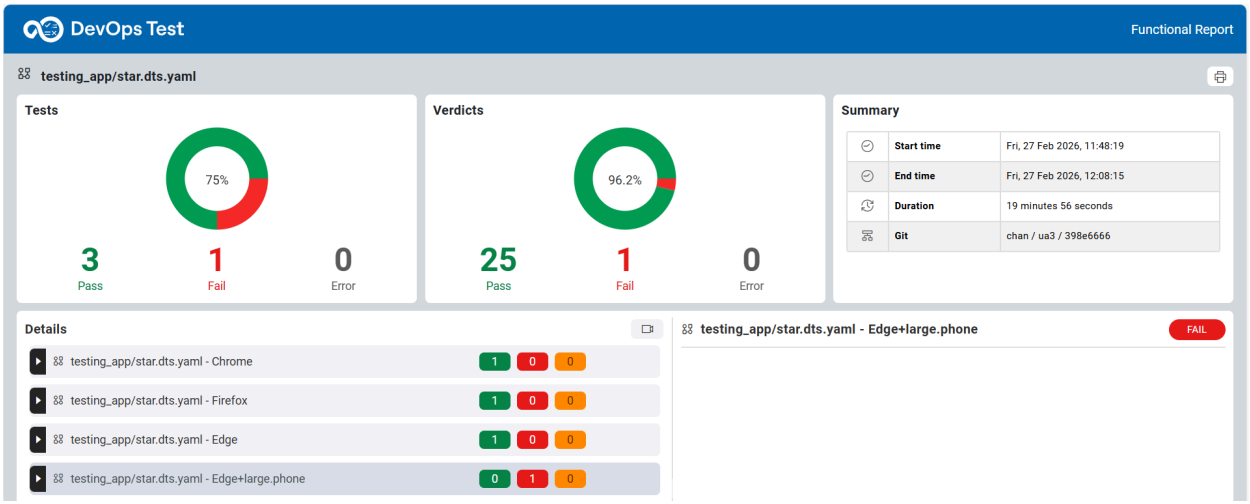
    You can also select the **Run all on Personal Agent** option if the personal agent is in a ready state.

    The **Results** tab displays the progress of the test run. After the test run is complete, the tab displays the verdict and links to view the functional report, test log, and results page.

    Alternatively, you can select the **Run with** option to choose specific browsers. You can select one or multiple browsers. The suite runs only on the selected browser for that run.

12. Click the **Functional Report**  icon to open the report in a separate window.

The functional report displays the results of the test run for each configured browser.



Results

You have run the test suite on multiple browsers and devices.

Accessing the console output

When you run suites, you can access the console output that provides information such as error messages, logs, and the test verdict. You can then use this output to troubleshoot issues by reviewing details, such as session IDs and error messages.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Ensure that your project contains a test.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Click the `.yaml` file in the list of Suites to open the suite.
   The test steps are displayed on the **Suite Editor** page.
5. Click the **Console** tab at the bottom of the **Test Editor** page.

   Alternatively, you can click the expand icon  .

   The **Console** output view is displayed.

6. Review the console output to identify the error messages and logs.
7. Perform the following actions to manage the console output:

Action	Description
Search a keyword in the console output	Enter a keyword in the **Search console** field, and the corresponding word is highlighted in the console output.
Select all console content	Right-click the console output area and, then select the **Select All** option to select all content.
Copy the selected content	Right-click the selected content and choose **Copy** to copy the content.
Expand console output view	Click the up arrow  to expand the console output view for more details.
Restore console output view	Click the down arrow  to return the console output page to its original size.
Close console output view	Click the drop-down icon  to close the console output view.

Results

You have accessed the console output content.

What to do next

You can perform any of the following tasks:

- Decide on the actions that you might want to take based on the logs.
- Rerun the test asset.

Stopping test runs within a suite

You can stop the tests that run within a suite when issues occur or if you decide to stop the run before completion.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Initiated a test run from the **Suite Editor** page.

About this task

You can stop multiple running tests from the **Suite Editor** page if the test run is in **In transition** or **Running** state.

1. Go to the **Suite Editor** page.
2. Click the **Stop playback** icon .
   The status of the tests is displayed as **Inconclusive** on the **Results** tab and **Stopped by User** on the progress bar.

Results

You have stopped test runs within a suite.

What to do next

You can re-initiate the test runs from the **Suite Editor** page.

Rerunning failed tests in a suite

When you run tests in a suite, a few tests might fail. You can rerun the failed tests instead of rerunning the suite.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Ensured that a suite is available in your project.

About this task

The **Run only failed** option is available only if the suite ran previously and contained failed tests. When the **Run only failed** option is available, you can select it to rerun the tests that failed.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Click the `.dts.yaml` file in the list of Suites to open the suite.
   The tests, suites or both are displayed on the **Suite Editor** page.
5. Click the icon, and then select the **Run failed tests > Default Cluster** option.

   You can also select the **Personal Agent** option if the personal agent is in a ready state.

   You can view the following stages on the progress bar during the test run:

- **In transition**: The test setup is in progress.
- **Running**: The test is actively running.
- **Complete**: The test run is completed.
- During the **Running** phase, you can view the following details:
  - A checkmark after each test or suite that passes, and a crossmark for any failed test or suite.
  - The **Results** tab displays the test as **In Progress** during the test run.
- After the test run is completed, you can view the following results:
  - The **Results** tab displays the test as **Pass** or **Fail**.

6. Click to open the **Results** page, where you can view the results, reports, and logs of the completed test run.

Results

You have rerun the failed tests in a suite.

What to do next

You can view the following information:

- The details of the suites and individual test results from the **Run Details** tab.
- The console output, which includes error messages and logs. See Accessing console output on page 230.
- The test results, reports, and logs from the **Results** page after the test completes the run.

Deleting a suite

You can delete a suite when it is no longer required in your project.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource. See Managing access to the server projects on page 1220.
- Ensured that your project contains a suite.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the suite assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The test navigator panel is displayed.
4. Select the suite that you want to delete from the test navigator panel.
   Result
   The suite opens in **Suite Editor**.

5. Click the 

button displayed on the **Suite Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   The **Edit branch** dialog is displayed if there is no edit branch active. Otherwise, go to step 7 on page 311.
6. Select an existing branch or create a new branch.
7. Click the **Delete** icon  .

   Alternatively, right-click the suite in the test navigator panel and select **Delete**.

   Result

   The **Delete Asset** dialog is displayed.
8. Click **Delete**.

Results

You have deleted a suite from your project.

Related information

Test navigator panel on page 200

Suite Editor page on page 203

Test variables

A test variable is a user-defined, name-value pair that stores and refers to information throughout a test. You can assign the value of a variable to a test element or assign the value of a UI object property to a test variable.

When you record a test, the URL of the application is stored in the default variable, APP_APP1_URL. You can create variables by using the following options:

- By using the **Add** 

option in the **Configure settings and data** pane.
- By using the **Add new variable** option when you assign the property of an object to a variable in the **Assign** step.
- By adding var_name: var_value under vars: section in the **Code** view of the test script in **Test Editor**.

You can use the variables for substitution. To substitute data from a test variable, you can use the **Details** section of the step details pane. You can perform the following actions with test variables:

- Provide a default value to the variable during variable creation.
- Change the value of a UI object by replacing the value with a variable. The UI object assumes the value of the variable when the test is run.
- Change the value of the variable using the 'Assign' statement. You can assign an object property to a variable.
- Modify the value of a variable by using the values extracted through RegEx patterns.

Types of variables

Variables are classified as input, local, or output variables based on their purpose and scope.

- **Input variables**  : Provide a way to receive values from calling tests or configurations. Use them to pass data into a test or module at runtime.
- **Local variables**  : Define values that are used within a test or a suite. These variables remain internal and are not accessible outside the test or suite in which they are declared.
- **Output variables**  : These variables are used to return values to the calling test or a suite. You can map these variables to existing or new variables to capture the returned data.

Variable state and promotion

Automated test recordings often contain sensitive information, such as passwords, personally identifiable information (PII), security keys, and authentication tokens. You can protect sensitive data by promoting a variable to an environment and marking the variable as a secret. When you promote a variable to an environment, you can mark it as a secret or a standard value. Promoting a variable as a secret automatically masks the value in functional reports, and execution output, ensuring that sensitive credentials remain secure across different application-under-test (AUT) environments. Also, when you promote a variable as a secret, in the configuration settings, the value field becomes blank and it displays a placeholder text 'Enter Value'. You can edit the value of secrets which are displayed as masked values from the **Security** page. The other standard promoted variables of type **Value** are used to reuse non-sensitive parameters that can be updated through the environments.

Each variable in the table includes a state indicator that helps you identify how the variable is used at run time.

Random number generation

You can generate random numbers and assign the value to a variable by defining a random expression. For example, you can define `${Math.floor(Math.random()*100)}` and assign it to a variable, `${value}`.

You can use a random expression in the following ways:

- Create a new variable in the **Configure settings and data** step, and assign the random expression to the variable.
- Create a new variable in the test, and insert an **Assign** step before any step where the random number is needed. Then, in the **Assign** step details, you can assign the random expression to the variable.

Assigning the random expression to the variable allows the random number to be stored in the variable and referenced throughout the test run.

Creating and assigning test variables

You can create a variable and initialize it to an alphanumeric value. Then, within the test, you can assign the value of the variable to a test element.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 313.
2. Select an existing branch or create a new branch.
3. Click the **Configure settings and data** step in **Test Editor**.
   The **Configure settings and data** pane is displayed.
4. Click **Add** icon 

in the **Variables** section.
5. Enter a name for the variable in **Name**.
   The name must start with an alphabet, $, or _ and the name must not be a reserved keyword or contain a space.
6. Enter a value for the variable.
   The newly created variable is listed in the **Variables** section.
   The default variable type is **Input**.
7. **Optional:** Click the input icon  to change the type to **Output** or **Local** in the **Modifier** column.
8. Perform the following steps to assign a variable to a test element:

   a. Select a step such as **Type** or **Select** step for which you want to replace the value with the value of a variable.
   The step details pane is displayed.

   b. Click the **Insert Variable** icon  to select a variable from the list or enter the variable name in the **Value** field for the selected **UI Object** in the following format:
   ${variable_name}

   For example, you can replace the text that is saved in the text field for `username` with the value of the variable, `user_name`.

   You can also replace the value of the object with the value of a variable under the **Object Identified By** section by clicking the **Insert Variable** icon  and selecting a variable from the list.

c. Close the step details pane.
The test step is updated with the variable that you chose.

9. Click the **Save** icon  to save the modifications.

Results

You have created a variable and assigned the value of the variable to a test element.

Note: The variables that you created in **Test Editor** are listed in the **Variables** section of the **Configure settings and data** pane as test variables.

What to do next

- Run the test in **Test Editor** and view the Functional Report to verify the test steps with the variable assignment. See Running a Web UI test on page 228.
- Start the test run from the **Execution** page, and in the **Execute test asset** dialog, navigate to the **Configure settings and data** option on the **Test Editor** page to view the Test variables defined in the test. See Configuring a run of a Web UI test on page 795 and Variables on page 804.

Related information

Viewing a Functional Report on page 994

Assigning an object property to a test variable

You can use the value of a UI object property and assign the value to a test variable in the Web UI tests. The process of assigning a variable to an object property is called a variable assignment.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

About this task

You can modify the value that is being assigned to the variable by using a regular expression. You can extract a substring from the content of an object property by using a regular expression, and then assign the substring to a variable. For example, you can extract *User Documentation* from the Content property of a hyperlink, and then assign the string to the *trimVar* variable. Further, you can include *trimVar* in the subsequent steps and run the test. You can view the variable value *User Documentation* in the Functional Report.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 315.
2. Select an existing branch or create a new branch.
3. Drag the **Assign** statement from the **Catalog > UI** pane and drop it under a test step as required.
   The **Assign** step is inserted.
4. Click the **Assign** step.
   The step details pane is displayed.
5. Hover over a UI element in the **Smartshot** view, right-click, and then select **Use this element as a step target**.
   Alternatively, you can drag and drop a UI element from the **Smartshot** view to the **Object identified by** section.

   The **UI Object** type is identified.
6. Enter a property name such as Content, Id, or XPath of the object that must be used for variable assignment in **UI Property**.
   If you entered the property name correctly, the value of the property is displayed under **Extracted Value**.
7. Select one of the following options from the **Assign to** list:
   - Select an existing variable.
   - Select **Add new variable**, enter a name for the variable, and then click **Add**.
8. **Optional:** Select the **Modify value** checkbox.
   The **Regex** field is displayed.
9. **Optional:** Enter a regular expression to extract a substring from the value of the object property in **Regex**.
   For example, if you want to extract the last 3 digits from `560064`, which is a Zip code, and assign it to the variable `'zipCode_3'`, you can use the expression `[0-9]{3}`.
10. Close the step details pane.
11. Click the **Save** icon  to save the modifications.

Results

You have assigned the value of a UI object property to a test variable in the Web UI test.

Note: The variables that you created in **Test Editor** are listed in the **Variables** section of the **Configure settings and data** pane as test variables.

What to do next

You can perform the following tasks:

- Run the test in **Test Editor** and view the Functional Report to verify the test steps with the variable assignment. See Running a Web UI test on page 228.
- Start the test run from the **Execution** page, and in the **Execute test asset** dialog, navigate to the **Configure settings and data** option on the **Test Editor** page to view the Test variables defined in the test. See Configuring a run of a Web UI test on page 795 and Variables on page 804.

Related information

Viewing a Functional Report on page 994

Adding an Assign step

You can use the **Assign** step to assign values from datasets to test variables or store values extracted from UI elements in variables during a test run.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test or suite.
- Ensured that the required variables have been created and are included in your Web UI test or suite.
- Ensured that your project contains a dataset.

About this task

You can use the **Assign** step to assign values to variables during a test run. You can assign values from datasets to support data-driven testing or extract values from UI elements and store them in variables. The assigned values can then be used in subsequent test steps to perform flexible data-driven testing across multiple scenarios

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

Alternatively, right-click the test in the test navigator panel and select **Edit**.

The **Edit branch** dialog is displayed.

If the **Edit branch** dialog is not displayed, then go to step 3 on page 317.

2. Select an existing branch or create a new branch.
3. Drag the **Assign** statement from the **Catalog > General** pane and drop it under a test step as required.
   The **Assign** step is inserted.

   Note:

   You can also use the **Assign** step within a **while** or **do-while** step, so that the **Assign** step retrieves the values from a dataset to process each row of the dataset iteratively and assign the retrieved values to the test variables.

4. Click the **Assign** step.

   The step details pane opens with the **Details** tab which was the default selection.

5. Select the required mode from the **Mode** list:
   - **UI**
   - **General**

   A confirmation dialog is displayed, indicating that switching modes clears the current step configuration.

6. Click **Yes**.

   Note:
   - The default mode depends on where the **Assign** step is added. When the **Assign** step is added within a **With** step, it opens in **UI** mode. When it is added outside a **With** step, it opens in **General** mode.
   - You can switch between the available modes by selecting a different option from the **Mode** list.
   - If you switch modes after configuring the step, select the required mode and then click **Yes** in the confirmation dialog. The existing configuration is cleared and the step opens in the selected mode.

7. You can perform the actions that are listed in the following table to configure assign during a test run:

Mode	Action
General	Perform the following steps: a. Select an existing dataset in the **Dataset** field. b. Click **+ Add Variable** to add a variable and assign the value from dataset. c. Select an existing variable from the **Select a variable to assign** drop down.

Mode	Action
	Note: You can also select **Create variable** from the variable list to create a new variable and assign the value from the dataset. The variable is displayed in the **Variables** section. d. Select the column in the dataset from which you want to assign the values during test run. e. Click the **Close** icon to close the run details pane.
UI	Perform the following steps: a. Hover over a UI element in the **Smartshot** view, right-click, and then select **Use this element as a step target**. Alternatively, you can drag and drop a UI element from the **Smartshot** view to the **Object identified by** section. The **UI Object** type is identified. b. Enter a property name such as Content, Id, or XPath of the object that must be used for variable assignment in **UI Property**. If you entered the property name correctly, the value of the property is displayed under **Extracted Value**. c. Select an existing variable, or Select **Add new variable**, enter a name for the variable, and then click **Add**.

8. Click the **Save** icon to save the modifications.

Results

You can configured the **Assign** step to assign values to test variables during a test run. Based on the selected mode, the Assign step assigns values from a dataset to variables or stores values extracted from UI elements in variables.

What to do next

You can run the test in **Test Editor** and view the Functional Report to verify the test steps with the variable assignment. See Running a Web UI test on page 228.

Assigning test variable values from a dataset by using the Run step

You can assign values from a dataset to test variables by using the **Run** step. This supports dynamic data substitution during test execution, allowing subsequent steps to use unique and data-driven values.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test or suite.
- Ensured that the required variables have been created and are included in your Web UI test or suite.
- Ensured that your project contains a dataset.

About this task

You can create datasets in Test Hub and assign their values to variables in a test or suite. This process helps you to run data-driven tests consistently with parameters and reduces the need to repeat steps. Dataset values are mapped to variables in a test or suite, to perform flexible data-driven testing across multiple scenarios.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test or suite in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 3 on page 319.
2. Select an existing branch or create a new branch.
3. Click the **Run** step to assign the dataset columns to variables.

   The run details pane opens with the **Details** tab which was the default selection. The variables in the test and suite are displayed in the **Inputs** section.
4. Select an existing dataset in the **Dataset** field.
5. Click the drop-down icon  in the variable.

   Result

   The variables in the parent test and column names in the selected datasets are listed.
6. Select the column in the dataset from which you want to assign the values during test run.
7. Click Close icon  to close the run details pane.
8. Click the **Save** icon  to save the modifications.

Results

You have assigned the dataset to test variables in a test or suite by using the **Run** step.

What to do next

You can run the test or suite in **Test Editor** sequentially or concurrently and view the Functional Report to verify the test steps with the variable assignment. See Running a Web UI test on page 228, Running a suite sequentially on page 299, or Running a suite concurrently on page 300.

Configuring test variables in the Run step of a test or suite

You can configure input and output variables in a **Run** step to pass values into and out of a test during test run.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test or suite.
- Ensured that the required variables have been created and are included in your Web UI test or suite.
- Ensured that your project contains a dataset.

About this task

When configuring input and output variables in a Run step, the following tooltips help indicate the status of each variable:

- **Using default variable**: The variable uses a predefined default value from the called test.
- **Value expected**: No value or mapping has been provided, and the variable does not have a default value. A value must be provided at runtime.
- **Value provided**: A value has been successfully assigned, either through direct input, variable mapping, or hard coding.
- **Using variable**: The variable is mapped to an existing variable in the parent test.
- **Value ignored**: The variable is configured as an output, but its returned value is not used or mapped in the calling test.
- **Value extracted**: The variable is configured as an output, and its value will be captured from the called test

These tooltips guide you in ensuring that all required values are correctly configured before running the test.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** is not displayed, then go to step 3 on page 320.
2. Select an existing branch or create a new branch.
3. Click the **Run** step in the **Test Editor**.

The run details pane opens with the **Details** tab selected by default. The variables in the test or suite are displayed in the **Inputs** and **Outputs** section.

4. You can perform following tasks to configure variables:

Task	Action
Define a value	Perform following steps: a. Click **Define variable** icon  . b. Type the value in the **Value** field. c. Click **Add**. The tooltip is updated as **Value provided**, and the define variable icon  is disabled.
Add a new variable in the suite	Perform the following steps: a. Select the drop-down icon  under the variable. b. Select **Add new variable**. The **Add Variable** dialog is displayed. c. Type the name of the variable in the **Name** field. d. Type the value in the **Variable** field. e. Click **Add**. A new variable is created and mapped to the selected variable.
Link the new variable to the parent variable	Perform the following steps: a. Click the drop-down icon under the variable. b. Select an existing variable from the parent test. The tooltip is updated as **Value provided**, and the define variable icon  is disabled.
Change the value of a variable	Type the new value directly under the variable name. The tooltip is updated as **Value provided**, and the define variable icon  is disabled.
Map output variables	Perform the following steps: a. Click the drop-down icon under the variable. b. Select an existing variable from the child test to capture the output value.

Task	Action
	The tooltip is updated as **Value extracted**, and the define variable icon is disabled. If no mapping is done, the tooltip is displayed as **Value ignored**.

5. Click the **Close** icon to close the run details pane.
6. Click the **Save** icon to save the modifications.

Results

You have configured input and output variables in the **Run** step of a test or suite.

What to do next

You can run the suite in **Test Editor** sequentially or concurrently and view the Functional Report to verify the test steps with the variable assignment. See Running a suite sequentially on page 299 or Running a suite concurrently on page 300.

Promoting test variables

You can promote a variable to map it to an environment value or to a secret that protects sensitive data through security groups. You can use the promote option when the value of a variable must be managed centrally across environments. When you promote a variable and mark it as a secret, the value is automatically masked in security group and reports.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Read and understood the information in Test variables on page 311.
- Ensured that your project contains a Web UI test and opened the test in **Test Editor**. See Editing a Web UI test on page 236.

About this task

When you promote a variable, the following changes occur based on its type:

If variable type is..	Then..
Value	• The **Source** column displays: `From Environment` • The variable is listed in the **Variables** section for the selected environment. • You can manage the value of the variable from the settings of the environment, and the same value is used when you run the test.
Secret	• The **Source** column displays: `From Secrets Group via Environment` • The variable is listed in the **Variables** section of the selected environment and the **Secrets** section of the selected security group. • The value of the variable is blank in the **Variables** section of the **Configure settings and data** pane and on the selected environment page. The value of the secret is masked in **Secrets** section of the selected security group. • You can manage the value of the variable from the settings of the security group, and the same value is used when you run the test.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 3 on page 323.
2. Select an existing branch or create a new branch.
3. Click the **Configure settings and data** step in **Test Editor**.
   The **Configure settings and data** pane is displayed.
4. Click the **Promote variable to environment** icon , and then perform the following steps in the **Promote Variable <variable_name>** dialog:

   a. Select an environment for mapping from the **Environment** drop-down list.

   b. Select the category as **Value** or **Secret** from the **Type** drop-down list.
   If you select the **Secret** option, then you must select the security group to add the variable from the **Security Group** drop-down list.

   c. Click **Promote**.

A confirmation message is displayed. The **Source** column for the promoted variable displays one of the following messages depending on the variable type:

- `From Environment` for a text value override from the environment
- `From Secrets Group via Environment` for a secret value override from the security group

5. Close the step details pane.
6. Click the **Save** icon to save the modifications.

Results

You have promoted a variable to map it to an environment value or to a secret-backed value.

What to do next

You must run the test in **Test Editor** and view the Functional Report to verify the values of the promoted variables. See Running a Web UI test on page 228.

Related information

Creating and assigning test variables on page 312

Creating an environment on page 338

Protecting test assets by using secrets on page 1235

Creating a dataset by using test variables

When you want to create a dataset by using variables that are configured in a test or a suite, you can do so in Test Editor. In the dataset, each variable gets displayed as a column, and you can update it by adding values to the columns.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Opened the test in **Test Editor**. See Editing a Web UI test on page 236.

1. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

The **Edit branch** dialog is displayed.

If the **Edit branch** dialog is not displayed, then go to step 3 on page 325.

2. Select an existing branch or create a new branch.
3. Click the **Configure settings and data** step in **Test Editor**.

   The **Configure settings and data** pane is displayed. If the test contains the configured variables, then they are listed under the **Data** section.
4. Click **+ Create Dataset** in the **Data** section.

   The **Create dataset** dialog is displayed.
5. Perform the following steps in the **Create dataset** dialog:
   a. Select an application for which you want to create a dataset from the **Application** drop-down list.
   b. Select an existing folder in **Location** to save the new dataset.
   c. Enter a name for the dataset in **Name**.
6. Click **Next**.

   All the variables available in the test are listed with a checkbox, and only the **Input** variables are selected.
7. Select the variables to be part of the dataset.
8. Click **Create**.

   A success message is displayed. The dataset created is listed under the logical folder `Datasets` within the selected application.

Results

You have created a dataset by using the variables configured in a test.

What to do next

You can view the dataset and add values to it. See Editing a dataset on page 433.

Related information

Adding an Assign step on page 242

Assigning test variable values from a dataset by using the Run step on page 318

Editing a dataset in Test Editor

You can now create and edit a dataset in **Test Editor** either through the catalog in the **Visual** view or through the written code in the **Code** view. However, for encrypted datasets, switching to the **Code** view is disabled to ensure data confidentiality.

Before you begin

1. Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
2. Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
3. Ensured that you have created a dataset or configured a repository that contains the dataset.

About this task

If you are a project **Owner** or **Tester** in HCL DevOps Test Hub (Test Hub), you can perform basic tasks in datasets by right-clicking any row, column, or cell in the dataset to organize your data more efficiently. For example, you can cut, copy, paste, insert, or delete rows or columns, show encrypted data, encrypt column data, or rename column names.

After you edit the dataset, you can save the changes made to the dataset, and then you can publish the dataset to the Git repository. If you save and close the edited dataset, the **Changes** page lists the edited dataset, and later you can publish it to the Git repository for other members to use.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.
4. Select the branch that contains your project with the datasets that you want to edit.

   The projects in the selected branch are displayed in the Test Editor navigator panel. The datasets are listed under the folder you created to contain the dataset within a logical folder named `Datasets`.
5. Click the dataset in the Test Editor navigator panel.

   The dataset opens in the right pane.

   The dataset is displayed in the **Visual** view, where you can add, modify, remove, import, or export data by using the toolbar actions or the right-click menu. If needed, you can switch to the **Code** view to edit the dataset in code form.

   Note: For encrypted columns, switching to the **Code** view is not allowed.

6. Click **Edit**.

   Alternatively, right-click the dataset in the Test Editor navigator panel, and then click **Edit**.

   The **Edit Branch** dialog is displayed.

   If the **Edit Branch** dialog is not displayed, then go to step 8 on page 326.
7. Select an existing branch or create a new branch.
8. Perform the following actions to use the top-level editing controls available in the toolbar:

Editing Options	Actions
Find and Replace	To find: a. Click the **Find and Replace** icon. b. Enter the content that you want to search for in the **Find** field. c. Select any of the following options to refine your search content:

Editing Options	Actions
	▪ Select the **Case sensitive** icon to search the content that matches the exact letter case entered in the **Find** field. ▪ Select the **Match entire cell contents** icon to search for cells that contain only the characters that you have entered in the **Find** field. ▪ Select the **Search using regular expression** icon to search the pattern that matches strings. For example, to search a cell that contains any number between 0 to 9, do the following: i. Enter \d in the **Find** field. ii. Select the **Search using regular expression** checkbox. iii. Click **Find**. d. Click the **Find** icon. If the text is found, the cell containing that text is selected. e. Click the **Find** icon again to find further instances of the search text. To find and replace: a. Click the **Find and Replace** icon. b. Enter the content that you want to search for in the **Find** field. c. Enter the content that you want to replace in the **Replace** field. d. Select any or all of the following options to find and replace the content more effectively: ▪ Select the **Case sensitive** icon to find the content that is the exact letter case of the content entered in the **Find** field. ▪ Select the **Match entire cell contents** icon to find and replace for cells that contain only the characters that you have entered in the **Find** and **Replace** fields. ▪ Select the **Search using regular expression** icon to find and replace the pattern that matches strings. e. Click the **Replace** icon to replace the individual instances. f. Click the **Replace All** icon to replace every instance of the content throughout the dataset.
Undo	a. Click the **Undo** icon. b. From the list, click the change that you want to undo. The **Undo** option reverts the change you made in the dataset.
Redo	a. Click the **Redo** icon. b. From the list, click the change that you want to redo.

Editing Options	Actions
Import	You can import variable data into a dataset from the following sources: ◦ When there is a large amount of data stored in a CSV file ◦ When you use the data fabrication feature to generate test data ◦ When you import data by using a database query **Importing data from a CSV file** a. Click the **Import** icon. b. Select **Import a CSV file** from the drop-down list. c. Click **Select file**, and choose the CSV file that contains variable data to import into the dataset. d. Enable the **First row contains headers** if your CSV file contains the header. e. Select one of the following import options to append or overwrite data in the dataset: ▪ Enable **Overwrite existing data** to add the rows and columns from the beginning of the dataset. ▪ Enable **Append existing data** to add rows and columns to the end of the dataset. f. Click **Import**. **Importing data by using data fabrication** a. Click the **Import** icon. b. Select **Fabricate data** from the drop-down list. c. Select a data definition from the drop-down list. d. Enter the number of rows that you want to generate in the **Number of Rows** field. e. Enter a seed value in the **Seed (optional)** field. f. Enable **Fabricate headers** if your schema contains the headers. g. Select one of the following options to append or overwrite data in the dataset:

Editing Options	Actions
	▪ Enable **Overwrite existing data** to add the rows and columns from the beginning of the dataset. ▪ Enable **Append existing data** to add rows and columns to the end of the dataset. h. Click **Import**. **Importing data by using a database query** a. Click the **Import** icon  . b. Select **Import a Database Query** from the drop-down list. c. Select a database query from the drop-down list. d. Enter the number of rows that you want to generate in the **Number of Rows** field. e. Select one of the following options to append or overwrite data in the dataset: ▪ Enable **Overwrite existing data** to add the rows and columns from the beginning of the dataset. ▪ Enable **Append existing data** to add rows and columns to the end of the dataset.
Export 	You can export variable data from a dataset to a CSV file to reuse the data in future tests. You must open the dataset from which you want to export the data. Click the **Export** icon  to download the dataset as a CSV file.
Configure	You can configure the dataset by clicking the **Menu** icon , and then selecting **Configure**. In the **Configure Dataset** dialog, you can change the row and column settings and configure the string values in the dataset that contain variable data for tests to use when they run. ◦ Select any of the separator values that you used in the CSV file. The available options are **Comma**, **Semicolon**, **Space**, **Tab**, and **Other**. If the CSV file uses any separator characters that are not listed, select **Other**, and then you can specify the character. For example, if the data in the CSV file is separated by a character #, then select the **Other** option and enter # in the field. ◦ Configure the following options to change the row and column settings:

Editing Options	Actions
	▪ **Column header** - Use an up-down control button to increment or decrement the value of the column header. ▪ **Data start point** - Use an up-down control button to increment or decrement the value of the data starting pointer. ▪ **Current row** - Use an up-down control button to increment or decrement the value of the current row. ◦ Configure the following options to change the string values in the dataset: ▪ **Treat as null** - Enter a string value that is to be treated as null when running the test. ▪ **Treat as empty** - Enter a string value that is to be treated as empty when running the test. For example, when you run the test, and the data *123* in the dataset is to be treated as empty, then you must specify *123* in the **Treat as empty** field. ▪ **Treat empty text as null** - Select this field when you want the dataset that contains any blank cells, and the value of those blank cells to be interpreted as null. ◦ Click **Update** to apply the changes.
Discard 	Click the **Menu** icon ⋮ and select **Discard** to discard the changes made to the dataset.

9. Perform the following actions using the editing options available in the column header:

Editing Options	Actions
Rename column 	Hover over the column header and click the **Rename column** icon  to update the column name.
Delete column 	Hover over the column header and click the **Delete column** icon  to delete the column.

10. Perform the following actions to use the cell-level editing options available in the right-click context menu.

Option	Actions
Cut	Right-click the cell and click the **Cut** icon to remove the content from the selected cell and paste it into another cell.
Copy	Right-click the cell and click the **Copy** icon to copy the content.
Paste	Right-click the cell and click the **Paste** icon to insert the copied or cut content.
Insert row above	Right-click the row and click **Insert row above** to add a new row above.
Insert row below	Right-click the row and click **Insert row below** to add a new row below.
Insert column left	Right-click the column and click **Insert column left** to add a column on the left.
Insert column right	Right-click the column and click **Insert column right** to add a column on the right.
Delete row(s)	Right-click the selected row and click **Delete row(s)**.
Delete column(s)	Right-click the selected column and click **Delete column(s)**.
Show encrypted data	Right-click the encrypted column and click the **Show encrypted data** icon to view the encrypted data. Right-click the decrypted column and select **Hide encrypted data** to encrypt the data again. **Important:** The **Show encrypted data** option is available only when you have added the encrypted dataset to the respective security group. For more information about adding an encrypted dataset to a security group, see Adding an encrypted dataset to a security group on page 447.
Encrypt column data	Perform the following steps to encrypt the column data: a. Right-click any cell in a column that you want to encrypt and select the **Encrypt column data** icon. The **Encrypt Column** window is displayed. b. Enter an encryption key in the **Encryption Key** field to encrypt the data in the column. **Remember:** When you have already encrypted other columns in the dataset, you must enter the same encryption key that you used

Option	Actions
	previously. You can use only one encryption key to encrypt columns in a dataset. **Important:** The encryption keys that you use to encrypt data in a dataset are not stored on the server, nor they can be retrieved from the server. Therefore, you must remember to store the encryption keys in a secure location. You must use the same encryption keys to perform the following operations: ▪ View the encrypted values ▪ Decrypt data ▪ Enable the use of the encrypted dataset during test runs c. Click **Encrypt Column**. Asterisks are displayed instead of actual data for the encrypted column.
Decrypt column data	Perform the following steps to decrypt the column data: a. Right-click the encrypted cells that display the contents with asterisks, and then select the **Decrypt column data** icon . The **Decrypt Column** window is displayed. b. Enter an encryption key in the **Encryption Key** field to decrypt the data in the column. c. Click **Decrypt Column**. Asterisks are replaced with the actual data in the decrypted column. When you run a test that uses a dataset that contains decrypted data, the variable data is substituted for the original data in the recorded test without prompting for the encryption.
Set as current row	During the test run, if you want variable data to be selected from the current row instead of the first row in a dataset, right-click any cell in a row and select the **Set as current row** icon . **When rows are deleted:**

Option	Actions
	If you delete any row between row 1 and the current row, the current row data is taken from the next row. For example, when you set the current row as 6, and then you delete any row between row 1 to row 6, the current row remains at row 6, but the content of row 7 is moved to row 6. **When rows are inserted:** If you insert any new row between row 1 to the current row, the current row data is taken from the previous row. For example, when you set the current row as 6, and then you insert any row between row 1 to row 6, the current row remains at row 6, but the content of row 5 is moved to row 6.

11. Click the **Save** icon to save the changes made to the dataset or the **Save All** icon in the Test Editor navigator panel to save all changes made to different datasets by using the same Edit branch.
12. Click the **Publish** icon to publish the dataset to the Git repository.

Results

You have edited the dataset in the **Test Editor**.

Multiple datasets in tests and test suites

You can configure multiple datasets in tests or test suites to provide additional data for variables during a test run. You can combine data from multiple datasets which supports flexible data-driven testing across different scenarios.

Overview

When working with datasets in a test or test suite, you can use a primary dataset along with additional datasets to provide values for variables.

The primary dataset is used as the main data source. Additional datasets provide supporting values when required. Therefore, you can use data from multiple datasets without duplicating test logic.

In **Test Editor**, you can add additional datasets when configuring tests and test suites.

You can configure multiple datasets in **Test Editor** as follows:

- Select a dataset as the primary dataset.
- Add one or more additional datasets, which are displayed indented under the primary dataset.

- Add additional datasets only after selecting the primary dataset.

Note: You cannot select the same dataset more than once.

Dataset behavior

The following behavior applies when you assign values to variables:

- Columns from both primary and additional datasets are available for mapping to variables.
- If a column name exists in multiple datasets, the value from the first matching dataset is used for mapping to variables. Datasets are considered in the configured order, and the primary dataset is considered first.
- If the same column exists in multiple datasets, the value from the first matching dataset is used.
- Additional datasets are used to provide values that are not available in the primary dataset.

Assigning test variable values from multiple datasets by using the Assign step

You can assign test variable values from multiple datasets to retrieve values from more than one dataset. This capability supports flexible data-driven testing across different scenarios and reduces the need to repeat test steps.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test.
- Ensured that the required variables have been created and are included in your Web UI test.
- Ensured that your project contains a dataset.

1. Open the required test in **Test Editor**.
2. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** is not displayed, then go to step 4 on page 334.
3. Select an existing branch or create a new branch.
4. Drag the **Assign** statement from the **Catalog > General** pane and drop it under a test step as required.
   The **Assign** step is inserted.

Note:

You can also use the **Assign** step within a **while** or **do-while** step, so that the **Assign** step retrieves the values from a dataset to process each row of the dataset iteratively and assign the retrieved values to the test variables.

5. Click the **Assign** step.

   The step details pane opens with the **Details** tab.

6. Perform the following steps to add multiple datasets:

   a. Select an existing dataset in the **Dataset** field.
   This dataset acts as the primary dataset. The **Add Dataset** option is enabled.

   b. Click **Add Dataset** and select additional datasets.

   You can add multiple datasets as required.

   If you want to remove a dataset, you can click **Remove Dataset** from the dataset list.

7. Click **Add Variable** to add a variable and assign the value from the dataset.
8. Select an existing variable from the **Select a variable to assign** drop down.
   Result
   The variable is displayed in the **Variables** section.
9. Click the drop-down icon for the variable.

   Note: You can also select **Create variable** from the list to create a new variable and assign the value from the dataset.

   Result
   The variables in the test and column names in the selected datasets are listed.
10. Select the column in the dataset from which you want to assign the values during test run.
11. Click the **Close** icon to close the step details pane.
12. Click the **Save** icon to save the modifications.

Results

You have assigned multiple datasets to test variables in a test by using the **Assign** step.

Assigning test variable values from multiple datasets by using the Run step

You can assign or modify test variable values from multiple datasets to retrieve values from more than one dataset. This capability supports flexible data-driven testing across different scenarios and reduces the need to repeat test steps.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test or suite.
- Ensured that the required variables have been created and are included in your Web UI test or suite.
- Ensured that your project contains a dataset.

1. Open the required test or suite in **Test Editor**.
2. Click the 

button displayed on the **Test Editor** page.

   The **Edit** button appears only when no active edit branch is selected.

   Alternatively, right-click the test in the test navigator panel and select **Edit**.

   The **Edit branch** dialog is displayed.
   If the **Edit branch** is not displayed, then go to step 4 on page 336.
3. Select an existing branch or create a new branch.
4. Click the **Run** step to assign the dataset columns to variables.

   The step details pane opens with the **Details** tab. The variables in the test and suite are displayed in the **Inputs** section.
5. Perform the following steps to add multiple datasets:

   a. Select an existing dataset in the **Dataset** field. This dataset acts as the primary dataset.
      This dataset acts as the primary dataset. The **Add Dataset** option is enabled.

   b. Click **Add Dataset** and select additional datasets. You can add multiple datasets as required.

      If you want to remove a dataset, you can click **Remove Dataset** from the dataset list.
6. Locate the variable in the **Inputs** variable section for which you want to assign a value.
7. Hover over the variable value field.

   A drop-down icon is displayed next to the field.
8. Click the drop-down icon  for the variable.
   Result
   The variables in the test and column names in the selected datasets are listed.
9. Select the column in the dataset from which you want to assign the values during the test run.

10. Click the **Close** icon to close the run details pane.
11. Click the **Save** icon to save the modifications.

Results

You have assigned multiple datasets to test variables in a test or suite by using the **Run** step.

Test environments

You can define custom environments to manage sets of variables such as application URLs and user credentials, and to configure profile overrides. When you run a test, the variables and override rules defined in the environment dynamically replace the values within your test script. You can easily change testing targets without modifying your test script by using environments because the environment-level configurations take precedence over the default values defined in the script.

You can create, edit, copy, move, and delete the environments. When you create an environment, a `.dte.yaml` file is created and listed under the **Environments** folder for the selected application in the test navigator panel.

Environments that you defined for an application are available for selection in the **Environment** drop-down list in the header of the **Test Editor** page. If no environments are defined in the selected project, the **Environment** list displays **No environments available**. The last environment selected by a user is retained as the pre-selected option in the list. This selection is used for authoring and execution flows unless you explicitly select a different environment. When you open a test that is associated with a different application, then the **Alert** icon is displayed in the **Environment** box indicating that the selected environment is not associated with the current asset's application or any of its dependent applications. You can select or unselect an environment from the list.

Environments that you defined for an application are also available for selection in the **ENVIRONMENT** tab of the **Execute test asset** dialog when you initiate a test run.

The environment page supports Value, Code, or Secret type of variables. The description of each type is as follows:

- **Value**: Stores the alphanumeric value that you enter as the variable value. The text is used as-is. For example, if you enter `1 + 2`, the value of the variable is the string `1 + 2`. The promoted variable as a value is listed on the environment page and you can manage the value from this page.
- **Code**: Evaluates the value as an inline JavaScript expression when the variable is referenced, and uses the result as the variable value. For example, enter `700 + Math.floor(Math.random() * 201)` to generate a random integer from 700 through 900. The variable uses the resulting number, not the JavaScript expression itself. You can also use this type for generated values such as ticket IDs or credit card numbers that follow a pattern.
- **Secret**: Stores sensitive data, such as passwords, authentication tokens, or credit card numbers. Secret values are managed thorough security groups and they are masked on the **Security** page. The promoted variable as a secret is listed on the environment page and the value of the secret is not displayed.

You can map an environment with a security group so that the variables promoted as secrets are listed in the mapped security group. You can also remove the mapping as required.

Profile overrides

In addition to managing variables, you can use environments to configure profile overrides. By using profile overrides, you can substitute a default profile, such as a specific browser or emulated device, with a different target profile, such as a cloud-native device or alternative browser, when you run tests.

In the **Execute test asset** dialog, you can either specify an **Environment** that carries override rules or select a browser or device profile in the **Browser** field. If you specify values in both the **Environment** and **Browser** fields, then the value in the **Browser** field takes priority.

Related information

Test navigator panel on page 200

Creating an environment

You can create an environment to define custom variables that override the variables defined in the test when you run a test or suite. You can also configure profile overrides so that cloud-native devices can replace the default browsers or device profiles when you run a test or suite.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Read and understood the information in Test environments on page 337.

About this task

You can use environments such as Dev, QA, or Staging to manage a set of variables and profiles with different values. When you run a test, the environment variables dynamically replace the matching variable values, and profile overrides replace the browser or device profiles defined in the test.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Navigate to **Author > Test Editor**.
   The **Test Editor** page and test navigator panel are displayed.
4. Perform any of the following actions:

- Click the **Action Menu** icon  and select **Environment**.
- Right-click the application folder in the test navigator panel and select **New > Environment**.

The **Edit branch** dialog is displayed.

If the **Edit branch** dialog is not displayed, then go to step 6 on page 339.

5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
6. Perform the following steps to configure the environment details in the **Create Environment** dialog.
   a. Select a target application from the **Application** drop-down list.
   b. Enter a unique name for your environment in the **Name** field.
7. Click **Done**.

   Result

   The new environment is created, and it opens in the **Test Editor** for further configuration.
8. Select a security group for the secrets to be added to the environment from the **Security Group** list.
9. Perform the following steps to add specific variables in the **Variables** section:
   a. Click **+ Create Variable**.
   b. Enter the name of the variable in the **Name** column.
      For example, `App_URL`, `username`, or `password`.
   c. Enter the corresponding value for the variable in the **Value** column.
   d. Optional: Click the variable type indicator in the first column to set the variable type.
      You can set the type as **Value**, **Code**, or **Secret**.
   e. Repeat these substeps to add all necessary variables for this environment.
10. Perform the following steps to define overrides for the default profiles in the **Profiles** section:

a. Click **+ Add Override**.

b. Select the name of the default browser or device profile from the **Profile** drop-down list.
For example, select `Phone (Small)` or `Chrome`.

c. Select a device from the list of devices for the configured device cloud from the **Override** drop-down list.

d. Repeat these substeps to add all necessary overrides for this environment.

11. Click the **Save** icon to save the details.

Results

You have created an environment.

What to do next

You can perform the following tasks:

- Run a server test asset. See Running server test assets on page 808 or Running server test assets on an agent on page 811.
- Run a server test suite asset. See Running server suite assets on page 816.

Modifying an environment

You can add, update, or remove variables and profile overrides that are part of the configured environments.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Read and understood the information in Test environments on page 337.
- Ensured that your project contains an environment.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Navigate to **Author > Test Editor**.
   Result

The **Test Editor** page and test navigator panel are displayed.

4. Double-click the environment that you want to modify in the test navigator panel, and then click **Edit**.
   Alternatively, right-click the environment in the test navigator panel and click **Edit**.
   The **Edit branch** dialog is displayed.
   If the **Edit branch** dialog is not displayed, then go to step 7 on page 341.
5. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
6. **Optional:** Select **Remove security group**, and then select a different security group for the secrets in the environment from the **Security Group** list.
7. Perform any of the following steps to modify your variables in the **Variables** section of the environment editor:

Action	Steps
Add a new variable	Click **+ Create Variable**, then enter the new variable name and value.
Edit an existing variable	Click inside the **Name** or **Value** field of the specific variable and type your updates.
Change variable type	Click the type indicator in the first column of the variable row, and then select the required type. The options are **Value**, **Secret**, and **Code**.
Delete a variable	Click the **Delete** icon next to the variable row that you want to remove.

8. Perform any of the following steps to modify overrides in the **Profiles** section of the environment editor:

Action	Steps
Add an override	Click **+ Add Override**, select the name of the default browser or device profile from **Profile**, and then select a device in the configured device cloud from **Override**.
Edit an existing override	Select different values from the **Profile** or **Override** field.
Delete an override	Click the **Delete** icon  next to the override row that you want to remove.

9. Click the **Save** icon  to save the details.

Results

You have modified the existing environment.

Catalog management

You can use the data catalog in HCL DevOps Test Hub (Test Hub) to browse and manage data generators used in test data creation. The catalog section provides access to built-in generators, allows you to create custom generators tailored to specific needs, and group them using custom categories.

You can access the data catalog for a project from **Author > Catalog** tab in the Test Hub. The catalog includes:

- **Basic Generators**: You can select any generator from the available generators listed in the **Catalog** tab of a data definition. The generators in turn contain different options and fields that generate data by using a built-in database in Test Hub. See Basic generators on page 342.

  Note: You must add at least one generator as a field in the data definition to generate test data.

- **Custom Generators**: Custom generators are modifications of the built-in generators that you modify any or all the fields in a generator to suit your specific requirements of your test data. The modified generator cannot be saved to the built-in generator and hence, you must save them as custom generators by providing a name to identify the custom generator. See Management of custom generators on page 391.
- **Custom Categories:** You can create a custom category in the catalog to contain the custom generators that you create. See Management of custom categories on page 398.

Basic generators

You can use the built-in basic generators in Data definitions with or without modifying the default configurations of the generators.

The Data fabrication feature of HCL DevOps Test Hub (Test Hub) contains a collection of built-in data generators available in the **Basic** category on the **Catalog** tab of a Data definition.

Basic generators are the built-in generators of the Data fabrication feature of Test Hub that you can use to generate diverse test data. These generators provide commonly used data types and patterns that you can select and configure in a Data definition to generate realistic test data.

The basic generators that you can use in Test Hub are as follows:

- Address
- Boolean
- Dataset
- DateTime
- Enumerator
- Function
- Name
- Number
- Phone
- Regex
- String
- Text
- UUID

Every basic generator has unique parameters with default features. When you click a basic generator that is added as a field in the Data definition, the corresponding unique parameters with their default configuration are displayed in a new dialog. The new dialog is named after the generator that you select.

You can change the default configurations of any basic generator only when it is added as a field in a Data definition. You can use the changed configurations for any basic generator in the Data definition by saving the Data definition. The changed configuration is applicable only in the Data definition that contains it and is not available for use from the **Catalog** tab.

If you want to save the modified generators, you can create them as custom generators and save them in a custom category in the **Catalog** tab.

You can find details about the parameters in each of the basic generators along with their default configuration and the configurations that you can change or modify. The parameters of the basic generators are as follows:

- Parameters of the Address generator on page 344
- Parameters of the Boolean generator on page 348
- Parameters of the Dataset generator on page 351
- Parameters of the DateTime generator on page 353
- Parameters of the Enumerator generator on page 357
- Parameters of the Function generator on page 361
- Parameters of the Name generator on page 367
- Parameters of the Number generator on page 371
- Parameters of the Phone generator on page 375

- Parameters of the Regex generator on page 379
- Parameters of the String generator on page 382
- Parameters of the Text generator on page 387
- Parameters of the UUID generator on page 390

Parameters of the Address generator

After you add the Address generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Address generator of a specific field in the Data definition in the Edit branch, the unique parameters of the Address generator that contains the existing configuration details for that field are displayed in the **Address** dialog.

The following table provides the list of UI elements of the **Address** dialog with their description, default features, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	**Description**	**Default feature**	**Action**
The **Generator** field	Displays the name of the selected generator, that is, **Address**.	The field is not editable.	NA
The **Distribution** field	Determines whether the addresses are randomly generated in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the addresses to follow a random distribution when generated in the test data. • Clear the **Random** option if you want the addresses to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for addresses using the **Percentage** field. For example, if you set the percentage distribution of `30%` to a configuration that contains addresses with the postal codes and `70%` to another configuration that contains the address with no postal codes, then the address column of the generated data displays the values accordingly.

UI element	Description	Default feature	Action
			The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Duplicates** field	Determines whether duplicate or unique addresses are present in the generated data. Contains the **Allowed** option.	The **Allowed** option is selected.	• Do not perform any action if you want the generated addresses to include duplicate entries. This means that some addresses might appear more than once in the generated data. • Clear the **Allowed** option if you want to ensure unique addresses with no duplicates. This will prevent any duplicate addresses from appearing in the generated data.
The **Blank percentage** field	Determines the frequency of blank addresses in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the secondary addresses fields in the generated data will be empty or blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated addresses when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of addresses to be generated. For example, if you enter `50`, the generated data will display the address column with a 50% occurrence rate.

UI element	Description	Default feature	Action
The **First Line** option	Determines whether the generated addresses include the first line of the address.	The **First Line** option is selected.	• Do not perform any action if you want the generated addresses to include the first line of the address. • Clear the **Allowed** option if you want to exclude the first line from the generated addresses.
The **Second Line** option	Determines whether the generated addresses include the second line of the address.	The **Second Line** option is selected.	• Do not perform any action if you want the generated addresses to include the second line of the address. • Clear the **Allowed** option if you want to exclude the second line from the generated addresses.
The **City** option	Determines whether the generated addresses include the city name.	The **City** option is selected.	• Do not perform any action if you want the generated addresses to include the name of the city. • Clear the **Allowed** option if you want to exclude the name of the city from the generated addresses.
The **Zip Code** option	Determines whether the generated addresses include the postal code.	The **Zip Code** option is selected.	• Do not perform any action if you want the generated addresses to include the postal code. • Clear the **Allowed** option if you want to exclude the postal code from the generated addresses.
The **Country** option	Determines whether the generated addresses include the country name.	The **Country** option is selected.	• Do not perform any action if you want the generated addresses to include the name of the country. • Clear the **Allowed** option if you want to exclude the name of the country from the generated addresses.

UI element	Description	Default feature	Action
The **Add new configuration** tab	You can add additional address configurations with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage** field is editable when the **Random** option is not selected. • The selected options from the previous configuration remain selected, and the unselected options remain unselected for the following elements: ◦ First Line ◦ Second Line ◦ City ◦ Zip Code ◦ Country	Click the tab to create additional address configurations. For example, you can have one configuration for addresses with the postal code, and another configuration for addresses with no postal code. You can similarly customize the values, distribution, and percentage as required. **Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **Address** generator is complete, you can use the **Address** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon to save the new generator with a unique name in the catalog. You can reuse the same configuration at a later time in different scenarios without having to set it up again. See Creating custom generators and categories from Data definitions on page 393.
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.

UI element	Description	Default feature	Action
The close icon 	NA	NA	Click the icon to exit the **Address** generator dialog.

Parameters of the Boolean generator

After you add the Boolean generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Boolean generator of a specific field in the Data definition in the Edit branch, the unique parameters of the Boolean generator that contains the existing configuration details for that field are displayed in the **Boolean** dialog.

The following table provides the list of UI elements of the **Boolean** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **Boolean**.	The name is not editable.	NA
The **Distribution** field	Determines if the Boolean values are randomly generated or not in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the Boolean values, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the Boolean to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the Boolean values using the **Percentage** field. For example, if you set the percentage distribution of `100%` and provide a value as `True`, then all the generated Boolean values in the corresponding column will be true.

UI element	Description	Default feature	Action
			The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Blank percentage** field	Determines the frequency of blank values in the corresponding column of the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the Boolean values in the corresponding column of the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated Boolean values when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of the Boolean values to be generated. For example, if you enter `50`, the generated data will display the Boolean column with a 50% occurrence rate.
The **Format** section	Determines the additional details about the format of the generated Boolean values for the selected configuration.	NA	Provide information about the format of the Boolean values for the selected configuration. For more details about the section, see The Format section on page 350.
The **Add new configuration** tab	You can add additional configurations for boolean values with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage** field is editable when the	Click the tab to create additional configurations for Boolean values. This is useful when you need different sets of Boolean values in your synthetic data. For example, you can have one configuration for `Is married` with a fixed distribution, and another configuration for `Has children` with a random

UI element	Description	Default feature	Action
		Random option is not selected. • The **Value** and **Distribution** fields display the same values that were provided in the previous configuration.	distribution. You can similarly customize the values, distribution, and percentage as required. **Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **Boolean** generator is complete, you can use the **Boolean** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon. You can reuse the same configuration at a later time in different scenarios without having to set it up again from scratch. See Creating custom generators and categories from Data definitions on page 393.
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.
The close icon	NA	NA	Click the icon to exit the **Boolean** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **Dataset** generator should be equal to `100`.

The Format section

You can add additional details about the format of the generated Boolean values for a selected configuration in the **Format** section of the **Boolean** dialog. The following table provides the list of UI elements of the section with their description, default features, and actions that you can perform in the section to customize the configuration:

UI element	Description	Default feature	Action
The **Value** fields	Contains two editable fields.	Displays `Yes` in the first field and `No` in the second field.	You can modify the default values to any other suitable Boolean values.
The **Distribution** fields	Contains two editable fields. The field determines the proportion of the entered values when you do not select the **Random** option.	Editable when **Random** is not selected. Displays 60% and 40% as default values.	If you did not select the **Random** option, enter a number from `0` to `100` in the **Distribution** field for both values. Note: The total percentage of the distribution corresponding to both values must be equal to 100.
The **Random** option	Determines if the Boolean values are randomly generated or not.	The option is selected.	• Do not perform any action if you want the Boolean values, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the Boolean values to follow a fixed distribution and to be non-random.

Parameters of the Dataset generator

After you add the Dataset generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Dataset generator of a specific field in the Data definition in the Edit branch, the unique parameters of the Dataset generator that contains the existing configuration details for that field are displayed in the **Dataset** dialog.

The following table provides the list of UI elements of the **Dataset** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **Dataset**.	The name is not editable.	NA
The **Blank**	Determines the frequency of blank datasets in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`.

UI element	Description	Default feature	Action
percentage field			For example, if you enter `10`, approximately 10% of the fields corresponding to datasets in the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated rows with datasets.	Editable.	Enter a number from `0` to `100` in the field to set the percentage of rows with datasets to be generated. For example, if you enter `50`, the generated data will display the datasets column with a 50% occurrence rate.
The **Dataset** field	Allows you to browse and select a dataset file from the system.	NA	Click the **Browse** option to select a dataset file from your system. The selected dataset value is displayed in the **Dataset** field.
The **Column** list	Allows you to select the desired column from the dataset for data generation.	NA	Click the list to select the column from the dataset.
The **Add new configuration** tab	You can add additional dataset configurations with specific dataset and column selections, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage** field is editable. • The **Dataset** and **Column** fields display the	Click the tab to create additional dataset configurations, and then add the required components for the additional dataset configuration. For example, if the previous configuration included a dataset with the Address column, you can add a new dataset in another configuration to include another dataset with the Email column.

UI element	Description	Default feature	Action
		same values that were provided in the previous configuration.	**Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **Dataset** generator is complete, you can use the **Dataset** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon. You can reuse the same configuration at a later time in different scenarios without having to set it up again from scratch. See Creating custom generators and categories from Data definitions on page 393.
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.
The close icon	NA	NA	Click the icon to exit the **Dataset** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **Dataset** generator should be equal to `100`.

Parameters of the DateTime generator

After you add the DateTime generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the DateTime generator of a specific field in the Data definition in the Edit branch, the unique parameters of the DateTime generator that contains the existing configuration details for that field are displayed in the **DateTime** dialog.

The following table provides the list of UI elements of the **DateTime** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **DateTime**.	The name is not editable.	NA
The **Distribution** field	Determines if the date and time are randomly generated or not in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the date and time, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the date and time to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the date and time using the **Percentage** field. For example, if you set the percentage distribution of `30%` to a configuration that contains the dates of weekdays with a fixed time range and `70%` to another configuration that contains the dates of weekends with a different time range, then the enumerator column of the generated data displays the values accordingly. The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Duplicates** field	Determines if duplicate or unique date and time are present in the generated data. Contains the **Allowed** option.	The **Allowed** option is selected.	• Do not perform any action if you want the generated date and time to include duplicate entries. This means that some of the date and time values might appear more than once in the generated data.

UI element	Description	Default feature	Action
			• Clear the **Allowed** option if you want to ensure a unique date and time with no duplicates. This will prevent any duplicate date and time from appearing in the generated data.
The **Blank percentage** field	Determines the frequency of blank date and time values in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the date and time values in the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated date and time values when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of the date and time values to be generated. For example, if you enter `50`, the generated data will display the column corresponding to the date and time values with a 50% occurrence rate.
The **Pattern** list	Allows you to select the date pattern for data generation.	The default pattern is **mm/dd/yyyy**.	Select the required date pattern from the list of available options.

UI element	Description	Default feature	Action
The **Limit dates** section	Allows you to define limitations on the generated dates.	NA	• Either enter the dates in the **From** and **To** fields in the format of dd/mm/yyyy or select the dates from the corresponding calendar icons to reflect in the fields. • Select the days of the week (S, M, T, W, T, F, S) in the **Exclude** section to exclude those days from the **From** and **To** date fields. • Enter multiple dates in the **Specific dates to exclude** field, separated by commas, to exclude specific dates from the generation.
The **Limit times** section	Allows you to define limitations on the generated times.	NA	Enter the times in the **From** and **To** fields in the AM and PM format.
The **Add new configuration** tab	You can add additional date and time configurations with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage** field is editable when the **Random** option is not selected. • The following sections display	Click the tab and then add the required components for the additional date and time configuration. For example, you can have one configuration for weekdays with a fixed time range, and another configuration for weekends with a different time range. **Note:** You can click the delete icon to delete any additional configuration that you do not need.

UI element	Description	Default feature	Action
		the same values that were provided in the previous configuration: ◦ Limit dates ◦ Limit times	
The **Save to catalog** option	After the customization of the **DateTime** generator is complete, you can use the **DateTime** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon  . You can reuse the same configuration at a later time in different scenarios without having to set it up again from scratch. See Creating custom generators and categories from Data definitions on page 393.
The close icon 	NA	NA	Click the icon to exit the **DateTime** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **DateTime** generator should be equal to `100`.

Parameters of the Enumerator generator

After you add the Enumerator generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Enumerator generator of a specific field in the Data definition in the Edit branch, the unique parameters of the Enumerator generator that contains the existing configuration details for that field are displayed in the **Enumerator** dialog.

The following table provides the list of UI elements of the **Enumerator** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **Enumerator**.	The name is not editable.	NA
The **Distribution** field	Determines if the enumerator values are randomly generated or not in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the enumerator values, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the enumerator values to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the enumerator values using the **Percentage** field. For example, if you set the percentage distribution of `30%` to a configuration that contains the names of colors and `70%` to another configuration that contains the names of shapes, then the enumerator column of the generated data displays the values accordingly. The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Blank perc**	Determines the frequency of blank enumerator values in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the enumerator values in the generated data will be blank.

UI element	Description	Default feature	Action
entage field			
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated enumerated values when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of enumerator values to be generated. For example, if you enter `50`, the generated data will display the enumerator column with a 50% occurrence rate.
The **Format** section	Determines the additional details about the format of the generated enumerator values for the selected configuration.	NA	Provide information about the format of the enumerator values for the selected configuration. For more details about the section, see The Format section on page 360.
The **Add new configuration** tab	You can add additional enumerator value configurations with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage**	Click the tab and then add the required components for the additional configuration. For example, you can have one configuration for colors with a fixed distribution, and another configuration for shapes with a random distribution. You can similarly customize the values, distribution, and percentage as required.

UI element	Description	Default feature	Action
		field is editable when the **Random** option is not selected. • The **Value** and **Distribution** fields display the same values that were provided in the previous configuration.	**Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **Enumerator** generator is complete, you can use the **Enumerator** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon. You can reuse the same configuration at a later time in different scenarios without having to set it up again from scratch. See Creating custom generators and categories from Data definitions on page 393.
The close icon	NA	NA	Click the icon to exit the **Enumerator** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **Enumerator** generator should be equal to `100`.

The Format section

You can add additional details about the format of the generated enumerator values for a selected configuration in the **Format** section of the **Enumerator** dialog. The following table provides the list of UI elements of the section with their description, default features, and actions that you can perform in the section to customize the configuration:

UI element	Description	Default feature	Action
The **Value** fields	You can add multiple editable fields and provide desired values.	Displays three editable fields with `RED`, `GREEN`, and `BLUE` as the corresponding values.	You can modify the default values to any other suitable enumerator values or add additional values.
The **Distribution** fields	You can add multiple editable fields. The field determines the proportion of the entered values when you do not select the **Random** option.	Editable when **Random** is not selected. Displays 20%, 30%, and 50% as default values.	If you did not select the **Random** option, enter a number from `0` to `100` in the **Distribution** fields for all the values. **Note:** The total percentage of distribution corresponding to both values must be equal to 100.
The **Random** option	Determines if the enumerator values are randomly generated or not.	The option is selected.	• Do not perform any action if you want the enumerator values, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the enumerator values to follow a fixed distribution and to be non-random.

Parameters of the Function generator

After you add the Function generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Function generator of a specific field in the Data definition in the Edit branch, the unique parameters of the Function generator that contains the existing configuration details for that field are displayed in the **Function** dialog.

The following table provides the list of UI elements of the **Function** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Gene**	Displays the name of the selected generator, that is, **Function**.	The name is not editable.	NA

UI element	Description	Default feature	Action
rator field			
The **Distribution** field	Determines if the values are randomly generated or not in the function column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the values, when generated in the function column, to follow a random distribution. • Clear the **Random** option if you want the function values to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the Function values using the **Percentage** field. For example, if you set the percentage distribution of `30%` to a configuration that contains the names in uppercase and `70%` to another configuration that contains the names in lowercase, then the Function column of the generated data displays the values accordingly. The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Duplicates** field	Determines if duplicate or unique function values are present in the generated data. Contains the **Allowed** option.	The **Allowed** option is selected.	• Do not perform any action if you want the generated function values to include duplicate entries. • Clear the **Allowed** option if you want to ensure unique function values with no duplicates. This will prevent any duplicate values from appearing in the function column.

UI element	Description	Default feature	Action
The **Blank percentage** field	Determines the frequency of blank function values in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the function values in the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated function values when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of function values to be generated. For example, if you enter `50`, the generated data will display the function column with a 50% occurrence rate.
The **Function** section	Determines the additional details about the function for the selected configuration.	NA	Provide information about the function for the selected configuration. For more details about the section, see The Function section on page 365.
The **Add new configuration** tab	You can add additional function configurations with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage**	Click the tab and then add the required components for the additional configuration. For example, you can have one configuration for uppercase names with a 30% distribution and another configuration for lowercase names with a 70% distribution. You can similarly customize the values, distribution, and percentage as required.

UI element	Description	Default feature	Action
		field is editable when the **Random** option is not selected. • The **Function** field displays the same values that were provided in the previous configuration.	**Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **Function** generator is complete, you can use the **Function** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon. You can reuse the same configuration at a later time in different scenarios without having to set it up again from scratch. See Creating custom generators and categories from Data definitions on page 393.
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.
The close icon	NA	NA	Click the icon to exit the **Function** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **Function** generator should be equal to `100`.

The Function section

You can add additional details about the function for a selected configuration in the **Function** section of the **Function** dialog. The following table provides the list of UI elements of the section with their description, default features, and actions that you can perform in the section to customize the configuration:

UI element	Description	Default feature	Action
The **Function** field	Combines the built-in methods and the Data definition fields to create custom functions.	Empty	You can either enter a function or select values from the **Fields** and **Methods** scroll areas to create a function.
The **Items for function** field	Finds matching results in both the **Methods** and **Fields** scroll areas based on the entered text.	NA	You can type a few letters in the **Items for function** field, and matching results from both the **Methods** and **Fields** scroll areas will be displayed.
The **Methods** scroll area	Displays predefined Java methods that you can use to perform specific operations on entity fields within the function.	NA	When you double-click on an available method, the corresponding parameter is added to the **Function** field. When you hover over a method in the scroll area, the corresponding function, description, and example are displayed. The Data fabrication feature of HCL DevOps Test Hub (Test Hub) supports a variety of built-in methods. See The supported built-in methods on page 365.
The **Fields** scroll area	Displays the names of all the fields used in the Data definition. You can double-click a field to add it to the **Function** field and incorporate it into the custom function.	NA	When you double-click an item in the **Fields** scroll area, the corresponding parameter is added to the **Function** field.

The supported built-in methods

The Data fabrication feature of Test Hub supports a variety of built-in methods. The following table provides the list of built-in methods, their description, and relevant examples that you can use as functions:

Method	Description	Example
`replace(oldString, newString)`	Replaces all occurrences of the old string with the new string	`F$address.replace('road', 'street')`
`format(formatString, args)`	Returns a formatted string using format string and arguments	`String.format('Hello %s', F$name)`
`split(matchString)`	Splits the string around the given matching string	`F$name.split(" ")[0].concat('@gmail.com')`. In this example, the name is split where a white space is present and only the first part of the split is considered. Also, the first part of the split is appended with `@gmail.com`. If the input name is `Amy Smith`, then the result of the function is `Amy@gmail.com`.
`join(delimiter, stringElements)`	Joins string elements by a delimiter.	`String.join(',', F$id, F$name)`
`length()`	Returns the length of the string	`F$name.length`
`isBlank()`	Returns True only if the string has zero length or contains only white spaces	`F$address.isBlank`
`trim()`	Removes leading and trailing spaces of the string	`F$phone.trim`
`toUpperCase()`	Converts all characters into uppercase	`F$name.toUpperCase`
`toLowerCase()`	Converts all characters into lowercase	`F$name.toLowerCase`

Method	Description	Example
`concat`	Concatenates the specified string to the end of the selected string	`'Hello '.concat(F$name)`

Parameters of the Name generator

After you add the Name generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Name generator of a specific field in the Data definition in the edit branch, the unique parameters of the Name generator that contains the existing configuration details for that field are displayed in the **Name** dialog.

The following table provides the list of UI elements of the **Name** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **Name**.	The name is not editable.	NA
The **Distribution** field	Determines if the names are randomly generated or not in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the names, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the names to follow a fixed distribution and to be non-random, when you add a new configuration. The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.

UI element	Description	Default feature	Action
The **Duplicates** field	Determines if duplicate or unique names are present in the generated data. Contains the **Allowed** option.	The **Allowed** option is selected.	• Do not perform any action if you want the generated names to include duplicate entries. • Clear the **Allowed** option if you want to ensure unique names with no duplicates. This will prevent any duplicate names from appearing in the generated data. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the names using the **Percentage** field. For example, if you set the percentage distribution of `30%` to a configuration that contains the names with titles and `70%` to another configuration that does not contain the title, then the names column of the generated data displays the values accordingly.
The **Blank percentage** field	Determines the frequency of blank names in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the names in the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percent**	Determines the proportion of the generated names when	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the

UI element	Description	Default feature	Action
age field	you do not select the **Random** option.		**Percentage** field to set the percentage of names to be generated. For example, if you enter `50`, the generated data will display the names column with a 50% occurrence rate.
The **Title** option	Determines whether the generated names include the title.	The **Title** option is selected.	• Do not perform any action if you want the generated names to include the title. • Clear the **Allowed** option if you want to exclude the title from the generated names.
The **First Name** option	Determines whether the generated names include the first name.	The **First Name** option is selected.	• Do not perform any action if you want the generated names to include the first name. • Clear the **Allowed** option if you want to exclude the first name from the generated names.
The **Last Name** option	Determines whether the generated names include the last name.	The **Last Name** option is selected.	• Do not perform any action if you want the generated names to include the last name. • Clear the **Allowed** option if you want to exclude the last name from the generated names.
The **Add new configuration** tab	You can add additional name configurations with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage** field is editable when the	Click the tab and then add the required components for the additional configuration. For example, you can have one configuration for names with titles and another configuration for names without titles. You can similarly customize the values, distribution, and percentage as required.

UI element	Description	Default feature	Action
		Random option is not selected. • The selected options from the previous configuration remain selected, and the unselected options remain unselected for the following elements: ◦ Title ◦ First Name ◦ Last Name	**Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **Name** generator is complete, you can use the **Name** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon. You can reuse the same configuration at a later time in different scenarios without having to set it up again from the scratch. See Creating custom generators and categories from Data definitions on page 393.
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.
The close icon	NA	NA	Click the icon to exit the **Name** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **Name** generator should be equal to `100`.

Parameters of the Number generator

After you add the Number generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Number generator of a specific field in the Data definition in the Edit branch, the unique parameters of the Number generator that contains the existing configuration details for that field are displayed in the **Number** dialog.

The following table provides the list of UI elements of the **Number** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	**Description**	**Default feature**	**Action**
The **Generator** field	Displays the name of the selected generator, that is, **Number**.	The name is not editable.	NA
The **Distribution** field	Determines if the numbers are randomly generated or not in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the numbers, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the numbers to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the numbers using the **Percentage** field. For example, if you set the percentage distribution of `30%` to a configuration that contains numbers with 2 decimal values and `70%` to another configuration that contains numbers with no decimal values, then the number column of the generated data displays the values accordingly.

UI element	Description	Default feature	Action
			The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Duplicates** field	Determines if duplicate or unique numbers are present in the generated data. Contains the **Allowed** option.	The **Allowed** option is selected.	• Do not perform any action if you want the generated numbers to include duplicate entries. • Clear the **Allowed** option if you want to ensure unique numbers with no duplicates. This will prevent any duplicate numbers from appearing in the generated data.
The **Blank percentage** field	Determines the frequency of blank numbers in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the numbers in the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated numbers when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of numbers to be generated. For example, if you enter `50`, the generated data will display the number column with a 50% occurrence rate.

UI element	Description	Default feature	Action
The **Numbers** section	Determines either the range of numbers or the number of digits that appear in the generated data. You can select either the **Range** or **# of Digits** button and then enter the corresponding values in the **Min** and **Max** fields.	The **# of Digits** button is selected with a minimum digit of **1** and a maximum digit of **10**.	• Enter numeric values in the **Min** and **Max** fields to set the minimum and maximum number of digits respectively. The **# of Digits** button is selected as the default feature. • Click the **Range** button to switch to range-based number generation and then enter numeric values in the **Min** and **Max** fields to set the minimum and maximum range of numbers.
The **Decimal places** section	Determines the range of decimal places that are displayed in the generated numbers.	The **Min** and **Max** fields in the section are blank.	Enter numeric values in the **Min** and **Max** fields to set the minimum and maximum number of decimal places, respectively, in the generated numbers.
The **Add new configuration** tab	You can add additional configurations for numbers with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage**	Click the tab and then add the required components for the additional configuration. For example, you can have one configuration for numbers with fixed decimal places and another configuration with no decimal places. You can similarly customize the values, distribution, and percentage as required.

UI element	Description	Default feature	Action
		field is editable when the **Random** option is not selected. • The fields or options in the following sections display the same values that were provided in the previous configuration: ◦ Numbers ◦ Decimal places	**Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **Number** generator is complete, you can use the **Number** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon. You can reuse the same configuration at a later time in different scenarios without having to set it up again from the scratch. See Creating custom generators and categories from Data definitions on page 393.
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.
The close	NA	NA	Click the icon to exit the **Number** generator dialog.

UI element	Description	Default feature	Action
icon 			

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **Number** generator should be equal to `100`.

Parameters of the Phone generator

After you add the Phone generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Phone generator of a specific field in the Data definition in the edit branch, the unique parameters of the Phone generator that contains the existing configuration details for that field are displayed in the **Phone** dialog.

The following table provides the list of UI elements of the **Phone** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **Phone**.	The name is not editable.	NA
The **Distribution** field	Determines if the phone numbers are randomly generated in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the phone numbers, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the phone numbers to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the phone numbers using the **Percentage** field. For example, if you set the percentage distribution of `30%` to a configuration that

UI element	Description	Default feature	Action
			contains a specific pattern with the UK as the country and `70%` to another configuration that contains another pattern with France as the country, then the column of the phone number of the generated data displays the values accordingly. The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Duplicates** field	Determines if duplicate or unique phone numbers are present in the generated data. Contains the **Allowed** option.	The **Allowed** option is selected.	• Do not perform any action if you want the generated phone numbers to include duplicate entries. • Clear the **Allowed** option if you want to ensure unique phone numbers with no duplicates. This will prevent any duplicate phone numbers from appearing in the generated data.
The **Blank percentage** field	Determines the frequency of blank phone numbers in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the phone numbers in the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percent**	Determines the proportion of the generated phone numbers when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of phone numbers to be generated.

UI element	Description	Default feature	Action
age field			For example, if you enter `50`, the generated data will display the phone numbers column with a 50% occurrence rate.
The **Format** list	Determines the desired format for the generated phone numbers.	NA	Select a specific format from the list.
The **Country** list	Determines the country for which you want to generate phone numbers. <Not sure why this list is provided. Selecting or not selecting from this list has no changes in the generated data - Need to confirm with the team>	NA	Select a specific country name from the list.
The **Add new configuration** tab	You can add additional phone number configurations with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable.	Click the tab and then add the required components for the additional configuration. For example, you can have one configuration for colors with a fixed distribution, and another configuration for shapes with a random distribution. You can similarly customize the values, distribution, and percentage as required.

UI element	Description	Default feature	Action
		• The **Percentage** field is editable when the **Random** option is not selected. • The **Format** and **Country** fields display the same values that were provided in the previous configuration.	**Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **Phone** generator is complete, you can use the **Phone** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon. You can reuse the same configuration at a later time in different scenarios without having to set it up again from the scratch. See Creating custom generators and categories from Data definitions on page 393.
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.

UI element	Description	Default feature	Action
The close icon	NA	NA	Click the icon to exit the **Phone** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **Phone** generator should be equal to `100`.

Parameters of the Regex generator

After you add the Regex generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Regex generator of a specific field in the Data definition in the edit branch, the unique parameters of the Regex generator that contains the existing configuration details for that field are displayed in the **Regex** dialog.

The following table provides the list of UI elements of the **Regex** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **Regex**.	The name is not editable.	NA
The **Distribution** field	Determines if the Regex values are randomly generated in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the Regex values, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the Regex values to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the Regex values using the **Percentage** field.

UI element	Description	Default feature	Action
			For example, if you set the percentage distribution of `30%` to a configuration that contains Regex values of ten numeric values and `70%` to another configuration that contains Regex values of five alphabets, then the Regex column of the generated data displays the values accordingly. The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Duplicates** field	Determines if duplicate or unique Regex values are present in the generated data. Contains the **Allowed** option.	The **Allowed** option is selected.	• Do not perform any action if you want the generated Regex values to include duplicate entries. • Clear the **Allowed** option if you want to ensure unique Regex values with no duplicates. This will prevent any duplicate Regex values from appearing in the generated data.
The **Blank percentage** field	Determines the frequency of blank Regex values in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the Regex values in the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated Regex values when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of the Regex values to be generated.

UI element	Description	Default feature	Action
			For example, if you enter `50`, the generated data will display the column of the Regex values with a 50% occurrence rate.
The **Regex** field	Determines the custom regular expression pattern to generate data that matches the given pattern.	Displays **[a-z]{4}**.	• Edit the default Regex pattern and enter the desired custom Regex pattern. • Perform the following actions to use the helper, if you are not sure of the required Regex format: ◦ Click the **Helper** option. The **Regex helper** dialog appears. ◦ Enter the sample texts in the fields provided, and then click the **Create regex** option. The Regex format is displayed in the **Regex generated** field. ◦ Click **Save** to close the dialog and display the Regex value in the **Regex** field of the generator dialog.
The **Add new configuration** tab	You can add additional configurations of the Regex values with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage** field is editable when the **Random** option is not selected. • The **Regex** fields display	Click the tab and then add the required components for the additional configuration. For example, you can have one configuration that contains ten numerals, and another configuration that contains five alphabets. You can similarly customize the values, distribution, and percentage as required. You can click the delete icon to delete any additional configuration that you do not need.

UI element	Description	Default feature	Action
		the same values that were provided in the previous configuration.	
The **Save to catalog** option	After the customization of the **Regex** generator is complete, you can use the **Regex** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon . You can reuse the same configuration at a later time in different scenarios without having to set it up again from the scratch. See Creating custom generators and categories from Data definitions on page 393.
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.
The close icon	NA	NA	Click the icon to exit the **Regex** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **Regex** generator should be equal to `100`.

Parameters of the String generator

After you add the String generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the String generator of a specific field in the Data definition in the Edit branch, the unique parameters of the String generator that contains the existing configuration details for that field are displayed in the **String** dialog.

The following table provides the list of UI elements of the **String** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **String**.	The name is not editable.	NA
The **Distribution** field	Determines if the strings are randomly generated or not in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the strings, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the strings to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the strings using the **Percentage** field. For example, if you set the percentage distribution of `30%` to a configuration that contains strings with special characters and `70%` to another configuration that contains strings with no special characters, then the string column of the generated data displays the values accordingly. The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Duplicates** field	Determines if duplicate or unique strings are present in the generated data. Contains the **Allowed** option.	The **Allowed** option is selected.	• Do not perform any action if you want the generated strings to include duplicate entries. • Clear the **Allowed** option if you want to ensure unique strings with no duplicates. This will prevent any duplicate strings from appearing in the generated data.

UI element	Description	Default feature	Action
The **Blank percentage** field	Determines the frequency of blank strings in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the strings in the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated strings when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of strings to be generated. For example, if you enter `50`, the generated data will display the string column with a 50% occurrence rate.
The **Minimum characters** field	Determines the minimum number of digits for the string in the generated data.	Displays `10`.	Edit the default number and enter the custom minimum value, if required.
The **Maximum characters** field	Determines the maximum number of digits for the string in the generated data.	Displays `1,000`.	Edit the default number and enter the custom maximum value, if required.
The **Include special characters** option	Determines whether special characters must be included in the string.	The option is not selected.	Select the option if you want to include special characters in the string.

UI element	Description	Default feature	Action
The **Include digits** option	Determines whether digits must be included in the string.	The option is not selected.	Select the option if you want to include digits in the string.
The **Add new configuration** tab	You can add additional string configurations with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage** field is editable when the	Click the tab and then add the required components for the additional configuration. For example, you can have one configuration for strings with special characters and another configuration for strings with no special characters. You can similarly customize the values, distribution, and percentage as required.

UI element	Description	Default feature	Action
		Random option is not selected. • The fields or options in the following sections display the same values that were provided in the previous configuration: ◦ Minimum characters ◦ Maximum characters ◦ Include special characters ◦ Include digits	**Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **String** generator is complete, you can use the **String** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon. You can reuse the same configuration at a later time in different scenarios without having to set it up again from the scratch. See Creating custom generators and categories from Data definitions on page 393.
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.

UI element	Description	Default feature	Action
The close icon 	NA	NA	Click the icon to exit the **String** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **String** generator should be equal to `100`.

Parameters of the Text generator

After you add the Text generator in a field of a Data definition, you can customize the existing configurations of the field.

When you click the Text generator of a specific field in the Data definition in the edit branch, the unique parameters of the Text generator that contains the existing configuration details for that field are displayed in the **Text** dialog.

The following table provides the list of UI elements of the **Text** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **Text**.	The name is not editable.	NA
The **Distribution** field	Determines if the texts are randomly generated or not in the corresponding column of the test data. Contains the **Random** option.	The **Random** option is selected.	• Do not perform any action if you want the texts, when generated in the test data, to follow a random distribution. • Clear the **Random** option if you want the texts to follow a fixed distribution and to be non-random. When you clear the option, the **Percentage** fields of all the existing configurations become editable. You can set a fixed distribution for the enumerator values using the **Percentage** field. For example, if you set the percentage distribution of `30%` to a configuration that contains words with a count of 10 and `70%` to another configuration that

UI element	Description	Default feature	Action
			contains words with a count of 5, then the text column of the generated data displays the values accordingly. The percentage for each configuration is automatically rebalanced preserving the allocation in the existing configurations, when you add a configuration subsequently.
The **Duplicates** field	Determines if duplicate or unique texts are present in the generated data. Contains the **Allowed** option.	The **Allowed** option is selected.	• Do not perform any action if you want the generated texts to include duplicate entries. • Clear the **Allowed** option if you want to ensure unique texts with no duplicates. This will prevent any duplicate texts from appearing in the generated data.
The **Blank percentage** field	Determines the frequency of blank texts in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the texts in the generated data will be blank.
The **Name** field	Displays the custom name of the selected configuration.	Displays `Default` and is editable.	Enter an identifiable name to the selected configuration, if required.
The **Percentage** field	Determines the proportion of the generated texts when you do not select the **Random** option.	Editable when **Random** is not selected.	If you did not select the **Random** option for distribution, enter a number from `0` to `100` in the **Percentage** field to set the percentage of texts to be generated. For example, if you enter `50`, the generated data will display the text column with a 50% occurrence rate.
The **Text**	Determines the desired text type for the generated texts.	The **Words** option is selected.	Select a specific text type from the list. The available options are as follows:

UI element	Description	Default feature	Action
type list			• Words • Sentences • Paragraphs
The **Count** field	Determines the count with respect to the selected text type.	Displays `10`.	Edit the default count to enter the custom count of the selected text type.
The **Add new configuration** tab	You can add additional text configurations with specific components and settings, if required.	• The **Name** field displays `Configuration-1` and is editable. • The **Percentage** field is editable when the **Random** option is not selected. • The **Text type** and **Count** lists display the same values that were provided in the previous configuration.	Click the tab and then add the required components for the additional configuration. For example, you can have one configuration for words with a count of 10, and another configuration for words with a count of 5. You can similarly customize the values, distribution, and percentage as required. **Note:** You can click the delete icon to delete any additional configuration that you do not need.
The **Save to catalog** option	After the customization of the **Text** generator is complete, you can use the **Text** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon . You can reuse the same configuration at a later time in different scenarios without having to set it up again from the scratch. See Creating custom generators and categories from Data definitions on page 393.

UI element	Description	Default feature	Action
The preview icon	The preview icon is available for each configuration of a generator.	NA	Click the preview icon to view the sample test data for the selected configuration of a generator.
The close icon	NA	NA	Click the icon to exit the **Text** generator dialog.

Note: The total percentage displayed in the **Percentage** fields of all the configurations in the **Text** generator should be equal to `100`.

Parameters of the UUID generator

After you add the UUID (Universal Unique Identifier) generator in a field of a data definition, you can customize the existing configurations of the field.

When you click the UUID generator of a specific field in the Data definition in the edit branch, the unique parameters of the UUID generator that contains the existing configuration details for that field are displayed in the **UUID** dialog.

The following table provides the list of UI elements of the **UUID** dialog with their default parameters, and the actions that you can perform in the dialog to customize the parameters for the selected field:

UI element	Description	Default feature	Action
The **Generator** field	Displays the name of the selected generator, that is, **UUID**.	The name is not editable.	NA
The **Blank percentage** field	Determines the frequency of the unique identifiers in the generated data.	Displays `0` and is editable.	Enter a number from `0` to `100`. For example, if you enter `10`, approximately 10% of the unique identifiers in the generated data will be blank.

UI element	Description	Default feature	Action
The **Save to catalog** option	After the customization of the **UUID** generator is complete, you can use the **UUID** generator as the base generator to create a new custom generator.	NA	Click the **Save to catalog** icon  . You can reuse the same configuration at a later time in different scenarios without having to set it up again from the scratch. See Creating custom generators and categories from Data definitions on page 393.
The close icon 	NA	NA	Click the icon to exit the **UUID** generator dialog.

Management of custom generators

HCL DevOps Test Hub (Test Hub) provides flexibility to create, modify, and delete custom generators. You can access and modify custom generators to suit your specific data requirements in the **Catalog** page.

You can use the built-in Basic generators as the base to create custom generators that closely mimic the real-world data patterns and distributions. You can customize the parameters of the Base generators and save them as new custom generators to use them for future uses.

The creation, modification, and deletion of custom generators are restricted to the edit branch only. Also, only those members with either owner or tester roles within the project are authorized to create, modify, or delete these generators. This security measure can help to maintain data integrity and consistency while preventing unauthorized changes in Test Hub. After all the modifications are published, other project members can view the changes and updated assets.

Creating a custom generator

You can create custom generators in addition to the existing generators in the HCL DevOps Test Hub (Test Hub) catalog. You can customize the existing generators and save them as new custom generators to help you create complex real-world scenarios for synthetic data generation. Additionally, custom generators enable the generation of data for specialized fields or domains that may not be readily available in standard datasets.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Read information about the built-in basic generators. See Basic generators on page 342.
- Read about the parameters of the basic generators. See Basic generators on page 342.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Catalog**.
   The **Catalog** page is displayed.
4. Select the branch where you want to store the data generator from the **Branch** list.
5. Perform one of the following actions to open the **Add new generator** dialog:
   - Perform the following actions to use the base generator:
     a. Click a generator from the **Basic** category that you want to use as the base generator.

        The *Generator* dialog is displayed.
     b. Click the **Customize** icon  .
   - Click the  icon in the **Catalog** page, and then click **Add new generator**.

   Based on the selected **Branch**, the following dialog is displayed:
   - **Edit Branch**: If you selected a branch other than the Edit branch.
   - **Add new generator**: If you selected the Edit branch.
6. Perform one of the following actions in the **Edit branch** dialog, if you selected a branch other than the Edit branch:

Action	Steps	Result
Select an Edit branch	a. Select an Edit branch from the **Select branch** list if the Edit branch already exists. b. Click **Ok**.	The **Add new generator** dialog is displayed.
Create an Edit branch	a. Select **Create new branch**. b. Enter a name for the Edit branch. c. Click **Save**.	The **Add new generator** dialog is displayed.

7. Enter a unique name for the generator in the **Name** field of the **Add new generator** dialog.
8. Select a built-in generator from the **Base generator** list.
   Test Hub displays only the built-in Basic generators as the base generator.
9. Perform one of the following actions to select the category of the new generator in the **Category** list:
   - To select an existing category, go to step 10 on page 392.
   - To add a new category, go to step 11 on page 393.
10. Perform the following steps to select an existing category:

a. Select an existing category, if any, that is displayed in the **Category** list.

Note: You cannot create a new generator in the **Basic** category. Therefore, the **Basic** category is not displayed in the **Category** list.

b. Click **Create**.

The new generator is added under the existing category of the **Catalog** page.

11. Perform the following steps to add a new category:
    a. Click **Add new category** in the **Category** list, and then click **Next**.

    The **Add new category** dialog displays a new set of fields to capture more details about the new category.

    b. Enter a unique name in the **Category name** field where the new generator will be saved in the catalog of Test Hub.
    c. Select a color from the available range of colors so that you can identify the new category easily.
    d. Select the location to create the new category in the selected folder.

    For example, if you entered `MyFolder/MyCategory` in the **Category name** field, and then selected `Location` as the location to create the category in the folder, then the category is created under the `Location/MyFolder/MyCategory` path.

    e. Click **Create**.

    The new generator is added in the newly created category and is displayed in the **Catalog** page.

Results

You have created a custom generator in addition to the existing generators of the Test Hub catalog.

What to do next

You can perform any of the following tasks:

- Publish the custom generators to the project repository so that other members of the project can use the custom generator. See Publishing all changes in an Edit branch on page 493.
- Modify the parameters of custom generators. See Modifying custom generators on page 395.
- Delete custom generators. See Deleting custom generators on page 397.

Creating custom generators and categories from Data definitions

You can create custom generators and categories from existing Data definitions in HCL DevOps Test Hub (Test Hub). After you add an existing generator in a field of a Data definition, you can customize the generator parameters that are displayed for the field. You can then save the customized parameters in the Data fabrication catalog as a new generator. You can also save the new generator either in an existing category or save in a new category.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a data definition. See Creating an empty data definition on page 450.
- Read about the parameters of the basic generators. See Basic generators on page 342.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Open a Data definition in the Edit branch.
5. Click a generator in the **Generator** column of a field where you want to customize the configuration.
6. Perform any modification in the *Generator* dialog to customize the existing configuration of the selected field.
7. Click **Save to catalog** to save the customized parameters in the catalog of the Data fabrication feature as a new generator.
   The **Add new generator** dialog is displayed.
8. Enter a unique name for the new generator in the **Name** field.
   Test Hub uses the existing generator as the base generator. Therefore, the **Base generator** list is not editable.
9. Perform one of the following actions to select the category of the new generator:
   - To select an existing category, go to step 10 on page 394.
   - To add a new category, go to step 11 on page 394.
10. Perform the following steps to select an existing category:
    a. Select an existing category that is displayed in the **Category** list.
    b. Click **Create**.

       The new generator is added in the existing category and is displayed in the **Catalog** panel of the data definition.
11. Perform the following steps to add a new category:
    a. Click **Add new category** in the **Category** list, and then click **Next**.

       The **Add new generator** dialog displays a new set of fields to capture more details about the new category.
    b. Enter a unique name in the **Category name** field where the new generator will be saved in the catalog of Test Hub.
    c. Select a color from the available range of colors so that you can identify the new category easily.
    d. Select the location to create the new category in the selected folder.

For example, if you entered `MyFolder/MyCategory` in the **Category name** field, and then selected `Location` as the location to create the category in the folder, then the category is created under the `Location/MyFolder/MyCategory` path.

e. Click **Create**.

The new generator is added in the newly created category and is displayed in the **Catalog** panel of the data definition.

Results

You have created a custom generator using an existing generator from a Data definition. Also, you have learned to create a new category and save the custom generator under the category.

What to do next

You can save and publish the Data definition to the Git repository so that other members of the project can use the Data definition. See Publishing all changes in an Edit branch on page 493.

Modifying custom generators

You can modify the parameters of custom generators from the Catalog page of HCL DevOps Test Hub (Test Hub). You can make modifications to precisely align the test data that uses the custom generators with the desired dataset.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a custom generator. See Creating a custom generator on page 391.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Catalog**.
   The **Catalog** page is displayed.
4. Select the branch that contains the custom generator for modification from the **Branch** list.
5. Select the custom generator for modification.

   The selected *Generator* dialog that contains the following tabs is displayed:

Element	Description
The **Configuration** tab	Displays all the existing parameters of the selected custom generator.

Element	Description
The **Details** tab	Provides following details about the selected custom generator and its associated Git details: ◦ Name of the branch that contains the generator ◦ Description for the generator, if any ◦ Git-related details for the generator as follows: ▪ Asset ID: The ID uniquely identifies the generator. You also have the option to copy the ID for easy reference or use. ▪ Repo: The repository name and path where the generator is stored. ▪ Path: The exact location of the generator within the repository. ▪ Commit: The Git commit message, the name of the member who performed the changes, and the date and time of the commit. ▪ The copy button: You can click the button to copy the complete SHA commit ID for the generator. ▪ The commit ID button: You can click the button to view the changes that were made in the commit directly within Git. ▪ The browse button: You can click the button to view the specific location in the Git history where the commit was performed for the generator.
The **Usage** tab	Displays a list of all the Data definitions along with their location in the project that currently use the selected generator.

6. Perform one of the following actions in the **Generator** dialog:
   - Perform the required modifications to the existing configurations in the **Configuration** tab, if you are in the edit branch.
   - Perform the following actions, if you are in a branch other than the Edit branch:

     a. Click the **Edit** icon  .

     The **Edit branch** dialog is displayed.

     b. Perform one of the following actions in the dialog:

Action	Steps	Result
Select an Edit branch	i. Select an Edit branch from the **Select branch** list if the Edit branch already exists. ii. Click **Ok**.	The **Catalog** page is displayed in the Edit branch. You can open the custom generator and perform modifications to the existing generator configurations in the **Configuration** tab of the *Generator* dialog.

Action	Steps	Result
Create an Edit branch	i. Select **Create new branch**. ii. Enter a name for the Edit branch. iii. Click **Save**.	The **Catalog** page is displayed in the Edit branch. You can open the custom generator and perform modifications to the existing generator configurations in the **Configuration** tab of the *Generator* dialog.

7. Click the save icon in the *Generator* dialog to save the modified custom generator.

Note: If you want to continue using the modified generator to generate test data without publishing it, you must save the changes. However, if you want to collaborate with other members of your project, you must publish the changes.

Results

You have modified the parameters of a custom generator.

What to do next

You can perform any of the following tasks:

- Publish the custom generators to the project repository so that other members of the project can use the custom generator. See Publishing all changes in an Edit branch on page 493.
- Delete custom generators. See Deleting custom generators on page 397.

Deleting custom generators

You can delete custom generators to remove them when custom generators are outdated or no longer relevant to the current project. Additionally, when custom generators contain errors or inaccuracies that can lead to misleading or incorrect fabricated data, then you might need to remove them. This ensures that the project remains clutter-free, optimizes the performance of HCL DevOps Test Hub (Test Hub), and reduces storage requirements.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a custom generator. See Creating a custom generator on page 391.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

The **Overview** page of the project is displayed.

3. Click **Author > Catalog**.
   The **Catalog** page is displayed.
4. Select the branch that contains the custom generator to delete from the **Branch** list.
5. Select the custom generator for deletion.
   The selected *Generator* dialog is displayed.
6. Click the **Delete** icon .

Results

You deleted a custom generator from your project.

What to do next

You can publish the changes performed to the custom generator (that is, deletion of the custom generator) to the Git repository so that other members of the project do not see the deleted custom generator. See Publishing all changes in an Edit branch on page 493.

Management of custom categories

HCL DevOps Test Hub (Test Hub) provides flexibility to create and delete custom categories in the **Catalog** page. The **Catalog** page displays both the basic and custom categories of generators in a project.

The basic category consists of all the built-in basic generators to simplify the access when you work with Data definitions. When you create custom generators that are tailored to specific use cases, then you can save them under the custom categories and organize them in a way that aligns with the project requirements. You can locate and use the custom generators when you need.

The creation and deletion of custom categories are restricted to the Edit branch only. Also, only those members with either owner or tester roles within the project are authorized to create or delete these categories. This security measure can help to maintain data integrity and consistency while preventing unauthorized changes in Test Hub. After all the modifications are published, other project members can view the changes and updated assets.

Creating a new custom category

You can create custom categories in addition to the existing Basic category in the HCL DevOps Test Hub (Test Hub) catalog. You can save the custom generators under various custom categories with unique names.

Before you begin

You must be a member of the project with the **Owner** or **Tester** role.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.

3. Click **Author > Catalog**.
   The **Catalog** page is displayed.
4. Select the branch where you want to store the custom category from the **Branch** list.
5. Click the  icon in the **Catalog** page, and then click **Add new category**.
   Based on the selected **Branch**, the following dialog is displayed:
   - **Edit Branch**: If you selected a branch other than the Edit branch.
   - **Add new category**: If you selected the Edit branch.
6. Perform one of the following actions in the **Edit branch** dialog, if you selected a branch other than the Edit branch:

Action	Steps	Result
Select an Edit branch	a. Select an Edit branch from the **Select branch** list if the Edit branch already exists. b. Click **Ok**.	The **Add new category** dialog is displayed.
Create an Edit branch	a. Select **Create new branch**. b. Enter a name for the Edit branch. c. Click **Save**.	The **Add new category** dialog is displayed.

7. Perform the following steps in the **Add new category** dialog:
   a. Enter a unique name in the **Category name** field.
   b. Select a color from the available range of colors so that you can identify the new category easily.
   c. Select the location to create the new category in the selected folder.

      For example, if you entered `MyFolder/MyCategory` in the **Category name** field, and then selected `Location` as the location to create the category in the folder, then the category is created under the `Location/MyFolder/MyCategory` path.

   d. Click **Create**.

   The new category is added in the **Catalog** page.

Results

You created a custom category in the **Catalog** page.

What to do next

You can perform the following tasks:

- Add custom generators to the new category. See Creating a custom generator on page 391.
- Publish the changes to the Git repository so that other members of the project can use the new custom category. See Publishing all changes in an Edit branch on page 493.

Deleting custom categories

You can delete custom categories to remove them when custom categories are outdated or no longer relevant to the current project. Additionally, when custom categories contain inaccuracies that can lead to misleading or incorrect fabricated data, then you might need to remove them. This ensures that the project remains clutter-free, optimizes the performance of HCL DevOps Test Hub (Test Hub), and reduces storage requirements.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a custom category. See Creating a new custom category on page 398.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Catalog**.
   The **Catalog** page is displayed.
4. Select the branch that contains the custom category that you want to delete from the **Branch** list.
5. Click the **Delete** icon beside the category tab to delete the custom category.
   Based on the selected **Branch**, the following dialog is displayed:
   - **Edit Branch**: If you selected a branch other than the Edit branch.
   - **Delete category**: If you selected the Edit branch.
6. Perform one of the following actions in the **Edit branch** dialog, if you selected a branch other than the Edit branch:

Action	Steps	Result
Select an Edit branch	◦ Select an Edit branch from the **Select branch** list if the Edit branch already exists. ◦ Click **Ok**.	The **Delete category** dialog is displayed.
Create an Edit branch	◦ Select **Create new branch**. ◦ Enter a name for the Edit branch. ◦ Click **Save**.	The **Delete category** dialog is displayed.

7. Click **Delete** in the **Delete category** dialog.

Results

You deleted a custom category from your project.

What to do next

You can publish the changes performed to the custom category (that is, deletion of the custom category) to the Git repository so that other members of the project do not see the deleted category. See Publishing all changes in an Edit branch on page 493.

Data management

Effective data management is at the core of robust application testing. In HCL DevOps Test Hub (Test Hub), you can manage test data by using the **Data** tab, which provides a centralized workspace for handling various types of data. Whether you work with fabricated data, retrieve information from databases, or define structured schemas, this interface supports both creating new data and reusing existing assets across your testing requirements.

The **Data** tab contains the following core components:

- **Data Definition**: Data definitions represent the data structures that are composed of one or more generators. Test Hub provides you the interface to create and manage data definitions so that you can design data structures that closely resemble the real-world scenarios. You can configure generators, define rules, set restrictions and distributions if required, and generate test data using a data definition. You can generate realistic and diverse test data for comprehensive testing scenarios. See Management of data definition on page 450 and Catalog management on page 342.
- **Dataset**: Datasets provide tests with variable data during a test run. The test that uses a dataset at the run time replaces a value in the recorded test with variable test data that is stored in the dataset. In Test Hub, you can create a dataset and use its values during the run time to test various scenarios. See Management of datasets on page 424.
- **Database Query**: A database query is a request for data or information from a database. You can run queries on integrated databases to examine existing data structures and retrieve records. Then, use the information to guide the design of schemas or enhance your data definitions. See Management of database query on page 417.
- **Schema**: Schemas define how data objects relate to each other. In Test Hub, schemas provide a visual and configurable way to design relational data environments that closely mirror real-world systems. See Management of schemas on page 469.

Related information

Management of datasets on page 424

Management of data definition on page 450

Management of schemas on page 469

Management of database query on page 417

Generating fabricated test data on page 483

Data navigator panel

The home page of **Data** features a navigation panel called the data navigator panel, which serves as a central hub for managing and organizing data definitions, datasets, database queries, and schemas across multiple folders within your project.

Each folder in the panel acts as a container for project assets such as applications, datasets, data definitions, database query or schemas. You can select a dataset, data definition, database query, or schema to open it in the **Data** page, where you can view and manage its contents.

You can perform the following actions from the data navigator panel:

Actions	Descriptions
Search	You can search for a data definition or a dataset in a branch by entering the name of the data definition or dataset in the **Search** field.
View	You can view the contents of a data definition or a dataset for a selected folder in a project. See Viewing a data definition and its details on page 464 and Viewing a dataset on page 443.
New	You can create a dataset, data definition, database query, or a schema in a project. See Creating an empty data definition on page 450 and Creating a dataset on page 425.
Edit	You can modify a dataset, data definition, database query, or a schema in a project. See Modifying a data definition on page 462 and Editing a dataset on page 433.
Rename	You can change the name of a Dataset, Data definition, Database Query, or Schema within a project.
Save As	You can create a copy of an existing dataset, data definition, database query, or a schema with a new name in a project. You can select the **Save As** option, which appears when you right-click the data asset.
Delete	You can delete a dataset, data definition, database query, or a schema in a project. See Deleting a data definition on page 468 and Deleting a dataset on page 444.
Move	You can move a dataset, data definition, database query, or a schema from one applicationr to another within the same project. You can select the **Move** option, which appears when you right-click the test asset.
Compare	You can view and compare changes made in the edit branch with unpublished changes in the Web UI test or suite. You can select the **Compare** option, which appears when you right-click the test asset.
Collapse All	You can minimize all expanded datasets in the project view to simplify navigation and focus on top-level items.
Expand All	You can expand all sections or folders in the project view to quickly view the all the datasets.

You can use the following options to enhance your data navigation experience while managing and organizing data definitions and datasets across multiple folders of your project.

Icon	Description
	You can use the **Virtual Folder** icon to view folders at different levels. The available options are: • **Root Folders**: Displays virtual folders at the top, followed by the repository's folder structure. • **Embedded Folders**: Shows the repository's folder structure first, with files organized into virtual folders. • **None**: Excludes virtual folders entirely, and presents only the repository's orignal folder structure.
	You can use the **Link with editor** icon to connect editor tabs to their corresponding locations in the navigator. By using this icon, you can easily navigate and quickly access specific sections within your project.
	You can apply a filter to view folders by their asset type. You can select one or more of the following asset types and apply the filter: • Data Definitions • Database Queries • Datasets • Schemas
	You can use the **Save All** icon to save all changes made to multiple assets or folders within your project.

Notes:

- Only a *Project Owner* or a project member whose role is a *Tester* can edit a data definition or a dataset.
- To view the contents of an encrypted dataset, you must provide an encryption key that you set when you encrypted the dataset column.
- As a *Viewer*, when you try to modify a data definition or dataset, an error message is displayed because you do not have the permission to modify them.

To conclude, the data navigator panel provides easy access to projects, applications, datasets, data definitions, database query and schema in multiple branches.

Prerequisites for working with datasets and data definitions

You can find information about the tasks that you need to perform before you work with datasets and data definitions in HCL DevOps Test Hub (Test Hub).

Management of security groups

You can create a security group and provide access only for the required members of a project in Test Hub. You can add the encrypted datasets, restricted database connections, or API secrets that you create in a project to the security group and ensure that it can be accessed and used only by the members who have access to the security group. See Management of security groups on page 404.

Management of database connections

You can import the data that you have in an external database that can be used to create the test assets such as datasets and Data definitions in Test Hub.

You can establish and manage connections with the following external databases in **Test Hub**:

- Oracle
- PostgreSQL
- IBM DB2

See Management of database connections on page 408.

Management of database queries

You can create a database query to import the data that is present in an external database, after you establish a connection with the external database. You can query the required data from a database when you create a database query to import the useful data present in your existing databases to create the test assets. See Management of database query on page 417.

Management of security groups

You can find information about creating, editing, deleting, and granting access to a security group in HCL DevOps Test Hub (Test Hub).

You can find the following information about managing the security groups:

- Creating a security group on page 404
- Editing a security group on page 405
- Deleting a security group on page 406
- Granting access to security group on page 406

Creating a security group

As a project *Project Owner* or *Tester*, you must create a security group to organize encrypted datasets, external database connections, and API secrets so that project members can use them efficiently.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Created a project in Test Hub. See Managing DevOps Test Hub projects on page 1209.
- Configured the repository that contains the test assets in your project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Click **Create Security Group** and enter a name for the security group.
5. Click **Create**.
   The security group is added to the **Security Group** list.

What to do next

You can grant access to other project members to use the encrypted dataset. See Granting access to security group on page 406.

Editing a security group

You might want to change the name of a security group that you created, you can do so by editing the security group.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Created a security group. See Creating a security group on page 404.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Click a security group that you want to edit from the list under **Security Group**.
5. Click the **Edit** icon in line with the security group.
   The **Edit Security Group** dialog is displayed.
6. Edit the name of the security group, and then click **Save**.

Results

You edited the name for a security group in your project.

Deleting a security group

After you create a security group, you can delete a security group when it is not required in your test environment.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Created at least one dataset and encrypted the dataset with an encryption key. See Dataset encryption on page 439.
- Created at least two security groups. See Creating a security group on page 404.

About this task

You can delete a security group from your project any time after you create a security group. For example, you might want to delete the security group if the security group that was created earlier is not required.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Click a security group that you want to delete from the list under **Security Group**.

   Note: Before deleting a security group associated with encrypted datasets, you must remove or move the encrypted datasets under it to another security group so that the delete icon is enabled.

5. Click the **Delete** icon .
   The **Delete Security Group** dialog is displayed.
6. Click **Delete** to confirm the deletion of the security group.

Results

You deleted a security group.

Granting access to security group

You can grant or revoke access to the security group in your project to individual members with different roles or all members with a specific role. Without access to the security group, members cannot view, create, edit, delete, or use the security group.

Before you begin

You must be a *Project Owner* and have completed the following tasks:

- Created a project in Test Hub. See Managing DevOps Test Hub projects on page 1209.
- Added one or more users to your project. See Adding users to a project on page 1217.
- Created at least one security group. See Creating a security group on page 404.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Click a security group in the list under **Security Group** for which you want to grant the access.
5. Choose any of the following methods to grant access to a member:
   Choose from:

   - Select the checkbox for the specific role under **Grant Read Only Access to roles** to grant read-only access to the members of a specific role. For example, if you select **Tester** then all members in the project with a tester role are granted read-only access to the selected security group. You can select any role or all the roles listed.
   - To grant read/write access to specific members, perform the following steps:
     a. Click **Add Members** under **Members with Read/Write Access**.

        The **Additional Members** dialog is displayed.

     b. Enter the name or email ID to search the user or navigate through the list and select the member you want to grant access. You can select multiple members in the list.

        Important:

        Irrespective of the role that the member (**Owner**, **Tester**, or **Viewer**) was assigned in the project, access to the security group has to be specifically granted to the members from the **Security** page.

     c. Click **Add Members.**

        You granted read/write access to the added members.
6. Choose any of the following methods to revoke access to a member:
   Choose from:

- Clear the checkbox for the specific role under **Grant Read Only Access to roles,** to revoke read-only access to the members of a specific role. For example, if you clear the **Tester** checkbox then access for all members in the project with a tester role is revoked for the selected security group. You can select any role or all the roles listed.
- Click the **Delete** icon in line with the member name that you want to remove the specific members from the **Members with Read/Write Access** list.

Notes:

- Any member with access to the security group can remove access to other members specifically added or of all members with a specific role.
- Members with access to the security group can remove themselves from the access list provided that there is at least one member in the list. After removing themselves, members cannot add themselves back to the access list and must be added by any of the other members in the list.

Results

You have granted or revoked the security group access to project members.

Management of database connections

You can add a database connection to a project and import the data from an external database in HCL DevOps Test Hub (Test Hub).

You can also restrict the database connection when you add it to a project and then add it to a security group to grant or restrict access for other members of the project. After you add a database connection you can create the database query that can be used to create test assets.

You can find the following information about adding the database connection and managing them in Test Hub:

- Adding a database connection on page 408
- Adding a database connection to a security group on page 411
- Removing a database connection from a security group on page 412
- Editing a database connection on page 413
- Duplicating a database connection on page 414
- Deleting a database connection on page 416

Adding a database connection

You can add a database connection to your project in HCL DevOps Test Hub (Test Hub) to import the data from an external database and create the test assets. You can restrict the project members from using this database connection by adding it to a security group and providing permission only to the required members.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Ensured that you have a valid Java Database Connectivity (JDBC) URL and credentials for the database.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Infrastructure > Database Connections**.
   Result
   The **Database Connections** page is displayed.
4. Click **New Connection**.
   Result
   The **New Connection** dialog is displayed.
5. Perform the following actions in the **Details** tab:

   a. Enter a name for the database connection in the **Name** field.

   b. Select the database from the **Driver** list.
      The following are the supported databases:
      - Oracle
      - PostgreSQL
      - IBM DB2

   c. Enter the URL for the database in the **URL** field.

   d. Enter the username for the database in the **Username** field.

   e. Enter the password for the database in the **Password** field.

   f. Enter the schema name for the database in the **Schema** field.

   g. Add a description for the database connection in the **Description** field.

6. Select the **Restrict access using Security Groups** checkbox if you want to restrict other members of the project from using this database connection.

   Later you can grant access to the required members by adding the connection to a security group.

Note: The option, **Restrict access using Security Groups** is not selected. This is the default selection.

7. Perform the following steps to add the SSL certificate for encrypting the data exchanged between Test Hub and the database.
    a. Click the **SSL** tab.
    b. Click **Select File** and browse to the location on your computer where you have the Java Keystore File.
    c. Select the file, and then click **Open**.
    Result
    The selected file name is displayed.
    d. Enter the password for the Java Keystore File in the **Password** field.
8. Click **Test** to test the connection with the database.
Result
A message is displayed that confirms the connection with database.
9. Click **Save** to save the database.

A message is displayed that confirms database connection is added.

You can view the following details of the database connection on the **Database Connections** page:

Column header	Description	
Name	Displays the name of the database connection that you added.	
URL	Displays the URL of the database connection that you added.	
User	Displays the name of the user who added the database connection.	
Description	Displays the description that you entered when you created the database connection.	
Actions	 or 	Click the lock icon to go to the **Security** page. You can grant access to the database connection by adding it to a security group. **Note:** The unlock icon is displayed to indicate that the database connection is not restricted for other members of the project, and they can use database connection to create the test assets.
		Click the edit icon to edit the database connection. See Editing a database connection on page 413.

Column header	Description	
		Click the duplicate icon to duplicate the database connection. See Duplicating a database connection on page 414.
		Click the delete icon to delete the database connection. See Deleting a database connection on page 416.

Results

You have added a database connection to your project.

What to do next

You can add the database to a security group to provide access to other members of the project, if you have restricted it when you created the database connection. See Adding a database connection to a security group on page 411.

You can create a database query using the database connection that you added to your project. See Creating a database query on page 417.

Adding a database connection to a security group

You can add the database connection to a security group to grant access to the security group for other project members in HCL DevOps Test Hub (Test Hub). After the access is provided, they can use the database connection to create the database query and import the data from the database.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Added at least one database connection and restricted the access to other members of the project. See Adding a database connection on page 408
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created at least one security group. See Creating a security group on page 404.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.

Result

The **Security** page is displayed.

4. Click a security group for which you want to add the database connection in the list under **Security Group**.
5. Click the **Add Connection** icon to select the database that you want to add to the security group.

   Result

   The **Add Connection** dialog is displayed.
6. Click the **Database Connection** field to view the available database connection in the project.
7. Identify and select the database connection that you want to add to the security group.
8. Enter the password that you use to connect to the database in the **Password** field.

   Tip: You can click the view icon to view the entered password and ensure that you entered the correct password.

9. Click **Add**.

   Result

   The database connection that is added to the security group is displayed.

Results

You have added a database connection to the security group.

What to do next

You can create a database query by using the database connection that you added to your project. See Creating a database query on page 417.

Removing a database connection from a security group

You can remove a database connection from a security group when you are no longer required to provide access to other members of your project.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Added at least one restricted database connection to a security group. See Adding a database connection to a security group on page 411.

1. Log in to Test Hub.

   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.

3. Click **Manage > Security**.
   Result
   The **Security** page is displayed.
4. Click a security group in the list under **Security Group** from which you want to delete the database connection.
5. Click the **Delete** icon in line with the database connection that you want to delete from the security group.
   Result
   The **Remove database connection** dialog is displayed.
6. Click **Remove**.

Note: Removing the database connection from the security group also removes the password stored in HCL DevOps Test Hub (Test Hub) for that database. You must enter the password again to gain access to the database connection while adding it to a different security group.

Results

You have removed the database connection from a security group.

Editing a database connection

You can edit the database connection, when you want to change the database details that you added earlier or if you want to update the SSL certificate file.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Added at least one database connection. See Adding a database connection on page 408
- Ensured that you have credentials for the database that you want to edit in a HCL DevOps Test Hub (Test Hub) project.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Infrastructure > Database Connections**.
   Result
   The **Database Connections** page is displayed.
4. Click the Edit icon inline with the database that you want to edit.
   Result
   The **Edit connection** dialog is displayed.
5. Perform the following actions in the **Details** tab:

a. Edit the details of the connection.

b. Enter the password for the database in the **Password** field.

c. Select the **Restrict access using Security Groups** checkbox if you want to restrict other members of the project from using this database connection.

Later you can grant access to the required members by adding the connection to a security group.

Note: The option, **Restrict access using Security Groups** is not select. This is the default selection.

6. Perform the following steps to change or add the SSL certificate and encrypt the data exchanged through the database.

a. Click the **SSL** tab.

b. Click the **x** icon to remove the existing Java Keystore file.

c. Click **Select File** and browse to the location on your computer where you have the Java Keystore File.

d. Select the file, and then click **Open**.
Result
The selected file name is displayed.

e. Enter the password for the Java Keystore File in the **Password** field.

7. Click **Test** to test the connection with the database.
Result
A message is displayed that confirms the connection with database.
8. Click **Save** to save the database connection.

A message is displayed that confirms database connection is modified and the edited database connection is displayed on the **Database Connections** page.

Results
You have edited the database connection in your project.

Duplicating a database connection

When you want create a similar database connection with different user credentials, to create the similar database connection and restrict or unrestricted it, or to create a similar database connection by encrypting the data exchanged between HCL DevOps Test Hub (Test Hub) and database, you can create a new database connection using an existing database connection.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Added at least one database connection. See Adding a database connection on page 408.
- Ensured that you have a valid URL and credentials for the database that you want to add or duplicate in a Test Hub project.

About this task

You can duplicate database connection and create a new one in the following scenarios:

- Creating the same database connection with or without restricting the access to other members of the project.
- Adding the same database connection to a different security group which consists of different datasets, secrets and have access to different group of people.
- Adding the same database with different user credentials.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Infrastructure > Database Connections**.
   Result
   The **Database Connections** page is displayed.
4. Click the **Duplicate** icon  inline with the database connection that you want to duplicate.
   Result

   The **Duplicate connection** dialog is displayed.

   The value in the **URL** field, and the selections in the **Driver** field and the **Restrict access using Security Groups** checkbox are auto-populated with the current values and selections in the database connection that you duplicate.
5. Perform the following actions in the **Details** tab:

   a. Enter a different name for the database connection in the **Name** field.

   b. Enter the username for the database in the **Username** field.

   c. Enter the password for the database in the **Password** field.

   d. Select the **Restrict access using Security Groups** checkbox if you want to restrict other members of the project from using this database connection.

Later you can grant access to the required members by adding the connection to a security group.

Note: The option, **Restrict access using Security Groups** is not select. This is the default selection.

6. Perform the following steps to change or add the SSL certificate for encrypting the data exchanged through the database.
    a. Click the **SSL** tab.
    b. Click the **x** icon to remove the existing Java Keystore file.
    c. Click **Select File** and browse to the location on your computer where you have the Java Keystore File.
    d. Select the file, and then click **Open**.
    Result
    The selected file name is displayed.
    e. Enter the password for the Java Keystore File in the **Password** field.
7. Click **Test** to test the connection with the database.
    Result
    A message is displayed that confirms the connection with database.
8. Click **Save** to save the database connection.
    Result

    A message is displayed that confirms database connection is added and the new database connection created by using the existing database connection is displayed on the **Database Connections** page.

Results

You have created a new database connection by using the existing database connection in your project.

Deleting a database connection

You can delete the database connection from a project when it is no longer required.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Added at least one database connection. See Adding a database connection on page 408.

1. Log in to Test Hub.
    The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

The **Overview** page of the project is displayed.

3. Click **Infrastructure > Database Connections**.
   Result
   The **Database Connections** page is displayed.
4. Click the **Delete** icon inline with the database connection that you want to delete.
   Result
   The **Delete Connection** dialog is displayed.
5. Click **Ok** to confirm the deletion.
   Result
   The **Connection deleted successfully** message is displayed.

Results

You have deleted a database connection from your project.

Management of database query

You can find information about creating, editing, duplicating, and deleting the database query in HCL DevOps Test Hub (Test Hub).

You can find the following information about creating and managing the database queries:

- Creating a database query on page 417
- Editing a database query on page 419
- Duplicating a database query on page 421
- Deleting a database query on page 423
- Publishing all changes in an Edit branch

Creating a database query

When you have the data that is useful for creating the test assets is stored in a database, you can import the data from the database and create a database Query in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Added at least one database connection. See Adding a database connection on page 408
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.

2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.

3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.

4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.

5. Perform any of the following actions:
   - Click **Add files** and select **Database Query** on the **Test Editor** page.
   - Click the **Action Menu** icon  and select **Database Query**.
   - Right-click the application in the Data navigator panel, select **New > Database Query**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 418.

6. Select your action to either select an existing Edit branch or create an Edit branch:

   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, which you selected, is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

   Result

   The **Create database query** dialog is displayed.

7. In the **Create database query** dialog, perform the following tasks:
   a. Select the **Application** for which you want to create a dataset.
   b. Select **Location** from the drop-down list to store the dataset.
   c. Type a name for the database query.
   d. Click **Save**.

   Result

   The database query is created and displayed on the **Data** page. The database query created is listed under the `Database Queries` container within the selected application.

8. Perform the following steps in the **Data** page to select the table and query the data from the database:

    a. Select the database from the **Connection** list.

    b. Select the table in the selected database from the **Table** list.

    Result

    The data from the selected table is displayed.

    c. Select the required column of the table from which you want to import the data.

    d. Click **View** to switch between the **Visual** and **Code** view.

    Note: Note: The default view is **Visual** view. In the **Code** view you can type the query to get data from the database.

    e. Click the **Preview** icon to view the queried data from the database in tabular format in the bottom window.

    f. Click **Save** icon to save the queried data from the database.

Results

You have created a database query in your project.

What to do next

Publish the database query to the Git repository so that other members of the project can use the database query. See Publishing all changes in an Edit branch.

You can create a dataset using the database query. See Creating a dataset by using a database query on page 431

You can create a Data definition using the database query. See Creating a data definition by importing fields from a database query on page 456.

Editing a database query

You can edit the database query when you want to query different data from the database or if you want to query the data from a different database.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created at least one database query. See Creating a database query on page 417.
- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Navigate to the **Database Queries** container, then right-click the database query you want to edit and select **Edit**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 420.
6. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
7. Perform the following steps on the **Data** page to select a different table or query the different data from the same database or different database:

   a. Select a different table, modify the query, or different connection.

   b. Click **View** to switch between the **Visual** and **Code** view.

Note: The default view is **Visual** view. In the **Code** view you can type the query to get data from the database.

c. Click the **Preview** icon to view the queried data from the database in tabular format in the bottom window.

d. Click the **Save** icon to save the queried data from the database.

Results

You have edited a database query in your project.

What to do next

You can publish the database query to the Git repository so that other members of the project can use the database query. See Publishing all changes in an Edit branch.

Duplicating a database query

You can create a new database query by using the existing database query, when you want the same database query with a different name or only slight modification required in the existing database query.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created at least one database query. See Creating a database query on page 417.

About this task

When you want to create a database query by using the existing database query, you must have an Edit branch. You can create the Edit branch during the creation of the database query. All your dataset resides in the Edit branch until you publish or discard the changes.

After you create a database query using the existing database query, you can either publish your changes to a remote Git repository or you can discard the changes made to the database query. The Edit branch automatically disappears from the **Branch** list as well as from the remote Git repository and sets the global branch to the previously selected branch in the following conditions:

- When you publish changes to the remote Git repository.
- When you discard the changes made in the dataset.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.

2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.

3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.

4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.

5. Navigate to the logical folder **Datasets** in the Data navigator, right click the database query that you want to save with a different name, and then click **Save As**.
6. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
7. Enter a different name for the database query in the **Name** field.
8. Select a folder in the **Location** field.
9. Click **Save**.
   Result

   A message is displayed that confirms database query is created.

   The database query is displayed on the **Data** page. The dataset created is listed under the logical folder `Datasets` under the selected physical folder that you selected. The name of the database query is suffixed with the `.dtd.yaml` file extension.

10. Make any required changes to the existing query, and then click **Save** icon to save the queried data from the database.

Results

You have created a database query using the existing database query in your project.

What to do next

You can publish the database query to the Git repository so that other members of the project can use the database query. See Publishing all changes in an Edit branch.

Deleting a database query

You can delete a database query when it is no longer required.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created at least one database query. See Creating a database query on page 417.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform one of the following actions:
   a. Select the database query to open it on the **Data** page. Click the **Delete** icon  .
   b. Right-click the database query you want delete and select **Delete**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 424.
6. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, which you selected, is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:

a. Select **Create new edit branch**.
b. Enter a name for the Edit branch in the **Branch Name** field.
c. Click **Save**.

Result

The **Delete database query** dialog is displayed.

7. Click **Delete** to confirm the deletion.

Result

A message is displayed that confirms database query is deleted.

Results

You have deleted the selected database query from your project repository.

What to do next

You can publish the database query to the Git repository so that other members of the project can use the database query. See Publishing all changes in an Edit branch.

Management of datasets

You can find information about creating datasets in HCL DevOps Test Hub (Test Hub), applying a dataset security group, editing datasets, and managing datasets that are in your project repository.

A dataset contains variable data that are used by a test at test run time. You create the datasets and associate them with your tests in the desktop products. You can decide whether you want to use the dataset associated with the test to be used at run time or override that associated dataset with another dataset at run time. You can create datasets in the desktop products itself and select the datasets as overrides in the desktop products.

When you want to run those tests from Test Hub and you want the tests to use a different set of variable data, you can create datasets in Test Hub. You can then select them to override the dataset associated with the tests that are in the project repository.

You can create a dataset in Test Hub by using any of the following methods:

- Creating an empty dataset. See Creating a dataset on page 425.
- Importing data from a CSV file. See Creating a dataset by using a CSV file on page 427.
- Creating a dataset with fabricated data. See Creating a dataset by using a Data definition on page 429.

When you want to edit the data in a dataset, you must open it for editing. See Editing a dataset on page 433.

After you edit the dataset, you must publish the changes, so that the dataset is added to the project repository. See Publishing changes in datasets.

If the data in your dataset contains sensitive information, you can encrypt the dataset columns. See Encrypting a dataset column on page 440.

If a dataset has encrypted columns and you want to view an encrypted dataset, see Viewing an encrypted dataset on page 446.

When you want to remove the encryption for dataset columns, you can decrypt the dataset column. See Decrypting a dataset column on page 442.

You can assign a security group for encrypted datasets. See Managing an encrypted dataset on page 447.

You can change the security group of an encrypted dataset. See Changing security group for an encrypted dataset on page 446.

Creating a dataset

You can create datasets in HCL DevOps Test Hub (Test Hub) to replace the values of existing dataset with the values of new dataset during a test or schedule run.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:
   - Click **Add files** and select **Dataset** on the **Test Editor** page.
   - Click the **Action Menu** icon  and select **Dataset**.
   - Right-click the application in the data navigator panel, select **New > Dataset**.

   The **Edit Branch** dialog is displayed.

If the **Edit branch** dialog is not displayed, then go to 7 on page 426.

6. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, which you selected, is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

   Result

   The **Create dataset** dialog is displayed.
7. In **Step 1** of the **Create dataset** dialog, perform the following tasks:
   a. Select the **Application** for which you want to create a dataset.
   b. Select **Location** from the drop-down list to store the dataset.
   c. Type a name for the dataset.
   d. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create dataset** dialog, perform the following tasks:
   a. Select **Source** as **Create an Empty dataset**.
   b. Click **Create**.

   Result

   The dataset is displayed on the **Data** page. The dataset created is listed under the logical folder `Datasets` within the selected application.
9. Enter the variable data in the table to match the data that you want to use in tests.
10. Click the **Save** icon to save the dataset.

Results

You created a dataset in your project.

What to do next

You can perform the following tasks:

- Add, modify, or remove data in the dataset. See Editing a dataset on page 433.
- Publish the dataset to the Git repository so that other members of the project can use the dataset. See Publishing changes in datasets.

Related information

Creating a dataset by using a CSV file on page 427

Creating a dataset by using a Data definition on page 429

Creating a dataset by using a CSV file

When you use variable data that is contained in a CSV file, you can import the data from the CSV file and create a dataset in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Saved a CSV file that contains variable data on your computer.
- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:
   - Click **Add files** and select **Dataset** on the **Test Editor** page.
   - Click the **Action Menu** icon  and select **Dataset**.
   - Right-click the application in the data navigator panel, select **New > Dataset**.

The **Edit Branch** dialog is displayed.

If the **Edit branch** dialog is not displayed, then go to step 7 on page 428.

6. Select your action to either select an existing Edit branch or create an Edit branch:
    - Perform the following steps to select an existing Edit branch:
        a. Select **Use an existing edit branch**.
        b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, which you selected, is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

        c. Click **Ok**.
    - Perform the following steps to create the Edit branch:
        a. Select **Create new edit branch**.
        b. Enter a name for the Edit branch in the **Branch Name** field.
        c. Click **Save**.

    Result

    The **Create dataset** dialog is displayed.
7. In **Step 1** of the **Create dataset** dialog, perform the following tasks:
    a. Select the **Application** for which you want to create a dataset.
    b. Select **Location** from the drop-down list to store the dataset.
    c. Type a name for the dataset.
    d. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create dataset** dialog, perform the following tasks:
    a. Select **Source** as **Import a CSV file**.
    b. Click **Select File** and choose the CSV file that contains variable data to import into the dataset.
    c. **Toggle on** the **First row contains headers** option if your CSV file includes column headers.
    d. Click **Create**.

Results

You have created the dataset by using the CSV file in your project.

What to do next

You can perform the following tasks:

- Add, modify, or remove data in the dataset. See Editing a dataset on page 433.
- Publish the dataset to the Git repository so that other members of the project can use the dataset. See Publishing changes in datasets.

Related information

Creating a dataset on page 425

Creating a dataset by using a Data definition on page 429

Creating a dataset by using a Data definition

When you created a Data definition in HCL DevOps Test Hub (Test Hub) to fabricate test data, you can create a dataset by using the Data definition.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created a data definition. See Creating an empty data definition on page 450.
- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:
   - Click **Add files** and select **Dataset** on the **Test Editor** page.
   - Click the **Action Menu** icon  and select **Dataset**.
   - Right-click the application in the data navigator panel, select **New > Dataset**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 430.
6. Select your action to either select an existing Edit branch or create an Edit branch:

- Perform the following steps to select an existing Edit branch:
    a. Select **Use an existing edit branch**.
    b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

    Note: A warning message is displayed if the Edit branch, which you selected, is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

    c. Click **Ok**.
- Perform the following steps to create the Edit branch:
    a. Select **Create new edit branch**.
    b. Enter a name for the Edit branch in the **Branch Name** field.
    c. Click **Save**.

Result

The **Create dataset** dialog is displayed.

7. In **Step 1** of the **Create dataset** dialog, perform the following tasks:
    a. Select the **Application** for which you want to create a dataset.
    b. Select **Location** from the drop-down list to store the dataset.
    c. Type a name for the dataset.
    d. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create dataset** dialog, perform the following tasks:
    a. Select **Source** as **Fabricate data**.
    b. Select the Data definition to use from the **Select Data Definition** list.
    c. Enter the number of rows in the **Number of Rows** field.
    d. Enter a seed value in the **Seed** field.
    e. Click the **Fabricate headers** toggle to generate a header row.
    f. Click **Create**.

Results

You have created the dataset by using a Data definition in your project.

What to do next

You can perform the following tasks:

- Add, modify, or remove data in the dataset. See Editing a dataset on page 433.
- Publish the dataset to the Git repository so that other members of the project can use the dataset. See Publishing changes in datasets.

Related information

Creating a dataset on page 425

Creating a dataset by using a CSV file on page 427

Creating a dataset by using a database query

When you use variable data that is contained in a database, you can create a database query to import the required data from a database and use the data that is present in the database query to create a dataset in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created at least one database query. See Creating a database query on page 417.
- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:
   - Click **Add files** and select **Dataset** on the **Test Editor** page.
   - Click the **Action Menu** icon  and select **Dataset**.
   - Right-click the application in the data navigator panel, select **New > Dataset**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 432.
6. Select your action to either select an existing Edit branch or create an Edit branch:

- Perform the following steps to select an existing Edit branch:
    a. Select **Use an existing edit branch**.
    b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

    Note: A warning message is displayed if the Edit branch, which you selected, is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

    c. Click **Ok**.
- Perform the following steps to create the Edit branch:
    a. Select **Create new edit branch**.
    b. Enter a name for the Edit branch in the **Branch Name** field.
    c. Click **Save**.

Result

The **Create dataset** dialog is displayed.

7. In **Step 1** of the **Create dataset** dialog, perform the following tasks:
    a. Select the **Application** for which you want to create a dataset.
    b. Select **Location** from the drop-down list to store the dataset.
    c. Type a name for the dataset.
    d. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create dataset** dialog, perform the following tasks:
    a. Select **Source** as **Use a Database Query**.
    b. Select the database query to use from the **Select Database Query** list.
    c. Enter the number of rows in the **Number of Rows** field.
    d. Enter a seed value in the **Seed** field.
    e. Click **Create**.

Results

You have created the dataset by using a database query in your project.

What to do next

You can perform the following tasks:

- Add, modify, or remove data in the dataset. See Editing a dataset on page 433.
- Publish the dataset to the Git repository so that other members of the project can use the dataset. See Publishing changes in datasets.

Related information

Creating a dataset on page 425

Creating a dataset by using a CSV file on page 427

Editing a dataset

You can add, modify, remove, import, or export data from a dataset.

Before you begin

You must have created a dataset or configured a repository that contains the dataset.

About this task

If you are a project **Owner** or **Tester** in HCL DevOps Test Hub (Test Hub), you can perform basic tasks in the datasets by right-clicking any row, column, or cell in the dataset to organize your data in a better way. For example, you can perform tasks such as updating data in a cell, inserting or deleting rows and columns, or renaming column names.

After you edit the dataset, you can save the changes made to the dataset, and then you can publish the dataset to the Git repository. If you save and close the edited dataset, the **Changes** page lists the edited dataset and later you can publish it to the Git repository for other members to use.

When you have a CSV file that has data separated from a character, then you can import that CSV file into the dataset. You can select any of the following separator characters from the **Configure Dataset** window, and the selection can be the separator character that you used in the CSV file:

- Comma
- Semicolon
- Space
- Tab
- Other

Consider that you have the data in the CSV file in the following format:

Name;CCNum
John;1234 5678 1234 5678
Bob;1122 3344 5566 7788
Amy;2233 4455 6677 8899

When you import the CSV file in the dataset, and then select the separator value as **Semicolon**, the data in the dataset is displayed as follows:

	Name	CCNum
1	John	1234 5678 1234 5678
2	Bob	1122 3344 5566 7788
3	Amy	2233 4455 6677 8899

If you want the data in its original format, that is, a semicolon (;) character to separate the data, then you can choose any other separator value from the **Configure Dataset** dialog.

Note: The default separator value is **Comma**.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch that contains your project with the dataset that you want to edit.

   The projects in the selected branch are displayed in the Data navigator panel. The dataset are listed under the folder you created to contain the dataset within a logical folder named `Datasets`.
5. Click the dataset in the data navigator.

   The dataset opens in the right pane.
6. Click **Edit**.

   Alternatively, right-click the datset in the data navigator panel, and then click **Edit**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 8 on page 435.
7. Select your action to either select an existing Edit branch or create an Edit branch:

   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

c. Click **Ok**.

- Perform the following steps to create the Edit branch:
  a. Select **Create new edit branch**.
  b. Enter a name for the Edit branch in the **Branch Name** field.
  c. Click **Save**.

8. Perform the following actions to use the options available for editing:

Options	Actions
Find and Replace	To find: a. Click the **Find and Replace** icon. b. Enter the content that you want to search in the **Find** field. c. Select any or all the following options to find the search content more effectively: ▪ Select the **Case sensitive** icon to search the content that is the exact letter case of the content entered in the **Find** field. ▪ Select the **Match entire cell contents** icon to search for cells that contain only the characters that you have entered in the **Find** field. ▪ Select the **Search using regular expression** icon to search the pattern that matches strings. For example, to search a cell that contains any number between 0 to 9, do the following: i. Enter \d in the **Find** field. ii. Select the **Search using regular expression** checkbox. iii. Click **Find**. d. Click the **Find** icon. If the text is found, the cell containing that text is selected. e. Click the **Find** icon again to find further instances of the search text. To find and replace: a. Click the **Find and Replace** icon. b. Enter the content that you want to search in the **Find** field. c. Enter the content that you want to replace in the **Replace** field. d. Select any or all the following options to find and replace the content more effectively:

Options	Actions
	▪ Select the **Case sensitive** icon  to find the content that is the exact letter case of the content entered in the **Find** field. ▪ Select the **Match entire cell contents** icon  to find and replace for cells that contain only the characters that you have entered in the **Find** and **Replace** fields. ▪ Select the **Search using regular expression** icon  to find and replace the pattern that matches strings. e. Click the **Replace** icon ✓ to replace the individual instances. f. Click the **Replace All** icon ✓✓ to replace every instance of the content throughout the dataset.
Undo 	a. Click the **Undo** icon  . b. Select the recent changes from the list that you want to undo, and then click the list. The **Undo** option reverts the change you made in the dataset.
Redo 	a. Click the **Redo** icon  . b. Select the recent changes from the list that you want to redo, and then click the list.
Import 	You can import variable data into a dataset from the following sources: ◦ When there is a large amount of data stored in a CSV file. ◦ When you use the data fabrication feature to generate test data. **Importing data from a CSV file** a. Click the **Import** icon  . b. Select **Import a CSV file** from the drop-down list. c. Click **Select file** and choose the CSV file that contains variable data to import into the dataset. d. Enable the **First row contains headers** by using the toggle icon if your CSV file contains the header. e. Select one of the following options to append or overwrite data in the dataset:

Options	Actions
	▪ Enable **Overwrite existing data** by using the toggle icon to add the rows and columns from the beginning of the dataset. ▪ Enable **Append existing data** by using the toggle icon to add rows and columns to the end of the dataset. f. Click **Import**. **Importing data by using data fabrication** a. Click the **Import** icon  . b. Select **Fabricate data** from the drop-down list. c. Select a Data definition from the drop-down list. d. Enter the number of rows that you want to generate in the **Number of Rows** field. e. Enter a seed value in the **Seed (optional)** field. f. Enable **Fabricate headers** by using the toggle icon if your schema contains the headers. g. Select one of the following options to append or overwrite data in the dataset: ▪ Enable **Overwrite existing data** by using the toggle icon to add the rows and columns from the beginning of the dataset. ▪ Enable **Append existing data** by using the toggle icon to add rows and columns to the end of the dataset. h. Click **Import**.
Export 	You can export variable data from the dataset into a CSV file to reuse in future tests when required. You must have opened a dataset from which you want to export the data. Click the **Export** icon  to download the dataset as a CSV file.
Set as current row	During the test run, if you want variable data to be selected from a current row instead of the first row in a dataset, right-click any cell in a row and select **Set as current row**. **When rows are deleted:** If you delete any row between row 1 and the current row, the current row data is taken from the next row. For example, when you set the current row as 6, and then

Options	Actions
	you delete any row between row 1 to row 6, the current row remains at row 6, but the content of row 7 is moved to row 6. **When rows are inserted:** If you insert any new row between row 1 to the current row, the current row data is taken from the previous row. For example, when you set the current row as 6, and then you insert any row between row 1 to row 6, the current row remains at row 6, but the content of row 5 is moved to row 6.
Dataset Configuration	You can configure the dataset by clicking the **Menu** icon ⋮, and then selecting **Configure**. In the **Configure Dataset** dialog, you can change the row and column settings and configure the string values in the dataset that contains variable data for tests to use when they run. ◦ Select any of the separator values that you used in the CSV file. The available options are **Comma**, **Semicolon**, **Space**, **Tab**, and **Other**. In the CSV file, if you have any other separator characters other than the available options, then you can select the **Other** option, and then can specify a value. For example, if the data in the CSV file is separated by a character #, then select the **Other** option and enter # in the field. ◦ Configure the following options to change the row and column settings: ▪ **Column header** - Use an up-down control button to increment or decrement the value of the column header. ▪ **Data start point** - Use an up-down control button to increment or decrement the value of the data starting pointer. ▪ **Current row** - Use an up-down control button to increment or decrement the value of the current row. ◦ Configure the following options to change the string values in the dataset: ▪ **Treat as null** - Enter a string value that is to be treated as null when running the test. ▪ **Treat as empty** - Enter a string value that is to be treated as empty when running the test.

Options	Actions
	For example, when you run the test and the data *123* in the dataset is to be treated as empty, then you must specify *123* in the **Treat as empty** field. ▪ **Treat empty text as null** - Select this field when you want the dataset that contains any blank cells, and the value of those blank cells to be interpreted as null. ◦ Click **Update** to apply the changes.
Discard	Click the **Menu** icon and select **Discard** to discard the changes made to the dataset.

9. Click the **Save** icon to save the changes made to the dataset or the **Save All** icon in the Data navigator panel to save all changes made to different datasets by using the same Edit branch.

Note: When editing multiple assets (such as datasets, data definitions or schemas) simultaneously, right-click a test asset in the Data page to manage open tabs using the following options:

To...	Do this...
Keep your current file open and close the open files on the right.	Right-click, and then click the **Close Tabs to right** option.
Keep your current file open and close the open files on the left.	Right-click, and then click the **Close Tabs to left** option.
Keep your current file open and close all the other open files.	Right-click, and then click the **Close Others** option.
Close all open file tabs.	Right-click, and then click the **Close All Tabs** option.

10. Click the **Publish** icon to publish the dataset to the Git repository.

Results

You have edited the dataset.

Dataset encryption

Encrypted datasets are useful when you want to run tests that contain confidential information such as a set of passwords or account numbers.

When you run a test in HCL DevOps Test Hub (Test Hub) that uses an encrypted dataset, then you must provide an encryption key for decrypting the encrypted data in columns so that the data can be used in the test. If the test uses data from the multiple encrypted dataset columns, you must enter the same encryption key for every encrypted dataset column that the test uses.

When you run the test that uses the dataset with the encrypted column, the value of the column is decrypted at a run time. The data in the column is sent as a clear-text string in requests to the server. The actual values of the encrypted dataset variables are not displayed in the test log. The test log displays asterisks for the encrypted dataset variables.

You can use only one encryption key to encrypt data in the columns in any dataset.

You can set the dataset security group (encrypted password) for a dataset. The dataset security group that you set is applied for the dataset if it is saved in different branches of the project repository and you need not set a password for the dataset in each branch. If you want to use different passwords, then you can create a different dataset by cloning the dataset that you want.

Important:

The encryption keys that you use to encrypt data in a dataset are not stored on the server, nor they can be retrieved from the server. Therefore, you must remember to store the encryption keys in a secure location. You must use the same encryption keys to perform the following operations:

- View the encrypted values
- Decrypt data
- Enable the use of the encrypted dataset during test runs

Encrypting a dataset column

To secure test data, you must encrypt datasets in HCL DevOps Test Hub (Test Hub). You can encrypt data in the columns of a dataset by using an encryption key. When you run a test that utilizes a dataset with encrypted variables, you must enter the encryption key for the encrypted column that the test uses.

Before you begin

You must have created a dataset. See Creating a dataset on page 425.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.

4. Select the branch that contains your project with the dataset that you want to edit.

   The projects in the selected branch are displayed in the Data navigator panel. The dataset are listed under the folder you created to contain the dataset within a logical folder named `Datasets`.

5. Click the dataset in the data navigator.

   The dataset opens in the right pane.

6. Click **Edit**.

   Alternatively, right-click the datset in the data navigator panel, and then click **Edit**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 8 on page 435.

7. Select your action to either select an existing Edit branch or create an Edit branch:

   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

8. Right-click any cell in a column that you want to encrypt and select **Encrypt column data**.
   Result
   The **Encrypt Column** window is displayed.
9. Enter an encryption key in the **Encryption Key** field to encrypt the data in the column.

   Remember: When you have already encrypted other columns in the dataset, you must enter the same encryption key that you used previously. You can use only one encryption key to encrypt columns in a dataset.

   Important:

   The encryption keys that you use to encrypt data in a dataset are not stored on the server, nor they can be retrieved from the server. Therefore, you must remember to store the encryption keys in a secure location. You must use the same encryption keys to perform the following operations:

- View the encrypted values
- Decrypt data
- Enable the use of the encrypted dataset during test runs

10. Click **Encrypt Column**.
    Result
    Asterisks are displayed instead of actual data for the encrypted column.
11. Click the **Save** icon to save the changes to the dataset.

Results

You have encrypted the dataset column in your project.

What to do next

You can publish the dataset to the Git repository so that other members of the project can use the dataset by clicking the **Publish** icon. See Publishing changes in datasets.

Decrypting a dataset column

To view the content of an encrypted dataset in HCL DevOps Test Hub (Test Hub), you can decrypt the dataset column. Removing encryption from a dataset revokes the protection offered to the test data.

Before you begin

You must have created at least one dataset and encrypted the dataset with an encryption key. See Creating a dataset on page 425 and Encrypting a dataset column on page 440.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch that contains your project with the dataset that you want to edit.

   The projects in the selected branch are displayed in the Data navigator panel. The dataset are listed under the folder you created to contain the dataset within a logical folder named `Datasets`.
5. Click the dataset in the data navigator.

   The dataset opens in the right pane.
6. Click **Edit**.

Alternatively, right-click the datset in the data navigator panel, and then click **Edit**.

The **Edit Branch** dialog is displayed.

If the **Edit branch** dialog is not displayed, then go to step 8 on page 435.

7. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.
8. Right-click encrypted cells that display the contents with asterisks, and then select **Decrypt column data**.
   Result
   The **Decrypt Column** window is displayed.
9. Enter the encryption key that you used to encrypt the data in the column in the **Encryption Key** field.
10. Click **Decrypt Column**.
    Result
    Asterisks are replaced with the actual data in the decrypted column.
11. Click the **Save** icon to save the changes to the dataset.

Results

The encryption is removed from the selected column in the dataset. When you run a test that uses a dataset that contains decrypted data, the variable data is substituted for the original data in the recorded test without prompting for the encryption.

What to do next

You can publish the dataset to the Git repository so that other members of the project can use the dataset by clicking the **Publish** icon. See Publishing changes in datasets.

Viewing a dataset

When test assets includes datasets, you can view the contents of a dataset from HCL DevOps Test Hub (Test Hub). The datasets residing in the Git repository are listed in the **Datasets** page.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Created or joined a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created at least one dataset and encrypted the dataset with an encryption key. See Creating a dataset on page 425 and Encrypting a dataset column on page 440.
- Created a security group for an encrypted dataset. See Managing an encrypted dataset on page 447.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch that contains your project with the dataset that you want to edit.

   The projects in the selected branch are displayed in the Data navigator panel. The dataset are listed under the folder you created to contain the dataset within a logical folder named `Datasets`.
5. Click the dataset in the data navigator.

   The dataset opens in the right pane.

Results

You have viewed the contents of the dataset.

Deleting a dataset

You can delete the dataset when it is not required in your test environment.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Created at least one dataset in your Git repository that you have configured.

About this task

You can delete datasets in Test Hub in the following scenarios:

- Datasets that you created in the desktop clients and that are cloned to the repository in your project on Test Hub.
- Datasets that you created in Test Hub and that are published to the repository in your project.
- Datasets that are in the `.csv` file format.

You cannot delete the following datasets:

- Datasets that you created or edited in Test Hub and that are not published to the repository in your project.
- Datasets that are in the `.sit` file format.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch that contains your project with the datasets that you want to edit.

   The projects in the selected branch are displayed in the Data navigator panel. The datasets are listed under the folder you created to contain the dataset within a logical folder named as `Datasets`.
5. Right-click the dataset in the data navigator, and then click the **Delete** option.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 446.
6. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

7. Perform the following steps in the **Delete <Dataset>** dialog that is displayed, where *<Dataset>* displays the name of the selected dataset.
    a. Select the option to publish the deletion from the edit branch to the main branch.
    b. Click **Delete**.

    The selected dataset is deleted and the deletion is published from the edit branch to the main branch.

Results

You deleted a dataset.

Viewing an encrypted dataset

You can use the **Dataset** page to view the contents of an encrypted dataset from HCL DevOps Test Hub (Test Hub).

Before you begin

You must have opened the dataset to view the contents.

1. Perform the following steps, if you have created a security group but yet to add the encrypted dataset to it:
    a. Find the encrypted dataset that you are interested in, and then right-click it.
    b. Click the **Security** option  in the menu.
    The **Change the security group for the dataset** dialog is displayed.
    c. Select a security group from the list and enter the encryption key set for the dataset.
    d. Click **Save** to save the security group details.
2. Right-click the encrypted column and click **Show encrypted data**.
    You can right-click the decrypted column and select **Hide encrypted data** to encrypt the data again.

    Important: The **Show encrypted data** option is available only when you have added the encrypted dataset to the respective security group.

Results

You have viewed the contents of an encrypted dataset.

Changing security group for an encrypted dataset

To move the encrypted dataset from one security group to another dataset in HCL DevOps Test Hub (Test Hub), you can change the security group for an encrypted dataset.

Before you begin

You must have created at least two or more security groups. See Creating a security group on page 404.

1. Go to the **Datasets** page and find an encrypted dataset for which you want to change the security group.
2. Click the **Lock** icon from the **Actions** column of a dataset.
3. In the **Change the security group for the Dataset** window:
    a. Choose the security group from the list.
    b. Enter the encryption key for the dataset that was set while encrypting the dataset column.
    c. **Save** the security group details.

Remember: You can perform this task only if you know the encryption key set for the dataset while encrypting the dataset column.

Results

You have changed the security group for an encrypted dataset.

Related information

Managing an encrypted dataset on page 447

Creating a security group on page 404

Managing an encrypted dataset

You can use encrypted datasets to limit access to confidential information such as account number or passwords. You can arrange data by an appropriate category so that project members can use datasets more effectively in certain tests and protect them.

About this task

A dataset can contain classified information that other members can access with permission. As a project owner, you can group encrypted datasets into different security groups and enable project members to view and edit datasets and run tests associated with the encrypted datasets. After you created a security group, you can change the security group for a dataset. You can also delete a security group if you do not require it in your test environment.

Notes:

- You must grant access and provide an encryption key of an encrypted dataset to other members of the project to work with the encrypted dataset.
- A project member who has been added as a *Tester* role can work with the encrypted dataset.

Adding an encrypted dataset to a security group

As a project *Project Owner* or *Tester*, you can organize encrypted datasets by adding it to a security group so that project members can use and protect datasets more efficiently.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Created a project in Test Hub. See Managing DevOps Test Hub projects on page 1209.
- Configured the repository that contains the test assets in your project. See Adding repositories to a server project on page 1214.
- Created at least one dataset and encrypted the dataset with an encryption key. See Dataset encryption on page 439.
- Created at least one security group. See Creating a security group on page 404.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Click **Add Dataset** to select the encrypted dataset to become part of the security group.
   The **Add Dataset** dialog is displayed.
5. Perform the following steps:
   a. Click the **Encrypted dataset** field to view the encrypted dataset.
   b. Identify and select the encrypted dataset that you wanted to add to the security group.
   c. Enter the encryption key that you used to encrypt the dataset in the **Password** field.

      Tip: You can click the view icon to view the entered encryption key and ensure that you entered the correct key.

   d. Click **Add**.

      The encrypted dataset that is added to the security group is displayed.
   e. You can add other encrypted datasets to the same security group, if required.

What to do next

You can grant access to other project members to use the encrypted dataset. See Granting access to security group on page 406.

Moving an encrypted dataset to another security group

When you add many encrypted datasets to the same security group, you can move some of them to another security group or when you want to delete a security group but you want to retain the encrypted datasets associated with it, you can move them to different security group before deleting the security group.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Created at least one dataset and encrypted the dataset with an encryption key. See Dataset encryption on page 439
- Created at least two security groups. See Creating a security group on page 404.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Click a security group from the list under **Security Group** from which you want to move the encrypted dataset.
5. Click the **Edit** icon inline with the encrypted dataset that you want to move to another security group.
   The **Move dataset to a different Security Group** dialog is displayed.
6. Click a security group from the list under **Security Group** to which you want to move the encrypted dataset.
7. Enter the encryption key that you used to encrypt the dataset in the **Password** field.
8. Click **Save**.

   The encrypted dataset is moved to the security group that you selected and the same is displayed on the **Security** page.

Results

You have moved the encrypted dataset from one security group to another.

Removing a dataset from the security group

You can remove a dataset from a security group when it is no longer required.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Created at least one dataset and encrypted the dataset with an encryption key. See Dataset encryption on page 439
- Created a security group and added the encrypted dataset to it. See Creating a security group on page 404.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.

3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Click a security group from the list under **Security Group** from which you want to delete the encrypted dataset.
5. Click the **Delete** icon in line with the encrypted dataset that you want to delete from the security group.
   The **Remove Dataset** dialog is displayed.
6. Click **Remove**.

Note: Removing dataset from the security group also removes the password stored in Test Hub for encrypted data. You must provide the password again while adding it to a different security group.

Results

You have removed the datasets from a security group.

Management of data definition

You can find information about creating, editing, and managing data definitions in HCL DevOps Test Hub (Test Hub), to support realistic and reusable test data generation.

You can create a data definition in Test Hub by using any of the following methods:

- Creating an empty data definition. See Creating an empty data definition on page 450.
- Importing data from a CSV file. See Creating a data definition by importing fields from a CSV file on page 452.
- Creating a data definition from a dataset. See Creating a data definition by importing fields from a dataset on page 454
- Creating a data definition from a database query. See Creating a data definition by importing fields from a database query on page 456.

When you want to edit the data in a data definition, you must open it for editing. See Modifying a data definition on page 462.

After you edit the data fefinition, you must publish the changes, so that the data definition is added to the project repository. See Publishing changes in datasets.

Creating an empty data definition

You can create an empty data definition in HCL DevOps Test Hub (Test Hub) as the first step to create a data definition. After you save the empty data definition within the selected application in your project repository, you can edit the data definition to add the different generators.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Read information about using data definitions to fabricate test data. See Overview of data fabrication.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:

   - Click the **Open action menu** icon  in the Data navigator, and then click **Data definition**.
   - Click **Add files** in the right panel, and then click **Data definition**.
   - Right-click the application in the data navigator panel, select **New > Data definition**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 452.
6. Select your action to either select an existing Edit branch or create an Edit branch:

   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:

    a. Select **Create new edit branch**.
    b. Enter a name for the Edit branch in the **Branch Name** field.
    c. Click **Save**.
7. In **Step 1** of the **Create Data Definition** dialog, perform the following tasks:
    a. Select the **Application** for which you want to create a data definition.
    b. Select **Location** from the drop-down list to store the dataset.
    c. Type a **Name** for the data definition.
    d. Add a description about the data definition in the **Description** field.
    e. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create Data Definition** dialog, perform the following tasks:
    a. Select **Source** as **Create an empty data definition**.
    b. Click **Create**.

    The new data definition opens in the **Data** page in the editing mode. The data definition created is listed under the logical folder `Data definitions` under the application selected.

Results

You created an empty data definition in your project.

What to do next

You can perform the following tasks:

- Add basic generators with their default settings to the data definition. You can then save and publish the data definition. See Saving a data definition by adding basic generators with their default settings on page 459.
- Add basic generators to the data definition, and then modify the configuration of the generators. You can then save and publish the data definition. See Saving a data definition by adding basic generators with their modified settings on page 460.

Related information

Creating a data definition by importing fields from a CSV file on page 452

Creating a dataset by using a Data definition on page 429

Creating a data definition by importing fields from a CSV file

When you want to create a data definition in HCL DevOps Test Hub (Test Hub) with specific fields for the test data, you can import a CSV file that defines the specific fields. After you import fields to the data definition, you can edit the data definition and add generators for the specific fields before you generate the test data.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Ensured that a CSV file is available on your computer, from which you want to import data.
- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:

   - Click the **Open action menu** icon  in the data navigator, and then click **Data definition**.
   - Click **Add files** in the right panel, and then click **Data definition**.
   - Right-click the application in the data navigator panel, select **New > Data definition**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 454.
6. Select your action to either select an existing Edit branch or create an Edit branch:

   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

7. In **Step 1** of the **Create Data Definition** dialog, perform the following tasks:
    a. Select the **Application** for which you want to create a data definition.
    b. Select **Location** from the drop-down list to store the dataset.
    c. Type a **Name** for the data definition.
    d. Add a description about the data definition in the **Description** field.
    e. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create Data Definition** dialog, perform the following tasks:
    a. Select **Source** as **Imports fields from a CSV file**.
    b. Click **Select file** and browse to the location on your computer where you have the CSV file.
    c. Click **Create**.

The new data definition opens in the **Data** page in the editing mode. The data definition created is listed under the logical folder `Data definitions` under the application selected.

Results

You created a data definition in your project, which has the fields from the CSV added as the fields of the new data definition but no generators are added.

What to do next

You can perform the following tasks:

- Add basic generators with their default settings to the data definition. You can then save and publish the data definition. See Saving a data definition by adding basic generators with their default settings on page 459.
- Add basic generators to the data definition, and then modify the configuration of the generators. You can then save and publish the data definition. See Saving a data definition by adding basic generators with their modified settings on page 460.

Related information

Creating an empty data definition on page 450

Creating a dataset by using a Data definition on page 429

Creating a data definition by importing fields from a dataset

When you want to create a data definition in HCL DevOps Test Hub (Test Hub) with specific fields for the test data, you can import the fields from a dataset. After you import fields to the Data definition, you can edit the data definition and add generators for the specific fields before you generate the test data.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Ensured that a dataset is available in your project repository, from which you want to import data.
- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:

   - Click the **Open action menu** icon  in the Data navigator, and then click **Data definition**.
   - Click **Add files** in the right panel, and then click **Data definition**.
   - Right-click the application in the data navigator panel, select **New > Data definition**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 456.
6. Select your action to either select an existing Edit branch or create an Edit branch:

   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

7. In **Step 1** of the **Create Data Definition** dialog, perform the following tasks:
    a. Select the **Application** for which you want to create a data definition.
    b. Select **Location** from the drop-down list to store the dataset.
    c. Type a **Name** for the data definition.
    d. Add a description about the data definition in the **Description** field.
    e. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create Data Definition** dialog, perform the following tasks:
    a. Select **Source** as **Import fields from a dataset**.
    b. Search for the dataset in your project repository by entering the file name in the **Dataset** field.

       Alternatively, browse to the location in your project repository, and then select the file.
    c. Click **Create**.

The new data definition opens in the **Data** page in the editing mode. The headers in the dataset file are added as the fields of the new data definition but no generators are added.The data definition created is listed under the logical folder `Data definitions` under the application selected.

Results

You created a data definition in your project, which has the fields from the dataset added as the fields of the new data definition but no generators are added.

What to do next

You can perform the following tasks:

- Add basic generators with their default settings to the data definition. You can then save and publish the data definition. See Saving a data definition by adding basic generators with their default settings on page 459.
- Add basic generators to the data definition, and then modify the configuration of the generators. You can then save and publish the data definition. See Saving a data definition by adding basic generators with their modified settings on page 460.

Related information

Creating an empty data definition on page 450

Creating a dataset by using a Data definition on page 429

Creating a data definition by importing fields from a database query

You can create data definition to generate a random test data that is required to run a test in HCL DevOps Test Hub (Test Hub) using the specific database fields.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created an application in your project. See Creating an application on page 204.
- Created at least one database query. See Creating a database query on page 417.

About this task

You must first create a database query and then create a data definition using the database query to import the specific fields from the database query. After you import fields to the data definition, you can edit the data definition and add generators for the specific fields before you generate the test data.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:

   - Click the **Open action menu** icon  in the data navigator, and then click **Data definition**.
   - Click **Add files** in the right panel, and then click **Data definition**.
   - Right-click the application in the data navigator panel, select **New > Data definition**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 458.
6. Select your action to either select an existing Edit branch or create an Edit branch:

   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

c. Click **Ok**.

- Perform the following steps to create the Edit branch:
  a. Select **Create new edit branch**.
  b. Enter a name for the Edit branch in the **Branch Name** field.
  c. Click **Save**.

7. In **Step 1** of the **Create Data Definition** dialog, perform the following tasks:
   a. Select the **Application** for which you want to create a data definition.
   b. Select **Location** from the drop-down list to store the dataset.
   c. Type a **Name** for the data definition.
   d. Add a description about the data definition in the **Description** field.
   e. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create Data Definition** dialog, perform the following tasks:
   a. Select **Source** as **Import fields from a database query**.
   b. Select the required database query file in your project.
   c. Click **Create**.

The new data definition opens in the **Data** page in the editing mode. The columns in the database query are added as the fields of the new Data definition but no generators are added. The data definition created is listed under the logical folder `Data definitions` under the application selected..

Results

You have created a data definition in your project by importing fields from a database query.

What to do next

You can perform the following tasks:

- Add basic generators with their default settings to the data definition. You can then save and publish the data definition. See Saving a data definition by adding basic generators with their default settings on page 459.
- Add basic generators to the data definition, and then modify the configuration of the generators. You can then save and publish the data definition. See Saving a data definition by adding basic generators with their modified settings on page 460.

Related information

Creating an empty data definition on page 450

Creating a dataset by using a Data definition on page 429

Saving a data definition by adding basic generators with their default settings

After you create a data definition in HCL DevOps Test Hub (Test Hub), you must add the basic generators with their default settings and save the data definition before you can generate test data.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created an empty data definition that is opened for editing.

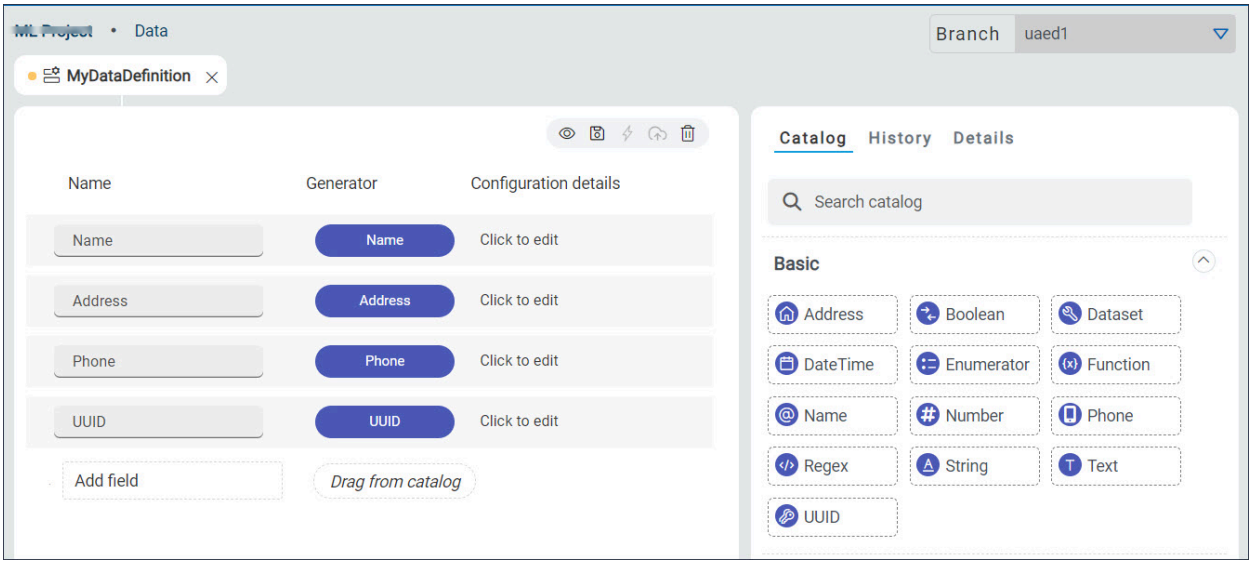
1. Drag a basic generator from the **Catalog** panel to the data definition.

   The basic generator is added as a field in the data definition.

2. Drag other basic generators from the **Catalog** panel to the data definition to complete the data structure that you want in the fabricated data.

   Do not change any of the fields in the data definition or the configuration of the basic generators.

   For example, if you want to fabricate data about persons with their name, address, phone number, and a unique ID for each name or record, you can drag the **Name**, **Address**, **Phone**, and **UUID** generators to the data definition, without making any changes to any of the fields in the data definition or in the basic generators.



3. Click the **Save** icon to save the changes made to the Data definition or the **Save All** icon to save all changes made to different Data definitions by using the same Edit branch.

Note: When you edit different files that include datasets or Data definitions simultaneously, you can select your action from the following options:

To...	Do this...
Keep your current file open and close the open files on the right.	Right-click, and then click the **Close Tabs to right** option.
Keep your current file open and close the open files on the left.	Right-click, and then click the **Close Tabs to left** option.
Keep your current file open and close all the other open files.	Right-click, and then click the **Close Others** option.
Close all open file tabs.	Right-click, and then click the **Close All Tabs** option.

Note: The changes you made are only visible and available to you in the Edit branch. You must publish the changes in the Edit branch to the branch in the project repository for the changes to be visible to the other members of the project.

4. Click the **Publish** icon to publish the changes made to the data definition to the branch in the project repository.

Results

You edited and saved a data definition after adding basic generators with their default settings.

What to do next

You can perform any of the following tasks:

- Preview the test data generated by using the generators added as fields in the data definition by clicking the **Preview** icon .
- Generate the test data by using the generators added as fields in the data definition by clicking the **Generate** icon .

Saving a data definition by adding basic generators with their modified settings

After you create an empty data definition in HCL DevOps Test Hub (Test Hub), you must add the basic generators with their modified settings, and save the data definition before you can generate test data.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created an empty data definition that is opened for editing.

1. Drag a basic generator from the **Catalog** panel to the data definition.

   The basic generator is added as a field in the data definition.
2. Drag other basic generators from the **Catalog** panel to the data definition to complete the data structure that you want in the fabricated data.
3. Perform the following steps to modify the parameter configurations of a basic generator in a field of the data definition:

   a. Click a generator in the **Generator** column of a field where you want to customize the configuration.
   The parameters of the generator are displayed in the **Generator** dialog.

   b. Change the parameter settings in the **Generator** dialog to customize the generator.

   c. Close the **Generator** dialog.
   The changes in the generator configuration are automatically saved.
4. Click the **Save** icon to save the changes made to the Data definition or the **Save All** icon to save all changes made to different Data definitions by using the same Edit branch.

Note: When you edit different files that include datasets or Data definitions simultaneously, you can select your action from the following options:

To...	Do this...
Keep your current file open and close the open files on the right.	Right-click, and then click the **Close Tabs to right** option.
Keep your current file open and close the open files on the left.	Right-click, and then click the **Close Tabs to left** option.
Keep your current file open and close all the other open files.	Right-click, and then click the **Close Others** option.

To...	Do this...
Close all open file tabs.	Right-click, and then click the **Close All Tabs** option.

Note: The changes you made are only visible and available to you in the Edit branch. You must publish the changes in the Edit branch to the branch in the project repository for the changes to be visible to the other members of the project.

5. Click the **Publish** icon to publish the changes made to the data definition to the branch in the project repository.

Results

You edited and saved a data definition after adding basic generators and then modifying the configuration of the generators.

What to do next

You can perform any of the following tasks:

- Preview the test data generated by using the generators added as fields in the data definition by clicking the **Preview** icon .
- Generate the test data by using the generators added as fields in the data definition by clicking the **Generate** icon .

Modifying a data definition

After you create and publish a data definition in HCL DevOps Test Hub (Test Hub), you can modify the fields or the configuration of the generators in an existing data definition when you want to change the structure of the test data.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a data definition. See Creating an empty data definition on page 450.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Click the data definition that you want to edit in the data navigator panel.

Alternatively, right-click the data definition in the data navigator panel, and then click **Edit**.

The selected data definition opens in the **Data** page.

5. Perform the actions that are listed in the following table to modify the data definition:

Modification	Action
Addition of new fields	Click the **Add field** option, and then enter a name for the field in the **Name** column. **Note:** A new row with an empty field is automatically displayed below the row that you edit.
Addition of generators in fields	Drag a generator from the **Catalog** panel into a field under the **Generator** column. **Note:** You can use the **Search catalog** field in the **Catalog** panel to find the required generator.
Replacement of generators in fields	Drag a generator from the **Catalog** panel into an existing generator field under the **Generator** column. The existing generator in the field is automatically replaced by the generator.
Modification of generator configurations	Click a generator in the **Generator** column. The generator configurations are displayed. Modify the options and settings in the generator configuration. **Note:** If you modify the configurations of a basic generator, the changes are applicable only when the generator is used as field in a Data definition. The basic generator that is available in the catalog is not modified nor saved.
Inclusion of additional configuration details	Click the **Add new configuration** tab. Provide a name, and enter or select the other settings as an additional configuration to the generator.
Arrangement of fields	Drag and drop the field rows in the Data definition panel to rearrange the fields in the order you want.
Deletion of fields	Click the delete icon in the row of a field to remove the field from the Data definition.
Deletion of Data definition	Click the **Delete** icon in the Data definition panel to delete the Data definition from the existing location.

6. Click the **Save** icon to save the changes made to the Data definition or the **Save All** icon in the Data navigator panel to save all changes made to different Data definitions by using the same Edit branch.

Note: When editing multiple assets (such as datasets, data definitions or schemas) simultaneously, right-click a test asset in the Data page to manage open tabs using the following options:

To...	Do this...
Keep your current file open and close the open files on the right.	Right-click, and then click the **Close Tabs to right** option.
Keep your current file open and close the open files on the left.	Right-click, and then click the **Close Tabs to left** option.
Keep your current file open and close all the other open files.	Right-click, and then click the **Close Others** option.
Close all open file tabs.	Right-click, and then click the **Close All Tabs** option.

Note: The changes you made are only visible and available to you in the Edit branch. You must publish the changes in the Edit branch to the branch in the project repository for the changes to be visible to the other members of the project.

7. Click the **Publish** icon to publish the changes made to the data definition to the branch in the project repository.

Results

You modified an existing data definition in your project.

What to do next

You can generate test data by using the modified Data definition. See Generating fabricated test data on page 483.

Viewing a data definition and its details

You can view the contents of a data definition on the Data page in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a data definition. See Creating an empty data definition on page 450.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch from the **Branch** list.
   The navigation panel displays all the existing projects in the selected branch.
5. Find the data definition that you want to view, and then click it.

   You can either enter the name of the data definition in the **Search** field or expand the project and manually search in the data navigator panel.

   The selected data definition opens in the **Data** page and the following elements are displayed:

Element	Description
The **Name** column	A list of fields and their custom names that exist in the Data definition is displayed in the column.
The **Generator** column	The generator that is selected for the corresponding field is displayed in the column. **Note:** If any of the fields does not have a corresponding generator, then you cannot preview or generate the test data for the Data definition.
The **Configuration details** column	The configuration details of each field are displayed in the column. **Note:** The column displays details only when additional configuration is added for the field. Also, if you have added three or more additional configurations, then the ellipses icon ... is displayed. You can hover over the icon to view the list of additional configuration names along with their distribution values. You can also click the column of a field to view the existing configuration details for the field in the corresponding *Generator* dialog.
Initials of members	The colored avatars, that represent the initials of the project members who are currently editing the data definition, are displayed in the page. You can hover over an avatar to view the name of the member and the branch that they are currently editing. This information can facilitate collaboration and communication among team members who are working on the same Data definition simultaneously. To

Element	Description
	understand how you can resolve conflicting edits, see Modifying a data definition on page 462.
The **Preview** icon 	You can click the icon to preview the test data in the preview dialog of the **Data** page. The preview dialog contains the generate icon  and the close icon  . **Notes:** ◦ If you are in an Edit branch, then you can access the **Generate** icon  in the preview dialog only if the Data definition is saved. ◦ If you modify the Data definition but not save it, you can still preview the corresponding test data. However, you can generate the test data only after you save the Data definition.
The **Save** icon 	You can click the icon to save the Data definition. If you modified the Data definition and you want to continue using the modified Data definition to generate test data without publishing it, you must save the changes. However, if you want to collaborate with other members of your project, you must publish the Data definition.
The **Generate** icon 	You can click the icon to generate the test data in the CSV format. See Generating standard test data by using data definitions and exporting it to a .zip file on page 484.
The **Delete** icon 	You can click the icon to delete the Data definition. See Deleting a data definition on page 468.
The **Catalog** tab	Provides a structured overview of the available built-in and custom generators for the selected project.
The **History** tab	Provides an overview of all the previously generated test data. The following information about the history of the generated data is displayed: ◦ The month, date, and year of the generated test data. ◦ The status of the download that indicates whether the data generation was successful or failed. ◦ The number of rows in the generated test data. ◦ The initials of the project member who downloaded the generated test data. You can also use the following options to work with the past data:

Element	Description
	◦ You can click the preview icon  to review the already generated data before taking any further actions. ◦ You can click the generate icon  to regenerate the same test data, if required. See Generating test data by using data definition history on page 488. ◦ You can click the delete icon  to delete specific test data history. You get a clutter-free view of the most relevant data and maintain your data generation history effectively.
The **Details** tab	Provides following details about the Data definition and its associated Git details: ◦ Name of the branch that contains the data definition. ◦ Description for the Data definition, if any. ◦ Git-related details for the Data definition as follows: ▪ Asset ID: The ID uniquely identifies the data definition. You also have the option to copy the ID for easy reference or use. ▪ Repo: The repository name and path where the Data definition is stored is displayed. ▪ Path: The exact location of the data definition within the repository is displayed. ▪ Commit: The Git commit message, the name of the member who performed the changes, and the date and time of the commit are displayed. ▪ The copy button: You can click the button to copy the complete SHA commit ID for the data definition. ▪ The commit ID button: You can click the button to view the changes that were made in the commit directly within Git. ▪ The browse button: You can click the button to view the specific location in the Git history where the commit was performed for the data definition.
The Data definition tabs	When you open a Data definition, the name of the currently opened Data definition is displayed as a tab in the **Data** page. Beside this tab, additional tabs with the names of all other data definitions that are currently open on the **Data** page are also displayed. You can use these tabs to navigate between different Data definitions easily. Also, when you hover over the tab of a Data definition, a tooltip is displayed that provides the location and repository details where the Data definition is stored.

Element	Description
The **Branch** list	The list displays the name of the branch where you currently opened the Data definition. You can also use the list to choose and select a different branch from the available options, if required. You can switch between branches easily.

Note: The **Save** icon and the **Delete** icon are displayed in the Data definition page only if you are in an Edit branch. If you are in a branch other than the Edit branch, then an **Edit** icon is displayed in the Data definition. You can click the **Edit** icon to perform any modification to the Data definition. Also, if you do not have either the owner or the tester role for the Data definition, you cannot save, modify, or delete the Data definition.

Results

You have viewed the contents of the data definition in your project.

What to do next

You can perform any of the following tasks:

- Modify the data definition. See Modifying a data definition on page 462.

Deleting a data definition

You can delete a data definition when it is outdated or no longer relevant to your project. Additionally, when data definitions contain errors or inaccuracies that can lead to misleading or incorrect fabricated data, then you might want to remove them. This ensures that the project remains clutter-free, optimizes the performance of HCL DevOps Test Hub (Test Hub), and reduces storage requirements.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a data definition. See Creating an empty data definition on page 450.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.

4. Select the data definition in the data navigator panel.

   Result

   The data dedfinition is displayed in the **Data** page

5. Click the **Delete** icon to delete the data definition.

   Alternatively, right-click the data definition in the data navigator panel, and then click **Delete**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 469.

6. Select your action to either select an existing Edit branch or create an Edit branch:

   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

     Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

7. Click the **Delete** button in the **Delete Data definition** dialog.

Results

You have deleted a data definition from your project.

What to do next

You can publish the changes performed to the data definition (that is, deletion of the data definition) to the Git repository so that other members of the project do not see the deleted data definition. See Publishing all changes in an Edit branch on page 493.

Management of schemas

A schema, also known as a database schema, defines how data is structured, organized, and managed in a database management system (DBMS). It outlines the format, storage, processing, security, and data access across various structural schema elements such as tables, views, and logical constraints.

The following sections provide an overview of how schema concepts are applied within Test Hub to support efficient data management and test design:

- Using schema in Test Hub on page 470
- Key benefits of schema in Test Hub on page 470
- Primary key on page 470
- Foreign key on page 470

Using schema in Test Hub

- You can create a new schema either as an empty structure or by connecting it to an existing database, allowing you to visualize how data is structured and related in real environments. See Creating an empty schema on page 470 and Creating a schema by using a database connection on page 473.
- After the schema is created, two separate tabs are displayed based on the database connection:
    - **Database View**
    - **Schema View**
- You can add data definitions and establish relationships between their fields in a schema.

Key benefits of schema in Test Hub

The schema in Test Hub offers the following advantages:

- It defines how data is stored and related within the database, making it easier to understand, manage, and maintain.
- You can group similar data to reuse it across different scenarios, even when it serves different purposes.
- You can enhance data integrity by using logical constraints such as tables, primary keys, and relationships that helps maintain data consistency and accuracy while reducing errors.

To use schema in Test Hub, you must be aware of the terms and concepts that are used to link database tables.

Primary key

A primary key is one or more columns in a database table that uniquely identify each record in that table. The primary key enforces data integrity by ensuring that no two rows have the same value(s) in the key column(s) and that these values are never null. Database systems use the primary key to efficiently retrieve, update, or relate records across multiple tables in a relational database.

Foreign key

A foreign key is a link between two tables that ensures the data in one table corresponds with the data in another table, thereby maintaining uniformity of the database information.

Creating an empty schema

You can create an empty schema to define a structured data storage. Initially, the schema is not connected to any database.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created an application in your project. See Creating an application on page 204.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:
   - Click **Add files** and select **Schema** on the **Test Editor** page.
   - Click the **Action Menu** icon  and select **Schema**.
   - Right-click the application in the data navigator panel, go to **New**, and select **Schema**.

   The **Edit Branch** dialog is displayed.

   If the **Edit branch** dialog is not displayed, then go to step 7 on page 472.
6. Select your action to either select an existing Edit branch or create an Edit branch:

   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

     Note: A warning message is displayed if the Edit branch, which you selected, is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

Result

The **Create a Schema** dialog is displayed.

7. In **Step 1** of the **Create a Schema** dialog, perform the following tasks:
    a. Select the **Application** for which you want to create a schema.
    b. Select **Location** from the drop-down list to store the schema.
    c. Type a unique **Name** for the schema.
    d. Type a brief description of the schema in the **Description** field.
    e. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create a Schema** dialog, perform the following tasks:
    a. Select **Source** as **Create an Empty Schema**.
    b. Click **Create**.

Results

You have created an empty schema.

What to do next

You can add a database connection to an empty schema. See Adding a database connection to the schema on page 472.

Adding a database connection to the schema

You can add a database connection to link a schema to a database for data generation and management.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Added a database connection in your project. See Adding a database connection on page 408.
- Created an empty schema in your project. See Creating an empty schema on page 470.

About this task

When you create an empty schema, it is not linked to any database. In the **Database View** tab, a message is displayed that indicates that the schema is empty and requires a database connection.

1. Select the **Schema** in the data navigator panel that is not connected to any database.
    Result
    The selected schema is displayed on the **Data** page.
2. Navigate to the **Database View** tab.
    If the schema is not connected to any database, a message appears `This schema is not linked to any database.`

3. Click **Add DB Connection**.
   The **Add a Database Connection** dialog is displayed.
4. Select a database from the **Select a Database Connection** list.
5. Click **Done**.

Results

You have added a database connection to the schema.

Related information

Creating a schema by using a database connection on page 473

Creating a schema by using a database connection

When your tests require data that reflects existing database models, you can create a schema by connecting to an existing database. You can retrieve and manage live data structures that help your test data reflect production schema complexity and relationships more accurately.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Read information about using the schema. See Management of schemas on page 469.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created an application in your project. See Creating an application on page 204.
- Added a database connection in your project. See Adding a database connection on page 408.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch in the project repository from the **Branch** list.

   The assets and resources in the selected branch are displayed in the data navigator panel.
5. Perform any of the following actions:

- Click **Add files** and select **Schema** on the **Test Editor** page.
- Click the **Action Menu** icon  and select **Schema**.
- Right-click the application in the data navigator panel, select **New > Schema**.

The **Edit Branch** dialog is displayed.

If the **Edit branch** dialog is not displayed, then go to step 7 on page 474.

6. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, which you selected, is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:
     a. Select **Create new edit branch**.
     b. Enter a name for the Edit branch in the **Branch Name** field.
     c. Click **Save**.

   Result

   The **Create a Schema** dialog is displayed.
7. In **Step 1** of the **Create a Schema** dialog, perform the following tasks:
   a. Select the **Application** for which you want to create a schema.
   b. Select **Location** from the drop-down list to store the schema.
   c. Type a unique **Name** for the schema.
   d. Type a brief description of the schema in the **Description** field.
   e. Click **Next** to proceed to Step 2.
8. In **Step 2** of the **Create a Schema** dialog, perform the following tasks:
   a. Select **Source** as **Select a Database Connection**.
   b. Select a database from the **Select a Database Connection** list.
   c. Click **Create**.

Results

You have created a schema by using an existing database connection.

Related information

Managing data definitions in a schema on page 475

Managing data definitions in a schema

You can manage and organize data definitions within a schema to support consistent and accurate test data generation.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Ensured that your project contains a schema.

About this task

You can access a schema in your project and perform actions such as creating or importing data definitions, linking fields, modifying generator settings, and saving schema changes. You can design data relationships by using the **Schema View**, or use the **Database View** to align data definitions with your database structure.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.
   The **Data** page and the data navigator panel are displayed.
4. Select the schema that you want to edit in the data navigator panel.

   The selected schema opens in the **Data** page.
5. You can perform the actions that are listed in the following table to modify data definitions in a **Schema View**:

Task	Action
Create an inline data definition	Perform the following steps: a. Click **Add Data Definition**. b. Type a name in the **Name** field under the **Create Inline Data Definition** section. c. Click **Add**. A table with an empty field is displayed in the schema view.
Import existing data definitions	Perform the following steps: a. Click the **Add Data Definition**. b. Select the data defintions from the **Available Data Definitions** section. c. Click **Add**.

Task	Action
	Note: You can use the **Search** field in the **Add Data Definition** dialog to find the required data definition.
Add a relationship between fields	Perform the following steps: a. Hover over the field line until the **Drag to connect to a field** tooltip appears. b. Drag to the destination field in another data definition **Note:** The data type of the source field and the destination field must be the same.
Define the relationship ratio	Perform the following steps: a. Click the **Destination** field. b. Enter the distribution ratio (For example: One-to-One or One-to-Many) in the **Relationship configuration** dialog.
Convert between Referenced and Inline DDFs	Perform the following steps: a. Click the **Actions-menu** icon on the data definition. ▪ Select **Convert to Referenced**, if the data definition is **inline** ▪ Select **Convert to Inline**, if the data definition is referenced.
Rename inline data definition	Perform the following steps: a. Click the **Actions-menu** icon on the data definition. b. Select **Rename**. c. Type a new name in the **Rename Data Definition** dialog. d. Click **Rename**.
Add new fields	Perform the following steps: a. Click the **Add field** icon. b. Type a field name in the **Name** column. An empty row is automatically inserted below the current row for your next entry.
Add generators in fields	Drag a generator from the **Catalog** panel into a field under the **Generator** column.

Task	Action
	Note: You can use the **Search Catalog** field in the **Catalog** panel to find the required generator.
Replace generators in fields	Drag a generator from the **Catalog** panel into an existing generator field under the **Generator** column. The existing generator in the field is automatically replaced by the generator.
Modify generator configurations	Perform the following steps: a. Click a generator in the **Generator** column. b. Update the configuration settings in the panel. **Note:** If you modify the configurations of a basic generator, the changes are applicable only when the generator is used as a field in a data definition. The basic generator that is available in the catalog is not modified or saved.
View sample data for a generator in Schema View	Perform the following steps: a. Click the generator label next to a field in the table in **Schema View**. The properties of the generator is displayed. b. Click the **View sample data**. A list of five sample records is displayed, fetched from the source database. c. **Optional**: Click the **Reload Sample Data** icon to refresh the data from the source.
Include additional configuration details	Perform the following steps: a. Click the **Add new configuration** tab. b. Type a name and set other settings as an additional configuration to the generator.
Reorder fields	Drag and drop rows in the table to rearrange the fields.
Delete fields	Click the **Delete** icon in the field row to remove the field.
Remove one or more data definitions from Schema View	Perform the following steps:

Task	Action
	a. Click the **Actions-menu** icon  on the data definition. b. Select **Remove from schema**. The data definition is removed from the schema. Alternatively, select the data definition and click **Remove (1)**. c. Select the data definitions or that you want to remove and click **Remove (#)**. The number in parentheses indicates the number of data definitions that are currently selected. For example, **Remove (2)** or **Remove (5)**. To remove all the data definitions from the Schema View, perform the following steps: a. Select the data definitions in the Schema View by pressing **Ctrl+A** to select all, or by pressing **Ctrl** and clicking on each DDF that you want to select. b. Click **Remove All** to remove all the data definition from the schema. The **Remove All Data Definitions** dialog box is displayed. c. Click **Yes** to confirm. **Note:** You cannot undo this action after the data definitions are removed.
Update data definitions in Schema View	Perform the following steps: a. Click the **Update** icon  to update the default row counts and default distribution ratios for the data definitions within the schema. The **Update Schema** dialog box is displayed, with the **Default row counts** and **Default distribution ratios** selected. b. Click **Update**.

6. You can perform the actions that are listed in the following table to modify data definitions in a **Database View**:

Task	Action
Export a data definition to a schema	Perform the following steps: a. Click the **Actions-menu** icon  on the data definition. b. Select **Add table to schema**.

Task	Action
	Note: The exported data definition changes to green color.
Select the linked data definitions	Perform the following steps: a. Click the **Actions-menu** icon on the data definition. b. Select **Select linked tables**. **Note:** The linked data definitions are changed to blue color.
Add missing columns	Perform the following steps: a. Select the data definition exported to schema. b. Click the **Actions-menu** icon on the data definition. c. Select **Add missing columns to schema**.
Remove unnecessary fields	Perform the following steps: a. Select the data definition exported to schema. b. Click the **Actions-menu** icon on the data definition. c. Select **Remove unnecessary fields**.

7. Click the **Save** icon to save the changes made to the schema.

Results

You have updated the data definition within a schema.

Related information

Publishing schema changes

Auto-assigning generators and AI-provided data configurations to data definitions in the Schema View

You can use AI assistance to automatically assign generators and AI-provided data configurations to the fields in data definitions within the Schema View.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.
- Ensured that your project contains a Web UI test or suite.
- Ensured that the required variables have been created and are included in your Web UI test or suite.
- Ensured that your project contains a dataset.
- Ensured that you have configured the required AI provider credentials. For more information, see Adding cloud credentials of AI providers. on page 596

About this task

When a schema contains multiple data definitions and fields, manually assigning generators and configuring them can be time-consuming. AI-based inference analyzes the available field information and recommends suitable generators and data configurations for the selected data definitions.

Note: Existing generators remain unchanged unless you select the option to overwrite them.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.
   The **Data** page and the data navigator panel are displayed.
4. Select the schema that you want to edit in the data navigator panel.

   The selected schema opens in the **Data** page.
5. Click the **Edit** button.
   The **Edit branch** dialog is displayed.
6. Select your action to either select an existing Edit branch or create an Edit branch:
   - Perform the following steps to select an existing Edit branch:
     a. Select **Use an existing edit branch**.
     b. Select an Edit branch from the **Select branch** list if Edit branches already exist for your repository.

        Note: A warning message is displayed if the Edit branch, you selected is behind the source branch. To update the selected edit branch and maintain synchronization with the source branch, you can select the **Update branch option when switching** checkbox.

     c. Click **Ok**.
   - Perform the following steps to create the Edit branch:

a. Select **Create new edit branch**.
b. Enter a name for the Edit branch in the **Branch Name** field.
c. Click **Save**.

7. Click the **Update Schema** icon .
8. Select the **Assign generators to fields** checkbox.

   The **AI Provider** option is displayed.

9. Select the configured **AI Provider** from the drop-down list.

   Note: When you select an **AI provider**, sample data from the selected tables is shared with the AI service to configure generators.

10. **Optional:** Select the **Overwrite existing generators** checkbox to overwrite the existing generators.
11. Click **Update**.

    Note: Verify the AI-provided generator configurations to ensure their accuracy, as AI-produced data might contain errors.

Results

You have automatically assigned generators and AI-provided data configurations to the fields in the selected data definitions.

Deleting a schema

You can delete a schema when it is outdated or no longer relevant to your project. Additionally, if a schema contains errors or inaccuracies that might lead to the generation of misleading or incorrect fabricated data, then you might want to remove them. By removing such shemas, you can maintain a clean and organized workspace, optimize the performance of HCL DevOps Test Hub (Test Hub), and reduce unnecessary storage usage.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220
- Ensured that your project contains a schema.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.

3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.

4. Select the schema that you want to delete from the data navigator panel.

   The selected schema opens on the **Data** page.

5. Click the **Delete** icon .

   Alternatively, right-click the schema in the data navigator panel, and then click **Delete**.

   The **Delete Schema** dialog is displayed.

6. Click **Delete**.

Results

You have deleted a schema from your project repository.

Publishing changes in data definitions, datasets, or schemas

After creating or modifying a data definition, dataset, or schema in HCL DevOps Test Hub (Test Hub), you can publish changes from an Edit branch to the source branch in the project repository. After you publish the changes, other members of the project can view and work with the updated datasets.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created or modified a schema, dataset, or data definition.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.

3. Go to **Author > Data**.
   Result
   The **Data** page and data navigator panel are displayed.
4. Select the data asset (schema, data definition, or dataset) you want to publish from the data navigator panel.
   Result
   The selected data asset is displayed on the **Data** page.
5. Click the **Save** icon to save changes.
6. Click the **Publish** icon .
   The **Publish** dialog is displayed.
7. Type the description in the **Description of change** field.

8. Click **Publish**.
   Alternatively, you can publish the changes from the **Changes** tab.
   If the publish succeeds, the Edit branch is automatically deleted and removed from the **Branch** list and the changes are merged into the previously selected remote branch.
   If the publish fails due to a conflict with the remote branch, click **Push** in the **Publish** dialog to push the edit branch to the remote repository. Then, you can create a pull request and resolve the conflicts manually in Git before you merge the changes into the target branch.

Results

You have published the changes in an Edit branch to the remote branch in the project repository.

Related information

Publishing all changes in an Edit branch on page 493

Generating fabricated test data

You can generate fabricated test data in HCL DevOps Test Hub (Test Hub) to support a wide range of test scenarios. The generated data is saved as a CSV file on your system and can be used to override existing datasets during test execution.

Data fabrication is the process of creating synthetic test data that mimics real-world information without exposing actual user or production data. You can simulate various testing scenarios using structured, configurable, and reusable datasets. In HCL DevOps Test Hub (Test Hub), this process involves defining data structures, applying data generation logic through built-in or custom generators, and producing test-ready datasets that enhance both test coverage and data privacy.

By combining datasets, data definitions, database query and schemas, Test Hub helps you to create realistic, privacy-compliant, and reusable test datasets that improve test coverage and accuracy.

You can generate fabricated test data in the following ways:

- Generating test data by using data definitions and exporting it to a `.zip` file.
- Generation of test data using Data definition history.
- Generation of schema based test data and exporting it as a `.zip` file.
- Generating test data by using data definitions or schema and exporting it to a database.

Generating test data by using data definitions and exporting it to a `.zip` file

You can create a data definition, add fields, and configure generators to produce realistic test data. Before generating, you can preview the data to ensure it meets your expectations. If the fields have no generators, you can still save the definition, but preview and generation are only available after at least one generator is configured.

Note: While configuring the generator parameters for a field in a Data definition, you must use the **Allowed** option of the **Duplicates** field of the generator with caution, particularly when you configure the data generator with a narrow

criteria for data generation. If you enable the **Allowed** option in scenarios where duplications are not feasible within the defined criteria, then the preview or generation of test data might not be successful.

Also, the data distribution in a generated test data closely aligns with the configured weightage that you provide in the **Distribution** field of the generator. Therefore, the accuracy of the test data improves as more number of records are generated. See Basic generators on page 342.

See Generating standard test data by using data definitions and exporting it to a .zip file on page 484.

Generation of test data using Data definition history

You can use the **History** tab in HCL DevOps Test Hub (Test Hub) to view all previously generated test data for a Data definition. From this tab, you can select and regenerate any earlier version of test data without reconfiguring the Data definition. This helps ensure consistency during repeated test runs, simplifies troubleshooting, and supports better collaboration when multiple team members need access to the same test dataset.

See Generating test data by using data definition history on page 488.

Generation of schema based test data and exporting it as a `.zip` file

You can generate test data based on a schema in HCL DevOps Test Hub (Test Hub) and export it as a `.zip` file containing CSV datasets. This allows you to use the fabricated data outside Test Hub or share it across different environments.

See Generating and exporting schema-based test data to a .zip file on page 490.

Generating test data by using data definitions or schema and exporting it to a database

You can generate test data from a schema in HCL DevOps Test Hub (Test Hub) and export it directly to a connected database. This enables seamless population of test environments with realistic, structured data for end-to-end testing.

See Generating test data by using data definitions or schema and exporting it to a database on page 491.

Generating standard test data by using data definitions and exporting it to a `.zip` file

You can generate synthetic standard test data by using data definitions in HCL DevOps Test Hub (Test Hub) to simulate realistic test scenarios, preview data, and export it for use in your testing workflows.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a data definition. See Creating an empty data definition on page 450.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

The **Overview** page of the project is displayed.

3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.

4. Select the branch from the **Branch** list.
   The navigation panel displays all the existing projects in the selected branch.
5. Find the data definition that you want to view, and then click it.

   Note: You can either enter the name of the data definition in the **Search** field or expand the project and manually search in the Data navigator panel.

   Result
   The selected data definition opens in the **Data** page.
6. Click the **Preview** icon to preview the test data that you want to generate.

   The test data is generated in the preview dialog.

   Note: You can preview a test data to ensure whether the data is in the expected format. You can have a glimpse of the generated test data that can help you to identify any potential issues, inconsistencies, or errors in the Data definitions. If you modify the data definition but not save it, you can still preview the corresponding test data. However, you can generate the test data only after you save the data definition.

7. Click the **Standard** tab in the preview section to view the generated standard test data.
8. Click the **Generate** icon either from the **Data** page or from the preview page.
   The **Generate data** dialog is displayed.

   Note: If you are in an Edit branch, then you can access the **Generate** icon from the preview dialog only if the data definition is saved.

9. Perform the following actions in the **Generate data** dialog:
   a. Enter a name in the **Name** field to identify the test data when generated.
   b. Enter the number of rows that you want to generate in the **Number of Rows** field.
   c. **Optional**: Enter the seed value in the **Seed** field.

      Whenever you generate test data to perform application testing, the generated data is different. You can produce the same set of data multiple times by setting the seed value.

      For example, assume that you have used the seed value as `1` to generate test data from a data definition. When you want to create another test data with the same set of values, then you can provide the seed value as `1` during the creation of the test data.

d. Clear the **Include headers** option if you do not want to include headers in the generated test data.
e. Click the **Download** option to initiate the generation of the test data.

Results

The generated test data is downloaded into your local file system as a Comma Separated Values (CSV) file.

Note: You can use other applications that support viewing or editing of the CSV files. You must ensure to apply the relevant formats to the fields to view the correct format of the generated test data.

What to do next

You can view the information about the history of the generated data in the **History** tab of the **Data** page.

Generating negative test data for a data definition

You can generate synthetic negative test data for a data definition in HCL DevOps Test Hub (Test Hub) to simulate invalid and boundary test scenarios for application forms.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a data definition. See Creating an empty data definition on page 450.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch from the **Branch** list.
   The navigation panel displays all the existing projects in the selected branch.
5. Find the data definition that you want to view, and then click it.

   Note: You can either enter the name of the data definition in the **Search** field or expand the project and manually search in the Data navigator panel.

   Result

   The selected data definition opens in the **Data** page.
6. Click the **Preview** icon to preview the test data that you want to generate.

The test data is generated in the preview dialog.

Note: You can preview test data to ensure that the data is in the expected format. You can have a glimpse of the generated test data that can help you to identify any potential issues, inconsistencies, or errors in the Data definitions. If you modify the data definition and do not save it, you can still preview the corresponding test data. However, you can generate the test data only after you save the data definition.

7. Click the **Negative** tab in the preview section to view the generated negative test data.
8. Click the **Generate** icon either from the **Data** page or from the preview page.
   The **Generate data** dialog is displayed.

   Note: If you are in an Edit branch, then you can access the **Generate** icon from the preview dialog only if the data definition is saved.

9. Perform the following actions in the **Generate Data** dialog:
   a. Select **Download CSV file** in the **Select a destination** field.

      Note: The **Export to database** option is not applicable for negative data generation.

   b. Enter a name in the **Name** field to identify the test data when generated.
   c. Select **Negative** in the **Generation Type** list to generate negative data.
   d. In the **Fields with negative/boundary content** field, select the field for which you want to generate the negative data.

      If multiple fields are available, you can generate negative data for all fields or select specific fields.

   e. **Optional**: Enter the seed value in the **Seed** field.

      Whenever you generate test data to perform application testing, the generated data is different. You can produce the same set of data multiple times by setting the seed value.

      For example, assume that you have used the seed value as `1` to generate test data from a data definition. When you want to create another test data with the same set of values, then you can provide the seed value as `1` during the creation of the test data.

   f. **Optional**: Select the following options as required:
      - **Include Headers** - Includes field name as column headers in the generated output.
      - **One per Row** - Generates one invalid value per row while keeping the remaining field values as standard data.
      - **Include Metadata** - Includes metadata information for each row in the generated output, such as the field containing the negative data and the property used to generate the negative data.

   g. Click the **Download** option to initiate the generation of the negative test data.

Results

The generated negative test data is downloaded into your local file system as a Comma-Separated Values (CSV) file.

Note: You can use other applications that support viewing or editing of the CSV files. You must ensure to apply the relevant formats to the fields to view the correct format of the generated test data.

What to do next

You can view the information about the history of the generated data in the **History** tab of the **Data** page.

Generating test data by using data definition history

You can generate fabricated test data in HCL DevOps Test Hub (Test Hub) by using the history of existing data definitions. This approach supports better collaboration by allowing multiple team members to work with consistent test data at different stages of the project.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a data definition. See Creating an empty data definition on page 450.
- Generated test data by using a data definition. See Generating standard test data by using data definitions and exporting it to a .zip file on page 484.

Note: If a project member has generated a test data from a data definition, then the other project members can view the generated test data only when the changes are pushed to the repository.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch from the **Branch** list.
   The navigation panel displays all the existing projects in the selected branch.
5. Find the data definition that you want to use for regenerating test data, and then click it.

Note: You can either enter the name of the data definition in the **Search** field or expand the project and manually search in the data navigator panel.

Result

The selected data definition opens in the **Data** page.

6. Click the **History** tab.

   A list of all the previously generated test data are displayed, along with the following information:
   - The month, date, and year of the generated test data.
   - The status of the download action that indicates whether the data generation was successful or failed.
   - The number of rows in the generated test data.
   - The initials of the project member who downloaded the generated test data.
7. Perform the following actions in the **History** tab:
   a. Find the required test data.

   b. **Optional**: Click the **Preview** icon to preview the test data before regeneration.

      The test data is generated in the preview dialog.

      Note: You can preview the previosuly generated test data to ensure whether the data is in the expected format. You can have a glimpse of the regenerated test data that can help you to identify any potential issues, inconsistencies, or errors in the data definitions.

   c. Click the **Generate** icon .

      The **Generate data** dialog is displayed.

      Note: If you are in an Edit branch, then you can access the **Generate** icon from the preview dialog only if the data definition is saved.

8. Perform the following actions in the **Generate data** dialog:
   a. Enter a name in the **Name** field to identify the test data when generated.
   b. Enter the number of rows that you want to generate in the **Number of Rows** field.

   c. **Optional**: Enter the seed value in the **Seed** field.

      Whenever you generate test data to perform application testing, the generated data is different. You can produce the same set of data multiple times by setting the seed value.

      For example, assume that you have used the seed value as `1` to generate test data from a data definition. When you want to create another test data with the same set of values, then you can provide the seed value as `1` during the creation of the test data.

d. Clear the **Include headers** option if you do not want to include headers in the generated test data.
e. Click the **Download** option to initiate the generation of the test data.

Results

The generated test data is downloaded into your local file system as a Comma Separated Values (CSV) file.

Note: You can use other applications that support viewing or editing of the CSV files. You must ensure to apply the relevant formats to the fields to view the correct format of the generated test data.

Generating and exporting schema-based test data to a `.zip` file

You can generate schema-based test data and export it as a `.zip` file that contains `.csv` files. The exported data remains consistent with the schema structure, which supports accurate testing, data validation, and you can use this data to perform data-driven testing across multiple scenarios in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Ensured that your project contains a schema.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch from the **Branch** list.
   The navigation panel displays all the existing projects in the selected branch.
5. Select the schema from which you want to generate and export test data in the data navigator panel.

   The selected schema is displayed in the **Data** page.

   Note: You can either enter the name of the Schema in the **Search** field or expand the project and manually search in the data navigator panel.

6. Click the **Preview** icon to preview the test data that you want to generate.

   The test data is generated in the preview dialog.

Note: You can preview test data to ensure whether the data is in the expected format. You can review the generated test data that can help you to identify any potential issues, inconsistencies, or errors in the schema. If you modify the schema and do not save it, you can still preview the corresponding test data. However, you can generate the test data only after you save the schema.

7. Click the **Generate** icon from the **Data** page.
   The **Generate data** dialog is displayed.
8. Select **Download as a zip file** from the **Select destination** list.
9. Type the file name in **File Name** field.
10. **Optional**: Enter the seed value in the **Seed** field.
    For example, consider that you have used the seed value of `1` to generate data from a schema. When you want to create another data with the same set of values, then you can provide the seed value as `1` during the creation of the test data.
11. Enter the number of rows that you want to generate in the **Generate** column.
12. **Optional**: Enter a value in the **Target** field to customize the number of rows to be generated in the tables.
    The default value in the **Target** field is set to 100%.

    For example, if a table in the database contains 10 rows and if you set the **Target** field to 200%, then the generated data displays 20 rows.
13. Click **Download** to initiate the generation of the data.
    Result
    The generated data is downloaded into your local file system as a `.zip` file.

Results

You have generated realistic and schema-aligned test data and exported it as a `.zip` file that contains `.csv` files.

What to do next

You can create a dataset by using the downloaded `.csv` files. See Creating a dataset by using a CSV file on page 427.

Generating test data by using data definitions or schema and exporting it to a database

You can generate test data by using data defintions or and schema, and export it directly to a connected database in HCL DevOps Test Hub (Test Hub). This process supports populating or updating database tables with realistic and schema-aligned test data used in development, testing, or validation workflows.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Ensured that your project contains data definition or schema.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Author > Data**.

   The **Data** page and the data navigator panel are displayed.
4. Select the branch from the **Branch** list.
   The navigation panel displays all the existing projects in the selected branch.
5. Select the data definition or schema from which you want to generate and export test data in the data navigator panel.

   The selected data definition or schema is displayed in the **Data** page.

   Note: You can either enter the name of the data definition or schema in the **Search** field or expand the project and manually search in the data navigator panel.

6. Click the **Preview** icon to preview the test data that you want to generate.

   The test data is generated in the preview dialog.

   Note: You can preview test data to ensure whether the data is in the expected format. If you modify the data definition or schema and do not save it, you can still preview the corresponding test data. However, you can generate the test data only after you save.

7. Click the **Generate** icon on the **Data** page.
   The **Generate data** dialog is displayed.
8. Perform the following steps to generate test data by using data definitions:
   a. Select the appropriate database from the **Select a Database Connection** list.

   b. Select a table from the **Table** dropdown.

      The selected table name appears in the **Table** field.

      If the selected table structure does not match the data definition for data generation, a **Mismatch** indicator is displayed next to the table name, and the **Export** button is disabled.

      The **Current Contents** field displays the existing number of rows in the selected table.

   c. Enter the number of rows you want to generate in the **Number of Rows** field.
9. Perform the following steps to generate schema based test data:
   a. Select the appropriate database from the **Select a Database Connection** list.

      The **Tables** column lists the available tables from the schema.

The **Current** column displays the number of existing rows in each table.

b. Enter the number of rows to generate for each table in the **Generate** column.

If there is a mismatch between the selected table and the schema used for data generation, a **Schema mismatch** indicator is dispalyed, and you cannot generate data for that table.

c. Enter a value in the **Target** field to customize the number of rows to be generated in the tables.

The default value in the Target field is set to 100%.

For example, if a table in the database contains 10 rows and if you set the Target field to 200%, then the generated data displays 20 rows.

10. **Optional**: Enter the seed value in the **Seed** field.
Whenever you generate data to perform application testing, the generated data is different. You can produce the same set of data multiple times by setting the seed value.
For example, consider that you have used the seed value of `1` to generate data from a data defintion or schema. When you want to create another data with the same set of values, then you can provide the seed value as `1` during the creation of the test data.
11. **Optional**: Select the **Overwrite** checkbox to overwrite the rows in the tables for generating data.
This option is not selected initially.
For example, if a table in the database contains 10 rows and if you select the **Overwrite** checkbox, then the generated data replaces the existing 10 rows with new data generated.
12. Click the **Export** option to initiate the generation of the test data.
The generated data is exported to the database.

Results

You have generated the test data by using data definition or schema and exported it to directly to the connected database.

What to do next

You can create a dataset by using the database query. See Creating a dataset by using a database query on page 431.

Publishing all changes in an Edit branch

After you create or modify any data or test assets in an Edit branch of a project in HCL DevOps Test Hub (Test Hub), you can publish all the changes that are in an Edit branch to the source branch in the project repository. After publishing, other members in the project can view and use the resources.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created or modified a test or data asset.
- Read and understood the information in Edit branch overview on page 198 and Create or select an Edit branch on page 212.

About this task

When you create or modify multiple assets in a project, the **Changes** page lists all the changed assets in the selected branch. In this scenario, you cannot publish or discard a single change. You must either publish or discard all changes that are made to all the listed assets.

Note: When you rename a folder in the test navigator panel, the **Changes** page displays separate entries for asset creation and asset deletion. However, when you click **Details** on the **Changes** page, a single entry for the rename is displayed. This is a known behavior in the current release.

When you review assets on the **Changes** page before publishing, each executable asset displays a colored status indicator in the parent row as follows:

- **Green** – The most recent run passed.
- **Red** – The most recent run failed.
- **Orange** – The most recent run resulted in an error.
- **Grey** – No run result is available for this revision.

The indicator helps you view the most recent run result for that specific asset revision. The colored bar on the left side of the parent row represents the most recent run result.

The **Changes** page also provides you with information about the number of commits that the selected branch is ahead and behind. After you publish an asset to the Git repository, all the other project members who have access to that asset can view the updated asset. You can perform the following actions in the **Changes** page:

Options	Actions
Edit	Click **Edit** to make any updates to the asset before publishing.
Discard	Click **Discard** to remove the changes that you made to the asset.
Save as	Click **Save as** to save a copy of the asset. **Note:** When you try to publish a modified asset with conflicting edits, an error about conflicting changes is displayed. However, you can use the **Save as** option to save a copy of the asset edits under a new asset name. You must discard the edits that you made to the original asset.

Options	Actions
	For example, consider a scenario when Amy Smith and John Doe edit the same Data definition, and Amy Smith edits and publishes the Data definition. On the **Changes** page, a message that indicates the number of commits ahead and behind in the branch is displayed when John Doe tries to publish the same Data definition. You can then use the **Save as** option to resolve this issue.
Compare	Click **Compare** to view the differences between the asset on the edit branch and the asset on the main branch.
Update	Click **Update** to merge the changes from remote branch that are not in your Edit branch.
Details	Click **Details** to see the changes.
Reset	Click **Reset** to discard the changes
Push	Click **Push** to upload the edit branch to remote repository when a conflict prevents publishing.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Go to **Author > Changes**.
4. Select the Edit branch from the **Branch** list.
   All the updated assets that are present in the selected branch are displayed in the **Changes** page.
5. Click **Publish**.
   The **Publish** dialog is displayed.
6. Enter a description for the changes that you performed to the asset, and then click **Publish**.
   If the publish succeeds, the Edit branch is automatically deleted and removed from the **Branch** list and the changes are merged into the previously selected remote branch.
   If the publish fails due to a conflict with the remote branch, click **Push** in the **Publish** dialog to push the edit branch to the remote repository. Then, you can create a pull request and resolve the conflicts manually in Git before you merge the changes into the target branch.

Results

You have published all the changes in an Edit branch to the remote branch in the project repository.

Chapter 6. Test Execution Specialist Guide

This guide describes tasks that you must complete before you can configure and run tests in HCL DevOps Test Hub (Test Hub). You can find information about configuring runs for the different test types that are supported. You can also find information about other tasks that you can perform on the Resource Monitoring Service, Virtualization, and Integrations with third-party applications. This guide is intended for testers or test execution specialists.

System modeling

You can use the system model feature on HCL DevOps Test Hub (Test Hub) to create a model that simulates an application or system under test. You can define components to represent the actual components in the system under test, define their relationships and then view the components in a system under test as a diagrammatic representation.

Introduction to a system model

A system model is a logical presentation of the components contained in a system under test and the relationships between components. You can create a system model when you want to visualize the different components and their relationships as a diagrammatic representation.

You can create only one system model within a team space on Test Hub. A team space repository is required to contain the system model that you create. Before you can create a system model, you must add a team space repository. The system model that is created or edited is stored in the team space repository. All projects within a team space share the same system model. All members of a team space can create, edit, and view the system model.

After you log in to Test Hub, you can go to the **System Model** page to create a system model for your team space. Alternatively, you can go to any of the projects in your team space and create a system model from the **System Model** page within a project. The system model is accessible from all the projects in your team space.

Components in a system model

Components are the basic building blocks that are used to represent a piece of software in an application.

You can model a system by using components. You can create components that represent the type of asset or resource used by the system or application under test and can be of the following types:

Component type	Description
Database	The component type represents the database resource. You can select this component type when you want to represent a database asset or resource in the system model.
Service	The component type represents the virtual service resource. You can select this component type in the system model to represent a virtual service in the team space repository or associate a virtual service resource that is in the project repository.
UI	The component type represents the user interface (UI) resource. You can select this component type when you want to represent the UI resource in the system model.

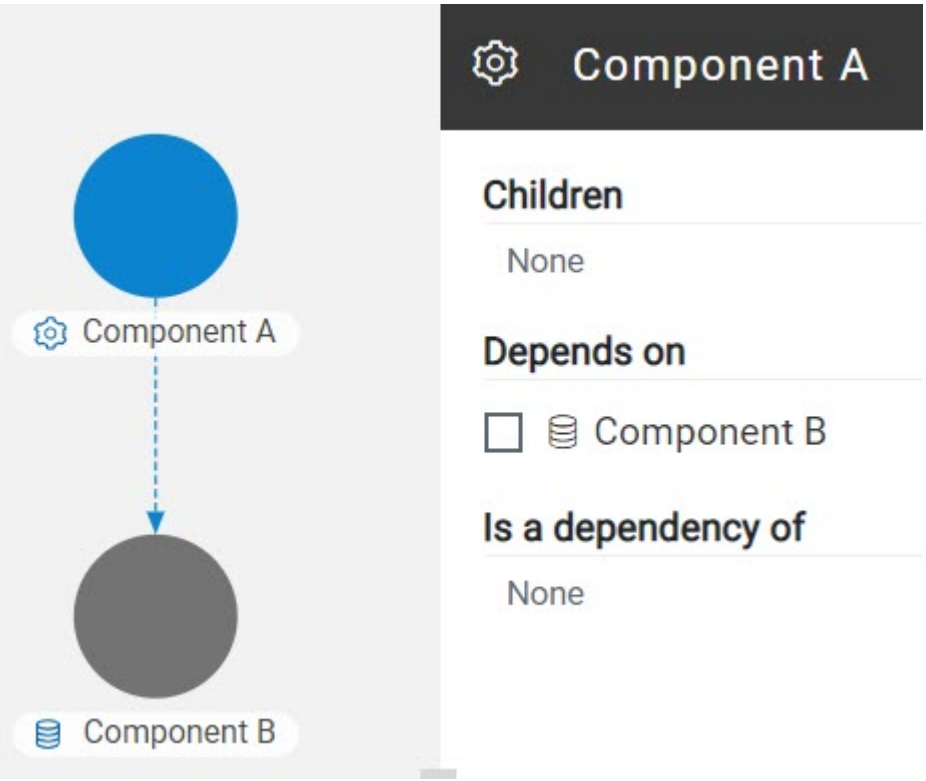
Component relationships

You can create multiple components to depict the different resources in the system or application under test. You can create a component as a *child* component of a *parent* component. You can also create multiple components as children components of the same parent component.

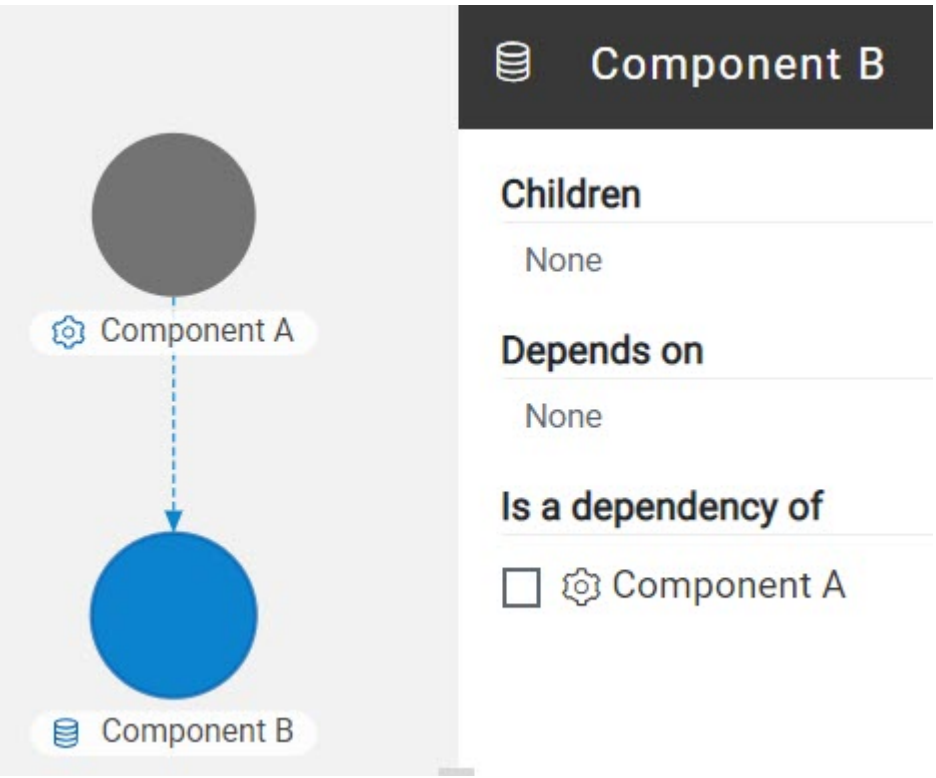
You can define the relationship between components if one component depends on another or if a component has a dependency of another component. The **Depends on** or **Is a dependency of** relationship depends on the component that you select in the system model after you define the relation.

For example, if you define *Component A* to depends on *Component B*, then the relationship is displayed with a line arrow from *Component A* to *Component B*, and the arrow points to *Component B*.

When you click the *Component A* in the system model page, you can view that the *Component B* is listed under the **Depends on** section for *Component A* in the right pane.



When you click the *Component B* in the system model, you can view that the *Component A* is listed under the **Is a dependency of** section for *Component B* in the right pane.



Component associations

When you work with the system model from your project in a team space, you can associate test resources that are in your project repository with components in the system model. You can associate all types of test resources such as Suites or tests, and virtual service resources with the components.

After associating test resources, you can view them when you select the component with which they are associated. You can select the associated test resources to start a run of the Suites or tests or start an instance of the virtual services resource.

Using the system model UI

The following table lists the icons used in a system model and the icons that are displayed on the **System Model** page:

Icon	Description
	Represents a system model.
	Represents Suites and tests in a repository.
	Represents a database resource. The component name is prefixed with this icon when you select the component type as database.
	Represents a service resource. The component name is prefixed with this icon when you select the component type as service.

Icon	Description
	Represents a user interface (UI) resource. The component name is prefixed with this icon when you select the component type as UI.
	Represents the virtual services.
	Represents the dependency between the components.
	Represents the inherited dependencies.

The following table lists the icon that you can use to perform different actions in the system model:

Icon	Used for...
	Adding components, child components, or dependent components.
	Searching for components in the system model.
	Publishing the changes made to the system model, changes made to components, or changes made to component associations. The changes are published to the team space repository.

You can perform the following operations when you view the model that is displayed on the **System Model** page:

Operation	Action
Pan the view of the model	1. Click the area of the model that you want to pan and move the area by keeping the left mouse button pressed. 2. Release the left mouse button to stay on the view that you panned.
Zoom in the view of the model	1. Click the **Zoom-in** icon  . The view is enlarged or zoomed in. **Note:** You can repeatedly click the **Zoom-in** icon to further enlarge the view. Alternatively, you can use the scroll button on the mouse or device to zoom in the view.

Operation	Action
Zoom out the view of the model	1. Click the **Zoom-out** icon  . The view is diminished or zoomed out. **Note:** You can repeatedly click the **Zoom-out** icon to further diminish the view. Alternatively, you can use the scroll button on the mouse or device to zoom out the view.
Reset the view	1. Click the **Menu** icon on the **System Model** page. 2. Select **Reset Zoom**. The view is reset to the default view without any zoom applied. If the view was zoomed in or zoomed out, the view is reset to 100% of the size.
Position the model view	1. Click the **Menu** icon on the **System Model** page. 2. Select from the following options: ◦ **Center**: Select this option to align the view of the model to the center of the pane. ◦ **Fit to View**: Select this option to fit the view of the model in the pane on the **System Model** page.

Tasks in system modeling

When you want to model the system under test for the first time after you install the server software, you must first add the team space repository. See Adding a repository to a team space on page 502.

After a repository is configured as the team space repository, you can work with the system model. See Tasks for working with a system model on page 501.

When you want to change, update or delete the repository, you can work with the team space repository. See Tasks for working with the team space repository on page 501.

Task flows for working with a system model

You can find information about the tasks that you can perform when you want to create a system model, components, associate the components, create linkages among the components, and then publish the system model to the team space

repository. You can also find information about the tasks that you must perform when you want to modify an existing system model.

Tasks for working with a system model

You can find the tasks when you want to create a system model for the application under test in your team space.

Tasks	Go to...
Add a repository to the team space.	Adding a repository to a team space on page 502
Create a system model.	Creating a system model on page 505
Create components in a system model.	Creating components on page 506
Create child components to existing components in a system model.	Creating child components on page 507
Define the relationship between the components in a system model.	Creating linkages between components on page 510
Delete the relationship between the components in a system model.	Deleting linkages between components on page 511
Modify components in a system model.	Modifying components on page 513
Delete components in a system model.	Deleting components on page 515
Publish changes made to the system model.	Publishing changes to the system model on page 516
Associate resources with components in a system model.	Associating resources with components on page 517
View resources associated with components in a system model.	Viewing resources that are associated with components on page 519
Remove resources that are associated with components in a system model.	Removing resources that are associated with components on page 522
Publish changes of associating or dissociating resources in a project with components in the system model.	Publishing changes of associating or dissociating resources with components on page 524
View system model.	Viewing the system model on page 526
Delete system model.	Deleting a system model on page 526

Tasks for working with the team space repository

You can find the tasks that you can perform when you want to work with the team space repository.

Tasks	More information
Add a repository to the team space.	Adding a repository to a team space on page 502
Delete a team space repository.	Deleting a repository from a team space on page 527

Adding a repository to a team space

You must add a repository to the team space in HCL DevOps Test Hub (Test Hub) when you want to create, modify, or store a system model.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* or an *Architect* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you obtained valid credentials to access the Git repository that you want to add.

About this task

As a *Team Space Owner* or an *Architect*, you can add a repository to the team space. You can add a repository to the team space in any of the following methods:

- Add an existing repository that can contain test assets, which are created in the desktop products.
- Add a bare repository.

Attention: The repository that you want to add must not contain any projects or `.project` files under the root directory.

You can add a Git repository using SSH. When you add a repository, the Git repository is cloned to the team space. While you add a repository, you must provide the necessary authentication credentials that are set for the Git repository. If the authentication type is SSH, then you must provide the Git URL, a deploy key, and a passphrase.

For example:

- `git@github.com:account/testRepo.git`
- `ssh://git@10.111.111.333:2222/~/account/project.git`

Important: You can add only one repository to the team space. After you add a repository to the team space, the option to add another repository is not available. You must delete an existing repository before you can add a different repository.

1. Log in to Test Hub.
   Result

The **Projects** page of the initial team space is displayed.

2. Click the **Switch Team Space** icon from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **Manage > Configuration**.

   Result

   The **Configuration** page of the team space is displayed
4. Click **Add Repository**.

   Result

   The **Add Repository** dialog is displayed.
5. Perform the action based on the selected repository type:

   The following table describes the actions required for each repository type:

Repository type	Action
Remote Repository	a. Enter the URL of the Git repository that you want to add to the team space in the **Git URL** field. The required fields are displayed based on the type of Git URL that you entered. b. Enter the required credentials based on any of the following authentication methods configured in the repository. To gain access to the repository, you must use any one of the following authentication methods: ▪ SSH: Deploy key and passphrase ▪ HTTPS: Username and password ▪ HTTP: Username and password **Notes:** ▪ You must have defined the authentication type and set the authentication credentials in the Git repository. ▪ If you use SSH to connect to your remote repository and Test Hub displays an `Auth Fail` exception while using the deliver changes option, you can resolve this exception error by regenerating your SSH keys by using the -m PEM option.
Local Repository	Enter a unique repository name in the **Repository Name** field. Optional: Modify the default branch name in the **Default Branch Name** field. By default, the branch name is `main`.

Repository type	Action
	Notes: ◦ You cannot change the Local repository names after creation. ◦ You can create and edit branches, create and edit assets, and publish changes locally in a local repository. ◦ You can later push a local repository to a supported remote repository provider. ◦ Ensure that the branch you are pushing from your local repository does not already exist in the remote repository. This helps avoid merge conflicts. ◦ Once a local repository is pushed to a remote repository, it is converted into a remote repository and behaves like any other remote repository in Test Hub.

6. Click **Add**.

Results

The Git repository is added to the team space on Test Hub.

Note: The time to clone the repository can take a few to several minutes and depends on the size of the repository.

What to do next

You can perform the following actions on the repository that you added:

- Update the authentication credentials if they are changed in the Git repository configuration. See Updating the authentication credentials of the repository on page 1230.
- Refresh a repository to fetch and synchronize changes from the remote repository. See Refreshing repositories manually on page 1231.
- Configure a webhook to notify the server if there is a push event in the remote repository. See Creating webhooks on page 1234.
- Add a system model to the repository. See Creating a system model on page 505.
- Delete a repository that is added to the team space, if it is no longer required. See Deleting a repository from a team space on page 527.

Related information

Managing repositories on page 1226

Creating a system model

Before you can create components in a system model in HCL DevOps Test Hub (Test Hub), you must create an empty system model and publish the system model to the team space repository.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as an *Architect* in the team space, a *Project Owner* or a *Tester* in the project. See Managing members and their roles in a team space on page 1207 or Managing access to the server projects on page 1220.
- Added a repository to the team space. See Adding a repository to a team space on page 502.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon from the left navigation pane. Then, use the **Switch to** option to open the team space for which you want to create a system model.
3. Select an option to create a system model based on your role in the team space or in a project in the team space:
   - If you are a *Team space owner* or an *Architect*, go to Step 4 on page 505.
   - If you are not a *Team space owner* or an *Architect* but a project *Owner* or *Tester*, go to Step 5 on page 505.
4. Perform the following steps, if you are a *Team space owner* or an *Architect*:
   a. Click **System Model** in the left navigation pane.

      The **System Model** page is displayed.

   b. Click the **Add system model** icon .

      An empty system model is created.

   c. Go to Step 6 on page 506.
5. Perform the following steps, if you are not a *Team space owner* or an *Architect* but a *Project owner*:
   a. Open your project that is displayed under **My projects** tab.

      The **Overview** page of the project is displayed.

   b. Click **System Model** in the left navigation pane.

      The **System Model** page is displayed.

c. Click the **Add system model** icon  .

An empty system model is created.

d. Go to Step 6 on page 506.

6. Click the  icon, and then click the **Publish** option.

Note: If you do not want to commit the changes that you made to the system model, you can select the **Discard** option.

7. Enter a commit message in the **Description of change** field, and then click **Publish**.

The empty system model is committed and published to the team space repository. When the system model is published to the team space repository, other members of the team space can work with the same system model from any of their projects in the team space.

Results

You have created an empty system model and published it to the team space repository.

What to do next

You can perform any of the following tasks:

- Create components in the system model. See Creating components on page 506.
- Create child components for the components that you added to the system model. See Creating child components on page 507.

Creating components

After you create an empty system model in the team space in HCL DevOps Test Hub (Test Hub), you can create components that represent components of the system under test.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as an *Architect* in the team space, a *Project Owner* or a *Tester* in the project. See Managing members and their roles in a team space on page 1207 or Managing access to the server projects on page 1220.
- Added a repository to the team space. See Adding a repository to a team space on page 502.
- Created and published the system model. See Creating a system model on page 505.

1. Log in to Test Hub.
The **Projects** page of the initial team space is displayed.

2. Click the **Switch Team Space** icon from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.

4. Perform the following steps to add a component to the system model:

   a. Click the **Add component** icon in the **Components** row on the right pane.

      The **Add a component** dialog is displayed.

   b. Enter a name for the component in the **Name** field.

   c. Select the type of component that you want to add from the following options displayed in the **Type** field.

      - **Service**
      - **Database**
      - **UI**

   d. Click **Add**.

      The component is displayed as a circular block on the system model pane with the icon and name of the component below the block. The added component is also displayed under the **Components** list on the right pane.

      You can create components of other types or the same type in the system model by following the preceding steps.

Results

You have created components in the system model.

What to do next

You can perform any of the following tasks:

- Publish the changes that you made to the system model. See Publishing changes to the system model on page 516.
- Create child components for the components that you added to the system model. See Creating child components on page 507.
- Create linkages between components in the system model. See Creating linkages between components on page 510.

Creating child components

You can create child components in the system model in HCL DevOps Test Hub (Test Hub) when you want to represent the sub-components of a component in a system under test.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as an *Architect* in the team space, a *Project Owner* or a *Tester* in the project. See Managing members and their roles in a team space on page 1207 or Managing access to the server projects on page 1220.
- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.

About this task

You can create multiple components in your system model and multiple child components under any component in the system model. After you create a child for a component, the child is represented as a circular block within the circular component block in the system model.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed with the existing components.
4. Create child components by using any of the following options:
   - To create child components from the left pane on the **System Model** page, go to Step 5 on page 508.
   - To create child components from the right pane on the **System Model** page, go to Step 6 on page 509.
5. Perform the following steps to create child components from the left pane:
   a. Click the component under which you want to create child components.

      The details of the component are displayed in the right pane.

   b. Click the **Add child** icon in the **Children** row on the right pane.

      The **Add a child to...** dialog is displayed.

   c. Enter a name for the component in the **Name** field.

   d. Select the type of component that you want to add from the following options displayed in the **Type** field.

      - **Service**
      - **Database**
      - **UI**

   e. Click **Add**.

The child component is displayed as a circular block within the circular block of the parent component on the system model pane. The added child component is also displayed under the **Children** list under the component on the right pane.

You can create child components of other types or the same type for the same or other components in the system model by following the preceding steps.

6. Perform the following steps to create child components from the right pane of the stem model page:
    a. Click the component name listed under the **Components** in the right pane.
    b. Click the **Add child** icon in the **Children** row on the right pane.

    The **Add a child to...** dialog is displayed.
    c. Enter a name for the component in the **Name** field.
    d. Select the type of component that you want to add from the following options displayed in the **Type** field.
        - **Service**
        - **Database**
        - **UI**
    e. Click **Add**.

    The child component is displayed as a circular block within the circular block of the parent component on the system model pane. The added child component is also displayed under the **Children** list under the component on the right pane.

    You can create child components of other types or the same type for the same or other components in the system model by following the preceding steps.

Results

A component is created and it is associated as a child with the selected component and the name of the child is displayed under the **Children** section on the right pane.

What to do next

You can perform any of the following tasks:

- Publish the changes that you made to the system model. See Publishing changes to the system model on page 516.
- Create linkages between components in the system model. See Creating linkages between components on page 510.

Creating linkages between components

You can create linkages between components in HCL DevOps Test Hub (Test Hub) to display the relationship between them. The linkages can be between components or between child components of the same or different parent components.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as an *Architect* in the team space, a *Project Owner* or a *Tester* in the project. See Managing members and their roles in a team space on page 1207 or Managing access to the server projects on page 1220.
- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.
- Created child components. See Creating child components on page 507.

About this task

You can create multiple components in your system model. You can also create multiple child components under any component in the system model. You can specify the dependency among the components.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
4. Perform the following steps to create a relationship between the components in the system model:

   a. Click the component that you want.

      The details of the component are displayed on the right pane.

   b. Perform the actions in the following table to create dependencies between the components:

Option	Action
Depends on	Perform the following steps, if the component that you selected is dependent on another component: i. Click the **Add dependency** icon  in the **Depends on** row on the right pane. The **Add dependency** dialog is displayed.

Option	Action
	ii. Select the component from the list of components displayed under **Depends on**. iii. Click **Add**. The components linked are listed under the **Depends on** section on the right pane. The linkage between the first to the second component that you selected is shown with an arrow that starts from the first and ends in the second component.
Is a dependency of	Perform the following steps, if the component that you selected has a dependency on another component: i. Click the **Add dependent** icon ⊕ in the **Is a dependency of** row on the right pane. The **Add dependent** dialog is displayed. ii. Select the component from the list of components displayed under the **Is a dependency of** section. iii. Click **Add**. The components linked are listed under **Is a dependency of** section on the right pane. The linkage between the first to the second component that you selected is shown with an arrow that starts from the second and ends in the first component.

Results

You have created linkages between components in the system model.

What to do next

You can publish the changes that you made to the system model. See Publishing changes to the system model on page 516.

Deleting linkages between components

When you do not want to retain the linkages between components that you created in the system model in HCL DevOps Test Hub (Test Hub), you can delete the linkages.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as an *Architect* in the team space, a *Project Owner* or a *Tester* in the project. See Managing members and their roles in a team space on page 1207 or Managing access to the server projects on page 1220.
- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.
- Created child components. See Creating child components on page 507.
- Created linkages between components in the system model. See Creating linkages between components on page 510.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
4. Perform the following steps to delete a relationship between the components in the system model:

   a. Click the component that you want.

      The details of the component are displayed on the right pane.

   b. Select from the following types of relationships that you want to delete:

When...	Action
The component depends on another component.	Perform the following steps, if the component that you selected is dependent on another component: i. Select the component from the list of components that are displayed under the **Depends on** section on the right pane. ii. Click the **Delete selected** icon inline with the **Depends on** row. The **Delete relationship** dialog is displayed with the details of the component and the dependent component. iii. Click **Delete**. The component linkage is removed from the system model pane.
The component has a dependency on another component.	Perform the following steps, if the component that you selected has a dependency on another component: i. Select the component from the list of components that are displayed under the **Is a dependency of** section on the right pane. ii. Click the **Delete selected** icon inline with the **Is a dependency of** row.

When...	Action
	The **Delete relationship** dialog is displayed with the details of the component and the component dependency. iii. Click **Delete**. The component linkage is removed from the system model pane.

Results

You have created linkages between components in the system model.

What to do next

You can publish the changes that you made to the system model. See Publishing changes to the system model on page 516.

Modifying components

At any point in time, after you create a component in HCL DevOps Test Hub (Test Hub), you can modify and update the details of the components such as the component type, and the relationships with other components.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as an *Architect* in the team space, a *Project Owner* or a *Tester* in the project. See Managing members and their roles in a team space on page 1207 or Managing access to the server projects on page 1220.
- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.
- Created child components. See Creating child components on page 507.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
4. Click the component that you want to edit.

   The component details are displayed on the right pane.
5. Perform the actions for the task to modify components as listed in the following table:

Task	Action
Change the component type.	Perform the following steps: a. Identify the component that you want to modify from the components listed under **Components** on the right pane. b. Click the component checkbox. The **Edit selected** icon is enabled. c. Click the **Edit** icon. The **Edit component** dialog box is displayed. d. Select the component type. e. Click **Modify** The type of the component is modified.
Change the name of a component.	Perform the following steps: a. Identify the component that you want to modify from the components listed under **Components** on the right pane. b. Click the component checkbox. The **Edit selected** icon is enabled. c. Click the **Edit** icon. The **Edit component** dialog box is displayed. d. Edit the component name in the **Name** field. e. Click **Modify**. The name of the component is modified.
Change the parent of a component.	Perform the following steps: a. Identify the component that you want to modify from the components listed under **Components** on the right pane. b. Click the component checkbox. The **Change parent** icon is enabled. c. Click the **Change parent** icon. The **Change parent** dialog box is displayed.

Task	Action
	d. Select another parent for the selected components from those that are listed under the **Parent** list. e. Click **Change**. The parent of the component is changed.

Results

You have edited components and updated the details successfully.

What to do next

You can publish the changes that you made to the system model. See Publishing changes to the system model on page 516.

Deleting components

When you do not require a component in HCL DevOps Test Hub (Test Hub), you can delete the component from the system model.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as an *Architect* in the team space, a *Project Owner* or a *Tester* in the project. See Managing members and their roles in a team space on page 1207 or Managing access to the server projects on page 1220.
- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
4. Click the component that you want to delete.

   The component details are displayed on the right pane.
5. Click the **Delete component** icon  in the row of the component name.

The **Delete component** dialog is displayed.

6. Click **Delete**.

Results

You have deleted a component from the system model.

What to do next

You can perform any of the following tasks:

- Publish the changes that you made to the system model. See Publishing changes to the system model on page 516.
- Create components in the system model. See Creating components on page 506.

Publishing changes to the system model

When you create, delete, or modify components in a system model in HCL DevOps Test Hub (Test Hub), the changes are saved locally and are not visible to the other project members or team space members. You must publish the system model changes to the team space repository to make your changes visible to all users of the team space.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as an *Architect* in the team space, a *Project Owner* or a *Tester* in the project. See Managing members and their roles in a team space on page 1207 or Managing access to the server projects on page 1220.
- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.
- Created child components. See Creating child components on page 507.
- Created linkages between components in the system model. See Creating linkages between components on page 510.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
4. Click the  icon, and then click the **Publish** option.

Note: If you do not want to commit the changes that you made to the system model, you can select the **Discard** option.

5. Enter a commit message in the **Description of change** field, and then click **Publish**.

Results

You have published the changes that you made in the system model to the team space repository. Other members of the team space can view and use the system model.

Associating resources with components

After creating components in the system model in HCL DevOps Test Hub (Test Hub), you can navigate to a project in the team space to associate the Suites, tests, or virtual service resources in the project repositories with components in the system model.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
4. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
5. Click the component that you want to associate with the Suites, Tests, or virtual service resources.

   The component details are displayed in the right pane.
6. Click the **Associate** tab, and then perform the following steps:
   a. Click **Add**.
   b. Select the type of resource that you want to associate with the component from the following options:

- Suites and Tests
- Virtual services

Note: You can select either of the options or both the options.

The **Edit branch** dialog is displayed.

c. Select your action that depends on the existence of an *Edit* branch from the following options:

- Select an existing *Edit* branch that you created previously from the list of branches listed in the **Select branch** field, and then click **Save**.
- Click **Create new branch**, enter a name for the branch, and then click **Save**, if you want to create a new branch.

The **Associate Assets** dialog is displayed. The resources that are contained in the branch of the project repository are listed. The name of the component and the branch are also displayed.

The **Suites and Tests** option is selected and the assets are listed if you selected the **Suites and Tests** option.

The **Virtual services** option is selected and the virtual service resources are listed if you selected the **Virtual services** option.

If you selected both the types of resources, then assets and virtual services in the branch are listed.

d. Identify the assets or virtual resources that you want to associate with the component in the system model by performing any of the following actions:

- Search for the resource by entering the asset name in the **Search** field.
- Find the assets by scrolling through the list of assets.

e. Select the resources, and then click **Add**.

The selected resources are associated with the component.

You can view the associated resources that are displayed in the **Associate** tab and the **Overview** tab. The test assets are listed under the **Suites and Tests** section and the virtual service resources are listed under the **Virtual services** section.

If you want to associate other resources to the same component, you can add them by clicking the **Add association** icon in the resource row on the right pane. You can then select the resource and add the resource to the selected component.

Results

You have associated Suites, tests, or virtual service resources with a component in the system model.

What to do next

You can perform any of the following tasks:

- Publish the changes made in the system model for associating resources with components. See Publishing changes to the system model on page 516.

Viewing resources that are associated with components

After you associate Suites, tests, or virtual services resources with components in the system model in HCL DevOps Test Hub (Test Hub), you can view the associated resources from the **System Model** page. You can opt to run an instance of the associated resource or stop a running instance of the associated resource.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.
- Associated resources in a project with components in the system model. See Associating resources with components on page 517.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
4. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
5. Click the component that you associated with the resources in the project repository.

   The resources associated with the component are displayed under the resource type in the **Associate** tab.
6. Click the **Overview** tab, if it is not already displayed.

   The resources that are associated with the component are displayed under the **Suites and Tests** or **Virtual Services** section.
7. View and perform the actions for the resource type that you want from the following options:
   - For **Suites and Tests** resources, go to 8 on page 520.
   - For **Virtual Services** resources, go to 9 on page 520.

8. Identify the test resource from those that are listed under the **Suites and Tests** section, and perform the action that you want as described in the following table:

Task	Description	Action
Collapse icon	You can hide the resource sections by clicking the **Collapse** icon . **Note:** When the section is hidden, you can view the number that is displayed of the resources that are associated.	Click the icon to toggle the display to show or hide the resource list in the **Suites and Tests** section.
Type	You can view the type of Suite or test.	Hover over the Suites and Tests icon under the **Type** column to view the type of Suite or test.
Name	You can view the name of the associated Suite or test and the path in the repository.	There is no action to perform.
Status	You can view the details of the test runs.	Click each of the squares to view the details of the specific test run. For example, if the status of a test run is represented as shown in the image  , then it indicates that the Suite or test has run only two times. After you click any colored square, the **Results** page is displayed.
Action	You can start the execution of the Suite or test from the **Execution** page when you click the **Show in execution page** icon .	Click the **Show in execution page** icon to view the Suite or test on the **Execution** page. The Suite or test is displayed on the **Execution** page.

9. Identify the virtual service resource from those that are listed under the **Virtual Services** section, and perform the action that you want as described in the following table:

Field	Description	Action
Collapse icon	You can hide the resource sections by clicking the **Collapse** icon . **Note:** When the section is hidden, you can view the number that is displayed	Click the icon to toggle the display to show or hide the resource list in the **Virtual Services** section.

Field	Description	Action
	 of the resources that are associated.	
Show all icon 	You can opt to view the virtual services that are associated with the component either on the **Resources** or **Instances** page. You can start an instance of the virtual service from the **Resources** page while you can stop a running instance from the **Instances** page.	Perform the following steps: a. Click the **Show all** icon  . b. Select any of the following actions: ▪ To start an instance of the virtual service: i. Select **Show all in resources page** to go to the **Resources** page. ii. Click the **Execute** icon  in the row of the virtual service. iii. Configure the settings for the run of the virtual service, if you want to change any of the settings, else click **Execute.** The virtual service starts to run. ▪ To stop a running instance of the virtual service: i. Select **Show all in instances page** to go to the **Instances** page. ii. Identify the instance that you want to stop and click the **Stop** icon  in the **Actions** column of the selected virtual service instance. iii. Click **Ok** in the **Stop virtual service instance** dialog that is displayed. The running virtual service instance is stopped.
Name	You can view the name of the associated virtual service and the path in the repository.	There is no action to perform.
Instances	You can view the state of the running instance of the virtual service resource. The state is displayed as `Running` only if the instance is running.	There is no action to perform.
Action	You can go to the **Resources** page to view the virtual service resource	Click the **Show resource** icon  to view the virtual service on the **Resources** page.

Field	Description	Action
	when you click the **Show resource** icon  . You can start an instance of the virtual service from the **Resources** page.	The virtual service resource is displayed on the **Resources** page.
Expand icon	You can expand the virtual service card only if the virtual service is a running instance. In the expanded card, you can view the details such as the Environment, the total number of requests received by the virtual service, and the relative time from when the virtual service had started. You can stop a running instance of the virtual service from the **Instances** page.	Perform the following steps: a. Expand the virtual service card for a running instance. You can view the details of the instance. b. Click the **Show instance** icon  to go to the **Instances** page to view the running instance of the virtual service. The running instance of the virtual service is displayed on the **Instances** page.

Results

You have viewed the resources that are associated with a component from the **System Model** page.

Removing resources that are associated with components

When you do not want to retain the resource that you associated with a component in HCL DevOps Test Hub (Test Hub), you can delete the resource association from the system model.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.
- Associated resources in a project with components in the system model. See Associating resources with components on page 517.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
4. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
5. Click the component that you associated with the resources in the project repository.

   The resources associated with the component are displayed under the resource type in the **Associate** tab.
6. Select the resource that you want to remove from the component by clicking the checkbox of the resource.

   You can select any or all resources for removal.
7. Click the **Delete selected** icon in the row of the resource type.

   The **Disassociate...** dialog is displayed for the type of resource that you selected. For example, the **Disassociate Suites and Tests** dialog is displayed when you selected the test assets.
8. Click **Delete**.

   The **Edit branch** dialog is displayed.
9. Select your action that depends on the existence of an *Edit* branch from the following options:

   - Select an existing *Edit* branch that you created previously from the list of branches listed in the **Select branch** field, and then click **Save**.
   - Click **Create new branch**, enter a name for the branch, and then click **Save**, if you want to create a new branch.

   The dialog that is displayed depends on the type of resource you selected and can be any of the following types:
   - The **Dissociate suites and tests** dialog is displayed, if you selected Suites or tests as the resources to dissociate.
   - The **Dissociate virtual services** dialog is displayed, if you selected the virtual services as the resources to dissociate.
10. Click **Delete**.

Results

You have removed resources that were associated with components in the system model.

What to do next

You must publish the dissociation changes in the system model to the project repository if you want the changes to be reflected in the system model for all members of the project. See Publishing changes of associating or dissociating resources with components on page 524.

Publishing changes of associating or dissociating resources with components

After you associate or dissociate resources in a project with a component in the system model in HCL DevOps Test Hub (Test Hub), you must publish the changes made to the branch in the project repository so that the association changes are visible to the other members of the project.

Before you begin

You must have completed the following tasks:

- Created and published the system model. See Creating a system model on page 505.
- Created components in the system model. See Creating components on page 506.
- Associated resources in a project with components in the system model. See Associating resources with components on page 517.
- Dissociated resources from components in the system model. See Removing resources that are associated with components on page 522.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
4. Click the  icon.

   You can view the options for unpublished changes made to the components and associations in the system model. The options available for association changes are as follows:
   - See unpublished commits
   - Discard
   - Publish
5. Perform any of the following actions to publish the association or dissociation changes:
   - Review the unpublished changes before you publish them. Go to Step 6 on page 524.
   - Publish the changes. Go to Step 7 on page 525.
6. Perform the following steps to review the unpublished changes:

a. Select the **See unpublished commits** option.

The **Your system model changes** page is displayed. You can find the following information about the changes that you made to the system model when you associated resources with components or deleted the association:

- The *Edit* branch that you selected to contain the changes is displayed in the **Branch** field. You can select the *Edit* branch that you used before associating resources to components, if it is not already displayed as selected.
- The projects that contained the resources, which you associated with components in the system model are listed under the **Asset name** column.
- The type of change is listed under the **Changes** column.
- The name of the member who made the change is displayed under the **Last changed by** column.
- The relative time when the change was made is displayed under the **Last updated** column.
- The number of changes that the *Edit* branch is ahead of from the project repository branch is indicated by the **commits ahead** option. You can view the details by clicking the **commits ahead** option and in the dialog that displays, you can discard the pending commits by clicking the **Reset** option.
- The number of changes that the *Edit* branch is behind from the project repository branch is indicated by the **commits behind** option. The *Edit* branch can be behind the project repository branch if another member has published or committed other changes in the system model to the same project repository branch. You can view the details by clicking the **commits behind** option. You can merge all the changes to your *Edit* branch so that the *Edit* branch is updated with all the changes made in the project repository branch.

b. Click **Publish**.

c. Enter a commit message in the **Description of change** field, and then click **Publish**.

Note: If you do not want to commit the changes that you made to the system model, you can select the **Discard** option.

7. Perform the following steps to publish the unpublished changes:

a. Click the **Publish changes** option.

b. Enter a commit message in the **Description of change** field, and then click **Publish**.

Note: If you do not want to commit the changes that you made to the system model, you can select the **Discard** option.

Results

You have published the association changes that you made in the system model to the repository in the project. Other members of the team space can view the association changes and use the system model.

Viewing the system model

You can view an existing system model that is in a team space in HCL DevOps Test Hub (Test Hub).

Before you begin

You must be a licensed user of Test Hub and a member of a team space.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.

Results

You viewed an existing system model in the team space from the **System Model** page.

Deleting a system model

When you want to create a new system model in HCL DevOps Test Hub (Test Hub), you must first delete the existing system model, as only one system model is supported per team space.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **System Model** in the left navigation pane.

   The **System Model** page is displayed.
4. Click the **Delete** icon  .

   The **Delete system model** dialog is displayed.
5. Click **Delete**.

Results

You deleted a system model.

What to do next

You must publish the changes to the team space repository. See Publishing changes to the system model on page 516.

Deleting a repository from a team space

You can delete a repository from a team space in HCL DevOps Test Hub (Test Hub) when you do not want to use a team space repository.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* or an *Architect* in the team space. See Managing members and their roles in a team space on page 1207.
- Added a repository to the team space. See Adding a repository to a team space on page 502.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane. Then, use the **Switch to** option to open the team space to which a team space respository is added.
3. Click **Manage > Configuration**.
   Result
   The **Configuration** page of the team space is displayed.
4. Click the **Delete** icon  in the action menu list.

   The **Delete repository** dialog is displayed.
5. Select the **I understand...** checkbox, and then click **Delete**.

Results

You have deleted the team space repository.

Restriction: When you delete the team space repository, the system model if it exists is also deleted, and you or other members of the team space cannot add a system model.

Prerequisites to running tests

Before you can run a test asset in a project on HCL DevOps Test Hub (Test Hub), you must read the information about the different tests assets. You can add a remote static agent, or Kubernetes cluster to the project and then select it as an alternate location to run the test assets.

You can find the following information about the prerequisite tasks:

- Test run considerations for AFT Suites or Functional tests on page 528
- Test run considerations for API Suites or API tests on page 529
- Test run considerations for browser-based Web UI tests on page 532

- Test run considerations for JMeter tests on page 532
- Test run considerations for JUnit tests on page 534
- Test run considerations for Postman tests on page 536
- Test run considerations for schedules on page 538
- Test run considerations for Selenium tests on page 539
- Test run considerations for using mobile device clouds on page 540
- Management of agents on page 542
    - Test run considerations for running tests on remote performance agents on page 544
- Management of remote Kubernetes clusters on page 572
    - Considerations for using a remote Kubernetes cluster on page 573
- Management of cloud credentials on page 587

Related information

Test run configurations on page 610

Test run considerations for AFT Suites or Functional tests

Before you configure an AFT Suite or a Functional test run in HCL DevOps Test Hub (Test Hub), you must first read the considerations that you need to take into account.

When you want to run an AFT Suite or Functional test in a project on Test Hub, you can check if the AFT Suite or Functional test meets any of the following conditions:

If...	Then...
The AFT Suite or Functional test uses the settings configured in an `AFT XML` file for a remote agent	You must ensure that you have configured the location element in the `AFT XML` to point to a remote agent. For more information about setting the location element, refer to Using an XML file to run multiple Web UI tests and Compound Tests simultaneously.
You want to run an AFT Suite or Functional test on a remote agent	You must run the test on the remote agent before you commit the test asset to the remote repository. When you run the test on the remote agent, the following events occur:

If...	Then...
	• The agent is added as the location in the `AFT XML` file, on which the test is to be run. • The agent is displayed as the agent on which the test is to be run, under the **Host** column within the **Location** tab in the **Execute test asset** dialog.
You commit the test asset to the remote repository without running the test on the remote agent	The agent is not displayed under the **Host** column within the **Location** tab in the **Execute test asset** dialog.

Before you configure the run for an AFT Suite or a Functional test, you must complete the following tasks, if they are applicable:

- Add the remote agents, on which the test is to be run, to the project on Test Hub. See Adding a performance agent to a project on page 546.
- After you add the agent, you can select the agent configured in the `AFT XML` file as the location for the test run or select the remote agents that you add to the project, when you configure a run for an AFT Suite.

  The remote agents are shown as the agents available for selection under the **Override** column within the **Location** tab in the **Execute test asset** dialog.

  You can then select the remote agent as the location to run the AFT Suite or Functional test when you are configuring the test run. Alternatively, you can enter the argument `-swaplocation <configured_agent_location>:<overriding_agent_location>` in the **Program Arguments** field in the **Advanced Setting** dialog when you are configuring the test run.

For running an AFT Suite, see Configuring an AFT Suite run on page 618.

Related information

Test run configurations on page 610

Test run considerations for API Suites or API tests

Before you configure an API Suite or API test run, you must first read the considerations that you need to take into account.

You can find the following information about API Suites:

- Important information about API Suites or tests on page 530
- Important information about report configuration in API Suites or tests on page 531

Important information about API Suites or tests

When you want to run an API Suite or test from a project on HCL DevOps Test Hub (Test Hub) and the test asset in HCL DevOps Test Integrations and APIs (Test Integrations and APIs) meets any of the following conditions:

- The Suite or test refers to local stubs that use a transport other than HTTP or MQ.
- The Suite or test refers to local stubs that use the HTTP transport.
- The transport used in the tests or Suite is configured with the hostname set to *localhost* for the HTTP/TCP proxy.
- The Suite or test has tests that use a transport and the transport requires third-party application `Jar` files for a successful run.

You must then refer to the following table for the next steps:

If...	Then...
The test asset refers to local stubs that use a transport other than HTTP or MQ.	You must perform any of the following actions before you commit the test asset to the remote repository: • Remove the reference to the local stub from the test asset. • Publish the stubs to Test Virtualization Control Panel. To publish and edit stubs, see Publishing stubs.
The tests refer to local stubs that use the HTTP transport.	You must perform the following actions before you commit the test asset to the remote repository: 1. Create a test or test Suite that contains the stub and tests that run on the stub. 2. Configure the physical HTTP transport with the following parameters in the **Client** tab in the **Physical View**: ◦ Set the `hostname or IP address of the HTTP Proxy server` in the **Proxy Host** field. ◦ Set *3128* as the proxy server port in the **Proxy Port** field. 3. Set the reference to the stub for the tests in the test suite scenario from the **Scenario Editor** dialog from the **Test Factory** view by selecting the *Live system* option in the **Satisfied by** column to run the stubs on the HTTP proxy registered with Test Hub. 4. Save the project.

If...	Then...
The transport used in the Suite or tests is configured with the hostname set to *localhost* for the HTTP/TCP proxy.	You must replace the hostname with *fully-qualified-domain-name* or *IPAddress* of the proxy host in the tests before you commit the test asset to the remote repository.
You plan to run the API Suite or test on the computer where Test Hub is installed and the API Suite or test that use a transport from any of the following third-party applications: • Camel • CentraSite • CICS • Coherence • Database • IMS • Integra • JMS • SAP RFC • Software AG Universal Messaging • TIBCO EMS • TIBCO Rendezvous • TIBCO SmartSockets • WebSphere Application Server Service Integration Bus (SiBus) • WebLogic • Software AG webMethods • WebSphere MQ	You must complete the following tasks: 1. Identify the location of the third-party application JAR files. You can use Library Manager to know the location where the third-party application JAR files are saved. 2. Copy the third-party application JAR files to the computer where Test Hub is installed. See Copying third-party application Jars to Kubernetes on page 165.

Important information about report configuration in API Suites or tests

You must configure test Suites or tests in Test Integrations and APIs to use a results database so that the details of the test results can be captured and displayed in the Test Hub API results reports. See Configuring the project results database.

If your server projects use either a Microsoft SQL Server, MYSQL, or both as a project results database, you must copy the Microsoft SQL Server and MySQL database JAR files to the JDBC folder in the path `/data/JDBC` on Test Hub. See Copying third-party application Jars to Kubernetes on page 165.

The database JAR files that you need are based on the version of Microsoft SQL Server or MQSQL that you use with Test Integrations and APIs. See Adding Microsoft SQL Server and MySQL drivers.

To run an API Suite, see Configuring an API Suite or an API test run on page 652.

Test run considerations for browser-based Web UI tests

Before you configure a Web UI test or a Compound Test that contains Web UI tests, which use either the Chrome or Edge browsers, you must first read the considerations that you need to take into account.

When you want to run a Web UI test or a Compound Test that contains Web UI tests, which use either the Chrome or Edge browser from HCL DevOps Test Hub (Test Hub), you must consider the following conditions:

- You must perform any of the following actions:
  - Accept the license agreement of the built-in Chrome and Edge browsers at the time of signing in or logging in to Test Hub.
  - Accept the license agreement of the built-in Chrome and Edge browsers, if it is not accepted at the time of signing in or logging in. See Management of third-party licenses on page 160.

Test run considerations for JMeter tests

Before you can run JMeter tests on HCL DevOps Test Hub (Test Hub), you must read the considerations that you must take into account and complete the tasks indicated.

Before you run JMeter tests that you create in JMeter, you must have completed the following tasks:

- You must ensure that the JMeter extension on Test Hub is enabled. You must enable the JMeter extension either at the time of installation of Test Hub or before you add the repository that contains JMeter tests to the server project. See Enabling server extensions on page 171.
- You must have committed the following JMeter assets or resources as a JMeter project to a remote Git repository:

  Note: You must have created a project directory and added the associated tests or assets in sub directories, that might be required for the JMeter tests to run successfully.

  - The `jmeterRoot.jprj` project file in the project directory.
  - The JMeter tests in a sub directory with a name you provide within the project directory.
  - Optionally, the Java Key Store that is required at test run time in the `certs` sub directory.
  - Optionally, the JMeter properties files that are required at test run time in the `properties` sub directory.
  - Optionally, the library jar files that are required at test run time in the `lib` sub directory.
  - Optionally, the test dependencies that are required at test run time in the `deps` sub directory.

  For example, the project directory can be *myproject* that contains the `jmeterRoot.jprj` project file. In the project directory, you can create the following sub directories:
  - The `mytests` sub directory that contains the JMeter tests.
  - Optionally, the `certs` sub directory that contains the Java Key Store.

    For example, the path to the key store can be specified in the `system.properties` file as follows:

    ```
    javax.net.ssl.trustStore=$(deploy.home)/certs/simpleStore.jks
    ```

Where <*simpleStore.jks*> is the Java Key Store and the <*deploy.home*> is the keyword to access the key store at the test run time.
    - Optionally, the `properties` sub directory that contains properties files such as `system.properties`, `jmeter.properties`, `reportGenerator.properties`, or `user.properties`.
    - Optionally, the `lib` sub directory that contains the **lib** and **lib/ext** content to be used at test run time. For example, the **amq client** and the **plugins** required to parse the **jmx** that contains **amqp** calls.
    - Optionally, the `deps` sub directory that contains the test dependencies.

  The resulting project directory can be as follows:

```
myproject (that contains the jmeterRoot.jprj)
myproject/mytests
myproject/certs
myproject/properties
myproject/lib
myproject/deps
```

- You must ensure that you use a relative path when JMeter tests refer to other test assets. For example, if the JMeter test that is in the `myproject/mytests` folder refers to a `mycsv.csv` file that is in the `myproject/deps` folder, then the `mycsv.csv` is referred as follows: `../deps/mycsv.csv` in the JMeter test.
- You can create the JMeter project file named as `jmeterRoot.jprj`.

  The project file is a java properties file. The JMeter project file is required so that Test Hub can identify the asset as a JMeter project. You can specify additional information in the project file. You can specify the names of the sub directories created and through keywords you can specify the path in the properties files whether they are relative to the root directory or to the tests sub directory.

  For example, if you have created the following sub directories in the project directory called *myproject* that contains the `jmeterRoot.jprj` project file:
    - `myproject/mytests`
    - `myproject/properties`

  If the JMeter tests are contained in the following path: `myproject/mytests/testplan/test1.jmx`, and some dependencies that are also required to be deployed at test run time exists in the following path: `myproject/mytests/testplan/localdir`. You must specify these file paths in the project file as follows:

```
testDir.localtest.sub.dir=localdir
aSecondRootDir.sub.dir=dir2
athirdRootDir.sub.dir=dir3
```

Related information

Configuring a run of a JMeter test on page 867

Troubleshooting issues on page 1273

Test run considerations for JUnit tests

Before you can run JUnit tests on HCL DevOps Test Hub (Test Hub), you must read the considerations that you must take into account and complete the tasks indicated.

Before you run JUnit tests, you must have completed the following tasks:

- You must ensure that the JUnit extension on Test Hub is enabled. You must enable the JUnit extension at the time of installation of the server software or before you add the repository that contains the JUnit tests to the server project. See Enabling server extensions on page 171.
- You must have created the JUnit tests that use JDK 8, as part of a Maven V3.6.3 project. The Maven project must contain the Maven Surefire plugin, which is required to run the JUnit tests contained in the Maven project. The Maven project as a `pom.xml` must be committed to the remote repository.



Note: If you create the JUnit tests on other versions of JDK, you must copy the JDK version to the `userlibs` pod under the `/data/junit-ext/jdks` directory. See Copying third-party application Jars to Kubernetes on page 165.

Related information

Configuring a run of a JUnit test on page 876

Troubleshooting issues on page 1273

Test run considerations for using Machine learning capabilities

Before you configure a Performance Test or Schedule run to use Machine learning (ML) capabilities, you must first read the considerations that you need to take into account.

When you want HCL DevOps Test Hub (Test Hub) to analyze Performance tests or Schedules by using ML capabilities, you must ensure that Performance tests or Schedules conform to the following criteria:

- Must contain HTML tests.
- Must contain stress or performance user profiles.
- Must contain the following minimum stress test requirements:
  - 20 users or more.
  - A minimum of three different user groups and each group has an activity of more than two minutes.
  - Stages that have a reasonably high number of samples within each stage.
  - Stages do not have an increasing number of users.
  - Tests run for longer durations so that a trend can be observed.

You can refer to the following table to view the criteria for analysis of the parameters:

Parameter analyzed	Criteria for analysis
Response Time Lock-Step Pattern	The analyzer identifies the **Response Time Lock-Step Pattern** parameter in the overall page response time observed against the user count based on the following criteria: • Tests have a minimum of 20 users. • Tests run for longer durations for a trend to be observed. • Tests contain a minimum of three different user groups and each group has an activity of more than two minutes.
Response Time Standard Deviation Pattern	The analyzer attempts to detect the response time of pages that are more than thrice the value of the standard deviation calculated for the page response time as the **Response Time Standard Deviation Pattern** parameter.
Throughput Drop Pattern	The analyzer attempts to detect sudden drops in network throughput as the **Throughput Drop Pattern** parameter that is based on the following criteria: **Note:** Sudden drops in throughput might be related to the performance tool itself, issues with network connectivity, or issues with signal-scalability of the system under test. • Tests have a minimum of 20 users. • Tests have stages with a reasonably high number of samples within a stage. • The length and intensity of the throughput drops that are less than 20% of the median observed within a stage are considered by the analyzer. When no throughput is received within 60 seconds, such drops are considered in the analysis. Short shark-tooth patterns in drops are not considered by the analyzer. • Parts of the stage with increasing users are not considered for analysis. For example, when the test adds one user every minute and the user range is between one user to 500 users, the **Throughput Drop Pattern** parameter is not analyzed.
Performance Agent Health	The analyzer inspects the health of agents that are participating in a performance execution and identifies potential configuration issues based on the following criteria: • **Minimum requirements** ◦ At least 10 users ◦ At least 10-minute run time ◦ Requires schedule with at least 1 static agent (Performance Agent Health insight is not generated if you are running the test on local host) • **Blocking threads**

Parameter analyzed	Criteria for analysis
	◦ Considered as Blocking threads when 90% of available threads are used for 20% of the run time • **Pegged CPU** ◦ Considered as Pegged CPU when 90% of CPU is utilized for 10% of the run time • **Pegged CPU for interval** ◦ Considered as Pegged CPU for interval when there are intervals of high CPU utilization greater than 95% For example, above 95% of CPU utilized for 50% of a 10-minute interval or 98% of CPU utilized over a shorter interval. ◦ Considered as Pegged CPU for interval when there are spikes in CPU utilization during long test runs The agents that are not configured properly can distort performance assessments of the system under test. The analyzer can provide multiple insight types. The insight recommendation texts are parametrized and they provide specific recommendations such as identifying the agents with issues. You can view the following insights: • Pegged CPU for run insight • Pegged CPU for interval insight • Blocked thread insight

Related information

Machine learning capabilities in DevOps Test Hub on page 1027

Test run configurations on page 610

Test run considerations for Postman tests

Before you can run Postman tests on HCL DevOps Test Hub (Test Hub), you must read the considerations that you must take into account and complete the tasks indicated.

Before you run Postman tests that you create in Postman, you must have completed the following tasks:

- You must ensure that the Postman extension on Test Hub is enabled. You must enable the Postman extension at the time of installation of the server software or before you add the repository that contains Postman collections to the server project. See Enabling server extensions on page 171.

  Restriction: If you add a Git repository with Postman collections to a server project before you enable the Postman microservice, the Postman collections might not be displayed in the **Execution** page for you.

- You must have exported the following resources from Postman and saved the resources on your computer:
  - Collections and their associated variables and environments from Postman in which all collections are exported as a single file.

    Note: You can export as many collections as you want.

    For more information, refer to Exporting a collection from Postman and Exporting data dumps.
  - Environments configured in Postman, if you are exporting only the environments as separate JSON files.

    For more information, refer to Exporting environments from Postman.
- You must have committed the following Postman assets or resources to the remote repository:
  - Collections and their associated variables and environments that you exported. You can export all resources that are in your workspace in Postman. You can also export the collections along with the variables as a single file and the environments separately as another file.

    For example, you can save the exported file as `MyCollectionsVarEnv.json` that contains the collections, variables, and environments or you can export only the collections and variables as `MyCollectionsVar.json`.
  - Environments that you exported, if you are exporting only the environments as separate JSON files. For example, you can save the exported file as `MyEnv.json`.
  - An empty text file with the `.postman` extension that is contained in the same directory as the other Postman resources, so that the contents in the directory are attributed as Postman resources. For example, you can create `MyProject.postman`.

    Note: The `.postman` file enables Test Hub to identify the test resources as Postman resources.

    For example, if you have saved the Postman resources that you downloaded from Postman as follows:
    - `MyCollectionsVarEnv.json` or `MyCollectionsVar.json` that contains the Postman collections.
    - `MyEnv.json` that contains the environments associated with the Postman collection.

    The project directory can be as follows:

    ```
    /Project
    /Project/MyProject.postman
    /Project/MyCollectionsVarEnv.json
    ```

Alternatively, the directory can be as follows:

```
/Project
/Project/MyProject.postman
/Project/MyCollectionsVar.json
/Project/MyEnv.json
```

Related information

Configuring a run of a Postman test on page 886

Troubleshooting issues on page 1273

Test run considerations for schedules

Before you configure a Rate Schedule or VU Schedule run, you must first read the considerations that you need to take into account.

Important information about JMeter tests in schedules

When you want to run JMeter tests as part of the VU Schedule or Rate Schedule on HCL DevOps Test Hub (Test Hub), you must have completed the following tasks:

- Installed the HCL DevOps Test Performance Agent on a computer.
- Installed the JMeter application on the computer where you have installed the HCL DevOps Test Performance Agent.
- Set the environment variable **JMETER_HOME** that points to the JMeter installation directory.

Important information about Resource Monitoring sources in schedules

When you want to run schedules on Test Hub, you must have completed the following tasks in Test Performance:

- Created a performance schedule of **Rate** or **VU** type.
- Added Resource Monitoring sources, or the labels of Resource Monitoring sources to a schedule. Refer to Adding resource monitoring sources to a performance schedule by using labels.

Before you configure a run of a Schedule, you must complete the following tasks in Test Hub:

- Add Resource Monitoring sources to your project in Test Hub. See Monitoring an NGINX server on page 1050.
- Enable Test Hub as Resource Monitoring service only if you want to replace Resource Monitoring labels set in Test Performance with the labels that you created in Test Hub. See Controlling resource monitoring sources in a schedule on page 1079.
- Perform the following steps when you want to override the Resource Monitoring labels in the test asset with the Resource Monitoring labels that you create in Test Hub:
    1. Click the **Resource Monitoring** tab.

Note:

The **Resource Monitoring** tab displays only if you enabled the **Resource Monitoring from Service** option in Test Performance when you created the test asset.

2. Click  , and press the `Ctrl` + `Space bar` keys, or enter the initial letter of a label to select a label in the list.

Note:

You can select or add labels only if you added the labels in the **Resource Monitoring Sources** page in Test Hub.

For running a Rate Schedule or VU Schedule, see Configuring a run of a Rate Schedule or VU Schedule on page 896.

Related information

Test run configurations on page 610

Test run considerations for Selenium tests

Before you can run Selenium tests on HCL DevOps Test Hub (Test Hub), you must read the considerations that you must take into account and complete the tasks indicated.

Before you run Selenium tests, you must have completed the following tasks:

- You must ensure that the JUnit extension on Test Hub is enabled. You must enable the JUnit extension at the time of installation of the server software or before you add the repository that contains the Selenium tests to the server project. See Enabling server extensions on page 171.
- You must have created the Selenium tests that use JDK 8, as part of a Maven V3.6.3 project. The Maven project must contain the Maven Surefire plugin, which is required to run the Selenium tests contained in the Maven project. The Maven project as a `pom.xml` must be committed to a remote Git repository.

  Note: If you create the Selenium tests on other versions of JDK, you must copy the JDK version to the `userlibs` pod under the `/data/junit-ext/jdks` directory. See Copying third-party application Jars to Kubernetes on page 165.

- You must ensure that the following flags are configured in the Selenium test:
  - Firefox is configured as the web browser to run the test.
  - Firefox is enabled to accept all certificates (secured and unsecured).
  - Firefox is configured to not display alerting messages when the test runs in the Firefox browser.
- You must set the system property to point to the Firefox driver (`linux_x64_geckodriver`) that is provided along with the JUnit server extension and is available in the `/tmp` folder.

For example,

```
System.setProperty("webdriver.gecko.driver", "/tmp/linux_x64_geckodriver")
```

Related information

Configuring a run of a Selenium test on page 909

Troubleshooting issues on page 1273

Test run considerations for using mobile device clouds

Before you configure runs of tests that use mobile device clouds as the location, you must first read the considerations that you need to take into account.

Before you can use mobile devices that are connected to a mobile device cloud from HCL DevOps Test Hub (Test Hub), you must complete the following tasks:

- Create any of the following types of tests that are supported to be run from Test Hub.

Test type	Description
AFT Suite	The AFT Suite must specify the path to the tests, device names, and the device cloud location to run the tests. AFT Suites that specify multiple tests or devices on the same device cloud are supported. **Restriction:** AFT Suites that specify a single test on multiple devices or multiple tests on multiple devices when the devices are connected to multiple device clouds are not supported.
Compound Tests that contain mobile tests or Web UI tests that use a device cloud	The Compound Tests must specify multiple mobile tests or multiple Web UI tests that run on a device connected to a single device cloud. **Restriction:** Compound Tests that specify tests on single or multiple devices that are connected to multiple device clouds are not supported.
Individual mobile test or Web UI test that use a device cloud	The individual test must be either a mobile test or a Web UI test that runs on a device connected to a single device cloud.

- Ensure that you have valid credentials for the mobile device clouds, which are supported to be connected from Test Hub. You must have the details as listed in the following table for the mobile device cloud that you want to connect and use:

Attribute	**Description**	**Mobile device cloud**				
		BitBar Cloud	**BrowserStack Cloud**	**LambdaTest Cloud**	**pCloudy Cloud**	**Perfecto Mobile Cloud**
Host	The hostname of the mobile device cloud instance.	Required	Required	Required	Required	Required
API key	The API key of your cloud account to authenticate the connection.	Required	Required	Required	Required	Not applicable
Security token	The token used to authenticate the connection.	Not applicable	Not applicable	Not applicable	Not applicable	Required
Project	The project name to use on the mobile device cloud.	Required	Required	Required	Not applicable	Not applicable
Username	The username of your cloud account.	Not applicable	Required	Required	Required	Not applicable

- Complete the following prerequisite tasks to enable the devices on the mobile device cloud for testing mobile apps or mobile web browsers:

Device cloud	**Testing on Android OS**	**Testing on iOS**
BitBar Cloud BrowserStack Cloud Perfecto Mobile Cloud	◦ Uploaded the `.apk` file of the Android application that you want to test. ◦ Installed the Android application that you want to test on the Android device.	◦ Installed the `.ipa` or `.app` file of the iOS application that you want to test. ◦ Installed the iOS application that you want to test on the iOS device.

Device cloud	Testing on Android OS	Testing on iOS
pCloudy Cloud LambdaTest Cloud	◦ Uploaded the `.apk` file of the Android application that you want to test. ◦ Installed the Android application that you want to test on the Android device. ◦ Verified that the device you want to test is available and in the *Released* state on the cloud platform.	◦ Uploaded the `.ipa` file of the iOS application that you want to test. ◦ Installed the iOS application that you want to test on the iOS device. ◦ Verified that the device you want to test is available and in the *Released* state on the cloud platform.

- Ensure that as a *Project Owner* or *Tester*, you are an authorized user of the cloud credential in your team space. See Viewing details of cloud credentials on page 601.
- Ensure that you create a variables file that contains the details of the agent or device cloud, if you do not want to use the cloud credentials on the server.

Management of agents

You can find information about managing agents that you install on remote computers and then configure the remote agents to register with HCL DevOps Test Hub (Test Hub). After the agents are registered, you can add them to your projects and then select them as locations to run tests or virtual services.

When you plan to run any of the tests from a project on Test Hub on a remote performance agent, see Management of performance agents on page 542.

When you plan to run any of the supported virtual services from a project on Test Hub on a remote API agent, see Management of API agents on page 551.

When you plan to run Web UI tests that are created in Test Hub on a remote agent. See Running server test assets on an agent on page 811.

Management of performance agents

You can manage the performance agents that you want to use as locations to run tests from HCL DevOps Test Hub (Test Hub). When you install the agents by using your offline user token, the agents are registered with the team space in Test Hub. You can view the registered agents before you add them to your projects. You can then select them as locations to run tests.

When you plan to run any of the tests on a remote performance agent from a project in Test Hub, you must have completed the following tasks:

- Installed the performance agent of the same version as the version of Test Hub. Refer to Installing DevOps Test Performance Agent.
- Configured the performance agent with your offline user token so that the performance agent can register with Test Hub.

  Note: When you configure the performance agent by using your offline user token, Test Hub considers you as the owner of the performance agent.

- Verified that you have write permissions to the `majordomo.config` file so that certain transactions can be written by Test Hub to the `majordomo.config` file.

  Note: You can verify if you have the *Read* and *Write* permissions by checking the properties of the `majordomo.config` file.

- Viewed the performance agents that are registered with Test Hub. See Viewing performance agents that are registered with DevOps Test Hub on page 545.
- Added the remote performance agents that are registered with Test Hub to your project on Test Hub. See Adding a performance agent to a project on page 546.
- After you add the remote performance agents to your project, Test Hub determines the capabilities that are provided by the remote performance agent and displays the details and capabilities of the performance agent on the **Agents and Intercepts** page. You can also define the capabilities that you want to add, and then view, edit, or delete the capabilities that you defined for the performance agents in your project. See Working with the capabilities of a performance agent on page 548.

Connecting a Docker performance agent to Test Hub

You can run a Performance agent in Docker container and connect it to Test Hub.

Before you begin

You must have the following details to connect the Docker performance agent to Test Hub:

- `TESTHUB_SERVER`: Server host name.
- `TESTHUB_TOKEN`: Offline user token that is obtained in following context.
- `TESTHUB_PORT`: `443`
- `TESTHUB_TEAM_SPACE`: `Initial`
- `TESTHUB_PATH_CONTEXT`: Server API context path.
- `Docker_Image`: Download the `HCL_DT_AGENT_<version_number>_DOCKER_IMG.zip` file from My HCLSoftware and refer to README document in the zip file to load the docker image `hcl-devops-test-performance-<version_number>.tar`.

1. Log in to Test Hub.
2. Click the **User** icon , and select the **Create Token** option.

Result

The token is generated automatically and displayed in the **Create Offline User Token** dialog.

3. Click **Copy** to copy the token and save it in a secure location.
4. Run the following command where the Docker image of the performance agent is located:

```
TESTHUB_SERVER=<server_url>
TESTHUB_TOKEN=<access_token>
TESTHUB_PORT=443
TESTHUB_TEAM_SPACE=Initial
TESTHUB_PATH_CONTEXT="/test"
docker run --detach --name Test_performance_agent_docker \
--env TESTHUB_SERVER=$TESTHUB_SERVER \
--env TESTHUB_PORT=$TESTHUB_PORT \
--env TESTHUB_TOKEN=$TESTHUB_TOKEN \
--env TESTHUB_TEAM_SPACE=$TESTHUB_TEAM_SPACE \
--env TESTHUB_PATH_CONTEXT=$TESTHUB_PATH_CONTEXT hcl-devops-test-performance:12.0.0
```

5. Go to **Infrastructure > Agents and Intercepts** and verify whether your agent is connected to Test Hub.
6. Run the following commands to stop and remove the Docker container:

   `docker stop <assigned_name_of_the_container>`

   `docker rm <assigned_name_of_the_container>`
7. Run the following command to view all the help options and available environment variables:

   `docker run --rm hcl-devops-test-performance:12.0.0`

Results

You have connected the Docker performance agent to Test Hub.

Test run considerations for running tests on remote performance agents

Before you configure a test to run on a remote performance agent from HCL DevOps Test Hub (Test Hub), you must read the considerations that you must take into account.

Supported tests

You can run the following tests that are supported to be run on a remote performance agent:

- AFT Suite
- Compound Tests that contains performance tests
- Compound Tests that contains Web UI tests
- Compound Tests that contains traditional HTML tests
- Rate Schedule
- VU Schedule

Prerequisite tasks

Before you can run any of the supported tests on a remote performance agent from a project on Test Hub, you must have completed the following tasks:

- Installed the HCL DevOps Test Performance agent on the remote computer. See Management of agents on page 542.
- Configured the agent with your offline user token so that the agent can connect to Test Hub.
- Added the remote agents to your project on Test Hub. See Adding a performance agent to a project on page 546.

Note: If you want to run an AFT Suite (that contains Web UI tests) on a performance agent, you must start the performance agent as a non-admin user so that certain browsers that are configured can start correctly during the test run time.

Before you add the tests to the remote repository, you must have completed the following tasks:

- Associated the performance agents with the performance schedule in Test Performance.
- Specified the hostname of the remote workstation in the AFT XML file.

You can continue to configure a run for any of the following supported test types:

- Configuring an AFT Suite run on page 618
- Configuring a run of a Compound Test that contains performance tests on page 743
- Configuring a run of a Compound Test that contains Web UI tests on page 704
- Configuring a run of a Compound Test that contains HTML tests on page 665
- Configuring a run of a Rate Schedule or VU Schedule on page 896

Viewing performance agents that are registered with DevOps Test Hub

After you install the remote performance agents and configure them with your offline user token, the performance agents register with HCL DevOps Test Hub (Test Hub) and you can view the registered performance agents on the **Agents and Intercepts** page.

Before you begin

You must have completed the following tasks:

- Installed the performance agent of the same version as the version of Test Hub.
- Configured the performance agent with your offline user token so that the performance agent can register with Test Hub.

  Note: When you configure the performance agent to connect to the server, you must specify the offline user token that you generated on Test Hub. This token defines your ownership over the agent.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

   The **Agents and Intercepts** page is displayed.

You can view the API agents, intercepts, Kubernetes clusters, performance agents, or DevOps Test Runtime agents that are registered with Test Hub.

3. You can view the agents that you own in any of the following ways:
    - Search for the agent by entering the name of the agent in the **Search** field.

      Note: You can enter either the full name or any text that is in the name. The search is enabled for case sensitive text that you can enter.

    - Sort the **Type** column to sort the items and display the agents in the rows at the top of the table.
    - Sort the **Agents** column and identify the agents by their names or by their owner.

   You can view the following details about the agents that are registered with Test Hub:
    - The projects to which the agent is added, are displayed in the **Projects** column.
    - The status of the agent is displayed in the **Status** column.

      Note: You can add agents that are in the `Ready` state to your project, and then use them as locations to run tests or virtual services.

Results

You have viewed the registered agents from the **Agents and Intercepts** page.

What to do next

You can perform any of the following tasks:

- You can add the performance agents that you own to your project. See Adding a performance agent to a project on page 546.
- You can add the API agents that you own to your project. See Adding a performance agent to a project on page 546.

Adding a performance agent to a project

If you have performance tests that distribute workload across different workstations or Accelerated Functional Testing (AFT) Suites that need to run on different platforms and browsers, you can add HCL DevOps Test Performance Agents to your projects on HCL DevOps Test Hub (Test Hub) to run tests on remote workstations.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Installed the agent by using your offline user token, which makes you the owner of the agent, or you are assigned the role of a *Team Space Owner*.

About this task

Only you can add or remove a performance agent that you own from the project. However, any member of your project with the permission to run the tests can initiate the run of test assets that are associated with the performance agent.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open your project, and then click **Infrastructure > Agents and Intercepts**.

   The **Agents and Intercepts** page is displayed.
3. Click **Add > Add performance agent**.

   The performance agents that you configured with your user token are displayed.

   Note: You can add the performance agents that you own and that are in the *Ready* state. You can add any number of performance agents to your project.

4. Select the performance agents that you want to use in the project, and then click **Add**.

   The performance agents that you added to the project are displayed.
5. Perform the following steps to view details and capabilities of the performance agents:

   a. Expand the performance agent to view the details panel.

      You can find the following details of the performance agent displayed:

Parameter	Description
Version	The version of the installed performance agent.
Address	The IP address of the computer on which the performance agent is installed.
License type	The license type that is enabled for you to use the performance agent.
Load generation service	The status of the performance agent whether it can be used for load generation service.
Last contact	The time of the last contact with the performance agent by Test Hub.

b. Expand the capabilities panel to view the system and application capabilities.

You can find the following capabilities of the performance agent displayed:

Capabilities		Description
System capabilities	Architecture	The architecture of the CPU processor of the computer on which the performance agent is installed.
	MAC Address	The MAC address of the computer on which the performance agent is installed.
	Operating system	The operating system of the computer on which the performance agent is installed.
	RAM	The memory that is allotted to RAM in the computer on which the performance agent is installed.
Application capabilities	Browser	The browsers along with their version that are installed on the computer on which the agent is installed.
User Defined capabilities	User Defined	The capabilities that you add for a performance agent. You can also tag a performance agent so that you can identify the performance agent to select as a location.

c. Check for the status of the performance agent from the **Status** column.

Note: You can add performance agents that are in any state to the project.

Results

You have added remote performance agents to your project and viewed the performance agent capabilities that is displayed for each agent that you own.

What to do next

You can perform any of the following tasks:

- Run the tests on the performance agent that is added to the project from the **Execution** page. See Test run configurations on page 610.
- Define the capabilities that you want to add, and then view, edit, or delete the capabilities that you defined for the performance agents in your project. See Working with the capabilities of a performance agent on page 548.

Working with the capabilities of a performance agent

When you installed performance agents V10.1.2 or later, and added them to your project on HCL DevOps Test Hub (Test Hub), you can view capabilities of the performance agents from the **Infrastructure** page. When you installed performance agents V10.2.0 or later, you can add, view, edit, or delete properties and capabilities of performance agents from the **User Defined** panel of that performance agent.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Installed the agent by using your offline user token, which makes you the owner of the agent, or you are assigned the role of a *Team Space Owner*.
- Installed a performance agent. See Management of agents on page 542.
- Added performance agents to your project. See Adding a performance agent to a project on page 546.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open your project, and then click **Infrastructure > Agents and Intercepts**.

   The agents that you added to your project are displayed on the **Agents and Intercepts** page.
3. Expand the performance agent.

   The **Details** and **Capabilities** tabs are displayed.
4. Click **Capabilities** to view the capabilities of the performance agent.
5. Click **User Defined**.

   The **User Defined Capabilities** dialog is displayed.
6. Complete the steps for the task that you want to perform as listed in the following table:

Task	Action
Adding capabilities	Perform the following actions: a. Click **Add new**. b. Enter the name of the capability in the **Capability Name** field. c. Perform any of the following actions in the **User Defined Category** field: ▪ Select an existing category from the list. ▪ Enter a name of the category for the capability, if the category does not exist.  **Note:** You can also click **Add new** to add a category. The category for the capability is either selected or created and the fields for the property name and value are displayed.

Task	Action
	Note: The property for a capability is a *name-value* pair. d. Enter a name for the property in the **Property Name** field and its corresponding value in the **Property Value** field for the capability. **Note:** You can add multiple properties for the same capability under the same category or under different categories by clicking **Add new property**, and then by entering the name and value for the property. e. Click **Save**. The capabilities that you added are displayed.
Viewing capabilities that you added	The **User Defined Capabilities** panel displays the capabilities that you added to the performance agent.
Editing capabilities that you added	Perform the following actions: a. Click the **Edit** icon in the row of the capability that you want to edit. The user-defined capability is enabled for editing. You can edit the name of the capability, or the property name or property value. b. Edit any of the items that you want for the user-defined capability. c. Click **Update**. The updated capability of the performance agent is displayed.
Deleting capabilities that you added	a. Click the **Delete** icon in the row of the capability that you want to delete. b. Click **Delete**. The capability that you selected is deleted along with all the properties that you added for the capability.
Deleting a property defined for a capability	a. Click the **Edit** icon in the row of the capability that you want to edit and remove a defined property. The user-defined capability is enabled for editing. b. Select and delete the contents in the **Property Name** field and its corresponding **Property Value** field. c. Click **Update**. The property is removed for the capability.

Results

You have added, viewed, edited, or deleted a property or the user-defined capability of a performance agent.

What to do next

You can run the tests on the performance agent from the **Execution** page. See Test run configurations on page 610.

Management of API agents

You can manage the HCL DevOps Test Integrations and APIs (Test Integrations and APIs) agents that you want to use as locations to run tests from HCL DevOps Test Hub (Test Hub). When you install the agents by using your offline user token, the agents are registered with the team space in Test Hub. You can view the registered agents before you add them to your projects. You can then select the agents as locations to run virtual services.

Note: The API agent is referred interchangeably as either the Test Integrations and APIs Agent or API agent in the documentation.

Before you run any of the supported virtual services on a remote Test Integrations and APIs Agent from a project on Test Hub, you must have completed the following tasks:

- Installed the Test Integrations and APIs Agent of the same version as the version of Test Hub. Refer to Installing DevOps Test Integrations and APIs Agent.
- Configured the Test Integrations and APIs agent so that the Test Integrations and APIs agent can register with Test Hub.

  Note: When you configure the Test Integrations and APIs agent by using the offline user token, Test Hub considers you as the owner of the API agent.

- Viewed the API agents that are registered with Test Hub. See Viewing API agents that are registered with DevOps Test Hub on page 552.
- Added the remote API agents that are registered with Test Hub to your project on Test Hub. See Adding an API agent to a project on page 554.

If you do not want to retain the API agents in a project or if you want to add the API agents to a different project, you can remove the API agents from a project. When you remove an API agent from a project, you cannot select that API agent to run virtual services in the project. See Removing an API agent from a project on page 555.

If you want to remove a registered API agent from a team space in Test Hub when the API agent is not available for use, you can unregister a remote API agent from a team space. See Unregistering a remote API agent from DevOps Test Hub on page 556.

Considerations for running virtual services on remote API agents

Before you configure a run of a virtual service resource on a remote API agent from HCL DevOps Test Hub (Test Hub), you must read the considerations that you must take into account.

Supported virtual services

You can run all virtual services that virtualize services on a remote API agent registered with a team space in Test Hub except the following services:

- Virtualized DPL calls in CICS® Transaction Server.
- Virtualized services for IBM® Db2® on IBM® z/OS®.

Prerequisite tasks

Before you run any of the supported virtual services on a remote API agent from a project on Test Hub, you must have completed the following tasks:

- Installed the Test Integrations and APIs Agent on the remote computer. See Management of API agents on page 551.
- Configured the API agent with your offline user token so that the API agent can connect to Test Hub.
- Added the remote API agents to your project on Test Hub. See Adding an API agent to a project on page 554.

Licensing

When a virtual service is run on an API agent, the license server configured on the remote agent computer determine the license server that is used by the API agent. This might be different from the license server that is configured for the team space.

Viewing API agents that are registered with DevOps Test Hub

After you install the remote HCL DevOps Test Integrations and APIs (Test Integrations and APIs) agents and configure them with your offline user token, Test Integrations and APIs agents register with HCL DevOps Test Hub (Test Hub). You can then view the registered API agents on the **Agents and Intercepts** page.

Before you begin

You must have completed the following tasks:

- Installed the Test Integrations and APIs agent. See Management of API agents on page 551.
- Configured the Test Integrations and APIs agent so that the Test Integrations and APIs agent can register with Test Hub. See Management of API agents on page 551.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

   The **Agents and Intercepts** page is displayed.

   You can view the API agents, intercepts, Kubernetes clusters, performance agents, or DevOps Test Runtime agents that are registered with Test Hub.

3. You can view the API agents that you own in any of the following ways:
   - Search for the API agent by entering the name of the API agent in the **Search** field.

     Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

   - Sort the **Type** column to sort the items and display the API agents in the rows at the top of the table.
   - Sort the **Agents** column and identify the API agents by their names or by their owners.
4. Perform the following steps to view the details and capabilities of the API agents:
   a. Expand the API agent to view the details panel.

      You can find the following details of the API agent that is displayed:

Parameter	Description
Agent ID	The ID of the installed API agent.
Identifier	The unique identifier of the installed API agent that is configured in the `Agent.config` file for the API agent.
Hostname	The hostname of the computer on which the API agent is installed.
Version	The version of the installed API agent.
Operating system	The operating system of the computer on which the API agent is installed.
Configuration location	The path to the location of the `Agent.config` file on the computer where the API agent is installed.
Tags	The agent attributes or tags that exist in the `Agent.config` file.

   b. Expand the capabilities panel to view the system capabilities.

      You can find the following capabilities of the API agent displayed:

Capabilities	Description
System capabilities	The capabilities of the API agent.

   c. Check for the status of the API agent from the **Status** column.

Results

You have viewed the registered API agents from the **Agents and Intercepts** page.

What to do next

You can add the API agents that you own to your project. See Adding an API agent to a project on page 554.

Adding an API agent to a project

You must add the registered API agents to your projects before you select them as the location to run the virtual service resources that are in your project on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Installed the agent by using your offline user token, which makes you the owner of the agent, or you are assigned the role of a *Team Space Owner*.

About this task

An owner of an API agent or team space owner can add or remove an API agent from the project. However, any member of the project with the permission to run the virtual services can initiate the run of the virtual service resources on API agents that are added to the project.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open your project, and then click **Infrastructure > Agents and Intercepts**.

   The **Agents and Intercepts** page is displayed.
3. Click **Add > Add API agent**.

   The API agents that you configured with your user token are displayed.

   Note: You can add the API agents that you own. You can add any number of API agents to your project.
4. Select the API agents that you want to use in the project, and then click **Add**.

   The API agents that you added to the project are displayed.
5. Perform the following steps to view the details and capabilities of the API agents:
   a. Expand the API agent to view the details panel.

      You can find the following details of the API agent that is displayed:

Parameter	Description
Agent ID	The ID of the installed API agent.
Identifier	The unique identifier of the installed API agent that is configured in the `Agent.config` file for the API agent.

Parameter	Description
Hostname	The hostname of the computer on which the API agent is installed.
Version	The version of the installed API agent.
Operating system	The operating system of the computer on which the API agent is installed.
Configuration location	The path to the location of the `Agent.config` file on the computer where the API agent is installed.
Tags	The agent attributes or tags that exist in the `Agent.config` file.

b. Expand the capabilities panel to view the system capabilities.

You can find the following capabilities of the API agent displayed:

Capabilities	Description
System capabilities	The capabilities of the API agent.

c. Check for the status of the API agent from the **Status** column.

Results

You have added remote API agents to your project and viewed the details and capabilities of the API agent that is displayed for each agent that you own.

What to do next

Configure a run of the virtual service resources that are in your project and select an API agent as the location to run the virtual service. See Configuring a run of a virtual service on page 946.

Removing an API agent from a project

You can remove an API agent from your project in HCL DevOps Test Hub (Test Hub) when you no longer need the agent to run virtual services.

Before you begin

You must have added the registered API agent to a project in your team space. See Adding an API agent to a project on page 554.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open your project, and then click **Infrastructure > Agents and Intercepts**.

   The agents that you added to your project are displayed on the **Agents and Intercepts** page.
3. You can view the API agents that you own in any of the following ways:

- Search for the API agent by entering the name of the API agent in the **Search** field.

  Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

- Sort the **Type** column to sort the items and display the API agents in the rows at the top of the table.
- Sort the **Agents** column and identify the API agents by their names or by their owners.

4. Identify the API agent that you want to remove from the project, and then click the **Remove agent** icon .

Note: The icon is displayed in the **Actions** column in the row of the API agent only if you are the owner of the remote API agent.

Results

You have removed the registered API agent from your project. The API agent is no longer available as a location to you while you configure a run of virtual service resources that are in your project.

What to do next

You might have to add API agents to your project to use the remote API agents as a location to run tests, if there are no API agents added to your project.

Unregistering a remote API agent from DevOps Test Hub

You can unregister a remote API agent that is registered with HCL DevOps Test Hub (Test Hub) in a team space when you no longer require it.

Before you begin

You must have completed the following tasks:

- Installed the agent by using your offline user token, which makes you the owner of the agent, or you are assigned the role of a *Team Space Owner*.
- Configured the Test Integrations and APIs agent so that the Test Integrations and APIs agent can register with Test Hub. See Management of API agents on page 551.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

   The **Agents and Intercepts** page is displayed.

   You can view the API agents, intercepts, Kubernetes clusters, performance agents, or DevOps Test Runtime agents that are registered with Test Hub.

3. Identify the API agent that you want to unregister.

   Note: The icon is displayed in the **Actions** column in the row of the API agent only if you are the owner of the API agent or you are assigned the role of a *Team Space Owner*. The icon is enabled only if the API agent is in a *Lost Contact* state.

4. Unregister the API agent by completing the following steps:
   a. Click the **Unregister API agent** icon .
   b. Click **Unregister** in the **Unregister API agent** dialog box.

Results

You have unregistered the remote API agent from a team space in Test Hub. You cannot run tests on this remote API agent if it is added to your project.

Note: You might have to register it again with Test Hub, if you want to use the remote API agent in your projects.

Management of DevOps Test Runtime agents

When you want to run the server test assets that require resources not available in the cluster, such as the Safari browser, you can use the DevOps Test Runtime agents. You can install the agent on a machine with the required resources, and then select the agent as the location for test runs from HCL DevOps Test Hub (Test Hub). You can also manage the registered agents in Test Hub.

When you install the agents by using your offline user token, the agents are registered with the team space in Test Hub, and they are associated with the project that you specified while registering the agents. You can view the registered agents before you add them to the other projects. You can then select the agents as locations to run the server test assets authored in Test Hub.

Before you run any tests on a DevOps Test Runtime agent from a project on Test Hub, you must have completed the following tasks:

- Installed and registered the DevOps Test Runtime agent. See. Installing and registering DevOps Test Runtime agents on page 558.

  Note: When you configure the DevOps Test Runtime agent by using your offline user token, Test Hub considers you as the owner of the agent.

  Restriction: You can run only one agent on a system.

- Viewed the agents that are registered with Test Hub. See Viewing agents that are registered with DevOps Test Hub on page 563.
- Added the agents that are registered with Test Hub to your project on Test Hub. See Adding an agent to a project on page 564.

If the agent is in the **Ready** state, then you can stop the agent by performing the following actions:

1. Switch to the same command line interface where you started the agent.
2. Navigate to the directory where the runtime script is located.
3. Press `Ctrl+C` keys.

If you do not want to retain the agents in a project or if you want to add the agents to a different project, you can remove the agents from a project. When you remove an agent from a project, you cannot select that agent to run tests in the project. See Removing an agent from a project on page 565.

If you want to remove a registered agent from a team space in Test Hub when the agent is not available for use or in the **Lost Contact** state, you can unregister an agent from a team space. See Unregistering a remote agent from DevOps Test Hub on page 566.

Installing and registering DevOps Test Runtime agents

When you want to run Web UI tests authored in HCL DevOps Test Hub (Test Hub) on a Linux, Mac, or Windows system using an agent, you must first install and register the system as an agent by downloading and running the DevOps Test Runtime agent.

Before you begin

You must have completed the following tasks:

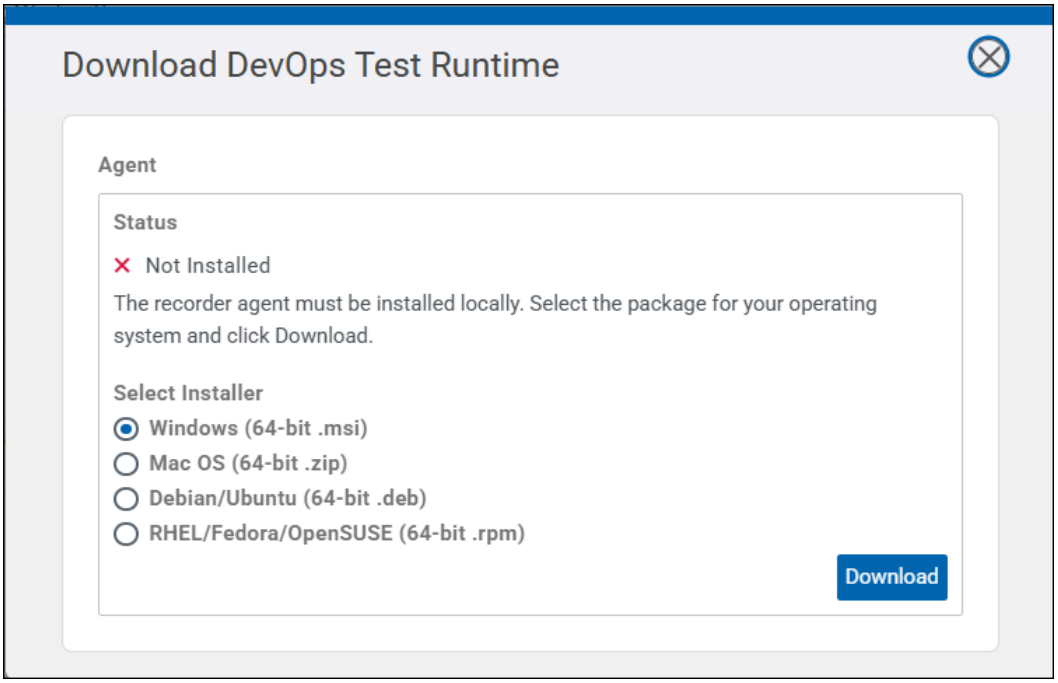
- Ensured that you are a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.
- Generated, copied, and then saved an offline token in a secure location.

1. Log in to Test Hub on the system that you want to register as an agent.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts**.

Alternatively, you can select a project for which the agent should be added and then click **Infrastructure > Agents and Intercepts**.

The **Agents and Intercepts** page is displayed.

3. Click **Download > DevOps Test Runtime**.
   The **Download DevOps Test Runtime** pop-up is displayed.



4. Perform the following steps depending on the operating system:

Operating System	Action
Windows (64-bit)	Perform the following steps to install the agent on Windows: a. Click **Download** to download the recording agent installer. The `devops-test-runtime.msi` file is downloaded. b. Double-click the `.msi` file to install the agent. The installation window with the progress bar is displayed. c. Click **Proceed** to accept the download of the proprietary certificate if prompted. d. Click **Verify** to confirm the agent installation. A confirmation message is displayed after verification is complete. e. Click **Done**. Go to step 5 on page 561.
Mac OS (64-bit)	Perform the following steps to install the agent on Mac:

Operating System	Action
	a. Click **Download** to download the recording agent installer. The `devops-test-runtime.zip` file is downloaded. b. Extract the contents of the `.zip` file: unzip devops-test-runtime.zip c. Quarantine the installer: xattr -dr com.apple.quarantine ./* d. Run the installer: ./install.sh e. Click **Verify** to confirm the agent installation. A confirmation message is displayed after verification is complete. f. Click **Done**. Go to step 5 on page 561.
Debian/Ubuntu (64-bit)	Perform the following steps to install the agent on Ubuntu: a. Click **Download** to download the recording agent installer. The `devops-test-runtime_12.0.0_amd64.deb` file is downloaded. b. Navigate to the directory where the installer is downloaded: cd <download-directory> c. Run the installer: sudo apt install ./devops-test-runtime_12.0.0_amd64.deb d. Click **Verify** to confirm the agent installation. A confirmation message is displayed after verification is complete. e. Click **Done**. Go to step 5 on page 561.
RHEL/Fedora/OpenSUSE (64-bit)	Perform the following steps to install the agent on Linux: a. Click **Download** to download the recording agent installer. The `devops-test-runtime-12.0.0-1.x86_64.rpm` file is downloaded. b. Navigate to the directory where the installer is downloaded:

Operating System	Action
	cd <download-directory> c. Run the following command: INGRESS_DOMAIN=<domain_name> dnf localinstall ./devops-test-runtime-12.0.0-1.x86_64.rpm d. Click **Verify** to confirm the agent installation. A confirmation message is displayed after verification is complete. e. Click **Done**. Go to step 5 on page 561.

5. Perform the following steps to register the system as an agent in Test Hub:
   You must provide the following details to register the system as an agent in Test Hub:
   - `<dth_url>`: The URL of Test Hub. For example, https://10.0.2.15.nip.io/test.
   - `<project_id>`: The ID of a project in Test Hub to which the agent must be associated. For example, `2600` is the project ID in https://10.0.2.15.nip.io/test/#/Initial/project/2600.
   - `<offline_token>`: An offline token generated from Test Hub for the logged-in user who is registering the agent to run tests.
   - `<agent_name>`: The name of the agent to be displayed in Test Hub. The name must be unique and must conform to the DNS name format, however, it need not be a real hostname. For example, my-agent.

Operating System	Action

Windows (64-bit .msi)	Perform the following steps:
	a. Navigate to the `C:Users\<user_name>\AppData\HCL\devops-test-runtime` directory where the runtime script is located.
	b. Run the following commands:
	`set DEVOPS_TEST_HUB_URL=<dth_url>`
	`set DEVOPS_TEST_PROJECT_ID=<project_id>`
	`set DEVOPS_TEST_OFFLINE_TOKEN=<offline_token>`
	`set DEVOPS_TEST_AGENT_NAME=<agent_name>`
	`.\devops-test-runtime.exe agent`
	Confirmation messages about the agent starting and registering are displayed. Go to step 6 on page 562.

Mac OS (64-bit .zip)	Perform the following steps:

Operating System	Action

	a. Navigate to the `~/Applications/devops-test-runtime.app/Contents/Resources` directory where the runtime script is located.
	b. Run the following commands:
	`export DEVOPS_TEST_HUB_URL=<dth_url>`
	`export DEVOPS_TEST_PROJECT_ID=<project_id>`
	`export DEVOPS_TEST_OFFLINE_TOKEN=<offline_token>`
	`export DEVOPS_TEST_AGENT_NAME=<agent_name>`
	`./devops-test-runtime agent`
	Confirmation messages about the agent starting and registering are displayed. Go to step 6 on page 562.

Debian/Ubuntu (64-bit .deb)	Perform the following steps:
	a. Navigate to the `/opt/hcl/devops-test` directory where the runtime script is located.
	b. Run the following commands:
	`export DEVOPS_TEST_HUB_URL=<dth_url>`
	`export DEVOPS_TEST_PROJECT_ID=<project_id>`
	`export DEVOPS_TEST_OFFLINE_TOKEN=<offline_token>`
	`export DEVOPS_TEST_AGENT_NAME=<agent_name>`
	`./devops-test-runtime agent`
	Confirmation messages about the agent starting and registering are displayed. Go to step 6 on page 562.

RHEL/Fedora/OpenSUSE (64-bit .rpm)	Perform the following steps:
	a. Navigate to the `/opt/hcl/devops-test` directory where the runtime script is located.
	b. Run the following commands:
	`export DEVOPS_TEST_HUB_URL=<dth_url>`
	`export DEVOPS_TEST_PROJECT_ID=<project_id>`
	`export DEVOPS_TEST_OFFLINE_TOKEN=<offline_token>`
	`export DEVOPS_TEST_AGENT_NAME=<agent_name>`
	`./devops-test-runtime agent`
	Confirmation messages about the agent starting and registering are displayed. Go to step 6 on page 562.

6. Log in to Test Hub and navigate to **Infrastructure > Agents and Intercepts** to view the agent details that you registered.

Results

You have installed and registered a DevOps Test Runtime agent.

What to do next

You can run the test and view the results. See Running server test assets on an agent on page 811.

Viewing agents that are registered with DevOps Test Hub

After you install and register the remote DevOps Test Runtime agents with HCL DevOps Test Hub (Test Hub), you can view the registered agents on the **Agents and Intercepts** page.

Before you begin

You must have installed and registered the DevOps Test Runtime agents. See Installing and registering DevOps Test Runtime agents on page 558.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

   The **Agents and Intercepts** page is displayed.

   You can view the API agents, intercepts, Kubernetes clusters, performance agents, or DevOps Test Runtime agents that are registered with Test Hub. The page displays the following columns:
   - **Type**: Type of the agent
   - **Agents**: IP address of the agent with the owner's name
   - **Projects**: Projects that are associated with the agent
   - **Status**: Status of the agent to indicate whether it is **Ready** or **Lost Contact**
   - **Actions**: Actions that are allowed on the agent
3. View the DevOps Test Runtime agents that are registered.
   If the list is long, you can search for the agent by entering the name of the agent in the **Search** field.

   Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

4. Expand the agent to view the details of the agent in four different tabs.
   You can find the following details:

Parameter	Description
DETAILS	
Agent ID	The ID of the installed DevOps Test Runtime agent.
Name	The name that you assigned to the agent during registration.
Hostname	The hostname of the computer on which the DevOps Test Runtime agent is installed.
Address	The IP address of the computer on which the DevOps Test Runtime agent is installed.
Version	The version of the installed DevOps Test Runtime agent.
Operating system	The operating system of the computer on which the agent is installed.

Parameter	Description
SYSTEM	
Architecture	The processor type of the system.
Operating system	The operating system of the computer on which the agent is installed.
RAM	The total RAM size of the system.
APPLICATION	
Browser	Browsers that are installed on the system are listed with the name and version details.

Results

You have viewed the registered DevOps Test Runtime agents from the **Agents and Intercepts** page.

What to do next

You can add the DevOps Test Runtime agents that you own to your project. See Adding an agent to a project on page 564.

Adding an agent to a project

You must add the registered DevOps Test Runtime agents to your projects before you select them as the location to run the tests.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Installed the agent by using your offline user token, which makes you the owner of the agent, or you are assigned the role of a *Team Space Owner*.

About this task

The owner of a DevOps Test Runtime agent or a team space owner can add or remove an agent from the project. However, any member of the project with the permission to run the tests can initiate the run on agents that are added to the project.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open your project, and then click **Infrastructure > Agents and Intercepts**.

   The **Agents and Intercepts** page is displayed.
3. Click **Add > Add DevOps Test Runtime agent**.
   The agents that you configured with your user token are displayed.

Note: You can add the agents that you own. You can add any number of agents to your project.

4. Select the agents that you want to use in the project, and then click **Add**.

   The agents that you added to the project are displayed.

Results

You have added remote agents to your project.

What to do next

You can perform the following tasks:

- View the details and capabilities of the agent that you added. See Viewing agents that are registered with DevOps Test Hub on page 563.
- Run a Web UI test that is in your project and select an agent as the location to run the test. See Running server test assets on an agent on page 811.

Removing an agent from a project

You can remove a DevOps Test Runtime agent from your project in HCL DevOps Test Hub (Test Hub) when you no longer need the agent to run tests.

Before you begin

You must have added the registered DevOps Test Runtime agent to a project in your team space. See Adding an agent to a project on page 564.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open your project, and then click **Infrastructure > Agents and Intercepts**.

   The agents that you added to your project are displayed on the **Agents and Intercepts** page.
3. View the DevOps Test Runtime agents that are registered.
   If the list is long, you can search for the agent by entering the name of the agent in the **Search** field.

Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

4. Identify the agent that you want to remove from the project, and then click the **Remove agent** icon .

Note: The icon is displayed in the **Actions** column in the row of the agent only if you are the owner of the remote agent.

Results

You have removed the registered agent from your project. The DevOps Test Runtime agent is no longer available as a location to you while you configure a test run in your project.

Unregistering a remote agent from DevOps Test Hub

You can unregister a remote DevOps Test Runtime agent that is registered with HCL DevOps Test Hub (Test Hub) in a team space when you no longer require it.

Before you begin

You must have installed and registered the DevOps Test Runtime agents, or you are assigned the role of a *Team Space Owner*. See Installing and registering DevOps Test Runtime agents on page 558.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

   The **Agents and Intercepts** page is displayed.

   You can view the API agents, intercepts, Kubernetes clusters, performance agents, or DevOps Test Runtime agents that are registered with Test Hub.
3. Identify the DevOps Test Runtime agent that you want to unregister.

Note: The icon is displayed in the **Actions** column in the row of the agent only if you are the owner of the agent or you are assigned the role of a *Team Space Owner*. The icon is enabled only if the agent is in a *Lost Contact* state.

4. Unregister the agent by completing the following steps:
   a. Click the **Unregister DevOps Test Runtime agent** icon .
   b. Click **Unregister** in the **Unregister DevOps Test Runtime agent** dialog.

Results

You have unregistered the remote agent from a team space in Test Hub. As a result, you cannot run tests on this remote agent if it is added to your project.

Management of the personal agent

A personal agent is a local agent that provides a dedicated, isolated runtime environment to run and diagnose Web UI tests and suites directly on your local machine.

A personal agent is a localized DevOps test runtime instance that is linked to your user account. The personal agent is designed for your individual use and remains invisible and inaccessible to other users. For clear identification, each agent is named using a unique `personal:username` format based on your profile credentials. There can be only one active personal agent per user across all machines. When the agent is ready, you can select the **Run on personal agent** option from the test editor to run the tests locally through that agent instead of a remote container. You can also choose the personal agent as a targeted location to run tests through the **Execute test asset** dialog on the **Execution** page.

The personal agent simplifies and improves the local debugging experience for Web UI tests and suites. As the agent allows you to bypass remote test containers and route tests directly to your local machine, you can visually observe browser behavior in real time and identify the exact point of test failure. This visibility helps you quickly resolve playback errors.

You can use the Personal Agent button located on the test editor page to download, start, or stop the agent. Once you stop the agent, the agent's state changes to inactive in the editor. When the agent is in the inactive state, the **Run on personal agent** option is automatically disabled in the editor interface, and it cannot be used for test runs. As the agent automatically associates with every project you belong to, you can switch between different project workspaces without the need to manually re-associate the agent. You can view the agent listed on the **Agents & Intercepts** page both inside a project and in the initial team space. To ensure privacy and prevent shared access, a different user in the same project can neither view the agent on the **Agents & Intercepts** page nor select it as a target location to run tests.

Installing and starting the personal agent

When you want to run server test assets or server suite assets locally on your system and view the live test run, you can install and use the personal agent. To run tests locally, you must first download the DevOps Test Runtime agent, and then install and start the agent in **Test Editor**.

Before you begin

You must have ensured that you are a Team Space Owner, Project Owner, or Member with a Tester role to create or edit a test resource.

1. Log in to Test Hub on the system that you want to register as an agent.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > project_name** to open the project that contains the test assets.
   The **Overview** page of the project is displayed.
3. Click **Author > Test Editor**.
   The **Test Editor** page and the test navigator panel are displayed.

4. Click the  button located in the page title bar next to the branch selector.
   The **Manage DevOps Test Runtime** dialog opens and checks the status of your local installation.
5. Perform one of the following steps depending on the installation status of the agent:

If...	Then...
The agent is not installed, and the screen displays the following options: ◦ Windows (64-bit .msi) ◦ Mac OS (64-bit .zip) ◦ Debian/Ubuntu (64-bit .deb) ◦ RHEL/Fedora/OpenSUSE (64-bit .rpm)	Perform the following steps to install the agent on Windows: a. Click **Download** to download the agent installer. The `devops-test-runtime.msi` file is downloaded. b. Double-click the `.msi` file to install the agent. The installation window with the progress bar is displayed. c. Click **Proceed** to accept the download of the proprietary certificate if prompted. d. Click **Verify** to confirm the agent installation. A browser notification might get displayed for you to confirm the opening of the agent. You must click **Open Link**. The **Launch** button is enabled. Go to step 6 on page 571.
	Perform the following steps to install the agent on Mac: a. Click **Download** to download the agent installer. The `devops-test-runtime.zip` file is downloaded. b. Extract the `.zip` file: unzip devops-test-runtime.zip c. Quarantine the installer: xattr -dr com.apple.quarantine ./* d. Run the installer: ./install.sh e. Click **Verify** to confirm the agent installation. The **Launch** button is enabled. Go to step 6 on page 571.
	Perform the following steps to install the agent on Ubuntu: a. Click **Download** to download the agent installer. The `devops-test-runtime_12.0.0_amd64.deb` file is downloaded.

If...	Then...
	b. Navigate to the directory where the installer is downloaded: cd <download-directory> c. Run the installer: sudo apt install ./devops-test-runtime_12.0.0_amd64.deb d. Click **Verify** to confirm the agent installation. The **Launch** button is enabled. Go to step 6 on page 571.
	Perform the following steps to install the agent on Linux: a. Click **Download** to download the agent installer. The `devops-test-runtime-12.0.0-1.x86_64.rpm` file is downloaded. b. Navigate to the directory where the installer is downloaded: cd <download-directory> c. Run the following command: sudo INGRESS_DOMAIN=<domain_name> dnf localinstall ./devops-test-runtime-12.0.0-1.x86_64.rpm d. Switch to the **Test Editor** page and click **Verify** to confirm the agent installation. The **Launch** button is enabled. Go to step 6 on page 571. **Note:** After you click **Verify**, if the dialog indicates that the agent is not installed and displays the **Download** button, you must refresh the **Test Editor** page, and then click the  button to launch the agent.
The agent is installed, but not updated, and the screen displays the following options: ◦ Windows (64-bit .msi) ◦ Mac OS (64-bit .zip) ◦ Debian/Ubuntu (64-bit .deb) ◦ RHEL/Fedora/OpenSUSE (64-bit .rpm)	Perform the following steps to update the agent on Windows: a. Click **Upgrade**. The installation screen with the progress bar is displayed. The latest DevOps Test Runtime is installed and configured. b. Click **Proceed** to accept the download of the proprietary certificate if prompted. The **Launch** button is enabled. Go to step 6 on page 571.

If...	Then...
	Perform the following steps to install the agent on Mac: a. Click **Download** to download the agent installer. b. Uninstall the existing agent: rm -rf ~/Applications/devops-test-runtime.app c. Extract the downloaded `.zip` file: unzip devops-test-runtime.zip d. Quarantine the installer: xattr -dr com.apple.quarantine ./* e. Run the installer: ./install.sh f. Click **Verify** to confirm the agent installation. The **Launch** button is enabled. Go to step 6 on page 571.
	Perform the following steps to install the agent on Ubuntu: a. Click **Download** to download the agent installer. b. Uninstall the existing agent: sudo apt remove devops-test-runtime c. Navigate to the directory where the installer is downloaded: cd <download-directory> d. Run the following command: sudo apt install ./devops-test-runtime_12.0.0_amd64.deb e. Click **Verify** to confirm the agent installation. The **Launch** button is enabled. Go to step 6 on page 571.
	Perform the following steps to install the agent on Linux: a. Click **Download** to download the agent installer. b. Uninstall the existing agent: sudo yum remove devops-test-runtime c. Navigate to the directory where the installer is downloaded: cd <download-directory> d. Run the installer:

If...	Then...
	sudo INGRESS_DOMAIN=<domain_name> dnf localinstall ./devops-test-runtime-12.0.0-1.x86_64.rpm e. Click **Verify** to confirm the agent installation. The **Launch** button is enabled. Go to step 6 on page 571.
The agent is installed.	The **Launch** button is displayed. Go to step 6 on page 571.

6. Click **Launch**.

   The personal agent button changes to  , and the tooltip displays the following message when hovered over: `Personal Agent Ready - Click to stop`. This message indicates that the agent is active.

Results

You have downloaded and installed a DevOps Test Runtime agent. The personal agent is listed and visible only to you when you navigate to **Infrastructure > Agents and Intercepts**, both in the initial team space and in the projects that you are associated with.

What to do next

You can run the Web UI tests or suites and view the results. See Running server test assets on page 808 and Running server suite assets on page 816.

Related information

Stopping or unregistering the personal agent on page 571

Creating a test by using existing steps of a test with the personal agent on page 226

Stopping or unregistering the personal agent

You can stop or unregister the personal agent when you no longer need the agent to run tests.

Before you begin

You must have added the personal agent in your team space, and it must be in the active state. See Installing and starting the personal agent on page 567.

Perform the steps by using one of the following two methods:

- Stop the agent through the **Test Editor** page.
  a. Log in to Test Hub on the system that you want to register as an agent.

     The **Projects** page of the initial team space is displayed.

b. Click **My projects > project_name** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.

c. Click **Author > Test Editor**.

   The **Test Editor** page and the test navigator panel are displayed.

d. Click the  button located in the page title bar next to the branch selector.

   The **Stop DevOps Test Runtime** dialog opens.

e. Click **Yes** to stop the agent.

   The personal agent button changes to  , and the tooltip displays the following message when hovered over:

   `Personal Agent Inactive - Click to launch`. This message indicates that the agent is inactive.

- Unregister the agent through the **Agents and Intercepts** page.
  a. Open the **Initial Team Space** page and then navigate to the **Infrastructure > Agents and Intercepts** page.
  b. View the personal agent that is listed with the personal:username name.

     If the list is long, you can search for the agent by entering the name of the agent in the **Search** field.

     Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

  c. Identify the agent, and then click the **Remove agent** icon  .

     The **Unregister DevOps Test Runtime** dialog opens.

  d. Click **Unregister** to remove the agent.

Results

You have stopped or unregistered the personal agent. The agent will not be available as a location to run tests till you start it again.

Management of remote Kubernetes clusters

You can find information about the tasks that you can perform on remote Kubernetes clusters. After you set up a remote Kubernetes cluster, you can register the external cluster with a team space on HCL DevOps Test Hub (Test Hub). You must then add the remote cluster to a project before you run test assets or resources on the remote Kubernetes cluster.

You can find the following information about managing the external Kubernetes cluster:

- Considerations for using a remote Kubernetes cluster on page 573
- Registering a remote Kubernetes cluster on page 576
- Viewing remote Kubernetes clusters on page 578
- Adding a remote Kubernetes cluster to a project on page 580
- Editing configurations of a remote Kubernetes cluster on page 582
- Removing a remote Kubernetes cluster from a project on page 584
- Unregistering a remote Kubernetes cluster from a team space on page 586

Considerations for using a remote Kubernetes cluster

When you want to use a remote Kubernetes cluster as a location to run the supported tests or test resources, you must read the considerations first.

Before you want to run any of the supported tests or test resources on HCL DevOps Test Hub (Test Hub) that are configured to run on a remote Kubernetes cluster, you must have completed the following tasks:

- Verified that the Kubernetes cluster that you want to use is a supported one by referring to the OS, platform, and containers that are specified for Test Hub in System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94.
- Kept a suitable `kubeconfig` file with access to the remote Kubernetes cluster in a folder on the computer that you want to use to access an instance of Test Hub.

You must be familiar with the administration of a Kubernetes cluster. When you as the cluster administrator, set up the remote Kubernetes cluster must consider the following requirements:

- You must have a `kubeconfig` file with access to the remote Kubernetes cluster in a folder on the computer that you want to use to access an instance of Test Hub.

  Note: If the `kubeconfig` file references certificate files by using filepaths then the certificate data must be embedded in the `kubeconfig` file to make it usable by Test Hub. If the `kubeconfig` file contains multiple contexts, then only the current context is required.

  You can use the following kubectl command to create a suitable `kubeconfig` file called `portable-config` in the current folder. The `portable-config` file contains embedded certificate data and the information used by the current context:

  ```
  kubectl config view --flatten --minify > portable-config
  ```

- You must ensure that the remote Kubernetes cluster is able to access the Test Hub image registry which is the same registry that you used to install Test Hub. There is no need to copy the Test Hub image to the remote Kubernetes cluster but the remote cluster must be able to access the image registry at test runtime.

  If the remote cluster that you want to use is a Lightweight Kubernetes (k3s) cluster, it might require a *Secret* to pull the Test Hub image. You must run the following command on the remote k3s cluster:

  ```
  kubectl create secret docker-registry hclcr.io \
  -n {remote-namespace} \
  --docker-server=hclcr.io \
  ```

```
--docker-username={okta-email-address} \
--docker-password={harbor-cli-secret} \
--docker-email={example@abc.com}
```

Notes:

- You must replace *{remote-namespace}* with the value of the *namespace* that you have used for the remote cluster.
- You must replace *{okta-email-address}* with the username of the Harbor repository and replace *{harbor-cli-secret}* with the secret key that you copied from the Harbor repository.
- If the username contains any special characters, such as *$*, you must enclose it within single quotation-marks.
- You can replace *{example@abc.com}* with the email address of the administrator, if required.

- You need not configure separate licenses for running on the remote cluster because the licenses are consumed from the pool of available licenses that are configured for the team space on Test Hub. The licenses are returned to the pool when the test run is completed on the remote cluster.
- You must ensure that the Test Hub domain is accessible from the remote cluster so that the pod in the cluster is able to communicate with Test Hub.
- You must ensure that the appropriate **PriorityClass** exists on the remote cluster if you want to control the priority given to the remote execution pod by the scheduler. The same **PriorityClass** that is used by the Test Hub execution pod is used by the remote execution pod. If the named **PriorityClass** does not exist in the remote cluster, then no **PriorityClass** is specified by the remote execution pod.
- You must create an HTTP or an HTTPS Emissary listener when you want to run virtual services on the remote cluster. You must create both listeners when you want to run virtual services that use both HTTP and HTTPS transports. For information about creating listeners, refer to https://www.getambassador.io/docs/emissary/latest/topics/running/listener. The Emissary listener associates with the hosts created by Test Hub in the target namespace for executions that require Ingress. For example, when you want to run virtual services on the remote cluster and run tests from the remote cluster that use static agents.
- You can enter the code snippets for *nodeSelectors* and *tolerations* in a `yaml` format, if you want to use *nodeSelectors* and *tolerations* on the remote cluster. You can enter these in the fields provided in the **Advanced** configuration dialog when you register a remote cluster with your team space on Test Hub.

  For more information about *nodeSelector*, see nodeSelector and for *tolerations*, see Tolerations.

Restrictions: The following restrictions apply when you want to run the supported tests on a remote Kubernetes cluster:

- Using the **UserLibs** that are required for running certain API Suites are not supported on the remote cluster.
- Running of Compound Tests that contain performance tests, which require the **NodePort** access to connect to static agents, are not supported on the remote cluster.

When you plan to run any of the test assets or resources on a remote Kubernetes cluster from a project on Test Hub, you must complete the following tasks:

	Tasks	More information...
1	Register a remote Kubernetes cluster with a team space on Test Hub.	Registering a remote Kubernetes cluster on page 576
2	View remote Kubernetes clusters added to a team space.	Viewing remote Kubernetes clusters on page 578
3	Add a remote Kubernetes cluster to a project in a team space.	Adding a remote Kubernetes cluster to a project on page 580
4.1	Configure a run for your test that you want to run on the remote Kubernetes cluster by performing the following actions: 1. Check for the prerequisite tasks that you must perform before you configure a test run for the test. 2. Select the task for the test that you want to configure for a run, and then follow the steps to run the test.	1. Test run configurations on page 610 2. Prerequisites to running tests on page 527 3. Test run configurations on page 610
4.2	Configure a run of the virtual service resources that you want to run on the remote Kubernetes cluster by performing the following actions: 1. Check for the prerequisite tasks that you must perform before you configure a run of the virtual services. 2. Select the virtual service resources in your project that you want to configure for a run, and then follow the steps to run them.	1. Prerequisites for running HTTP virtual services on page 938 2. Configuring a run of a virtual service on page 946

Related information

Test run configurations on page 610

Management of remote Kubernetes clusters on page 572

Registering a remote Kubernetes cluster

You must register a remote Kubernetes cluster to your team space on HCL DevOps Test Hub (Test Hub) before you add the remote cluster to your project.

Before you begin

You must have completed the following tasks:

- Verified that the Kubernetes cluster that you want to use is a supported one by referring to the OS, platform, and containers that are specified for Test Hub in System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94.
- Kept a suitable `kubeconfig` file with access to the remote Kubernetes cluster in a folder on the computer that you want to use to access an instance of Test Hub.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane of the team space.

   The **Agents and Intercepts** page is displayed.
3. Click **Add new > Add new cluster**.

   The **New cluster** dialog is displayed.
4. Perform the following steps to add the `kubeconfig` file:
   a. Click **Browse**, and locate the `kubeconfig` file of the Kubernetes cluster that you want to add.
   b. Select the `kubeconfig` file, and then click **Open**.

      The name of the `kubeconfig` file that you selected is displayed along with the address of the Kubernetes cluster that is specified in the file.
   c. Click **Next**.

   The address of the Kubernetes cluster that is configured in the `kubeconfig` file is displayed in the **Address** field.

Note: You can change the address of the Kubernetes cluster if it is different from the address in the `kubeconfig` file.

5. Perform the following steps to add a namespace and name for the cluster:
    a. Enter the namespace of the cluster in the **Namespace** field.
    b. Enter a name for the cluster in the **Name** field.

       The name helps you to identify the cluster.
    c. Click **Next**.
6. Perform the following steps to add the ingress details of the cluster:
    a. Select the ingress type of the cluster from the **Ingress type** list.

       You can select the ingress type from the following options that are displayed:

Option	Description	Result
None	When selected specifies that no ingress type is configured for the remote Kubernetes cluster.	You can run any of the supported tests on the remote cluster but not on the remote agents.
NGINX	When selected specifies that NGINX is configured as the ingress type for the remote Kubernetes cluster.	You can run any of the supported tests on the remote cluster or on any of the remote agents.

    b. Enter the ingress host pattern of the cluster in the **Ingress host pattern** field.

       The DNS used by a computer that runs a client application (such as a static agent) or the HTTP proxy that routes traffic to the virtual service needs to resolve such hostnames to the IP address of the ingress domain.

       For example, if the IP of the remote cluster is 12.12.12.123, then you must enter the ingress host pattern as **.12.12.12.123.nip.io*

       The *Unique ID* of the virtual service is prefixed to the ingress hostname when virtual services are routed from the Test Hub cluster.
    c. Enter the name of the secret in the remote namespace, which contains the certificate information.

       You must have created the secret and named it following the naming conventions per the Kubernetes guidelines. The certificate that you create is a wildcard certificate which contains the Ingress host pattern of the Kubernetes cluster. This wildcard certificate ensures that the client applications can trust the virtual services.

Note: If a secret name is not provided then the default certificate that is provided by NGINX is used, which might cause issues to authenticate the virtual service by the client applications.

d. Click **Next**.

7. Optionally, perform the following steps to enter the code snippets for *nodeSelectors* and *tolerations*, if applicable for the remote cluster:

a. Enter the code snippet for *nodeSelectors* in a `yaml` format in the **Node selectors** field.

b. Enter the code snippet for *tolerations* in a `yaml` format in the **Tolerations** field.

c. Select the **Copy pull secret to the cluster** checkbox, if it is not already selected.

Note: The remote Kubernetes cluster pulls the secret to access the Test Hub image registry which is the same registry that you used to install Test Hub. The remote Kubernetes cluster must be able to access the image registry at test runtime.

d. Click **Register**.

The remote cluster is added to the team space and is displayed with the icon in the **Type** column on the **Agents and Intercepts** page. You can identify the cluster with its name in the **Agents** column.

Note: The **Projects** column does not display any projects that are associated with the cluster yet.

Results

You have registered a remote Kubernetes cluster with your team space on Test Hub.

What to do next

You must add the remote cluster to your project in your team space before you use the remote cluster as a location to run test assets or resources in your project. See Adding a remote Kubernetes cluster to a project on page 580.

Related information

Management of remote Kubernetes clusters on page 572

Viewing remote Kubernetes clusters

You can view the clusters that you have registered with a team space on HCL DevOps Test Hub (Test Hub) to view their details or edit their configurations.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Registered a remote Kubernetes cluster with your team space on Test Hub. See Registering a remote Kubernetes cluster on page 576.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane of the team space.

   The **Agents and Intercepts** page is displayed.

   You can view the agents, intercepts, or clusters that are registered with the team space on Test Hub.
3. You can view the clusters that you own in any of the following ways:

   - Search for the cluster by entering the name of the cluster in the **Search** field.

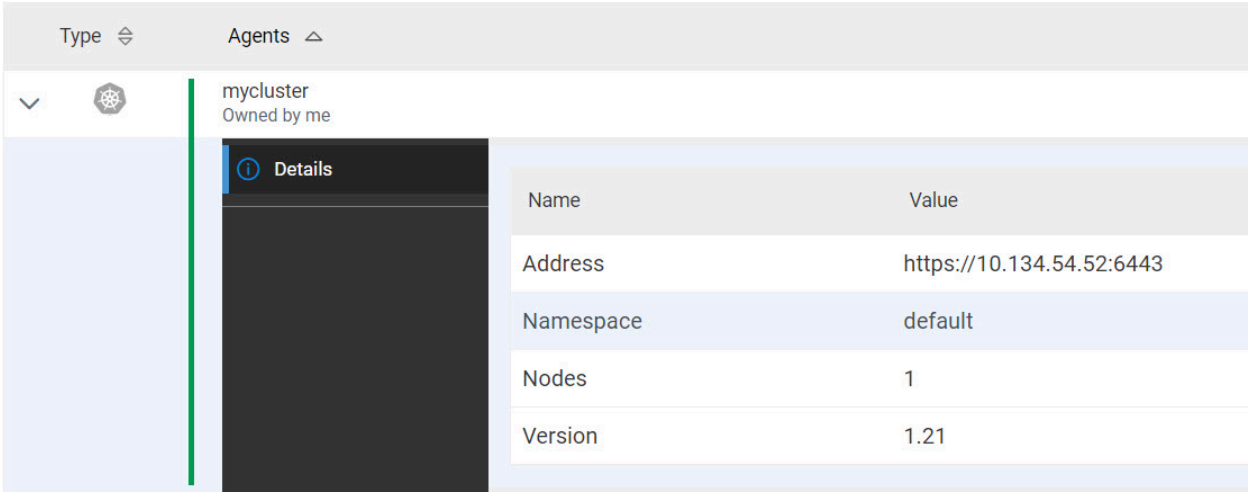
     Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

   - Sort the **Type** column to sort the items, and then identify the cluster by the name or owner.
   - Sort the **Agents** column to sort the items, and then identify the cluster by the name or owner.

   You can view the following details about the clusters that are in the team space:
   - The projects in the team space to which the cluster is added are displayed in the **Projects** column.
   - The status of the cluster is displayed in the **Status** column.

     Note: You can add clusters that are in the `Ready` state to your project, and then use them as locations to run tests.

   You can view the details of the cluster by clicking the **Expand** icon ›.



The following table describes the details about the cluster:

Name	Description
Address	Specifies the address of the remote Kubernetes API.
Namespace	Specifies the *namespace* of the remote Kubernetes cluster in which the test runs occur.
Nodes	Specifies the number of nodes in the remote Kubernetes cluster.
Version	Specifies the Kubernetes version of the remote Kubernetes cluster.

Results

You have viewed details of the remote clusters that are registered with a team space on Test Hub.

What to do next

You must add the remote cluster to your project in your team space before you can use the remote cluster as a location to run test assets or resources that are in your project. See Adding a remote Kubernetes cluster to a project on page 580.

Related information

Management of remote Kubernetes clusters on page 572

Adding a remote Kubernetes cluster to a project

You must add a remote Kubernetes cluster to your project after you register the cluster with your team space on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Registered a remote Kubernetes cluster with your team space on Test Hub. See Registering a remote Kubernetes cluster on page 576.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Been assigned a role as an *Owner* in the project.
- Created a project.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open your project.
   The **Overview** page is displayed.
3. Click **Infrastructure > Agents and Intercepts** in the navigation pane of the team space.

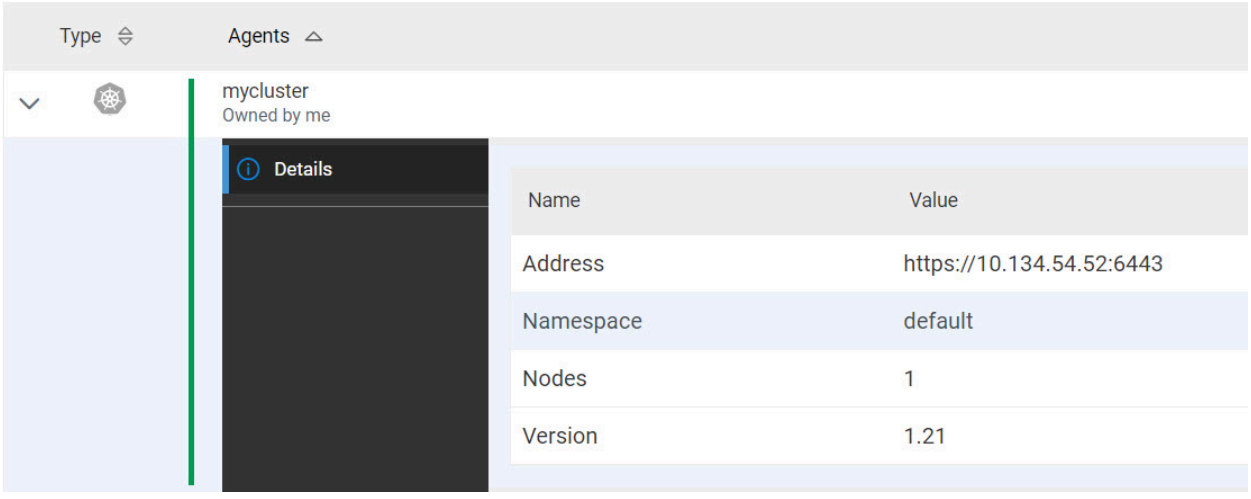
   The **Agents and Intercepts** page is displayed.
4. Click **Add > Add cluster**.

   The **Add cluster** dialog is displayed.
5. Select the cluster that you want to add to the project from the list of clusters that are registered with the team space, and then click **Add**.

   Note: You can add the clusters that are registered with the team space. You can add any number of clusters to your project.

   The clusters that you added to the project are displayed.

   You can view the details of the cluster by clicking the **Expand** icon ›.



The following table describes the details about the cluster:

Name	Description
Address	Specifies the address of the remote Kubernetes API.
Namespace	Specifies the *namespace* of the remote Kubernetes cluster in which the test runs occur.
Nodes	Specifies the number of nodes in the remote Kubernetes cluster.
Version	Specifies the Kubernetes version of the remote Kubernetes cluster.

What to do next

You can select any of the remote clusters as an alternate location to run the test assets or resources that are in your project while you configure a run from the **Execution** page.

Related information

Test run configurations on page 610

Management of remote Kubernetes clusters on page 572

Editing configurations of a remote Kubernetes cluster

You can edit the `kubeconfig` file, address, namespace, name, or ingress details of the remote Kubernetes cluster that you added to a team space on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Registered a remote Kubernetes cluster with your team space on Test Hub. See Registering a remote Kubernetes cluster on page 576.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane of the team space.

   The **Agents and Intercepts** page is displayed.

   You can view the agents, intercepts, or clusters that are registered with the team space on Test Hub.
3. You can view the clusters that you own in any of the following ways:

- Search for the cluster by entering the name of the cluster in the **Search** field.

  Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

- Sort the **Type** column to sort the items, and then identify the cluster by the name or owner.
- Sort the **Agents** column to sort the items, and then identify the cluster by the name or owner.

You can view the following details about the clusters that are in the team space:

- The projects in the team space to which the cluster is added, are displayed in the **Projects** column.
- The status of the cluster is displayed in the **Status** column.

  Note: You can add clusters that are in the `Ready` state to your project, and then use them as locations to run tests.

4. Identify the cluster that you want to modify the configurations, and then click the **Edit** icon .

   Note: The icon is displayed in the **Actions** column in the row of the cluster only if you own the cluster.

   The **Update cluster** dialog is displayed.

5. Perform the following steps to change the `kubeconfig` file:

   a. Click **Browse**, and locate the `kubeconfig` file of the Kubernetes cluster that you want to use.

   b. Select the `kubeconfig` file, and then click **Open**.

      The name of the `kubeconfig` file that you selected is displayed along with the address of the Kubernetes cluster that is specified in the file.

   c. Click **Next**.

6. Perform the following steps to change the namespace or name of the cluster:
   a. Edit or change the namespace of the cluster that you want to use, in the **Namespace** field.
   b. Edit or change the name of the cluster in the **Name** field.
   c. Click **Next**.
7. Perform the following steps to change the ingress details of the cluster:

   a. Select the ingress type of the cluster from the **Ingress type** list, if you want to use a different cluster.

   b. Enter or change the ingress host pattern of the cluster in the **Ingress host pattern** field, if you did not provide it when setting up the remote cluster or change it to match the cluster you want to use.

   c. Enter the secret if you did not provide it when setting up the remote cluster.

The secret is used to authenticate the remote cluster to use the licenses from the Team space License Server so that the inbound traffic is allowed into the ingress domain.

Note: The secret to use must be available with your Test Hub server administrator.

d. Click **Next**.

8. Optionally, perform the following steps to edit or change the code snippets for *nodeSelectors* and *tolerations*, if applicable for the remote cluster:
   a. Enter the code snippet for *nodeSelectors* in a `yaml` format in the **Node selectors** field.
   b. Enter the code snippet for *tolerations* in a `yaml` format in the **Tolerations** field.
9. Click **Update**.

The remote cluster is displayed with the updated configurations.

Note: The projects that are associated with the cluster are displayed in the **Projects** column and you need not add the cluster to the same project again.

Results

You have updated an existing remote cluster with the changed configurations.

What to do next

You can select the updated cluster as an alternate location to run the test assets or resources that are in your project while you configure a run from the **Execution** page.

Related information

Management of remote Kubernetes clusters on page 572

Removing a remote Kubernetes cluster from a project

You can remove a remote Kubernetes cluster from a project in a team space on HCL DevOps Test Hub (Test Hub) when you no longer need the cluster to run tests.

Before you begin

You must have completed the following tasks:

1. Registered a remote Kubernetes cluster with your team space on Test Hub. See Registering a remote Kubernetes cluster on page 576.
2. Added the Kubernetes cluster to a project in your team space. See Adding a remote Kubernetes cluster to a project on page 580.

Note: You can remove remote Kubernetes clusters that only you added to the project.

3. Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
4. Been assigned a role as an *Owner* in the project.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project in your team space.

   The **Overview** page is displayed.
3. Click **Infrastructure > Agents and Intercepts** in the navigation pane of your team space.

   The **Agents and Intercepts** page is displayed.

   You can view the agents, intercepts, or clusters that are added to your project.
4. You can view the clusters that you own in any of the following ways:
   - Search for the cluster by entering the name of the cluster in the **Search** field.

     Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

   - Sort the **Type** column to sort the items, and then identify the cluster by the name or owner.
   - Sort the **Agents** column to sort the items, and then identify the cluster by the name or owner.
5. Identify the cluster that you want to remove from the project, and then click the **Remove cluster** icon .

   Note: The icon is displayed in the **Actions** column in the row of the cluster only if you own the remote cluster.

   A message is displayed that the remote cluster is removed from the project successfully.

Results

You have removed the remote cluster from your project. The remote cluster is no longer available as a location when you configure a run of the test assets or resources that are in your project.

What to do next

You might have to add a remote cluster to your project to use the remote cluster as a location to run test assets or resources.

Related information

Management of remote Kubernetes clusters on page 572

Unregistering a remote Kubernetes cluster from a team space on page 586

Editing configurations of a remote Kubernetes cluster on page 582

Unregistering a remote Kubernetes cluster from a team space

You can unregister a remote Kubernetes cluster from a team space in HCL DevOps Test Hub (Test Hub) when you no longer require it.

Before you begin

You must have completed the following tasks:

1. Registered a remote Kubernetes cluster with your team space on Test Hub. See Registering a remote Kubernetes cluster on page 576.
2. Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
3. Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane of the team space.

   The **Agents and Intercepts** page is displayed.

   You can view the agents, intercepts, or clusters that are registered with the team space on Test Hub.
3. You can view the clusters that you own in any of the following ways:
   - Search for the cluster by entering the name of the cluster in the **Search** field.

     Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

   - Sort the **Type** column to sort the items, and then identify the cluster by the name or owner.
   - Sort the **Agents** column to sort the items, and then identify the cluster by the name or owner.
4. Identify the cluster that you want to remove.

Note: The and icons are displayed in the **Actions** column in the row of the cluster only if you own the remote cluster.

5. Unregister the cluster by completing the following steps:
    a. Click the **Unregister cluster** icon .
    b. Click **Unregister** in the **Unregister cluster** dialog box.

A message is displayed that the remote cluster is unregistered successfully.

Results

You have unregistered the remote cluster from a team space on Test Hub. You cannot run test assets or resources on this remote cluster if it is added to your project. You might have to add it again to your team space, if you want to use the remote cluster in your projects.

Related information

Management of remote Kubernetes clusters on page 572

Editing configurations of a remote Kubernetes cluster on page 582

Adding a remote Kubernetes cluster to a project on page 580

Removing a remote Kubernetes cluster from a project on page 584

Management of cloud credentials

You can find information about managing the credentials that are required to connect to cloud-based applications such as mobile device clouds and AI providers from HCL DevOps Test Hub (Test Hub).

Contents

- Mobile device clouds on page 588
- Mobile device clouds and Test Hub on page 588
- Mobile device clouds that are supported on page 588
- Details required of mobile device clouds on page 588
- Prerequisites for using mobile devices on mobile device clouds on page 589
- Overview of tasks to manage mobile device clouds on page 590
- AI providers on page 590
- Details required of AI providers on page 591

Mobile device clouds

Mobile device clouds are cloud computing services offered by cloud service providers who host real mobile devices in a cloud. You can purchase plans from the service providers to use the mobile devices in the cloud to test your mobile apps. The cloud service providers also offer different browsers in the cloud to test web applications.

To use the cloud services, you must subscribe to their services that suit your testing requirements.

Mobile device clouds and Test Hub

If you already use mobile device clouds to test your mobile apps, testers in your organization must be required to set up the connection to the mobile device clouds individually. They use the same credentials that you subscribed to as an organization. This can be a potential risk of exposure of your subscription details and without proper monitoring can lead to misuse of the credentials.

You can use Test Hub to connect to mobile device clouds. You must first configure a cloud credential by using the details of your cloud subscription account.

You can then add testers from your organization as authorized users who can use the credential to connect to mobile device clouds. Such members need not enter the subscription details to connect to the cloud. Only the authorized users of the credential can run their tests on the devices that are connected to the mobile device cloud. With this mechanism, you can prevent the exposure of the credentials of your subscription to testers and also prevent testers from creating separate connections to the same mobile device cloud.

Mobile device clouds that are supported

Test Hub supports connections to the following mobile device clouds:

- BitBar Cloud
- BrowserStack Cloud
- LambdaTest Cloud
- pCloudy Cloud
- Perfecto Mobile Cloud

Details required of mobile device clouds

You must ensure that you have the details and valid credentials of your subscription to the mobile device cloud that you want to connect and use. See the following table for the details required for the different mobile device clouds:

Attribute	**Description**	**Mobile device cloud**				
		BitBar Cloud	**BrowserStack Cloud**	**LambdaTest Cloud**	**pCloudy Cloud**	**Perfecto Mobile Cloud**
Host	The hostname of the mobile device cloud instance.	Required	Required	Required	Required	Required

Attribute	Description	Mobile device cloud				
		BitBar Cloud	**BrowserStack Cloud**	**LambdaTest Cloud**	**pCloudy Cloud**	**Perfecto Mobile Cloud**
API key	The API key of your cloud account to authenticate the connection.	Required	Required	Required	Required	Not applicable
Security token	The token used to authenticate the connection.	Not applicable	Not applicable	Not applicable	Not applicable	Required
Project	The project name to use on the mobile device cloud.	Required	Required	Required	Not applicable	Not applicable
Username	The username of your cloud account.	Not applicable	Required	Required	Required	Not applicable

Prerequisites for using mobile devices on mobile device clouds

When you want to use the available mobile devices that are connected to a mobile device cloud, you must ensure that you have completed certain tasks. Refer to the following table for details of the tasks that must be completed before tests can be run on the mobile devices connected to mobile device clouds:

Device cloud	Testing on Android OS	Testing on iOS
BitBar Cloud BrowserStack Cloud Perfecto Mobile Cloud	• Uploaded the `.apk` file of the Android application that you want to test. • Installed the Android application that you want to test on the Android device.	• Installed the `.ipa` or `.app` file of the iOS application that you want to test. • Installed the iOS application that you want to test on the iOS device.
pCloudy Cloud LambdaTest Cloud	• Uploaded the `.apk` file of the Android application that you want to test. • Installed the Android application that you want to test on the Android device. • Verified that the device you want to test is available and in the *Released* state on the cloud platform.	• Uploaded the `.ipa` file of the iOS application that you want to test. • Installed the iOS application that you want to test on the iOS device. • Verified that the device you want to test is available and in the *Released* state on the cloud platform.

Overview of tasks to manage mobile device clouds

You as a *Team Space Owner* can perform the following tasks:

- Configure the connection to mobile device clouds. You must create a *cloud credential* in the team space that represents a connection to a mobile device cloud. You must use the details of your mobile device cloud subscription account to configure the cloud credential to the mobile device cloud. See Adding cloud credentials of mobile device clouds on page 593.
- Add your team space members to access the mobile device cloud. The added members become the authorized users of the cloud credential, and then can connect to the mobile device cloud from Test Hub. See Adding authorized users to a cloud credential on page 600.
- View the details of the cloud credential that you added to the team space. See Viewing details of cloud credentials on page 601.
- Remove the authorized users of the cloud credential, when you want to reduce the number of users accessing the mobile device cloud. See Removing authorized users from a cloud credential on page 608.
- Edit an existing cloud credential that you configured to modify the name of the credential, project, or username of your cloud subscription account. See Editing details of cloud credentials on page 605.
- Delete an existing cloud credential that you configured, from the team space. You might want to delete the credential when your cloud subscription to the mobile device cloud has expired and you no longer need the connection. See Deleting an existing cloud credential on page 609.
- Export details of all existing cloud credentials that are configured in your team space, when you might want to use the details elsewhere. You can export details such as the name of the credentials, the credential type, and the email ID of authorized users of the credentials. See Viewing details of cloud credentials on page 601.

You as a *Project Owner* or *Tester* can perform the following tasks on cloud credentials or the mobile device clouds:

- Ensure that you are added as an authorized user of a cloud credential in the team space. See Viewing details of cloud credentials on page 601.
- Configure the following mobile tests to run on the mobile devices that are connected to mobile device clouds:
  - Configuring an AFT Suite to run on a mobile device by using a cloud credential on page 644.
  - Configuring a run of a Compound Test that contains mobile tests to run by using a cloud credential on page 693.

AI providers

AI providers are the organizations that build, develop, and offer AI technologies. They offer models and tools that power artificial intelligence in everything from virtual assistants to complex data analysis systems. You can purchase plans from the AI providers to use the services in the cloud or locally to generate Web UI tests. For the cloud-based provider such as OpenAI, you must subscribe to their services. For the locally hosted provider such as Ollama, you must download the software and set up a server.

Test Hub supports the following AI providers:

- OpenAI
- Ollama
- IBM watsonx
- Anthropic
- GitHub Copilot

Restriction: Only one test generation session through GitHub Copilot can be active at a time across the entire Test Hub server. Any additional Copilot requests are queued and wait until the current session completes.

Details required of AI providers

You must ensure that you have the details and valid credentials of the AI account that you want to connect and use. The following table lists the details required for the different accounts:

Attribute	Description	AI Agents				
		OpenAI	**Ollama**	**IBM watsonx**	**Anthropic**	**GitHub Copilot**
API End Point	The URL of the AI provider.	Optional	Required	Required	Optional	Not applicable
API key	The API key of the AI provider account to authenticate the connection.	Required	Optional	Required	Required	Not applicable
GitHub token	GitHub fine-grained Personal Access Token (PAT) configured to authenticate GitHub Copilot CLI commands. You must select **Copilot Requests** under **Permissions > Add permissions** while configuring the token. For more information on generating the token, refer to Creating a fine-grained personal access token.	Not applicable	Not applicable	Not applicable	Not applicable	Required
Project Id	A unique identifier assigned to the project that you have created.	Not applicable	Not applicable	Required	Not applicable	Not applicable

Attribute	Description	AI Agents				
		OpenAI	**Ollama**	**IBM watsonx**	**Anthropic**	**GitHub Copilot**
Model	The name of the large language model (LLM) that you want to use for test generation.	Not applicable	Optional	Required	Optional	Not applicable
Advanced						
Max tokens per session	The maximum cumulative number of tokens allowed during a single active conversation session. This limit includes tokens of both user prompts and generated responses.	Optional	Optional	Optional	Optional	Not applicable
Max interactions per session	The maximum number of user queries paired with bot responses permitted within a single session. This limit safeguards against infinite user loops, bot confusion, or prolonged conversations that degrade performance. When the limit is reached, the session closes or resets.	Optional	Optional	Optional	Optional	Not applicable
Max duration per session (in seconds)	The total time limit in seconds that a session remains active from start to finish before it expires or times out. This limit helps manage system memory and resource allocation by clearing inactive or overly extended user sessions.	Optional	Optional	Optional	Optional	Not applicable

Adding cloud credentials of mobile device clouds

You as a *Team Space Owner* can add a cloud credential in the team space by using the details of your mobile device cloud subscription. This cloud credential represents a connection between HCL DevOps Test Hub (Test Hub) and a mobile device cloud.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you have the details of your subscription and valid credentials of the mobile device cloud to which you want to connect. See Details required of mobile device clouds on page 588.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Cloud Credentials** on the navigation pane.

   The **Cloud Credentials** page is displayed.
3. Click **Add cloud credential**.

   The **Details** tab of the **Add cloud credential** dialog is displayed.
4. Perform the following steps in the **Details** tab:

   a. Enter a name in the **Name** field so that you can identify the mobile device cloud, which is associated with a specific subscription account.

   b. Select the mobile device cloud from the **Cloud Authentication** list.

   c. Click **Next**.

      The **Authentication** tab of the **Add cloud credential** dialog is displayed.
5. Enter the relevant details of your subscription account that you have in the mobile device cloud, in the **Authentication** tab.

   You can find information about the details that are required for each of the supported mobile device clouds and the action you need to perform for the required fields in the following table:

Field	Mobile device cloud					Action
	BitBar Cloud	**BrowserStack Cloud**	**LambdaTest Cloud**	**pCloudy Cloud**	**Perfecto Mobile Cloud**	
Host	Required	Required	Required	Required	Required	Enter the hostname of the mobile device cloud instance.
API key	Required	Required	Required	Required	Not applicable	Enter the API key of your cloud subscription account to authenticate the connection.
Security token	Not applicable	Not applicable	Not applicable	Not applicable	Required	Enter the token used to authenticate the connection.
Project	Required	Required	Required	Not applicable	Not applicable	Enter or select the project to use that is in the mobile device cloud.
Username	Not applicable	Required	Required	Required	Not applicable	Enter the username of your cloud subscription account.

6. Click **Add**.

The cloud credential for your specific subscription account with a mobile device cloud is displayed in a table on the **Cloud Credentials** page.

You can find the following details about the cloud credential on the **Cloud Credentials** page.

Column header	Details displayed
Provider	The name of the mobile device cloud service provider. You can view the Toggle icon associated with the service providers when you add a cloud credential. **Note:** If you are the owner of the cloud credential, you can switch to an On/Off position. However, when you are not the owner and you try to use the Toggle icon an error message is displayed.
Name	The name you entered for the cloud credential when you created the connection in the team space.

<table>
<tr><th>Column header</th><th>Details displayed</th></tr>
<tr><td>Credential Type</td><td>The type of cloud that is connected to Test Hub. Displays Device Cloud when you create a credential to connect to a mobile device cloud.</td></tr>
<tr><td>Members</td><td>The abbreviated names of the team space members who are authorized users of the cloud credential. The abbreviated name is the combination of the first letters of the first name and last name.</td></tr>
<tr><td>Actions</td><td>The icons for the different actions that you can perform on the credential. Refer to the following table for the icons and the actions:
<table>
<tr><th>Icon</th><th>Action</th><th>Available for</th></tr>
<tr><td>  </td><td>Click to view details of the credential of the cloud subscription account.</td><td>Team space owner and authorized user</td></tr>
<tr><td>  </td><td>Click to edit and update the credential for any of your cloud subscription details.</td><td>Team space owner</td></tr>
<tr><td>  </td><td>Click to remove the configured connection to the mobile device cloud account from the server.</td><td>Team space owner</td></tr>
<tr><td>  </td><td>Click to add team space members as authorized users of the credential.</td><td>Team space owner</td></tr>
</table></td></tr>
</table>

Results

You have added credentials of mobile device cloud accounts so that Test Hub can connect to the mobile device clouds.

What to do next

You can add team space members as authorized users of the credentials. See Adding authorized users to a cloud credential on page 600.

Related information

Management of cloud credentials on page 587

Adding cloud credentials of AI providers

You can add a cloud credential in the team space by using the details of your AI provider accounts so that Test Hub can connect to the providers.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you have the details of your subscription and valid credentials of the cloud account to which you want to connect. See Details required of AI providers on page 591.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Cloud Credentials** on the navigation pane.

   The **Cloud Credentials** page is displayed.
3. Click **Add cloud credential**.

   The **Details** tab of the **Add cloud credential** dialog is displayed.
4. Perform the following steps in the **Details** tab:
   a. Enter a name in the **Name** field so that you can identify the mobile device cloud, which is associated with a specific subscription account.
   b. Select an AI provider from the **Cloud Authentication** list.
   c. Click **Next**.

      The **Authentication** tab of the **Add cloud credential** dialog is displayed.
5. Enter the relevant details of your subscription account for AI providers in the **Authentication** tab.

   You can find information about the details that are required for each of the supported AI providers in the following table:

Attribute	**Description**	**AI Agents**				
		OpenAI	**Ollama**	**IBM watsonx**	**Anthropic**	**GitHub Copilot**
API End Point	The URL of the AI provider.	Optional	Required	Required	Optional	Not applicable
API key	The API key of the AI provider account	Required	Optional	Required	Required	Not applicable

Attribute	Description	AI Agents				
		OpenAI	**Ollama**	**IBM watsonx**	**Anthropic**	**GitHub Copilot**
	to authenticate the connection.					
GitHub token	GitHub fine-grained Personal Access Token (PAT) configured to authenticate GitHub Copilot CLI commands. You must select **Copilot Requests** under **Permissions > Add permissions** while configuring the token. For more information on generating the token, refer to the Related information section.	Not applicable	Not applicable	Not applicable	Not applicable	Required
Project Id	A unique identifier assigned to the project that you have created.	Not applicable	Not applicable	Required	Not applicable	Not applicable
Model	The name of the large language model (LLM) that you want to use for test generation.	Not applicable	Optional	Required	Optional	Not applicable
Advanced						
Max tokens per session	The maximum cumulative number of tokens allowed during a single active conversation session. This limit includes tokens of both user prompts and generated responses.	Optional	Optional	Optional	Optional	Not applicable
Max interactions per session	The maximum number of user queries paired with bot responses permitted	Optional	Optional	Optional	Optional	Not applicable

Attribute	Description	AI Agents				
		OpenAI	Ollama	IBM watsonx	Anthropic	GitHub Copilot
	within a single session. This limit safeguards against infinite user loops, bot confusion, or prolonged conversations that degrade performance. When the limit is reached, the session closes or resets.					
Max duration per session (in seconds)	The total time limit in seconds that a session remains active from start to finish before it expires or times out. This limit helps manage system memory and resource allocation by clearing inactive or overly extended user sessions.	Optional	Optional	Optional	Optional	Not applicable

6. Click **Add**.

The cloud credential for your specific subscription account is displayed in a table on the **Cloud Credentials** page.

You can find the following details about the cloud credentials on the **Cloud Credentials** page.

Column header	Details
Provider	The name of the AI provider. You can view the Toggle icon associated with the service providers when you add a cloud credential. **Note:** If you are the owner of the credential, you can switch to an On/Off position. However, when you are not the owner and you try to use the Toggle icon, an error message is displayed.
Name	The name you entered for the credential when you created the connection in the team space.

Column header	Details
Credential Type	The type of account that is connected to Test Hub. Displays *AI Provider* when you create a credential to connect to an AI account.
Members	The abbreviated names of the team space members who are authorized users of the cloud credential. The abbreviated name is the combination of the first letters of the first name and last name.
Actions	The icons for the different actions that you can perform on the credential. Refer to the following table for the icons and the actions:

Icon	Action	Available for
	Click to view details of the credential.	Team space owner and authorized user
	Click to edit and update the credential.	Team space owner
	Click to remove the configured connection to the AI provider account from the server.	Team space owner
	Click to add team space members as authorized users of the credential.	Team space owner

Results

You have added credentials of AI provider accounts in Test Hub.

What to do next

You can add team space members as authorized users of the credentials. See Adding authorized users to a cloud credential on page 600.

Related information

Management of cloud credentials on page 587

Creating a fine-grained personal access token

Adding authorized users to a cloud credential

After you add the cloud application account details to a cloud credential in your team space inHCL DevOps Test Hub (Test Hub), you can add other team space members as authorized users of this cloud credential. The authorized users can then use the connected cloud application account for tests.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Added the cloud credentials of your mobile device cloud or AI provider account to configure the connection between Test Hub and the cloud account. See Adding cloud credentials of mobile device clouds on page 593 and Adding cloud credentials of AI providers on page 596.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Cloud Credentials** on the navigation pane.

   The **Cloud Credentials** page is displayed with the cloud credentials that are added to the team space.
3. Identify the cloud credential by performing any of the following actions:
   - Scroll through the table of cloud credentials to identify the cloud credential with the name you provided.
   - Search for the cloud credential by entering the name of the credential in the **Search** field.
   - Sort the **Provider** column so that the names of the cloud providers are listed alphabetically in either increasing or decreasing order. Identify your cloud provider and look for the name of your credential.

   You identified the cloud credential that you want to use.
4. Perform the following steps to add team space members as authorized users:
   a. Click the **Add Members** icon in the row of the credential.

      The **Add New Member** dialog is displayed.

   b. Enter any text that is contained in the name or email ID of the team space member that you want to add in the field.

      The team space users are listed whose names or email ID matches the search text.

   c. Select the team space member.

      The selected member is added to the list of authorized users and is displayed under **Members with access**.

   d. Click **Close** to close the dialog.

The abbreviated name of the team space member is displayed under the **Members** column in the row of the cloud credential.

Note: The abbreviated name is the combination of the first letters of the first name and last name.

You can view the first name, last name, username, and the email ID of the added member when you hover over the abbreviated name.

Results

You as a *Team Space Owner* added team space members as authorized users of the cloud credential.

What to do next

You can inform the team space members that they are added as authorized users of the cloud credential. The authorized users can then use the credential to connect to the cloud account from Test Hub and run or generate tests. The authorized users of the credential can manage the following tasks:

- Configuring an AFT Suite to run on a mobile device by using a cloud credential on page 644.
- Configuring a run of a Compound Test that contains mobile tests to run by using a cloud credential on page 693.
- Configuring a run of a Compound Test that contains Web UI tests to run by using a cloud credential on page 732
- Configuring a run of a mobile test to run on a mobile device by using a cloud credential on page 771
- Configuring a run of a Web UI test to run by using a cloud credential on page 842
- Running server test assets by using a cloud credential on page 813
- Running server suite assets by using a cloud credential on page 819

Related information

Management of cloud credentials on page 587

Viewing details of cloud credentials

If you are a *Team Space Owner* in HCL DevOps Test Hub (Test Hub), you can view the details of the cloud credential when you want to decide your action for adding or removing team space members, changing your cloud subscription details, or deleting a credential. If you are a *Project Owner* or *Tester*, you can verify whether you are added as an authorized user to use the cloud credential.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space, a *Project Owner*, or a *Tester* in the project. See Managing members and their roles in a team space on page 1207.
- Added the cloud credentials of your mobile device cloud or AI provider account to configure the connection between Test Hub and the cloud account. See Adding cloud credentials of mobile device clouds on page 593 and Adding cloud credentials of AI providers on page 596.
- Added team space members as authorized users of the cloud credential. See Adding authorized users to a cloud credential on page 600.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Cloud Credentials** on the navigation pane.

   The **Cloud Credentials** page is displayed with the cloud credentials that are added to the team space.
3. Identify the cloud credential by performing any of the following actions:
   - Scroll through the table of cloud credentials to identify the cloud credential with the name you provided.
   - Search for the cloud credential by entering the name of the credential in the **Search** field.
   - Sort the **Provider** column so that the names of the cloud providers are listed alphabetically in either increasing or decreasing order. Identify your cloud provider and look for the name of your credential.

   You identified the cloud credential that you want to use.
4. Select the action that you want to perform:
   - Verify whether you are added as an authorized user of the cloud credential by the *Team Space Owner*, if you are a *Project Owner* or *Tester*, go to Step 5 on page 602.
   - View details of the cloud credential, if you are a *Team Space Owner*, *Project Owner*, or *Tester*, go to Step 6 on page 603.
   - Export the details of all the cloud credentials in your team space, if you are a *Team Space Owner*, go to Step 7 on page 604.
5. Perform the following steps to verify whether you are added as an authorized user of the cloud credential:
   a. Look for your abbreviated name under the **Members** column in the row of the identified cloud credential.

      Note: The abbreviated name is the combination of the first letters of the first name and last name.

   b. Hover over the abbreviated name to view the first name, last name, username, and the email ID to verify that they are your details.

      You have verified whether you are an authorized user of the cloud credential.

      Note: If you do not find your abbreviated name or details, you need to contact your team space owner to add you as an authorized user.

   c. Go to Results on page 604.

6. Click the **View** icon in the row of the credential that you want to view.

   The details of the cloud credential are displayed in the **View cloud credential** dialog.

   You can view the following details of the mobile device cloud or AI provider that was configured in the cloud credential:

Attribute	**Description**	**Mobile device cloud or AI provider details**									
		BitBar Cloud	**BrowserStack Cloud**	**LambdaTest Cloud**	**pCloudy Cloud**	**Perfecto Mobile Cloud**	**OpenAI**	**Ollama**	**IBM watsonx**	**Anthropic**	**GitHub Copilot**
Owner	The username and email ID of the team space owner who configured the cloud credential.	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Credential name	The name provided to the cloud credential by the team space owner when the credential was configured.	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Type	The name of the mobile device cloud or AI provider to which the connection was configured in the cloud credential.	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Host	The hostname or URL of the mobile device cloud instance.	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗
Project	The name of the project that you created in your cloud subscription account.	✓	✓	✗	✗	✗	✗	✗	✓	✗	✗
Username	The username of your cloud subscription account.	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗
API End Point *	The URL of the AI provider.	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗

Attribute	Description	Mobile device cloud or AI provider details									
		BitBar Cloud	BrowserStack Cloud	LambdaTest Cloud	pCloudy Cloud	Perfecto Mobile Cloud	OpenAI	Ollama	IBM watsonx	Anthropic	GitHub Copilot
Model	The name of large language model (LLM) that you want to use for test generation.	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗
API key	The API key of your cloud subscription account to authenticate the connection.	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓
Security token	The token used to authenticate the connection.	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗
Access to	The team space members that the team space owner added as authorized users of the cloud credential.	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

You can close the dialog by clicking **Close**.

7. Click the **Download CSV** option to export details of all existing cloud credentials that are configured in your team space when you want to use the details elsewhere, such as the name of the credentials, the credential type, and email ID of authorized users of the credentials members.

   You saved the details of all the credentials in your team space as a CSV file to your computer.

Results

You have completed the following tasks:

- If you are a *Project Owner* or *Tester*, you verified whether you are an authorized user of the cloud credential.
- If you are a *Team Space Owner*, *Project Owner*, or *Tester*, you have viewed the details of the cloud credential that you added.
- If you are a *Team Space Owner*, you saved the details of all cloud credentials that are in your team space to a file on your computer.

Related information

Management of cloud credentials on page 587

Editing details of cloud credentials

If any of the details of your cloud subscription account change, as a *Team Space Owner* in HCL DevOps Test Hub (Test Hub) you can view the details of the existing cloud credential and edit it to update the credential with the latest details.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Added the cloud credentials of your mobile device cloud or AI provider account to configure the connection between Test Hub and the cloud account. See Adding cloud credentials of mobile device clouds on page 593 and Adding cloud credentials of AI providers on page 596.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Cloud Credentials** on the navigation pane.

   The **Cloud Credentials** page is displayed with the cloud credentials that are added to the team space.
3. Identify the cloud credential by performing any of the following actions:
   - Scroll through the table of cloud credentials to identify the cloud credential with the name you provided.
   - Search for the cloud credential by entering the name of the credential in the **Search** field.
   - Sort the **Provider** column so that the names of the cloud providers are listed alphabetically in either increasing or decreasing order. Identify your cloud provider and look for the name of your credential.

   You identified the cloud credential that you want to use.
4. Click the **Edit cloud credential** icon  in the row of the credential that you want to edit.

   The **Update cloud credential** dialog is displayed.
5. Modify the name of the cloud credential in the **Details** tab, and then click **Next**.

   The **Authentication** tab of the **Update cloud credential** dialog is displayed.
6. Modify relevant fields of your subscription account if they have changed, in the **Authentication** tab.

   You can find information about the details that are required for the following account types:
   - Mobile device clouds: Click here on page 606 to refer to the table.
   - AI providers: Click here on page 606 to refer to the table.

   The following table lists details for the relevant and required fields for mobile device clouds.

Field	Mobile device cloud					Action
	BitBar Cloud	**BrowserStack Cloud**	**LambdaTest Cloud**	**pCloudy Cloud**	**Perfecto Mobile Cloud**	
Host	Required	Required	Required	Required	Required	Enter the hostname of the mobile device cloud instance.
API key	Required	Required	Required	Required	Not applicable	Enter the API key of your cloud subscription account to authenticate the connection.
Security token	Not applicable	Not applicable	Not applicable	Not applicable	Required	Enter the token used to authenticate the connection.
Project	Required	Required	Required	Not applicable	Not applicable	Enter or select the project to use that is in the mobile device cloud.
Username	Not applicable	Required	Required	Required	Not applicable	Enter the username of your cloud subscription account.

The following table lists details for the relevant and required fields for AI providers.

Attribute	Description	AI Agents				
		OpenAI	**Ollama**	**IBM watsonx**	**Anthropic**	**GitHub Copilot**
API End Point	The URL of the AI provider.	Optional	Required	Required	Optional	Not applicable
API key	The API key of the AI provider account to authenticate the connection.	Required	Optional	Required	Required	Not applicable
GitHub token	GitHub fine-grained Personal Access Token (PAT) configured to authenticate GitHub Copilot CLI commands. You must select **Copilot Requests** under **Permissions > Add permissions** while configuring the token.	Not applicable	Not applicable	Not applicable	Not applicable	Required

Attribute	Description	AI Agents				
		OpenAI	**Ollama**	**IBM watsonx**	**Anthropic**	**GitHub Copilot**
	For more information on generating the token, refer to the Related information section.					
Project Id	A unique identifier assigned to the project that you have created.	Not applicable	Not applicable	Required	Not applicable	Not applicable
Model	The name of the large language model (LLM) that you want to use for test generation.	Not applicable	Optional	Required	Optional	Not applicable
Advanced						
Max tokens per session	The maximum cumulative number of tokens allowed during a single active conversation session. This limit includes tokens of both user prompts and generated responses.	Optional	Optional	Optional	Optional	Not applicable
Max interactions per session	The maximum number of user queries paired with bot responses permitted within a single session. This limit safeguards against infinite user loops, bot confusion, or prolonged conversations that degrade performance. When the limit is reached, the session closes or resets.	Optional	Optional	Optional	Optional	Not applicable
Max duration per session (in seconds)	The total time limit in seconds that a session remains active from start to finish before it expires or times out. This limit helps	Optional	Optional	Optional	Optional	Not applicable

Attribute	Description	AI Agents				
		OpenAI	Ollama	IBM watsonx	Anthropic	GitHub Copilot
	manage system memory and resource allocation by clearing inactive or overly extended user sessions.					

7. Click **Update**.

   The existing cloud credential is updated with the changed details of your account.

Results

You edited an existing cloud credential in your team space and updated it with the changed details of your cloud subscription account.

Related information

Management of cloud credentials on page 587

Removing authorized users from a cloud credential

When you want to restrict the number of users that are using your cloud credential due to any change in your subscription fee or for other valid reasons, you might want to remove some of the authorized users of the cloud credential in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Added the cloud credentials of your mobile device cloud or AI provider account to configure the connection between Test Hub and the cloud account. See Adding cloud credentials of mobile device clouds on page 593 and Adding cloud credentials of AI providers on page 596.
- Added team space members as authorized users of the cloud credential. See Adding authorized users to a cloud credential on page 600.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Cloud Credentials** on the navigation pane.

   The **Cloud Credentials** page is displayed with the cloud credentials that are added to the team space.

3. Identify the cloud credential by performing any of the following actions:
   - Scroll through the table of cloud credentials to identify the cloud credential with the name you provided.
   - Search for the cloud credential by entering the name of the credential in the **Search** field.
   - Sort the **Provider** column so that the names of the cloud providers are listed alphabetically in either increasing or decreasing order. Identify your cloud provider and look for the name of your credential.

   You identified the cloud credential that you want to use.
4. Perform the following steps to remove authorized users of the cloud credential:
   a. Click the **Add Members** icon  in the row of the credential.

      The **Add New Member** dialog is displayed.

   b. Identify the member from the listed members displayed under **Members with access**.

   c. Click  in the row of the identified member.

      The selected member is removed from the list displayed under **Members with access**.

   d. Click **Close** to close the dialog.

      The abbreviated name of the authorized user is removed from the display under the **Members** column in the row of the cloud credential.

Results

You as a *Team Space Owner* removed authorized users from the cloud credential.

What to do next

You can inform the team space members that they are removed as authorized users of the cloud credential.

Related information

Management of cloud credentials on page 587

Deleting an existing cloud credential

When your cloud subscription to the mobile device cloud or AI provider expires and you no longer need the connection from HCL DevOps Test Hub (Test Hub), you can delete an existing cloud credential that you configured in your team space.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Added the cloud credentials of your mobile device cloud or AI provider account to configure the connection between Test Hub and the cloud account. See Adding cloud credentials of mobile device clouds on page 593 and Adding cloud credentials of AI providers on page 596.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Cloud Credentials** on the navigation pane.

   The **Cloud Credentials** page is displayed with the cloud credentials that are added to the team space.
3. Identify the cloud credential by performing any of the following actions:
   - Scroll through the table of cloud credentials to identify the cloud credential with the name you provided.
   - Search for the cloud credential by entering the name of the credential in the **Search** field.
   - Sort the **Provider** column so that the names of the cloud providers are listed alphabetically in either increasing or decreasing order. Identify your cloud provider and look for the name of your credential.

   You identified the cloud credential that you want to use.
4. Click the **Delete cloud credential** icon in the row of the credential that you want to delete.

   The **Delete cloud credential** dialog is displayed.
5. Click **Delete**.

   The existing cloud credential is deleted and removed from the display on the **Cloud Credentials** page.

Results

You deleted an existing cloud credential that you previously configured, from your team space.

Related information

Management of cloud credentials on page 587

Test run configurations

You can configure and run tests in HCL DevOps Test Hub (Test Hub) after you add the test resources to your project in your team space.

Before you configure a test run, you must have completed the following tasks:

- Created a project on Test Hub or you must have been granted access to a project with the *Tester* role assigned. See Managing DevOps Test Hub projects on page 1209.
- Created tests in the desktop clients and committed the test assets and test resources to a remote repository. You must have added the remote repository to the project.
- Read the considerations that you must take into account for certain types of tests. See Prerequisites to running tests on page 527.

You must be a project *Owner* or a member with the *Tester* role assigned to configure and run a test.

You can find information about how to configure and run the following types of test assets:

Type of Suite or test	Suite or test	More information
AFT Suite	Suite	• Configuring an AFT Suite run on page 618 • Configuring an AFT Suite to run on a mobile device by using a cloud credential on page 644
API Suite	Suite	• Configuring an API Suite or an API test run on page 652
Compound Test containing HTML tests	Suite	• Configuring a run of a Compound Test that contains HTML tests on page 665
Compound Test containing Web UI tests	Suite	• Configuring a run of a Compound Test that contains Web UI tests on page 704 • Configuring a run of a Compound Test that contains Web UI tests to run by using variables on page 717 • Configuring a run of a Compound Test that contains Web UI tests to run by using a cloud credential on page 732

Type of Suite or test	Suite or test	More information
Compound Test containing mobile tests	Suite	• Configuring a run of a Compound Test that contains mobile tests to run by using variables on page 678 • Configuring a run of a Compound Test that contains mobile tests to run by using a cloud credential on page 693
Compound Test containing performance tests	Suite	• Configuring a run of a Compound Test that contains performance tests on page 743
Compound test containing SAP GUI test		• Configuring a run of an SAP GUI test on page 822
JMeter test	Suite	• Configuring a run of a JMeter test on page 867
JUnit test	Suite	• Configuring a run of a JUnit test on page 876
Postman test	Suite	• Configuring a run of a Postman test on page 886
Rate Schedule	Suite	• Configuring a run of a Rate Schedule or VU Schedule on page 896
SAP GUI test	Test	• Configuring a run of an SAP GUI test on page 822
Selenium test	Suite	• Configuring a run of a Selenium test on page 909

Type of Suite or test	Suite or test	More information
VU Schedule	Suite	• Configuring a run of a Rate Schedule or VU Schedule on page 896
API test	Test	• Configuring an API Suite or an API test run on page 652
Functional test **Note:** The Functional test can contain any of the following individual tests: • Mobile test • Web UI test • Windows test	Test	• Configuring a run of a mobile test to run by using variables on page 756 • Configuring a run of a mobile test to run on a mobile device by using a cloud credential on page 771 • Configuring a run of a Web UI test on page 795 • Configuring a run of a Web UI test to run by using variables on page 827 • Configuring a run of a Web UI test to run by using a cloud credential on page 842 • Configuring a run of a Windows test on page 854
Performance test	Test	• Configuring a run of a Performance test on page 782
Server Suite asset	Suite	
Server Test asset	Test	• Running server test assets on page 808 • Running server test assets on an agent on page 811
Server Suite asset	Suite	Running server suite assets on page 816

Configuring a Suite or test for a quick run

You can configure any type of Suite or test to be run on HCL DevOps Test Hub (Test Hub) when you want to quickly ensure that the test runs correctly. You might not want to set the different options for the test nor want to schedule the run.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Prerequisites to running tests on page 527 if they apply to the test you want to configure for a run.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

The test assets in the selected tab are displayed.

5. Identify the test asset that you want to run by performing any of the following actions:
   - Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon  in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name

- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.
  - Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
  - Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.
- Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run, if you want to run a different version other than the latest version.
8. No action is required if you want to initiate the test run immediately after you click **Execute**.
9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page.
10. Click **Execute**.
    The test run is initiated.

Results

You have started a test run.

What to do next

You can perform any of the following tasks::

- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Viewing the progress of running test assets on page 926

Stopping test runs on page 932

Canceling scheduled test runs on page 934

Configuring an AFT Suite run

After you added the test resources that you created in the desktop client to the project, you can configure an AFT Suite to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Read Test run considerations for AFT Suites or Functional tests on page 528, if you want to configure a run for an AFT Suite that has the agent location configured in the `AFT XML` file.
- Read and completed the tasks mentioned in Test run considerations for browser-based Web UI tests on page 532, if you want to run tests that use Chrome or Edge browsers.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

The test assets in the selected tab are displayed.

5. Identify the test asset that you want to run by performing any of the following actions:
    - Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
Ju	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
Se	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.

- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

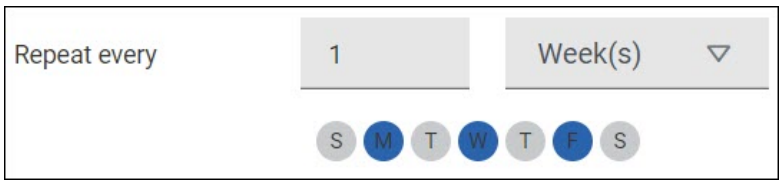
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour

Schedule	Actions
	▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

   If you are running an AFT Suite and the following conditions are true, then you must perform the next step:

- You do not want to run the test on the agent configured in the `AFT XML` file.
- You have not selected the agent from the **Override** column in the **Location** tab.

10. Click **Advanced** to make the following advanced configurations:

    a. Add the following setting in the **Program Arguments** field:

    ```
    -swaplocation <configured_agent_location>:<overriding_agent_location>
    ```

    For example, if the configured agent location is *91.2.352.24* and the remote agent where you want to run the test is *100.35.117.164*, then the entry in the **Program Arguments** field is as follows:

    ```
    -swaplocation 91.2.352.24:100.35.117.164
    ```

    If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

    The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

    You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

    Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

    c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

    The default value for the variables is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.

Export options	Results in...	Actions
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Select any of the following steps that depends on the location to run the test:
    - If you want to run the tests on the remote agents, go to Step 15 on page 629.
    - If you want to run the tests on a Kubernetes cluster, go to Step 16 on page 630.
15. Follow the instructions if you want to run a test on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

    a. Click the **LOCATION** tab, if it is not already open.

    The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability of the performance agent is displayed.

    Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

    The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

    b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.

If...	Then...	Action
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the agent.
The agent that is specified in the test assets is available and other agents added to the project.	Performance agents that have capabilities that match with the agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you hover over the performance agent in the **Override** column.	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

c. Click **Execute**.

The test run is initiated.

16. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the following remote host to your project that are then displayed under the **Override** column:

Kubernetes clusters that are registered with a team space.

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of an AFT Suite.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Unable to run AFT Suites on page 1277

Configuring an AFT Suite to run on a mobile device by using variables

After you added the test resources that you created in the desktop client to the project, you can configure an AFT Suite to run on a mobile device from HCL DevOps Test Hub (Test Hub). You must specify the variables for the mobile tests to run on mobile devices that are connected to an agent or a device cloud.

Before you begin

You must have completed the following tasks:

- Read Test run considerations for AFT Suites or Functional tests on page 528, if you want to configure a run for an AFT Suite that has the agent location configured in the `AFT XML` file.
- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Created a variable file if you want to import the variables file. The file must contain the details of the UI Test agent or the device cloud to which the mobile device is connected.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Ensured that you are a member of the cloud credential that is configured to connect to the mobile device cloud. See Viewing details of cloud credentials on page 601.

About this task

You can run an AFT Suite, which contains Web UI or mobile tests to run on a mobile device, emulator, or simulator that is connected to an agent or a device cloud. You must create the tests in Test UI, and then add the test asset to your project on Test UI. You can then run the tests on the mobile device, emulator, or simulator that is connected to any of the following agents or clouds:

- Remote UI Test agents. Requires the variables at test run time as specified in Variables for the UI Test agent on page 633.
- The BitBar Cloud. Requires the variables at test run time as specified in Table 6: Variables for the BitBar Cloud on page 633.
- The BrowserStack Cloud. Requires the variables at test run time as specified in Table 7: Variables for the BrowserStack Cloud on page 634.

- The pCloudy Cloud. Requires the variables at test run time as specified in Table 8: Variables for the pCloudy Cloud on page 634.
- The Perfecto Mobile Cloud. Requires the variables at test run time as specified in Table 9: Variables for the Perfecto Mobile Cloud on page 635.

You must provide the details of the UI Test agent or the mobile cloud to which the mobile devices are connected as variables in Step 10 on page 641. You can either enter the variables or use the file in which you entered the variables.

You must refer to the following tables for the variables that are required for a successful run:

Table 5. Variables for the UI Test agent

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the UI Test agent. For example, *Emulator:Pixel_3a_API_33*.
appium.server.host	Specify the hostname or IP address of the UI Test agent.
appium.server.port	Specify the port on the UI Test agent that is configured to communicate with Test Hub.

Table 6. Variables for the BitBar Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the BitBar Cloud. For example, *BitBar Cloud:Google Pixel 7 Pro*.
bitbar.apikey	Specify the user token generated for your BitBar Cloud account to authenticate your connection with the BitBar Cloud.
bitbar.host	Specify the hostname of the BitBar Cloud instance.
bitbar.project	Specify the name of the project that contains the recorded test.
bitbar.testrun	Specify a name for the test run that must be displayed in the BitBar Cloud dashboard for the test run.

Table 7. Variables for the BrowserStack Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the BrowserStack Cloud. For example, *BrowserStack Cloud:Google Pixel 7 Pro*.
browserstack.host	Specify the URL of the BrowserStack Cloud instance.
browserstack.username	Specify the user name of your BrowserStack Cloud account.
browserstack.apikey	Specify the API key of your BrowserStack Cloud account to authenticate your connection with the BrowserStack Cloud.
browserstack.project	Specify the name of the project that contains the recorded test. **Note:** This variable is optional.

Table 8. Variables for the pCloudy Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the pCloudy Cloud. For example, *pCloudy Cloud:Google Pixel 7 Pro*.
pcloudy.apikey	Specify the API key that is generated for your pCloudy Cloud account to authenticate your connection with the pCloudy Cloud.
pcloudy.username	Specify the username with which you can access the pCloudy Cloud instance.
pcloudy.host	Specify the hostname of the pCloudy Cloud instance.

Table 9. Variables for the Perfecto Mobile Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the Perfecto Mobile Cloud. For example, *Perfecto Mobile Cloud:Google Pixel 7 Pro*.
perfecto.securitytoken	Specify the user token generated for your Perfecto Mobile Cloud account to authenticate your connection with the Perfecto Mobile Cloud.
perfecto.host	Specify the hostname of the Perfecto Mobile Cloud instance.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **SUITES** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.

- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name
     - Test Asset Path
     - Last Result
     - Next Run
     - Components

  f. Click **Save** to save and apply the filter query to filter the assets based on the query.

     The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon  in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

◦ If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.

◦ If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

◦ Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

◦ Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

◦ If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.

◦ If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

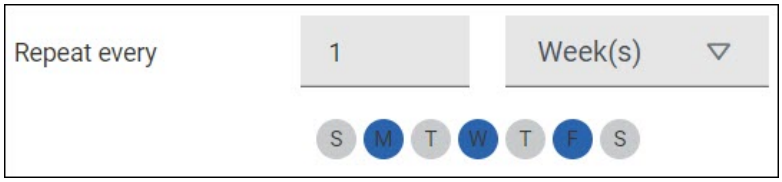
◦ No action is required if you want to initiate the test run immediately after you click **Execute**.

Important: Click **Execute** only after you have configured the other settings in this dialog.

Note: The default time for scheduling a run is **Now**.

◦ Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**.

Schedule	Actions
	The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Perform the following steps to provide the variables that specify the device cloud to which the mobile device is attached. You can either enter the variables that must be passed to the test at the test run time or import the file that contains the variables.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

      Note: You must have created a file with the variables before you can select the file.

11. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project.

Export options	Results in...	Actions
		iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

12. Perform any of the following actions that depend on whether you are using a remote performance agent or a mobile device cloud:
    - Go to Step 13 on page 642, if you are using an agent.
    - Go to Step 14 on page 644, if you are using a mobile device cloud.
13. Follow the instructions if you want to run the tests on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:
    a. Click the **LOCATION** tab, if it is not already open.

    The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability and capabilities of the agent are displayed.

Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the performance agent.
The performance agent that is specified in the test assets is available and there are other performance agents added to the project.	Performance agents that have capabilities that match with the performance agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently.	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

If...	Then...	Action
	You can find details of both performance agents for the matching capabilities when you hover over the agent in the **Override** column.	

c. Click **Execute**.

14. Click **Execute**.
    The test run is initiated.

Results

You have started or scheduled a test run of an AFT Suite to run on a device, emulator, or simulator that is connected to an agent or mobile device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Test run configurations on page 610

Management of cloud credentials on page 587

Resetting the configuration settings for a test run on page 931

Configuring an AFT Suite to run on a mobile device by using a cloud credential

After you added the test resources that you created in the desktop client to the project, you can configure an AFT Suite to be run on a mobile device by using a cloud credential from HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created a cloud credential in your team space that is configured to connect to the mobile device cloud. See Adding cloud credentials of mobile device clouds on page 593.
- Ensured that you are a member of the cloud credential that is configured to connect to the mobile device cloud. See Viewing details of cloud credentials on page 601.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **SUITES** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.

- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name
     - Test Asset Path
     - Last Result
     - Next Run
     - Components

  f. Click **Save** to save and apply the filter query to filter the assets based on the query.

     The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon  in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

◦ If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.

◦ If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

◦ Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

◦ Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

◦ If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
◦ If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

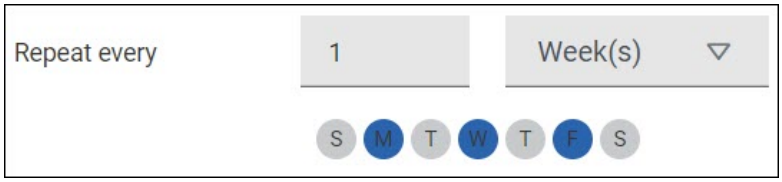
◦ No action is required if you want to initiate the test run immediately after you click **Execute**.

Important: Click **Execute** only after you have configured the other settings in this dialog.

Note: The default time for scheduling a run is **Now**.

◦ Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**.

Schedule	Actions
	The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Perform the following steps in the **ENVIRONMENT** tab:

    a. Select the cloud credential of the mobile device cloud from the **Device cloud credentials** list.

    The list displays the cloud credentials that are configured in the team space. The name of the cloud credential is appended to the name of the mobile device cloud in the list. For example, if the mobile device cloud provider is *BitBar* and the name of the credential is *BB_Account1*, then the name displayed in the list is *Bit Bar - BB_Account1*.

    Note: Access to credentials for mobile device cloud services depends on the status of the provider. If you disable a provider on the **Cloud Credentials** page, you cannot access that device cloud provider from the **Device cloud credentials** list.

    The devices that are available for testing on the selected mobile device cloud are not displayed because the devices are already specified in the AFT Suite.

    Note: If you are connected to either the BitBar Cloud or BrowserStack Cloud, the **Projects** field is displayed.

    b. Select the project on the mobile device cloud that you want to use or create a project in the **Projects** field. The project overrides the project that is configured in the AFT Suite.

11. **Optional:** Click the **RESULTS** tab, and then perform any the following actions:
    - Select an option if you want to export the test results to a Jira issue in your Xray project in Jira.

      You must have set up Jira as a change management system and configured the Xray integration to export the results.
    - Click **+ Create Property**, and then add values in the **Name** and **Value** columns.

The key-value pair of properties created here helps you add details about the test, and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed on the **Functional Report** page, **Test Log** page, and in the **Properties** column on the **Results** page.

12. Click **Execute**.
    The test run is initiated.

Results

You have started or scheduled a test run of an AFT Suite to run on a mobile device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Test run configurations on page 610

Management of cloud credentials on page 587

Resetting the configuration settings for a test run on page 931

Configuring an API Suite or an API test run

After you added the test resources that you created in the desktop client to the project, you can configure an API Suite or an API test to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Prerequisites to running tests on page 527, if they apply to the test that you want to configure for a run.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.

- Completed the following tasks if you are running API Suites or tests that use a transport and the transport requires third-party application `Jar` files for a successful run:
  - Read Test run considerations for API Suites or API tests on page 529.
  - Copied the third-party application `Jar` files to the third-party application folder on the computer where Test Hub is installed, if you are running the API Suite in Kubernetes. See Copying third-party application Jars to Kubernetes on page 165.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.

If...	Then...
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

The test assets in the selected tab are displayed.

5. Identify the test asset that you want to run by performing any of the following actions:
    - Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.

- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

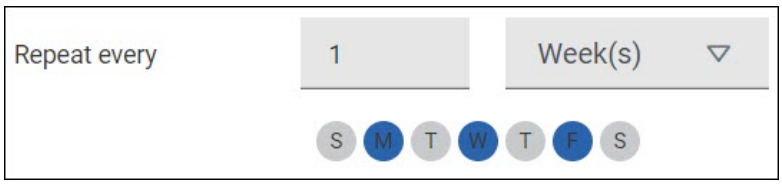
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour

Schedule	Actions
	▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

   For example, you can set a maximum Java heap size.

b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

   Perform the following actions, if applicable for your test:

   - Configuring test log reporting.

     If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

   - Overriding results database that is configured in the API Suite project.

     Note: You can use any of the databases that are specified in the system requirements as supported databases in Test Integrations and APIs.

     Enter the following arguments in the **Program Arguments** field, if you want to override the **Results Database**:

Program Arguments	Description
-resultDatabaseURL *<database_URL>*	Specifies the URL of the results database that overrides the results database that is configured for the project in Test Integrations and APIs.
-resultDatabaseDriver *<driver_name>*	Specifies the driver of the results database that overrides the results database that is configured for the project in Test Integrations and APIs. For example, the driver for a MySQL database is *com.mysql.cj.jdbc.Driver*.
-resultDatabaseUsername *<user_name>*	Specifies the username that is authorized to access the results database that overrides the results database that is configured for the project in Test Integrations and APIs.
-resultDatabasePassword *<encrypted_password>*	Specifies the password that is to be used to authenticate the user to access the results database that overrides the results database that is configured for the project in Test Integrations and APIs. **Note:** You can encrypt the password by using the **EncryptPassword** application that is available in the installation directory of Test Integrations and APIs.

     For example, if you want to override the results database that is configured in the Test Integrations and APIs project with a different results database that has the following details:

- MySQL database as the results database with the URL as *jdbc:mysql://localhost:3306/myresultsdb*
- *com.mysql.cj.jdbc.Driver* as the driver for the MySQL database
- *tester* as the username to access the MySQL database
- #com.ghc.*1!9a8521137E0A12994ACF4434E315A2B9* as the encrypted password to authenticate the user name *tester*

The entry in the **Program Arguments** field must be as follows:

```
-resultDatabaseURL  jdbc:mysql://localhost:3306/myresultsdb
-resultDatabaseDriver com.mysql.cj.jdbc.Driver
-resultDatabaseUsername tester
-resultDatabasePassword #com.ghc.1!9a8521137E0A12994ACF4434E315A2B9
```

Note: You must separate entries in the field with white spaces or with a new-line character.

11. Follow the instructions if the API Suite has an environment or secrets configured:
    a. Click the **ENVIRONMENT** tab, if it is not already open.
    b. Select the API test environment from the list if there are multiple environments configured in the test asset.
    c. Select the security group that contains the secrets to be used for the test run.

Notes:

- The test asset that was created in the desktop client and added to the Git repository must have the environments defined as part of the API test project.
- If the test asset contained secrets, then you must create those secrets in the security groups in the project.

The default value for the environment is the environment configured in the test asset. The default value for secrets is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you are running a test asset that contains datasets:
    a. Click the **DATA SOURCES** tab, if it is not already open.
    b. Consider the following information about datasets before you select a dataset:

    The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

    You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions.

Option	Description
	Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

    The default value for the variables is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

14. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project.

Export options	Results in...	Actions
		iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

15. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the following remote host to your project that are then displayed under the **Override** column:

Kubernetes clusters that are registered with a team space.

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of an API Suite.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Compound Test that contains HTML tests

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains HTML tests to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- When you want to run a Compound Test, you must have authored a Compound Test in HCL DevOps Test UI (Test UI) and must have completed the following tasks when you created the Compound Test:
  - Created a Compound Test under a Web UI project in your workspace.
  - Added a Functional HTML Test under the Compound Test.
  - Used the **startBrowser(Firefox,<URL>)** attribute with the browser value set to *Firefox* in the Functional test script.

    Restriction: Functional test scripts that contain the **startApp()** or **callScript()** attributes are treated as non-HTML scripts and are not supported to be used in test scripts in the HTML tests.

  - Linked the dataset that is used in the Functional HTML Test with the Web UI Compound Test only if you want to override this dataset. You must have created the test resources in HCL DevOps Test UI, before committing the test resources to the server project repository. You must have also created the dataset that you want to use to override the dataset linked to the test.
  - Ran the Compound Test that contains the HTML tests on Test UI.
  - Committed the projects that include the Web UI project (with the Compound Test) and the Functional test (with the HTML tests) to the remote repository.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.

Tab	Description
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

   The test assets in the selected tab are displayed.

5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

     Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

     Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

Choose one of the following visibility levels:

- **Private**: Makes the filter available only to the owner of the filter who created it.
- **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon  in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

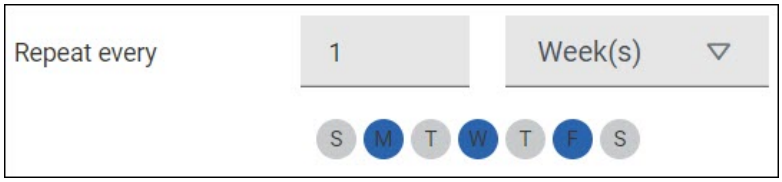
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps:

Schedule	Actions
	a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

       If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

    c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

       For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

       The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

       You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions.

Option	Description
	Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions in this step if the test requires a variable that must be passed to the test at the test run time.

    You must configure the supported browser by using a variable if the test has a browser configured, which is different from the one that is supported by Test Hub.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.

        i. Enter `RTW_WebUI_Browser_Selection` as the name.
        ii. Enter `Firefox` as the value, if you want to use Firefox as the browser and override the browser that is specified in the HTML test.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

    The default value for the variables is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the following remote host to your project that are then displayed under the **Override** column:

Kubernetes clusters that are registered with a team space.

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of a Compound Test that contains Functional HTML tests.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Compound Test that contains mobile tests to run by using variables

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains mobile tests to run by using variables from HCL DevOps Test Hub (Test Hub). You must specify the variables for the mobile tests to run on mobile devices that are connected to an agent or a device cloud.

Before you begin

You must have completed the following tasks:

- Created a Compound Test that contains the mobile tests to run on a mobile device in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Created a variable file if you want to import the variables file. The file must contain the details of the UI Test agent or the device cloud to which the mobile device is connected.
- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

You can run a Compound Test, which contains mobile tests to run on a mobile device, emulator, or simulator that is connected to an agent or a device cloud. You must create the tests in Test UI, and then add the test asset to your project on Test Hub. You can then run the mobile tests on the mobile devices that are connected to any of the following agents or clouds:

- Remote UI Test agents. Requires the variables at test run time as specified in Variables for the UI Test agent on page 679.
- The BitBar Cloud. Requires the variables at test run time as specified in Variables for the BitBar Cloud on page 679.
- The BrowserStack Cloud. Requires the variables at test run time as specified in Variables for the BrowserStack Cloud on page 680.
- The pCloudy Cloud. Requires the variables at test run time as specified in Variables for the pCloudy Cloud on page 680.
- The Perfecto Mobile Cloud. Requires the variables at test run time as specified in Variables for the Perfecto Mobile Cloud on page 681.

You must provide the details of the UI Test agent or the mobile cloud to which the mobile devices are connected as variables in Step 12 on page 689. You can either enter the variables or use the file in which you entered the variables.

You must refer to the following tables for the variables that are required for a successful run:

Table 10. Variables for the UI Test agent

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the UI Test agent. For example, *Emulator:Pixel_3a_API_33*.
appium.server.host	Specify the hostname or IP address of the UI Test agent.
appium.server.port	Specify the port on the UI Test agent that is configured to communicate with Test Hub.

Table 11. Variables for the BitBar Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the BitBar Cloud. For example, *BitBar Cloud:Google Pixel 7 Pro*.
bitbar.apikey	Specify the user token generated for your BitBar Cloud account to authenticate your connection with the BitBar Cloud.
bitbar.host	Specify the hostname of the BitBar Cloud instance.

Table 11. Variables for the BitBar Cloud (continued)

Name of the Variable	Action for the Value field
bitbar.project	Specify the name of the project that contains the recorded test.
bitbar.testrun	Specify a name for the test run that must be displayed in the BitBar Cloud dashboard for the test run.

Table 12. Variables for the BrowserStack Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the BrowserStack Cloud. For example, *BrowserStack Cloud:Google Pixel 7 Pro*.
browserstack.host	Specify the URL of the BrowserStack Cloud instance.
browserstack.username	Specify the user name of your BrowserStack Cloud account.
browserstack.apikey	Specify the API key of your BrowserStack Cloud account to authenticate your connection with the BrowserStack Cloud.
browserstack.project	Specify the name of the project that contains the recorded test.  **Note:** This variable is optional.

Table 13. Variables for the pCloudy Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the pCloudy Cloud. For example, *pCloudy Cloud:Google Pixel 7 Pro*.
pcloudy.apikey	Specify the API key that is generated for your pCloudy Cloud account to authenticate your connection with the pCloudy Cloud.

Table 13. Variables for the pCloudy Cloud (continued)

Name of the Variable	Action for the Value field
pcloudy.username	Specify the username with which you can access the pCloudy Cloud instance.
pcloudy.host	Specify the hostname of the pCloudy Cloud instance.

Table 14. Variables for the Perfecto Mobile Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the Perfecto Mobile Cloud. For example, *Perfecto Mobile Cloud:Google Pixel 7 Pro*.
perfecto.securitytoken	Specify the user token generated for your Perfecto Mobile Cloud account to authenticate your connection with the Perfecto Mobile Cloud.
perfecto.host	Specify the hostname of the Perfecto Mobile Cloud instance.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **SUITES** or the **ADVANCED** tab.

5. Identify the test asset that you want to run by performing any of the following actions:
    - Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon  in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name

- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.
  - Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
  - Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.
- Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:
- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

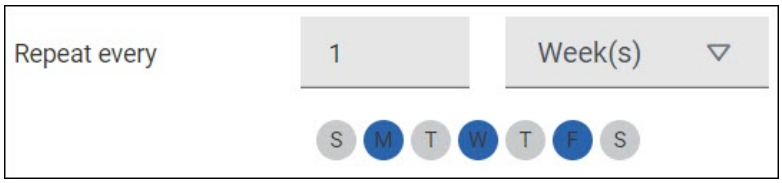
- No action is required if you want to initiate the test run immediately after you click **Execute**.

  Important: Click **Execute** only after you have configured the other settings in this dialog.

  Note: The default time for scheduling a run is **Now**.

- Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:

Schedule	Actions
	 e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter the following program argument that must be passed to the test run at run time in the **Program Arguments** field:

       Enter `-history testlog` in the **Program Arguments** field.

Important: If the `-history testlog` is not entered in the **Program Arguments** field, the unified report that is generated displays the test steps incorrectly.

c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

a. Click the **DATA SOURCES** tab, if it is not already open.

b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Perform the following steps to provide the variables that specify the UI Test agent or cloud to which the mobile device is attached. You can either enter the variables that must be passed to the test at the test run time or import the file that contains the variables.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

Note: You must have created a file with the variables before you can select the file.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:
    a. Click the **RESULTS** tab.
    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported.	Perform the following steps when:

Export options	Results in...	Actions
	Note: This export option is the default option.	▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Perform any of the following actions that depend on whether you are using a remote performance agent or a mobile device cloud:
    - Go to Step 15 on page 691, if you are using an agent.
    - Go to Step 16 on page 692, if you are using a mobile device cloud.
15. Follow the instructions if you want to run the tests on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

a. Click the **LOCATION** tab, if it is not already open.

The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability and capabilities of the agent are displayed.

Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.

If...	Then...	Action
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the performance agent.
The performance agent that is specified in the test assets is available and there are other performance agents added to the project.	Performance agents that have capabilities that match with the performance agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you hover over the agent in the **Override** column.	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

c. Click **Execute**.

16. Click **Execute**.
    The test run is initiated.

Results

You have started or scheduled a test run of a Compound Test that contains mobile tests to run on a mobile device that is connected to an agent or a device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Compound Test that contains mobile tests to run by using a cloud credential

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains mobile tests to run by using a cloud credential from HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Created a Compound Test that contains the mobile tests in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created a cloud credential in your team space that is configured to connect to the mobile device cloud. See Adding cloud credentials of mobile device clouds on page 593.
- Ensured that you are a member of the cloud credential that is configured to connect to the mobile device cloud. See Viewing details of cloud credentials on page 601.

About this task

You can run a Compound Test, which contains mobile tests to run on mobile devices that are connected to a device cloud. You must create the tests in Test UI, and then add the test asset to your project on Test Hub.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **SUITES** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

Choose one of the following visibility levels:

- **Private**: Makes the filter available only to the owner of the filter who created it.
- **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon  in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

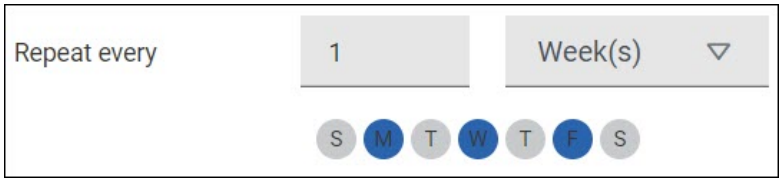
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps:

Schedule	Actions
	a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Perform the following steps in the **ENVIRONMENT** tab:

    a. Select the cloud credential of the mobile device cloud from the **Device cloud credentials** list.

    The list displays the cloud credentials that are configured in the team space. The name of the cloud credential is appended to the name of the mobile device cloud in the list. For example, if the mobile device cloud provider is *BitBar* and the name of the credential is *BB_Account1*, then the name displayed in the list is *Bit Bar - BB_Account1*.

    Note: Access to credentials for mobile device cloud services depends on the status of the provider. If you disable a provider on the **Cloud Credentials** page, you cannot access that device cloud provider from the **Device cloud credentials** list.

    If the connection to the mobile device cloud is successful, the **Mobile devices** field is displayed.

    b. Select the devices on which you want to run the tests from the list of devices that are displayed in the **Mobile devices** list.

    Note: If you are connected to either the BitBar Cloud or BrowserStack Cloud, the **Projects** field is displayed.

    c. Select the project on the mobile device cloud that you want to use or create a project in the **Projects** field. The project overrides the project that is configured in the test asset.

11. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

    For example, you can set a maximum Java heap size.

    b. Enter the following program argument that must be passed to the test run at run time in the **Program Arguments** field:

    Enter `-history testlog` in the **Program Arguments** field.

Important: If the `-history testlog` is not entered in the **Program Arguments** field, the unified report that is generated displays the test steps incorrectly.

c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you are running a test asset that contains datasets:

a. Click the **DATA SOURCES** tab, if it is not already open.

b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Click **Execute**.
The test run is initiated.

Results

You have started or scheduled a test run of a Compound Test that contains mobile tests to run on a mobile device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Test run configurations on page 610

Management of cloud credentials on page 587

Resetting the configuration settings for a test run on page 931

Configuring a run of a Compound Test that contains Web UI tests

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains Web UI tests to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Prerequisites to running tests on page 527, if they apply to the test that you want to configure for a run.
- Read and completed the tasks mentioned in Test run considerations for browser-based Web UI tests on page 532, if you want to run tests that use Chrome or Edge browsers.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Created a Compound Test that contains Web UI tests in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **SUITES** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

     Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

     Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:

a. Click **New filter**.
b. Enter a name for the filter.
c. Select the **Visibility** of the filter.

Choose one of the following visibility levels:

- **Private**: Makes the filter available only to the owner of the filter who created it.
- **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

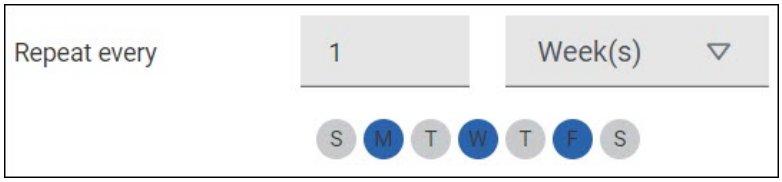
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps:

Schedule	Actions
	a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

       If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

    c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

       For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

       The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

       You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions.

Option	Description
	Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions in this step if the test requires a variable that must be passed to the test at the test run time.

    You must configure the supported browser by using a variable if the test has a browser configured, which is different from the one that is supported by Test Hub.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.

        i. Enter `RTW_WebUI_Browser_Selection` as the name.
        ii. Enter `Firefox` as the value, if you want to use Firefox as the browser and override the browser that is specified in the Web UI test.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

    The default value for the variables is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Select any of the following steps that depends on the location to run the test:
    - If you want to run the tests on the remote agents, go to Step 15 on page 714.
    - If you want to run the tests on a Kubernetes cluster, go to Step 16 on page 716.
15. Follow the instructions if you want to run the tests on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

a. Click the **LOCATION** tab, if it is not already open.

The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability and capabilities of the agent are displayed.

Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	**Then...**	**Action**
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the performance agent.
The performance agent that is specified in the test assets is available and there are other performance agents added to the project.	Performance agents that have capabilities that match with the performance agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

If...	Then...	Action
	match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you hover over the agent in the **Override** column.	

c. Click **Execute**.

16. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the following remote host to your project that are then displayed under the **Override** column:

Kubernetes clusters that are registered with a team space.

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of a Compound Test that contains Web UI tests.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Compound Test that contains Web UI tests to run by using variables

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains Web UI tests to run by using variables from HCL DevOps Test Hub (Test Hub). You must specify the variables for the Web UI tests to run on mobile devices that are connected to an agent or a device cloud.

Before you begin

You must have completed the following tasks:

- Created a Compound Test that contains Web UI tests to run on a mobile device in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Created a variable file if you want to import the variables file. The file must contain the details of the UI Test agent or the device cloud to which the mobile device is connected.
- Read and completed the tasks mentioned in Test run considerations for browser-based Web UI tests on page 532, if you want to run tests that use Chrome or Edge browsers.
- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.

- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

You can run the Compound Test that contains Web UI tests on a mobile device, emulator, or simulator that is connected to an agent or a device cloud. You must create the tests in Test UI, and then add the test asset to your project on Test Hub. You can then run the Compound Test that contains Web UI tests on a mobile device that is connected to any of the following agents or clouds:

- Remote UI Test agents. Requires the variables at test run time as specified in Variables for the UI Test agent on page 718.
- The BitBar Cloud. Requires the variables at test run time as specified in Variables for the BitBar Cloud on page 719.
- The BrowserStack Cloud. Requires the variables at test run time as specified in Variables for the BrowserStack Cloud on page 719.
- The pCloudy Cloud. Requires the variables at test run time as specified in Variables for the pCloudy Cloud on page 720.
- The Perfecto Mobile Cloud. Requires the variables at test run time as specified in Variables for the Perfecto Mobile Cloud on page 720.

You must provide the details of the UI Test agent or the mobile cloud to which the mobile devices are connected as variables in Step 12 on page 729. You can either enter the variables or use the file in which you entered the variables.

You must refer to the following tables for the variables that are required for a successful run:

Table 15. Variables for the UI Test agent

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use on the UI Test agent. For example, *Chrome(Emulator:Pixel_3a_API_33)*.
appium.server.host	Specify the hostname or IP address of the UI Test agent.
appium.server.port	Specify the port on the UI Test agent that is configured to communicate with Test Hub.

Table 16. Variables for the BitBar Cloud

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use in the BitBar Cloud. For example, *Chrome(BitBar Cloud:Google Pixel 7 Pro)*.
bitbar.apikey	Specify the user token generated for your BitBar Cloud account to authenticate your connection with the BitBar Cloud.
bitbar.host	Specify the hostname of the BitBar Cloud instance.
bitbar.project	Specify the name of the project that contains the recorded test.
bitbar.testrun	Specify a name for the test run that must be displayed in the BitBar Cloud dashboard for the test run.

Table 17. Variables for the BrowserStack Cloud

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use in the BrowserStack Cloud. For example, *Chrome(BrowserStack Cloud:Google Pixel 7 Pro)*.
browserstack.host	Specify the URL of the BrowserStack Cloud instance.
browserstack.username	Specify the user name of your BrowserStack Cloud account.
browserstack.apikey	Specify the API key of your BrowserStack Cloud account to authenticate your connection with the BrowserStack Cloud.
browserstack.project	Specify the name of the project that contains the recorded test.

Table 17. Variables for the BrowserStack Cloud (continued)

Name of the Variable	Action for the Value field
	Note: This variable is optional.

Table 18. Variables for the pCloudy Cloud

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use in the pCloudy Cloud. For example, *Chrome(pCloudy Cloud:Google Pixel 7 Pro)*.
pcloudy.apikey	Specify the API key that is generated for your pCloudy Cloud account to authenticate your connection with the pCloudy Cloud.
pcloudy.username	Specify the username with which you can access the pCloudy Cloud instance.
pcloudy.host	Specify the hostname of the pCloudy Cloud instance.

Table 19. Variables for the Perfecto Mobile Cloud

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use in the Perfecto Mobile Cloud. For example, *Chrome(Perfecto Mobile Cloud:Google Pixel 7 Pro)*.
perfecto.securitytoken	Specify the user token generated for your Perfecto Mobile Cloud account to authenticate your connection with the Perfecto Mobile Cloud.
perfecto.host	Specify the hostname of the Perfecto Mobile Cloud instance.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.

3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **SUITES** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
Ju	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
Se	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.

- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

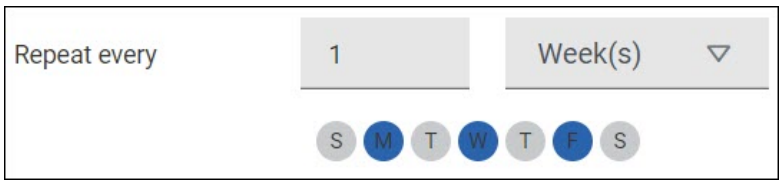
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour

Schedule	Actions
	▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

   For example, you can set a maximum Java heap size.

b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

   If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

   For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

       The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

       You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

       Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

    c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Perform the following steps to provide the variables that specify the UI Test agent or cloud to which the mobile device is attached. You can either enter the variables that must be passed to the test at the test run time or import the file that contains the variables.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported.	Perform the following steps when:

Export options	Results in...	Actions
	Note: This export option is the default option.	▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Perform any of the following actions that depend on whether you are using a remote performance agent or a mobile device cloud:
    - Go to Step 15 on page 730, if you are using an agent.
    - Go to Step 16 on page 731, if you are using a mobile device cloud.
15. Follow the instructions if you want to run the tests on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

a. Click the **LOCATION** tab, if it is not already open.

The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability and capabilities of the agent are displayed.

Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.

If...	Then...	Action
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the performance agent.
The performance agent that is specified in the test assets is available and there are other performance agents added to the project.	Performance agents that have capabilities that match with the performance agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you hover over the agent in the **Override** column.	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

c. Click **Execute**.

16. Click **Execute**.
    The test run is initiated.

Results

You have started or scheduled a test run of a Compound Test that contains Web UI tests to run on a mobile device that is connected to an agent or a device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Compound Test that contains Web UI tests to run by using a cloud credential

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains Web UI tests to run on devices by using a cloud credential from HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Created a Compound Test that contains the Web UI tests in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created a cloud credential in your team space that is configured to connect to the mobile device cloud. See Adding cloud credentials of mobile device clouds on page 593.
- Ensured that you are a member of the cloud credential that is configured to connect to the mobile device cloud. See Viewing details of cloud credentials on page 601.

About this task

You can run a Compound Test, which contains Web UI tests on a device cloud. You must create the tests in Test UI, and then add them to your project on Test Hub.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **SUITES** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

Choose one of the following visibility levels:

- **Private**: Makes the filter available only to the owner of the filter who created it.
- **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon  in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

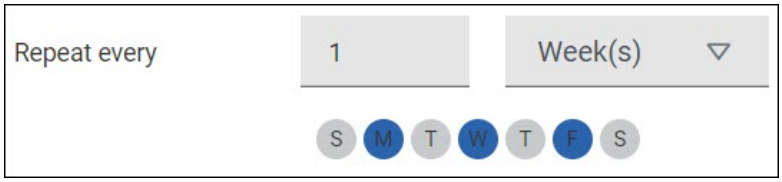
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps:

Schedule	Actions
	a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Perform the following steps in the **ENVIRONMENT** tab:

    a. Select the cloud credential of the mobile device cloud from the **Device cloud credentials** list.

    The list displays the cloud credentials that are configured in the team space. The name of the cloud credential is appended to the name of the mobile device cloud in the list. For example, if the mobile device cloud provider is *BitBar* and the name of the credential is *BB_Account1*, then the name displayed in the list is *Bit Bar - BB_Account1*.

    Note: Access to credentials for mobile device cloud services depends on the status of the provider. If you disable a provider on the **Cloud Credentials** page, you cannot access that device cloud provider from the **Device cloud credentials** list.

    If the connection to the mobile device cloud is successful, the **Mobile devices** field is displayed.

    b. Select the devices on which you want to run the tests from the list of devices that are displayed in the **Mobile devices** list.

    Note: If you are connected to either the BitBar Cloud or BrowserStack Cloud, the **Projects** field is displayed.

    c. Select the project on the mobile device cloud that you want to use or create a project in the **Projects** field. The project overrides the project that is configured in the test asset.

11. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

    For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

    If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you are running a test asset that contains datasets:

a. Click the **DATA SOURCES** tab, if it is not already open.

b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Click **Execute**.
The test run is initiated.

Results

You have started or scheduled a test run of a Compound Test that contains Web UI tests to run on a device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Test run configurations on page 610

Management of cloud credentials on page 587

Resetting the configuration settings for a test run on page 931

Configuring a run of a Compound Test that contains performance tests

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains performance tests to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Created performance tests under a Compound Test in HCL DevOps Test Performance (Test Performance).
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **SUITES** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:
   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

- **Private**: Makes the filter available only to the owner of the filter who created it.
- **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

   You can select a parameter from the following list:
   - Type
   - Test Asset Name
   - Test Asset Path
   - Last Result
   - Next Run
   - Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

   The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:
- Click the **Filter** icon  in the **ADVANCED** tab.

  The filter queries that you created and saved are displayed.
- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

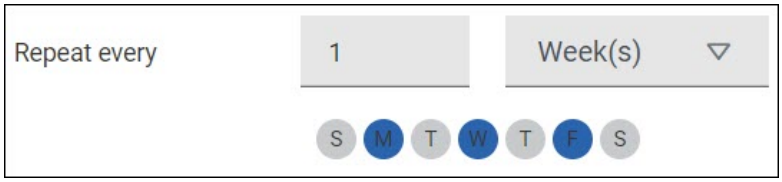
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps:

Schedule	Actions
	a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

       If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

    c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

       For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

       The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

       You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions.

Option	Description
	Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

    The default value for the variables is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project.

Export options	Results in...	Actions
		iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Select any of the following steps that depends on the location to run the test:
    - If you want to run the tests on the remote agents, go to Step 15 on page 753.
    - If you want to run the tests on a Kubernetes cluster, go to Step 16 on page 755.
15. Follow the instructions if you want to run a test on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

    a. Click the **LOCATION** tab, if it is not already open.

    The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability of the performance agent is displayed.

    Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

    The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the

default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the agent.
The agent that is specified in the test assets is available and other agents added to the project.	Performance agents that have capabilities that match with the agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you hover over the performance agent in the **Override** column.	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

c. Click **Execute**.

The test run is initiated.

16. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

    a. Click the **LOCATION** tab, if it is not already open.

    The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

    Important: You must have added the following remote host to your project that are then displayed under the **Override** column:

    Kubernetes clusters that are registered with a team space.

    Notes:

    - If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
    - If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

    b. Select the location where you want to run the test asset from the following options:

    - Select the **Default Cluster** when no remote clusters are available in your project.
    - Select the remote cluster from the list when a remote cluster is available in your project.
    - Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

    c. Click **Execute**.

    The test run is initiated.

Results

You have started or scheduled a test run of a Compound Test that contains performance tests.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.

- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a mobile test to run by using variables

After you added the test resources that you created in the desktop client to the project, you can configure a mobile test to run by using variables on a mobile device from HCL DevOps Test Hub (Test Hub). You must specify the variables for the mobile test to run on mobile devices that are connected to an agent or a device cloud.

Before you begin

You must have completed the following tasks:

- Created a mobile test in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Created a variable file if you want to import the variables file. The file must contain the details of the UI Test agent or the device cloud to which the mobile device is connected.
- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

You can run the mobile test after you create the test in Test UI, and then add them to your project on Test Hub. You can then run the mobile test on the mobile devices that are connected to any of the following agents or clouds:

- Remote UI Test agents. Requires the variables at test run time as specified in Variables for the UI Test agent on page 757.
- The BitBar Cloud. Requires the variables at test run time as specified in Variables for the BitBar Cloud on page 757.
- The BrowserStack Cloud. Requires the variables at test run time as specified in Variables for the BrowserStack Cloud on page 758.
- The pCloudy Cloud. Requires the variables at test run time as specified in Variables for the pCloudy Cloud on page 758.
- The Perfecto Mobile Cloud. Requires the variables at test run time as specified in Variables for the Perfecto Mobile Cloud on page 759.

You must provide the details of the UI Test agent or the mobile cloud to which the mobile devices are connected as variables in Step 12 on page 767. You can either enter the variables or use the file in which you entered the variables.

You must refer to the following tables for the variables that are required for a successful run:

Table 20. Variables for the UI Test agent

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the UI Test agent. For example, *Emulator:Pixel_3a_API_33*.
appium.server.host	Specify the hostname or IP address of the UI Test agent.
appium.server.port	Specify the port on the UI Test agent that is configured to communicate with Test Hub.

Table 21. Variables for the BitBar Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the BitBar Cloud. For example, *BitBar Cloud:Google Pixel 7 Pro*.
bitbar.apikey	Specify the user token generated for your BitBar Cloud account to authenticate your connection with the BitBar Cloud.
bitbar.host	Specify the hostname of the BitBar Cloud instance.

Table 21. Variables for the BitBar Cloud (continued)

Name of the Variable	Action for the Value field
bitbar.project	Specify the name of the project that contains the recorded test.
bitbar.testrun	Specify a name for the test run that must be displayed in the BitBar Cloud dashboard for the test run.

Table 22. Variables for the BrowserStack Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the BrowserStack Cloud. For example, *BrowserStack Cloud:Google Pixel 7 Pro*.
browserstack.host	Specify the URL of the BrowserStack Cloud instance.
browserstack.username	Specify the user name of your BrowserStack Cloud account.
browserstack.apikey	Specify the API key of your BrowserStack Cloud account to authenticate your connection with the BrowserStack Cloud.
browserstack.project	Specify the name of the project that contains the recorded test.  **Note:** This variable is optional.

Table 23. Variables for the pCloudy Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the pCloudy Cloud. For example, *pCloudy Cloud:Google Pixel 7 Pro*.
pcloudy.apikey	Specify the API key that is generated for your pCloudy Cloud account to authenticate your connection with the pCloudy Cloud.

Table 23. Variables for the pCloudy Cloud (continued)

Name of the Variable	Action for the Value field
pcloudy.username	Specify the username with which you can access the pCloudy Cloud instance.
pcloudy.host	Specify the hostname of the pCloudy Cloud instance.

Table 24. Variables for the Perfecto Mobile Cloud

Name of the Variable	Action for the Value field
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the Perfecto Mobile Cloud. For example, *Perfecto Mobile Cloud:Google Pixel 7 Pro*.
perfecto.securitytoken	Specify the user token generated for your Perfecto Mobile Cloud account to authenticate your connection with the Perfecto Mobile Cloud.
perfecto.host	Specify the hostname of the Perfecto Mobile Cloud instance.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **TESTS** or the **ADVANCED** tab.

5. Identify the test asset that you want to run by performing any of the following actions:
    - Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	VU Schedule			

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon  in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name

- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.
  - Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
  - Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.
- Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:
- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

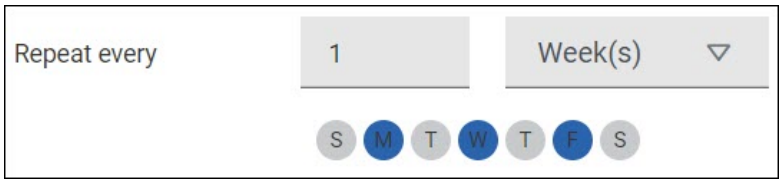
- No action is required if you want to initiate the test run immediately after you click **Execute**.

  Important: Click **Execute** only after you have configured the other settings in this dialog.

  Note: The default time for scheduling a run is **Now**.

- Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:

Schedule	Actions
	 e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

       If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

a. Click the **DATA SOURCES** tab, if it is not already open.

b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Perform the following steps to provide the variables that specify the UI Test agent or cloud to which the mobile device is attached. You can either enter the variables that must be passed to the test at the test run time or import the file that contains the variables.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

Note: You must have created a file with the variables before you can select the file.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:
    a. Click the **RESULTS** tab.
    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported.	Perform the following steps when:

Export options	Results in...	Actions
	Note: This export option is the default option.	▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Perform any of the following actions that depend on whether you are using a remote performance agent or a mobile device cloud:

    ◦ Go to Step 15 on page 769, if you are using an agent.
    ◦ Go to Step 16 on page 770, if you are using a mobile device cloud.

15. Follow the instructions if you want to run the tests on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

    a. Click the **LOCATION** tab, if it is not already open.

    The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability and capabilities of the agent are displayed.

    Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

    The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

    b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.

If...	Then...	Action
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the performance agent.
The performance agent that is specified in the test assets is available and there are other performance agents added to the project.	Performance agents that have capabilities that match with the performance agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you hover over the agent in the **Override** column.	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

c. Click **Execute**.

16. Click **Execute**.
    The test run is initiated.

Results

You have started or scheduled a test run of a mobile test to be run on a mobile device that is connected to an agent or device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a mobile test to run on a mobile device by using a cloud credential

After you added the test resources that you created in the desktop client to the project, you can configure a mobile test to run on a mobile device cloud by using a cloud credential from HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Created a mobile test in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created a cloud credential in your team space that is configured to connect to the mobile device cloud. See Adding cloud credentials of mobile device clouds on page 593.
- Ensured that you are a member of the cloud credential that is configured to connect to the mobile device cloud. See Viewing details of cloud credentials on page 601.

About this task

You can run the mobile test on a mobile device cloud. You must create the test in Test UI, and then add it to your project on Test Hub.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **TESTS** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

- **Private**: Makes the filter available only to the owner of the filter who created it.
- **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon  in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

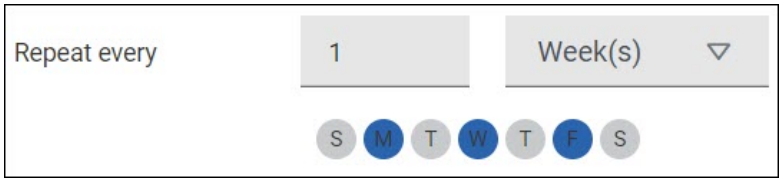
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps:

Schedule	Actions
	a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Perform the following steps in the **ENVIRONMENT** tab:

    a. Select the cloud credential of the mobile device cloud from the **Device cloud credentials** list.

    The list displays the cloud credentials that are configured in the team space. The name of the cloud credential is appended to the name of the mobile device cloud in the list. For example, if the mobile device cloud provider is *BitBar* and the name of the credential is *BB_Account1*, then the name displayed in the list is *Bit Bar - BB_Account1*.

    Note: Access to credentials for mobile device cloud services depends on the status of the provider. If you disable a provider on the **Cloud Credentials** page, you cannot access that device cloud provider from the **Device cloud credentials** list.

    If the connection to the mobile device cloud is successful, the **Mobile devices** field is displayed.

    b. Select the devices on which you want to run the tests from the list of devices that are displayed in the **Mobile devices** list.

    Note: If you are connected to either the BitBar Cloud or BrowserStack Cloud, the **Projects** field is displayed.

    c. Select the project on the mobile device cloud that you want to use or create a project in the **Projects** field. The project overrides the project that is configured in the test asset.

11. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

    For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

    If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you are running a test asset that contains datasets:

a. Click the **DATA SOURCES** tab, if it is not already open.

b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Click **Execute**.
The test run is initiated.

Results

You have started or scheduled a test run of a mobile test to run on a mobile device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Test run configurations on page 610

Management of cloud credentials on page 587

Resetting the configuration settings for a test run on page 931

Configuring a run of a Performance test

After you added the test resources that you created in the desktop client to the project, you can configure a Performance test to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Created a Performance test in HCL DevOps Test Performance (Test Performance).
- Read Test run considerations for using Machine learning capabilities on page 534, if you want to configure a run for either a Performance Test or a Schedule that must be analyzed by using Machine Learning capabilities in Test Hub.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.

- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **TESTS** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

Choose one of the following visibility levels:

- **Private**: Makes the filter available only to the owner of the filter who created it.
- **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon  in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

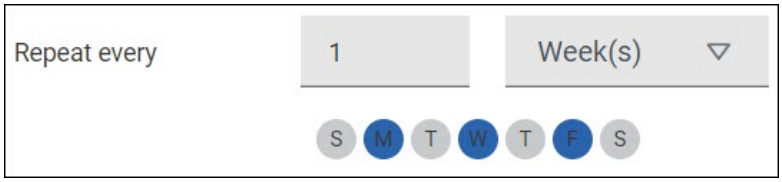
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps:

Schedule	Actions
	a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

       If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

    c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

       For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

       The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

       You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions.

Option	Description
	Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

    The default value for the variables is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project.

Export options	Results in...	Actions
		iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Select any of the following steps that depends on the location to run the test:
    - If you want to run the tests on the remote agents, go to Step 15 on page 792.
    - If you want to run the tests on a Kubernetes cluster, go to Step 16 on page 794.
15. Follow the instructions if you want to run a test on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

    a. Click the **LOCATION** tab, if it is not already open.

    The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability of the performance agent is displayed.

    Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

    The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the

default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the agent.
The agent that is specified in the test assets is available and other agents added to the project.	Performance agents that have capabilities that match with the agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you hover over the performance agent in the **Override** column.	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

c. Click **Execute**.

The test run is initiated.

16. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

    a. Click the **LOCATION** tab, if it is not already open.

    The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

    Important: You must have added the following remote host to your project that are then displayed under the **Override** column:

    Kubernetes clusters that are registered with a team space.

    Notes:

    - If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
    - If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

    b. Select the location where you want to run the test asset from the following options:

    - Select the **Default Cluster** when no remote clusters are available in your project.
    - Select the remote cluster from the list when a remote cluster is available in your project.
    - Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

    c. Click **Execute**.

    The test run is initiated.

Results

You have started or scheduled a test run of a Performance test.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.

- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Web UI test

After you added the test resources that you created in the desktop client to the project, you can configure a Web UI test to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Prerequisites to running tests on page 527, if they apply to the test that you want to configure for a run.
- Read and completed the tasks mentioned in Test run considerations for browser-based Web UI tests on page 532, if you want to run tests that use Chrome or Edge browsers.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Created a Web UI test in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

The test assets in the selected tab are displayed.

5. Identify the test asset that you want to run by performing any of the following actions:
    - Scroll through the list.



Tip: You can hover over the icon in the **Type** column to know the type of the test asset.



Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
JMeter	JMeter Test	✓		✓
Ju	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
Se	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.

- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name
     - Test Asset Path
     - Last Result
     - Next Run
     - Components

  f. Click **Save** to save and apply the filter query to filter the assets based on the query.

     The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon  in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

◦ If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.

◦ If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

◦ Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

◦ Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

◦ If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
◦ If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

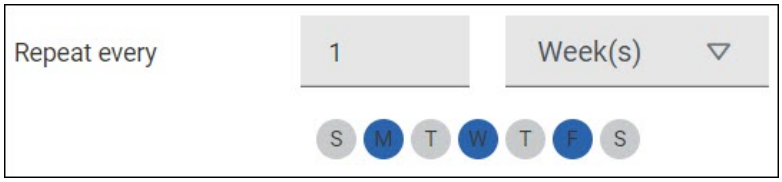
◦ No action is required if you want to initiate the test run immediately after you click **Execute**.

Important: Click **Execute** only after you have configured the other settings in this dialog.

Note: The default time for scheduling a run is **Now**.

◦ Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**.

Schedule	Actions
	The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

       If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

    c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

       For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

       The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.

Option	Description
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions in this step if the test requires a variable that must be passed to the test at the test run time.

    You must configure the supported browser by using a variable if the test has a browser configured, which is different from the one that is supported by Test Hub.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.

        i. Enter `RTW_WebUI_Browser_Selection` as the name.
        ii. Enter `Firefox` as the value, if you want to use Firefox as the browser and override the browser that is specified in the Web UI test.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

        Note: You must have created a file with the variables before you can select the file.

    The default value for the variables is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Select any of the following steps that depends on the location to run the test:
    - If you want to run the tests on the remote agents, go to Step 14 on page 805.
    - If you want to run the tests on a Kubernetes cluster, go to Step 16 on page 807.

14. Follow the instructions if you want to run the tests on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

    a. Click the **LOCATION** tab, if it is not already open.

    The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability and capabilities of the agent are displayed.

    Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

    The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

    b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the performance agent.
The performance agent that is specified in the test assets is available and there are other performance agents added to the project.	Performance agents that have capabilities that match with the performance agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list.	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

If...	Then...	Action
	▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you hover over the agent in the **Override** column.	

c. Click **Execute**.

15. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

a. Click the **RESULTS** tab.

b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported.	Perform the following steps when:

Export options	Results in...	Actions
	Note: This export option is the default option.	▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

16. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the following remote host to your project that are then displayed under the **Override** column:

Kubernetes clusters that are registered with a team space.

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of a Web UI test.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Running server test assets

You can run tests that are listed as server test assets created in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Prerequisites to running tests on page 527, if they apply to the test that you want to configure for a run.
- Read and completed the tasks mentioned in Test run considerations for browser-based Web UI tests on page 532, if you want to run tests that use Chrome or Edge browsers.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.

- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created a Web UI test in Test Hub. See Recording a Web UI test on page 211.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select one of the following tabs based on the type of test assets that you want to run:
   - SUITES
   - TESTS
   - ADVANCED

   The test assets in the selected tab are displayed.
5. Identify the Server Test asset that you want to run by scrolling through the list, searching for the test, or applying a filter.
   Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.
6. Click the **Execute** icon  in the row of the identified test asset.
   The **Execute test asset** dialog is displayed.
7. Select the version of the test resources that you want to run in the **Version** field.
8. Click **Schedule** and select the time if you want to schedule the test run. Else, skip the step.
9. Enter a label for the test run that helps you to identify the test on the **Results** page.
10. Follow the instructions to select a browser:

    a. Click the **ENVIRONMENT** tab, if it is not already open.

    b. Select an environment from the list that is available for the test run from the **Environment** drop-down list.
    The variables and profile overrides in the selected environment override the values defined in the test when you run a test or suite.

The environment most recently selected by the current user is displayed as the pre-selected option in the list. You can select a different environment for the current run.

c. Select a browser or device profile for the test run from the **Browser** drop-down list.
The browser that is used to record the test is selected as the default browser and the browser specified here overrides the default browser and the browser defined in the profile override.

d. Select a security group to be used for assigning the values for secrets from the **Security Group** drop-down list.

11. **Optional:** Click the **DATA SOURCES** and select a dataset tab if you are running a test asset that contains datasets.
12. **Optional:** Click the **VARIABLES** tab, and then add or upload variables if the test requires a variable that must be passed to the test at the test run time.
13. **Optional:** Click the **LOCATION** tab, and then select the agent that you have added in the **Override** field.
After selecting the agent, if the specified browser is not available to the agent, Test Hub displays a warning indicator in the **Environment** tab.
14. **Optional:** Click the **RESULTS** tab, and then perform any the following actions:
    - Select an option if you want to export the test results to a Jira issue in your Xray project in Jira.

      You must have set up Jira as a change management system and configured the Xray integration to export the results.
    - Click **+ Create Property**, and then add values in the **Name** and **Value** columns.

      The key-value pair of properties created here helps you add details about the test, and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed on the **Functional Report** page, **Test Log** page, and in the **Properties** column on the **Results** page.
15. Click **Advanced** to enter any of the following values that must be passed to the test run if they are applicable for the test:
    - **JVM Arguments**
    - **Program Arguments**
    - **Environment Variables**
16. Click **Execute**.
The test run is initiated.

Results

You have run a server test asset that is created in Test Hub.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.

- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Running server test assets on an agent

When you have server test assets that you want to run on a Linux, Mac, or Windows system by using an agent, you can register the agents and run the tests.

Before you begin

You must have completed the following tasks:

- Registered the Linux, Mac, or Windows system using an agent for running tests. See Installing and registering DevOps Test Runtime agents on page 558.
- Read and completed the tasks mentioned in Prerequisites to running tests on page 527, if they apply to the test that you want to configure for a run.
- Read and completed the tasks mentioned in Test run considerations for browser-based Web UI tests on page 532, if you want to run tests that use Chrome or Edge browsers.
- Created a Web UI test in Test Hub.

About this task

When you initiate a test run that requires a capability such as the Safari browser, and if you have not specified any agent to run, the test enters a **Waiting for agent** state. This state indicates that the job is valid but is waiting for a suitable agent to run it. If an agent that has the required capability is registered with the project, then the agent picks up the job and runs the test, which is in an **Waiting for agent** state. You can also cancel or stop the test runs that are in the **Waiting for agent** state.

Note: The default time limit for the agent to be in the **Waiting for agent** state is 15 minutes after which the test run attempt stops. If you started the test run from the command line, then the time limit is 60 mins.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run, as indicated in the following table:

If...	Then...
You want to run Suites.	Click the **SUITES** tab.
You want to run a Web UI test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

   The test assets in the selected tab are displayed.
5. Identify the test asset that you want to run by scrolling through the list, searching for the test, or applying a filter.
6. Click the **Execute** icon  in the row of the identified test asset.
   The **Execute test asset** dialog is displayed.
7. Select the version of the test resources that you want to run in the **Version** field.
8. Follow the instructions to select a browser:

   a. Click the **ENVIRONMENT** tab, if it is not already open.

   b. Select an environment from the list that is available for the test run from the **Environment** drop-down list.
      The variables and profile overrides in the selected environment override the values defined in the test when you run a test or suite.
      The environment most recently selected by the current user is displayed as the pre-selected option in the list. You can select a different environment for the current run.

   c. Select a browser or device profile for the test run from the **Browser** drop-down list.
      The browser that is used to record the test is selected as the default browser and the browser specified here overrides the default browser and the browser defined in the profile override.

   d. Select a security group to be used for assigning the values for secrets from the **Security Group** drop-down list.

9. **Optional:** Click the **DATA SOURCES** and select a dataset tab if you are running a test asset that contains datasets.
10. **Optional:** Click the **VARIABLES** tab, and then add or upload variables if the test requires a variable that must be passed to the test at the test run time.
11. Click the **LOCATION** tab, and then select the agent that you have added in the **Override** field.
    After selecting the agent, if the specified browser is not available to the agent, Test Hub displays a warning indicator in the **Environment** tab.
12. **Optional:** Click the **RESULTS** tab, and then perform any the following actions:
    - Select an option if you want to export the test results to a Jira issue in your Xray project in Jira.

      You must have set up Jira as a change management system and configured the Xray integration to export the results.
    - Click **+ Create Property**, and then add values in the **Name** and **Value** columns.

      The key-value pair of properties created here helps you add details about the test, and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed on the **Functional Report** page, **Test Log** page, and in the **Properties** column on the **Results** page.
13. Click **Execute**.
    The test run is initiated.

Results

You have run server test assets on an agent.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Running server test assets by using a cloud credential

You can run tests that are listed as server test assets created in HCL DevOps Test Hub (Test Hub) on a device cloud by using a cloud credential.

Before you begin

You must have completed the following tasks:

- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Created a cloud credential in your team space that is configured to connect to the mobile device cloud. See Adding cloud credentials of mobile device clouds on page 593.
- Ensured that you are a member of the cloud credential that is configured to connect to the mobile device cloud. See Viewing details of cloud credentials on page 601.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created a Web UI test in Test Hub. See Recording a Web UI test on page 211.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.

About this task

You can run tests on the supported device clouds such as BrowserStack Cloud, LambdaTest Cloud, and pCloudy Cloud.

Restriction: Tests that you record on a standard browser might fail when you play them back on a device cloud. To increase the probability of successful playback, record tests by using device emulation. See supported device profiles on page 211.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select one of the following tabs based on the type of test assets that you want to run:

- TESTS
- ADVANCED

The test assets in the selected tab are displayed.

5. Identify the Server Test asset that you want to run by scrolling through the list, searching for the test, or applying a filter.
   Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.
6. Click the **Execute** icon  in the row of the identified test asset.
   The **Execute test asset** dialog is displayed.
7. Select the version of the test resources that you want to run in the **Version** field.
8. Click **Schedule** and select the time if you want to schedule the test run. Else, skip the step.
9. Enter a label for the test run that helps you to identify the test on the **Results** page.
10. Follow the instructions to select a browser:

    a. Click the **ENVIRONMENT** tab, if it is not already open.

    b. Select an environment from the list that is available for the test run from the **Environment** drop-down list.
    The variables and profile overrides in the selected environment override the values defined in the test when you run a test or suite.

    c. Select a device from the list of devices for the configured device cloud in the **Browser** drop-down list.
    The name of the browser is appended with the name of the device and name of the cloud credential set by the user in the *Browser (Full device name : DeviceCloudName set by the user)* format. For example, if the mobile device is *Galaxy S10* and the cloud credential name is *Lambda*, then the name displayed in the list is *Chrome(Galaxy S10 : Lambda)*.

    d. Select a security group to be used for assigning the values for secrets from the **Security Group** drop-down list.

11. **Optional:** Click the **DATA SOURCES** and select a dataset tab if you are running a test asset that contains datasets.
12. **Optional:** Click the **VARIABLES** tab, and then add or upload variables if the test requires a variable that must be passed to the test at the test run time.
13. **Optional:** Click the **RESULTS** tab, and then perform any the following actions:
    - Select an option if you want to export the test results to a Jira issue in your Xray project in Jira.

      You must have set up Jira as a change management system and configured the Xray integration to export the results.

    - Click **+ Create Property**, and then add values in the **Name** and **Value** columns.

      The key-value pair of properties created here helps you add details about the test, and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed on the **Functional Report** page, **Test Log** page, and in the **Properties** column on the **Results** page.

14. Click **Advanced** to enter any of the following values that must be passed to the test run if they are applicable for the test:
    - **JVM Arguments**
    - **Program Arguments**
    - **Environment Variables**
15. Click **Execute**.
    The test run is initiated.

Results

You have run a server test asset that is created in Test Hub on a device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Running server suite assets

You can run a test suite that is listed as server suite asset created in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Prerequisites to running tests on page 527, if they apply to the test that you want to configure for a run.
- Read and completed the tasks mentioned in Test run considerations for browser-based Web UI tests on page 532, if you want to run tests that use Chrome or Edge browsers.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.

- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created Web UI tests, and then created a test suite in Test Hub. See Recording a Web UI test on page 211 and Creating a suite on page 292.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select one of the following tabs to select the test suite that you want to run:
   - SUITES
   - ADVANCED

   The test assets in the selected tab are displayed.
5. Identify the server suite asset that you want to run by scrolling through the list, searching for the test, or applying a filter.
   Clicking the test name displays the **Details** panel. You can view the details of the test asset such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.
6. Click the **Execute** icon  in the row of the identified test asset.
   The **Execute test asset** dialog is displayed.
7. Select the version of the test resources that you want to run in the **Version** field.
8. Click **Schedule** and select the time if you want to schedule the test run. Else, skip the step.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.
10. Follow the instructions to select a browser:
    a. Click the **ENVIRONMENT** tab, if it is not already open.
    b. Select an environment from the list that is available for the test run from the **Environment** drop-down list.
    The variables and profile overrides in the selected environment override the values defined in the test when you run a test or suite.
    c. Select a browser or device profile for the test run from the **Browser** drop-down list.
    If you do not specify the browser, the browsers that are configured for individual tests in the test suite are used for the test run.
    d. Select a security group to be used for assigning the values for secrets from the **Security Group** drop-down list.
11. **Optional:** Click the **DATA SOURCES** and select a dataset tab if you are running a test asset that contains datasets.
12. **Optional:** Click the **VARIABLES** tab, and then add or upload variables if the test requires a variable that must be passed to the test at the test run time.
13. **Optional:** Click the **LOCATION** tab, and then select the agent that you have added in the **Override** field.
    After selecting the agent, if the specified browser is not available to the agent, Test Hub displays a warning indicator in the **Environment** tab.
14. **Optional:** Click the **RESULTS** tab, and then perform any the following actions:
    - Select an option if you want to export the test results to a Jira issue in your Xray project in Jira.

      You must have set up Jira as a change management system and configured the Xray integration to export the results.
    - Click **+ Create Property**, and then add values in the **Name** and **Value** columns.

      The key-value pair of properties created here helps you add details about the test, and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed on the **Functional Report** page, **Test Log** page, and in the **Properties** column on the **Results** page.
15. Click **Advanced** to enter any of the following values that must be passed to the test run if they are applicable for the test:
    - **JVM Arguments**
    - **Program Arguments**
    - **Environment Variables**
16. Click **Execute**.
    The test run is initiated.

Results

You have run a server suite asset that is created in Test Hub.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Running server suite assets by using a cloud credential

You can run a test suite that is listed as server suite asset created in HCL DevOps Test Hub (Test Hub) on a device cloud by using a cloud credential. The tests in the suite run sequentially on device clouds.

Before you begin

You must have completed the following tasks:

- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Created a cloud credential in your team space that is configured to connect to the mobile device cloud. See Adding cloud credentials of mobile device clouds on page 593.
- Ensured that you are a member of the cloud credential that is configured to connect to the mobile device cloud. See Viewing details of cloud credentials on page 601.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created Web UI tests, and then created a test suite in Test Hub. See Recording a Web UI test on page 211 and Creating a suite on page 292.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.

About this task

You can run tests on the supported device clouds such as BrowserStack Cloud, LambdaTest Cloud, and pCloudy Cloud.

Restriction: Tests that you record on a standard browser might fail when you play them back on a device cloud. To increase the probability of successful playback, record tests by using device emulation. See supported device profiles on page 211.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select one of the following tabs to select the test suite that you want to run:
   - SUITES
   - ADVANCED

   The test assets in the selected tab are displayed.
5. Identify the server suite asset that you want to run by scrolling through the list, searching for the test, or applying a filter.
   Clicking the test name displays the **Details** panel. You can view the details of the test asset such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.
6. Click the **Execute** icon in the row of the identified test asset.
   The **Execute test asset** dialog is displayed.
7. Select the version of the test resources that you want to run in the **Version** field.
8. Click **Schedule** and select the time if you want to schedule the test run. Else, skip the step.
9. Enter a label for the test run that helps you to identify the test on the **Results** page.
10. Follow the instructions to select a browser:

a. Click the **ENVIRONMENT** tab, if it is not already open.

b. Select an environment from the list that is available for the test run from the **Environment** drop-down list.
The variables and profile overrides in the selected environment override the values defined in the test when you run a test or suite.

c. Select a device from the list of devices for the configured device cloud in the **Browser** drop-down list.
The name of the browser is appended with the name of the device and name of the cloud credential set by the user in the *Browser (Full device name : DeviceCloudName set by the user)* format. For example, if the mobile device is *Galaxy S10* and the cloud credential name is *Lambda*, then the name displayed in the list is *Chrome(Galaxy S10 : Lambda)*.

d. Select a security group to be used for assigning the values for secrets from the **Security Group** drop-down list.

11. **Optional:** Click the **DATA SOURCES** and select a dataset tab if you are running a test asset that contains datasets.
12. **Optional:** Click the **RESULTS** tab, and then perform any the following actions:
    - Select an option if you want to export the test results to a Jira issue in your Xray project in Jira.

      You must have set up Jira as a change management system and configured the Xray integration to export the results.
    - Click **+ Create Property**, and then add values in the **Name** and **Value** columns.

      The key-value pair of properties created here helps you add details about the test, and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed on the **Functional Report** page, **Test Log** page, and in the **Properties** column on the **Results** page.
13. Click **Advanced** to enter any of the following values that must be passed to the test run if they are applicable for the test:
    - **JVM Arguments**
    - **Program Arguments**
    - **Environment Variables**
14. Click **Execute**.
    The test run is initiated.

Results

You have run a server suite asset that is created in Test Hub on a device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.

- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of an SAP GUI test

When you want to run an SAP GUI test, then you must configure a run of SAP GUI test or the Compound Test that contains SAP GUI tests on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Created a Compound Test that contains SAP GUI tests in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub. Refer to Task flow for running SAP tests on DevOps Test Hub.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

You can run the SAP GUI test or Compound test that contains the SAP GUI test in Test Hub, only by using the remote performance agent installed with the SAP runtime environment. If you have not installed any remote performance agent with the SAP run time environment, you do not see any agents in the **Override** column and the **Execute** button is disabled so that you cannot run the test.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.

2. Open the project that contains the test assets, and then click **Execution**.
   The test assets in the selected branch of the repository are displayed on the **Execution** page.
3. Perform any of the following steps based on the test you want to run:
   - Click the **TESTS** tab to run the SAP GUI test.
   - Click the **SUITES** tab to run the compound test that contains the SAP GUI test.
4. Identify the test that you want to run and click the **Execute** icon in the row of the identified test asset.

   The **Execute test asset** dialog is displayed.

   Note: If you do not want to continue with the current settings, click **Reset** to reset the settings.

5. Select the version of the test resources that you want to run.
6. **Optional:** Click **Schedule** to select a time for scheduling the test run.
7. **Optional:** Click the **Add a label** icon , type a label to identify the test on the **Results** page, and then press the Enter key.
8. Perform the following steps to select the performance agent and run the SAP GUI tests:

   a. Click the **LOCATION** tab, if it is not already open.

      The performance agents installed with SAP GUI runtime are listed under the **Host** column.

      Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

   b. Select the performance agent where you want to run the test asset in the **Override** column.

9. **Optional:** Click **Advanced** to make the following advanced configurations:

   a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

      For example, you can set a maximum Java heap size.

   b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

      If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

   c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

      For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. **Optional:** Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

    The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

    You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

    Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

    c. Select the dataset that you want to use in the test run from any of the following options:

    - Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

      Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

    - Select the dataset from the list.

      Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

    - Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. **Optional:** Follow the instructions in this step if the test requires a variable that must be passed to the test at the test run time.

    You must configure the supported browser by using a variable if the test has a browser configured, which is different from the one that is supported by Test Hub.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.

        i. Enter `RTW_WebUI_Browser_Selection` as the name.

- To add new variables from your local computer or from the Git repository that is associated with your project, click the **Upload** icon  and choose either **Upload from local system** or **Browse from server** to select the variable file.

  Note: You must have created a file with the variables before you can select the file.

The default value for the variables is either null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. **Optional:** Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported.	Perform the following steps when:

Export options	Results in...	Actions
	Note: This export option is the default option.	▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

13. Click **Execute**.

Results

You have started or scheduled a test run of an SAP GUI test or Compound test that contains SAP GUI test.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Web UI test to run by using variables

After you added the test resources that you created in the desktop client to the project, you can configure a Web UI test to run by using variables from HCL DevOps Test Hub (Test Hub). You must specify the variables for the Web UI test to run on mobile devices that are connected to an agent or a device cloud.

Before you begin

You must have completed the following tasks:

- Created a Web UI test to be run on a mobile device in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Created a variable file if you want to import the variables file. The file must contain the details of the UI Test agent or the device cloud to which the mobile device is connected.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

You can run the Web UI test on a mobile device, emulator, or simulator that is connected to an agent or a device in a device cloud. You must create the test in Test UI, and then add the test asset to your project on Test Hub. You can then run the Web UI test on the devices that are connected to any of the following agents or clouds:

- Remote UI Test agents. Requires the variables at test run time as specified in Variables for the UI Test agent on page 829.
- The BitBar Cloud. Requires the variables at test run time as specified in Variables for the BitBar Cloud on page 829.
- The BrowserStack Cloud. Requires the variables at test run time as specified in Variables for the BrowserStack Cloud on page 829.
- The pCloudy Cloud. Requires the variables at test run time as specified in Variables for the pCloudy Cloud on page 830.
- The Perfecto Mobile Cloud. Requires the variables at test run time as specified in Variables for the Perfecto Mobile Cloud on page 830.

You must provide the details of the UI Test agent or the mobile cloud to which the mobile devices are connected as variables in Step 12 on page 839. You can either enter the variables or use the file in which you entered the variables.

You must refer to the following tables for the variables that are required for a successful run:

Table 25. Variables for the UI Test agent

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use on the UI Test agent. For example, *Chrome(Emulator:Pixel_3a_API_33)*.
appium.server.host	Specify the hostname or IP address of the UI Test agent.
appium.server.port	Specify the port on the UI Test agent that is configured to communicate with Test Hub.

Table 26. Variables for the BitBar Cloud

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use in the BitBar Cloud. For example, *Chrome(BitBar Cloud:Google Pixel 7 Pro)*.
bitbar.apikey	Specify the user token generated for your BitBar Cloud account to authenticate your connection with the BitBar Cloud.
bitbar.host	Specify the hostname of the BitBar Cloud instance.
bitbar.project	Specify the name of the project that contains the recorded test.
bitbar.testrun	Specify a name for the test run that must be displayed in the BitBar Cloud dashboard for the test run.

Table 27. Variables for the BrowserStack Cloud

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use in the BrowserStack Cloud. For example, *Chrome(BrowserStack Cloud:Google Pixel 7 Pro)*.

Table 27. Variables for the BrowserStack Cloud (continued)

Name of the Variable	Action for the Value field
browserstack.host	Specify the URL of the BrowserStack Cloud instance.
browserstack.username	Specify the user name of your BrowserStack Cloud account.
browserstack.apikey	Specify the API key of your BrowserStack Cloud account to authenticate your connection with the BrowserStack Cloud.
browserstack.project	Specify the name of the project that contains the recorded test.  **Note:** This variable is optional.

Table 28. Variables for the pCloudy Cloud

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use in the pCloudy Cloud. For example, *Chrome(pCloudy Cloud:Google Pixel 7 Pro)*.
pcloudy.apikey	Specify the API key that is generated for your pCloudy Cloud account to authenticate your connection with the pCloudy Cloud.
pcloudy.username	Specify the username with which you can access the pCloudy Cloud instance.
pcloudy.host	Specify the hostname of the pCloudy Cloud instance.

Table 29. Variables for the Perfecto Mobile Cloud

Name of the Variable	Action for the Value field
RTW_WebUI_Browser_Selection	Specify the web browser and the device to use in the Perfecto Mobile Cloud. For example, *Chrome(Perfecto Mobile Cloud:Google Pixel 7 Pro)*.

Table 29. Variables for the Perfecto Mobile Cloud (continued)

Name of the Variable	Action for the Value field
perfecto.securitytoken	Specify the user token generated for your Perfecto Mobile Cloud account to authenticate your connection with the Perfecto Mobile Cloud.
perfecto.host	Specify the hostname of the Perfecto Mobile Cloud instance.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **TESTS** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:
   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.

- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name
     - Test Asset Path
     - Last Result
     - Next Run
     - Components

  f. Click **Save** to save and apply the filter query to filter the assets based on the query.

     The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon  in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

◦ If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.

◦ If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

◦ Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

◦ Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

◦ If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.

◦ If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

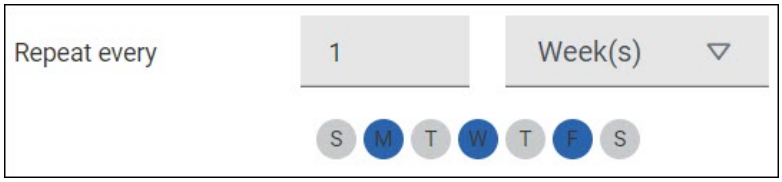
◦ No action is required if you want to initiate the test run immediately after you click **Execute**.

Important: Click **Execute** only after you have configured the other settings in this dialog.

Note: The default time for scheduling a run is **Now**.

◦ Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**.

Schedule	Actions
	The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

       If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

    c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

       For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

       The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.

Option	Description
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Perform the following steps to provide the variables that specify the UI Test agent or cloud to which the mobile device is attached. You can either enter the variables that must be passed to the test at the test run time or import the file that contains the variables.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed.

Export options	Results in...	Actions
	The results are exported to the selected issue that exists in your Xray project in Jira.	ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Perform any of the following actions that depend on whether you are using a remote performance agent or a mobile device cloud:
    - Go to Step 15 on page 840, if you are using an agent.
    - Go to Step 16 on page 842, if you are using a mobile device cloud.
15. Follow the instructions if you want to run the tests on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:
    a. Click the **LOCATION** tab, if it is not already open.

    The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability and capabilities of the agent are displayed.

Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	**Then...**	**Action**
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the performance agent.
The performance agent that is specified in the test assets is available and there are other performance agents added to the project.	Performance agents that have capabilities that match with the performance agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently.	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

If...	Then...	Action
	You can find details of both performance agents for the matching capabilities when you hover over the agent in the **Override** column.	

c. Click **Execute**.

16. Click **Execute**.
    The test run is initiated.

Results

You have started or scheduled a test run of a Web UI test to run on a device that is connected to an agent or a device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Web UI test to run by using a cloud credential

After you added the test resources that you created in the desktop client to the project, you can configure a Web UI test to run on a device cloud by using a cloud credential from HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Created a Web UI test in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Read Test run considerations for using mobile device clouds on page 540, if you want to configure a run on a mobile device cloud.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.
- Created a cloud credential in your team space that is configured to connect to the mobile device cloud. See Adding cloud credentials of mobile device clouds on page 593.
- Ensured that you are a member of the cloud credential that is configured to connect to the mobile device cloud. See Viewing details of cloud credentials on page 601.

About this task

You can run a Web UI test on a device that is connected to a device cloud. You must create the Web UI test in Test UI, and then add the test asset to your project on Test Hub.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Click the **TESTS** or the **ADVANCED** tab.
5. Identify the test asset that you want to run by performing any of the following actions:

◦ Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	VU Schedule			

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon  in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name

- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.
  - Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
  - Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.
- Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:
- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

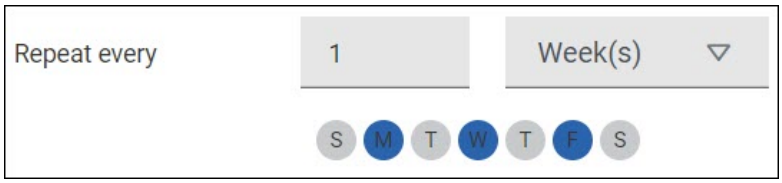
- No action is required if you want to initiate the test run immediately after you click **Execute**.

  Important: Click **Execute** only after you have configured the other settings in this dialog.

  Note: The default time for scheduling a run is **Now**.

- Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:

Schedule	Actions
	 e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Perform the following steps in the **ENVIRONMENT** tab:

    a. Select the cloud credential of the mobile device cloud from the **Device cloud credentials** list.

       The list displays the cloud credentials that are configured in the team space. The name of the cloud credential is appended to the name of the mobile device cloud in the list. For example, if the mobile device cloud provider is *BitBar* and the name of the credential is *BB_Account1*, then the name displayed in the list is *Bit Bar - BB_Account1*.

Note: Access to credentials for mobile device cloud services depends on the status of the provider. If you disable a provider on the **Cloud Credentials** page, you cannot access that device cloud provider from the **Device cloud credentials** list.

If the connection to the mobile device cloud is successful, the **Mobile devices** field is displayed.

b. Select the devices on which you want to run the tests from the list of devices that are displayed in the **Mobile devices** list.

Note: If you are connected to either the BitBar Cloud or BrowserStack Cloud, the **Projects** field is displayed.

c. Select the project on the mobile device cloud that you want to use or create a project in the **Projects** field. The project overrides the project that is configured in the test asset.

11. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

    For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

    If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

    c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

    For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.

Option	Description
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported.	Perform the following steps when:

Export options	Results in...	Actions
	Note: This export option is the default option.	▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Click **Execute**.
The test run is initiated.

Results

You have started or scheduled a test run of a Web UI test to run on a device that is connected to a device cloud.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Test run configurations on page 610

Management of cloud credentials on page 587

Resetting the configuration settings for a test run on page 931

Configuring a run of a Windows test

After you added the test resources that you created in the desktop client to the project, you can configure a Windows test to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Created a Windows test in HCL DevOps Test UI (Test UI) and added the test asset to the project repository on Test Hub.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

You can run the Windows tests after you create the tests in Test UI, and then add them to your project on Test Hub.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.

Tab	Description
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

   The test assets in the selected tab are displayed.

5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

- **Private**: Makes the filter available only to the owner of the filter who created it.
- **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon  in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

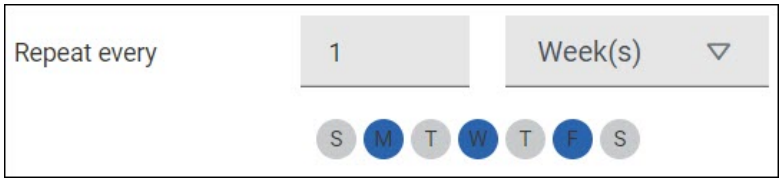
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps:

Schedule	Actions
	a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

       If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

    c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

       For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

       The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

       You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions.

Option	Description
	Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Perform the following steps to provide the variables that must be passed to the test at the test run time. You can either enter the variables or import the file that contains the variables.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.

        i. Enter `Windows_Device_Selection` as the name.
        ii. Enter `<IP_address>` as the value, which is the IP address of the computer that contains the UI Test agent.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed.

Export options	Results in...	Actions
	selected issue that exists in your Xray project in Jira.	ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Select any of the following steps that depends on the location to run the test:
    - If you want to run the tests on the remote agents, go to Step 15 on page 864.
    - If you want to run the tests on a Kubernetes cluster, go to Step 16 on page 866.
15. Follow the instructions if you want to run a test on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

    a. Click the **LOCATION** tab, if it is not already open.

    The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability of the performance agent is displayed.

Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the agent.
The agent that is specified in the test assets is available and other agents added to the project.	Performance agents that have capabilities that match with the agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

If...	Then...	Action
	hover over the performance agent in the **Override** column.	

c. Click **Execute**.

The test run is initiated.

16. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the following remote host to your project that are then displayed under the **Override** column:

Kubernetes clusters that are registered with a team space.

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of a Windows test.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a JMeter test

After you added the test resources that you created in JMeter to the server project, you can configure a JMeter test to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Test run considerations for JMeter tests on page 532.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that the JMeter extension on Test Hub is enabled before you configure a test run.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

Important: You can run the JMeter tests only on the default cluster of Test Hub or a remote Kubernetes cluster. You cannot run the JMeter tests on remote agents that are registered with Test Hub.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

   The test assets in the selected tab are displayed.

5. Identify the test asset that you want to run by performing any of the following actions:

◦ Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon  in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name

- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.
  - Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
  - Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.
- Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:
   - Expand the list in the **Version** field, find the version of the test resources, and then select the version.

     Use the following details about the version of the test resources that are displayed to identify the version that you want:
     - Commit message.
     - Tags labeled by the user for the version committed.
     - The user who committed the version to the repository.
     - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

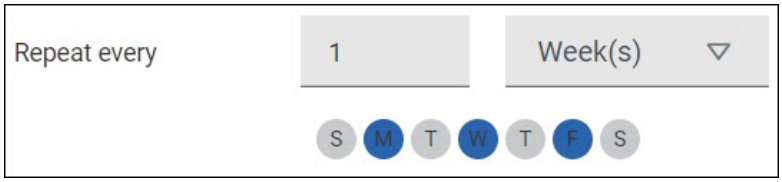
     The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
   - Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

     The version that matches the search criteria is displayed and it is selected for the test run.
8. Select the time for scheduling the test run from the following options:
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.
   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start.

Schedule	Actions
	d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       For example, you can set a maximum Java heap size.

    b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

    For example, enter **-J** to override any JMeter property that is defined in the properties file.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed.

Export options	Results in...	Actions
	selected issue that exists in your Xray project in Jira.	ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

12. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the registered remote Kubernetes clusters to your project that are then displayed under the **Override** column:

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of a JMeter test.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a JUnit test

After you added the test resources that contain the Maven project with the JUnit tests to the server project, you can configure a JUnit test run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Test run considerations for JUnit tests on page 534 before you configure a JUnit test run.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that the JUnit extension on Test Hub is enabled before you configure a test run. See Enabling server extensions on page 171.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

Important: You can run the JUnit tests only on the default cluster of Test Hub or a remote Kubernetes cluster. You cannot run the JUnit tests on remote agents that are registered with Test Hub.

After the test runs, you can view the results of the JUnit test in the following formats:

- Surefire
- JUnit XML Report

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.

Tab	Description
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

   The test assets in the selected tab are displayed.

5. Identify the test asset that you want to run by performing any of the following actions:

   - Scroll through the list.

   Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

   Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

- **Private**: Makes the filter available only to the owner of the filter who created it.
- **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon  in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

8. Select the time for scheduling the test run from the following options:

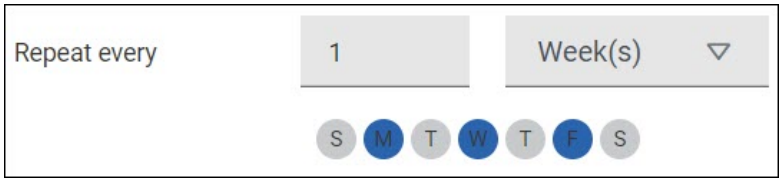
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option.

Schedule	Actions
	You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       You can enter a variable that points to the *MAVEN_OPTS* property.

b. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

   For example, if you have created the JUnit tests by using JDK V12, then you must enter the variable as **JDKOTHER** and the value as **jdkv12**, so that the test run uses the JDK in the path `/data/junit-ext/jdks/jdkv12`.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported.	Perform the following steps when:

Export options	Results in...	Actions
	Note: This export option is the default option.	▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

12. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the registered remote Kubernetes clusters to your project that are then displayed under the **Override** column:

Notes:
- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of a JUnit test.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Postman test

After you added the test resources that you created and exported from Postman to the server project, you can configure the Postman tests to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Test run considerations for Postman tests on page 536.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that the Postman extension in Test Hub is enabled before you configure a test run.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

Important: You can run the Postman tests only on the default cluster of Test Hub or a remote Kubernetes cluster. You cannot run the Postman tests on remote agents that are registered with Test Hub.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

   The test assets in the selected tab are displayed.
5. Identify the test asset that you want to run by performing any of the following actions:

◦ Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	VU Schedule			

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon  in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon  in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.
  e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:

     - Type
     - Test Asset Name

- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.
  - Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
  - Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.
- Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, `AFT XML` files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:
- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

8. Select the time for scheduling the test run from the following options:

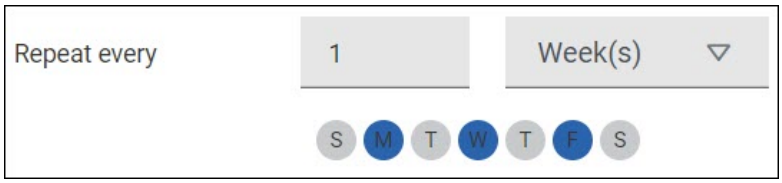
- No action is required if you want to initiate the test run immediately after you click **Execute**.

  Important: Click **Execute** only after you have configured the other settings in this dialog.

  Note: The default time for scheduling a run is **Now**.

- Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:

Schedule	Actions
	 e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced**, and then enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

    You can enter the command options that you use when you run Postman tests by using Newman, as the program arguments.

    For example, enter **--verbose**, if you want the details of the test run and each request sent.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

    c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
    The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

12. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the registered remote Kubernetes clusters to your project that are then displayed under the **Override** column:

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of a Postman test.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Rate Schedule or VU Schedule

After you added the test resources that you created in the desktop client to the project, you can configure a Rate Schedule or VU Schedule to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Created Schedules in HCL DevOps Test Performance (Test Performance) and added the test asset to the project repository on Test Hub.
- Read Test run considerations for schedules on page 538, if you want to configure a run for Rate Schedules or VU Schedules.
- Read Test run considerations for using Machine learning capabilities on page 534, if you want to configure a run for either a Performance Test or a Schedule that must be analyzed by using Machine Learning capabilities in Test Hub.
- Read and completed the tasks mentioned in Test run considerations for running tests on remote performance agents on page 544, if you want to use a remote agent as the location to run the test.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Identify the test asset that you want to run by performing any of the following actions:
    - Scroll through the list.

      Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

      Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.

- **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

d. Select an operator, and then add a rule or a group of rules.
e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

◦ Modify a saved filter query by performing the following steps:

- Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

- Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
- Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.

◦ Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

5. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

6. Select the version of the test resources that you want to run by completing any of the following actions:

Note: The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects that are created in any of the desktop products.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled by the user for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch of the repository. If you do not select any version, then the latest version is selected for the test run.

Notes:

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✖ icon. As a result, the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you chose for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✖ icon.

7. Select the time for scheduling the test run from the following options:

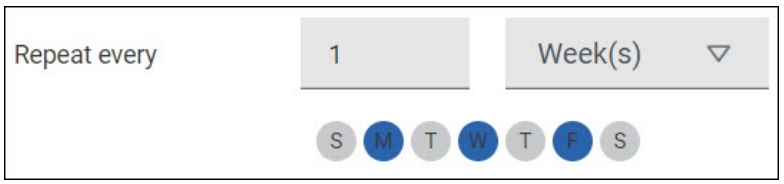
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.

   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour

Schedule	Actions
	▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options: ▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced** to make the following advanced configurations:

a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

   For example, you can set a maximum Java heap size.

b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

   If you want the test log for the test, you must enter `-history testlog` in the **Program Arguments** field.

c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

   For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Follow the instructions if you are running a test asset that contains datasets:

    a. Click the **DATA SOURCES** tab, if it is not already open.

    b. Consider the following information about datasets before you select a dataset:

       The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

       You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.

       Remember: You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

    c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.

  Note: If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.

  Note: If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.

  Important: If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

- Select a data definition from the **Override** list.
- Select one of the following options from the **Generation Type** list:

Option	Description
Standard	Select this option to generate standard test data. Specify the **Number of rows** to generate test data, and optionally, specify the **Seed value**.
Negative	Select this option to generate negative test data to test how the application handles error conditions. Select the **One per row** check box to generate one negative data per row, and optionally specify the **Seed value**.

- Click **Set** to apply the selected data generation settings.
- Click **Execute** if you want to run the test immediately or at the scheduled time. Otherwise, continue with the next step.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

    a. Click the **VARIABLES** tab, if it is not already open.

    b. Choose one of the following methods to add the variables:

    - To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
    - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

    Note: You must have created a file with the variables before you can select the file.

    The default value for the variables is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you want to override the Resource Monitoring labels in the test asset with the Resource Monitoring labels that you created in Test Hub:

    a. Click the **Resource Monitoring** tab.

    Note:

    The **Resource Monitoring** tab displays only if you enabled the **Resource Monitoring from Service** option in Test Performance when you created the test asset.

    b. Click , and press the `Ctrl` + `Space bar` keys, or enter the initial letter of a label to select a label in the list.

    Note:

    You can select or add labels only if you added the labels in the **Resource Monitoring Sources** page in Test Hub.

13. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

14. Select any of the following steps that depends on the location to run the test:
    - If you want to run the tests on the remote agents, go to Step 15 on page 906.
    - If you want to run the tests on a Kubernetes cluster, go to Step 16 on page 908.
15. Follow the instructions if you want to run a test on the remote performance agent that is connected either to the default Kubernetes cluster or the remote Kubernetes cluster:

a. Click the **LOCATION** tab, if it is not already open.

The performance agents that are configured in the test asset are listed under the **Host** column. The information about the availability of the performance agent is displayed.

Note: You must have added performance agents to your project from the **Agents and Intercepts** page for the performance agents to be displayed under the **Override** column.

The default value for the performance agents is null or an empty field if no performance agents were configured in the test asset. If the test asset contains performance agents that are configured, then the default performance agent is the first item to be displayed on the list of performance agents listed in increasing alphabetical order.

b. Select the performance agent where you want to run the test asset in the following scenarios:

If...	**Then...**	**Action**
The performance agent that is specified in the test assets is available and no other performance agents are added to the project.	The performance agent that is specified in the test assets is displayed in the **Override** column.	Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test.
The performance agent that is specified in the test assets is not available and no other performance agents are added to the project.	There is no performance agent displayed for override in the **Override** column.	You cannot run the test on the performance agent. You must wait until the performance agent is available to run the test on the agent.
The agent that is specified in the test assets is available and other agents added to the project.	Performance agents that have capabilities that match with the agent capabilities specified in the test assets are listed in the **Override** column as follows: ▪ Performance agents with capabilities that best match the capabilities of the agent in the asset are at the top of the list. ▪ Performance agents with capabilities that do not	Perform any of the following actions: ▪ Select the performance agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select a performance agent in the **Override** column as the location to run the test.

If...	Then...	Action
	match with the capabilities of the agent in the asset are listed subsequently. You can find details of both performance agents for the matching capabilities when you hover over the performance agent in the **Override** column.	

c. Click **Execute**.

The test run is initiated.

16. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the following remote host to your project that are then displayed under the **Override** column:

Kubernetes clusters that are registered with a team space.

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of a Rate Schedule or VU Schedule.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuring a run of a Selenium test

After you added the test resources that contain the Maven project with the Selenium tests to a server project, you can configure a Selenium test to be run on HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster.

Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in Test run considerations for Selenium tests on page 539 before you configure a Selenium test run.
- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Completed the Xray integration, if you want to export the test results to a Jira issue in your Xray project in Jira. See Integration with Xray test management for Jira on page 1188.
- Ensured that the JUnit extension on Test Hub is enabled before you configure a test run. See Enabling server extensions on page 171.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.

- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Added the remote Git repository that contains the test resources to the project. See Adding repositories to a server project on page 1214.

About this task

Important: You can run the Selenium tests only on the default cluster of Test Hub or a remote Kubernetes cluster. You cannot run the Selenium tests on remote agents that are registered with Test Hub.

After the test runs, you can view the results of the Selenium test in the following formats:

- Surefire
- JUnit XML Report

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the test assets, and then click **Execution**.
3. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

4. Select the tab based on the type of test assets that you want to run as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules.	Click the **SUITES** tab.
You want to run an individual API test, functional test, or performance test.	Click the **TESTS** tab.

If...	Then...
You want to find Suites and tests by using custom filters, and then run Suites and tests.	Click the **ADVANCED** tab.

The test assets in the selected tab are displayed.

5. Identify the test asset that you want to run by performing any of the following actions:
   - Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests that are in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select the **Visibility** of the filter.

     Choose one of the following visibility levels:

     - **Private**: Makes the filter available only to the owner of the filter who created it.
     - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
     - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

     Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

     Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

  d. Select an operator, and then add a rule or a group of rules.

e. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

f. Click **Save** to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.
  - Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
  - Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.
- Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you want to run.

Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.

6. Click the **Execute** icon in the row of the identified test asset.

The **Execute test asset** dialog is displayed.

Notes:

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

7. Select the version of the test resources that you want to run by completing any of the following actions:
   - Expand the list in the **Version** field, find the version of the test resources, and then select the version.

     Use the following details about the version of the test resources that are displayed to identify the version that you want:
     - Commit message.
     - Tags labeled by the user for the version committed.
     - The user who committed the version to the repository.
     - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

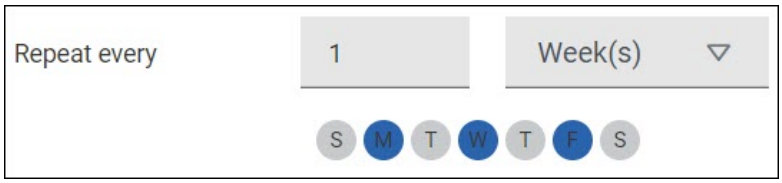
     The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.
   - Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

     The version that matches the search criteria is displayed and it is selected for the test run.
8. Select the time for scheduling the test run from the following options:
   - No action is required if you want to initiate the test run immediately after you click **Execute**.

     Important: Click **Execute** only after you have configured the other settings in this dialog.

     Note: The default time for scheduling a run is **Now**.
   - Select **Schedule** and perform the following actions if you want to schedule a single test run or configure recurring test runs:

Schedule	Actions
A single test run	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date.

Schedule	Actions
	c. Select the time at which the test run must start. d. Select the **Never** option in the **Repeat every** field. e. Click **Save**. The test run is scheduled to run at the selected time on the scheduled date.
Recurring test runs	Perform the following steps: a. Click the **Calendar** icon in the row of the **Start** field. b. Select the date. c. Select the time at which the test run must start. d. Set the frequency at which the test runs must run by entering the number in the **Repeat every** field, and then select the period from the list. You can select from the following options: ▪ Minute ▪ Hour ▪ Day ▪ Week ▪ Month For example, if you want the test run to be run every day at the set time, enter `1` and, then select the **Day(s)** option. You can also schedule the test to run at the set time on specific weekdays by selecting the **Week(s)** option. For example, the following image displays the days selected for the test runs as *Monday*, *Wednesday*, and *Friday*:  e. Set the option to end the recurring test runs from the following options:

Schedule	Actions
	▪ Select the **Never** option, if you do not want the recurring test runs to end. ▪ Select the **On** option, and then click the **Calendar** icon. You can select the date and time after which the scheduled test runs do not run. f. Click **Save**. The recurring test runs are scheduled to start the first run at the selected time on the scheduled date, and to repeat the run at the frequency that you set.

9. Enter a label for the test run that helps you to identify the test on the **Results** page.

   After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

   The default value for the **Label** field is null or an empty field.

   Important: The configuration that you set for the test run in the **Execute test asset** dialog is preserved when you run the same test again. Those changes are not visible when another user logs in to Test Hub. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

   If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

       You can enter a variable that points to the *MAVEN_OPTS* property.

    b. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

       For example, if you have created the Selenium tests by using JDK V12, then you must enter the variable as **JDKOTHER** and the value as **jdkv12**, so that the test run uses the JDK in the path `/data/junit-ext/jdks/jdkv12`.

    Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

    The default value for each of the fields for the advanced settings is null or an empty field.

    If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you want to export the test results to a Jira issue in your Xray project in Jira:

    a. Click the **RESULTS** tab.

    b. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: i. Select the **Jira Xray** option. The **Select issue** field is displayed. ii. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. iii. Select the issue to which you want to export the test results. iv. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: i. Select the **New Jira item** option. ii. Continue with the next step.
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ▪ You do not want to export the test results. ▪ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. i. Select the **None** option. ii. Continue with the next step.

    c. Click **+ Create Property**, and then add values in the **Name** and **Value** columns.
    The key-value pair of properties created here helps you add details about the test and they are displayed in test results. You can add multiple properties. For example, you can add `buildID` and `DTHub202609070223-2f3d81f80f` as values. The values entered in these fields are displayed in the **Properties** column on the **Results** page.

12. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

Important: You must have added the registered remote Kubernetes clusters to your project that are then displayed under the **Override** column:

Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the test asset.

c. Click **Execute**.

The test run is initiated.

Results

You have started or scheduled a test run of a Selenium test.

What to do next

You can perform any of the following tasks:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- Cancel a scheduled test run from the **Execution** page. See Canceling scheduled test runs on page 934.
- View all the states of the test asset by clicking the **Show in the Progress page** icon for the test asset for which you started or scheduled the run. See Viewing the state of test assets on page 922.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Related information

Resetting the configuration settings for a test run on page 931

Test run configurations on page 610

Configuration of test runs in a GitHub Actions Workflow

You can configure test runs of test resources that are in a project repository of HCL DevOps Test Hub (Test Hub) in a GitHub Actions Workflow. You can choose to run test resources that are in your server project in the GitHub Workflow when you do not want to run the test resources without opening the Test Hub project.

Before you configure a test run for any test resources that are in a project repository of Test Hub, you must complete certain tasks. See Prerequisites to run tests in a GitHub Actions Workflow on page 919.

After you complete the prerequisite tasks, you can proceed to configure a test run in a GitHub Workflow. See Configuring a test run in a GitHub Actions Workflow on page 920.

Prerequisites to run tests in a GitHub Actions Workflow

Before you configure a test run for any test resources that are in a project repository of HCL DevOps Test Hub (Test Hub), you must complete certain tasks to run the test resources in a GitHub Actions Workflow.

You must be familiar with creating projects and running test resources from Test Hub.

You must complete the following tasks to set up and use Test Hub:

- Install Test Hub. See Installation of DevOps Test Hub on page 115.
- Possess valid licenses to use Test Hub. See Management of licenses in DevOps Test Hub on page 182.
- Create a project on Test Hub. See Creating a project on page 1213.
- Add a repository that contains test resources to the project repository. See Adding repositories to a server project on page 1214.
- Generate the Offline User Token on Test Hub that is required for connecting to Test Hub from another application such as the GitHub Action. See Access management for DevOps Test Hub on page 1194.
- Open the project to view the details of the test resources that you want to run in a GitHub Actions Workflow. You must make a note of the test resource details from the **Execution** page of Test Hub.

  You must note the following details of the test resource that are to be provided in the YAML file for the GitHub Action:
  - The URL of the Test Hub instance.
  - The name of the team space that contains the project on Test Hub.
  - The name of the project on Test Hub that contains the test resources.
  - The branch name of the project repository that contains the test resources.
  - The ID of the test resource.

You must complete the following tasks outside of Test Hub:

- Create an empty Git repository on any GitHub Enterprise server, if you do not already use Git repositories to store the test resources that you create. For more information about creating Git repositories, refer to Creating a new repository.

  If you already use Git repositories to store test resources created, you can use the same Git repositories.

  Note: You must have the required privileges in the Git repository to create a GitHub Actions Workflow.

- Enable the GitHub Actions option in the Git repository, if it is not available as an option to use. For more information about enabling the GitHub Actions, refer to Learn about GitHub Actions.

You can continue to configure a run of a test, see Configuring a test run in a GitHub Actions Workflow on page 920.

Configuring a test run in a GitHub Actions Workflow

After you complete the prerequisite tasks to run tests in a GitHub Actions Workflow, you can configure the test run in a GitHub Actions Workflow. The test is in a project on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Completed the tasks that are mentioned in Prerequisites to run tests in a GitHub Actions Workflow on page 919.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Extended the CA list. See Extending the trusted certificate authority list on Linux on page 164 or Extending the trusted certificate authority list on Windows on page 164.

1. Perform the following steps to open the Git repository that you use or created, which has the GitHub Actions option enabled:
   a. Open the Git repository on your GitHub Enterprise server.
   b. Create a folder with the name as `.github/workflows` in the root folder of the repository.
   c. Create an empty YAML file in the `.github/workflows` folder.

      The YAML file is used to define the GitHub Action in the Actions Workflow.
2. Perform the following steps to define the YAML file for the GitHub Action:
   a. Open the GitHub Marketplace (https://github.com/marketplace?type=) in a web browser.
   b. Search for *HCL DevOps Test Hub GitHub Action*.
   c. Open the GitHub Action.
   d. Read through the information provided.

e. Copy the contents of the YAML file that is provided in the example by clicking the copy icon.

f. Open the YAML file that is in the `.github/workflows` folder of your repository and paste the copied contents.

g. Enter the valid details as values for the required attributes in the YAML file by following the actions in the following table:

Attribute	Action
Required attributes	
serverURL	Specify the URL of the Test Hub instance.
offlineToken	Specify the Offline User Token that is generated by Test Hub.
teamspace	Specify the name of the team space that contains your project on Test Hub.
project	Specify the name of your project on Test Hub.
branch	Specify the name of the branch in the repository that contains the test resources, which can be found in the **Details** pane of the test resource on the **Execution** page.
assetId	Specify the ID of the test resource, which can be found under the **Git details** section in the **Details** pane of the test resource on the **Execution** page.
Optional attributes	
environment	Specify the environment that is configured in the test.
datasets	Specify the dataset that must be used at test run time, which replaces the dataset that is configured in a test.
labels	Specify text labels to identify a specific test result when the test run is complete.
secretsCollection	Specify the name of the secret collection that is configured in a test.
variables	Specify the name of the variable and its value if your test requires variables during the test run time. **Note:** You can add multiple variables that are separated by a semicolon.

h. Save the YAML file.

3. Perform the following steps to set up a Runner for the GitHub Action in GitHub:

a. Click the Settings icon.

b. Click **Actions > Runners > New self-hosted runner**.

c. Follow the on-screen instructions to set up a self-hosted runner on your computer.

After the runner is installed on your computer, you must verify that the runner is ready to run jobs in the GitHub Actions Workflow and is in the *Active* state. The runner is listed in the **Runners** pane in the Git repository.

4. Perform the following steps to start the GitHub Action in the Workflow:

a. Click the **Actions** option in the Git repository.

The GitHub Action that you configured in the YAML file for Test Hub is displayed as an **Action** under **Workflows**.

b. Click the *HCL DevOps Test Hub GitHub Action*, and then click **Run workflow**.

Results

You have configured and run a resource that is in your project on Test Hub in a GitHub Action Workflow.

What to do next

You can perform any of the following tasks on the Git repository page by clicking the job that was completed:

- View the status of the test run by expanding the **Execute test** section.
- View the links to the test results and reports listed in the **Execute test** section.

You can perform any of the following tasks from Test Hub:

- Stop the test run at any point after the test run is initiated from the **Execution** page. See Stopping test runs on page 932.
- View the progress of the test from the **Progress** page. See Viewing the progress of running test assets on page 926.
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See Test results on page 983.

Management of running tests

Find information about the tasks that you can perform on a test that you configured for a run either while it runs or after it completes the run.

Viewing the state of test assets

After you initiate or schedule a test run on the **Execution** page, you can view the state of the test assets on the **Progress** page by clicking the **Show in the Progress page** icon that is enabled for that test asset on the **Execution** page. You can view all the states of the test asset from the past hour such as `In transition`, `Running`, `Scheduled`, `Completed`, or `Stopped`.

Before you begin

You must have initiated runs of the test assets in your project from the **Execution** page.

Note: The **Show in the Progress page** icon is enabled for the test asset only if a run of the test asset is either started or scheduled.

1. Click **Execution**, if you are not already on the **Execution** page.
2. Select the branch of the repository that contains the test assets.

   The test assets that are contained in the selected branch of the repository are displayed in the following tabs on the **Execution** page:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

3. Identify the test asset or resource that you want to run by performing any of the following actions:
   - Scroll through the list.

     Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

     Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** field. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all functional tests in the selected branch of the repository.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select an operator, and then add a rule, or a group of rules.
  d. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

e. Save the filter query to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

a. Click the **Filter** icon in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

b. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

4. Click the **Show in the Progress page** icon in the **Actions** column of the test asset.

The **Progress** page is displayed with the following details:

- The different states of the test asset runs that were initiated or completed in the past hour are displayed as follows:
  - `In transition`
  - `Running`
  - `Completed`
  - `Stopped by User`
  - `Canceled`
- The ID of the selected test asset is applied as a filter and is displayed in the filter row.

The applied filter is removed along with the displayed items for this filter when you perform any of the following actions on the **Progress** page:

- You click **Hide inactive**, and then click **Show all**.
- You visit any other page in the Navigation pane and return to the **Progress** page.

Results

You have viewed all the states of a specific test asset runs that were initiated during the past hour.

What to do next

You can choose to perform any of the following tasks from the **Progress** page:

- Stop test assets that are running. See Stopping test runs on page 932.
- Cancel scheduled runs. See Canceling scheduled test runs on page 934.
- View the results or the execution log of the test asset from the **Actions** column by clicking the **Open action menu** icon ⋮. See Checking logs on page 930.
- View the results, reports, and logs of the completed test run from the **Results** page. See Test results on page 983.

Viewing the progress of running test assets

After you initiate or schedule a test run on the **Execution** page, you can view the progress of the test assets that are running or scheduled to run, from the **Progress** page. You can also view the test assets that have completed their run, are stopped or cancelled, during the past hour.

Before you begin

You must have initiated runs of the test assets in your project from the **Execution** page.

1. Open your project and click **Progress**.

   Any one of the following views is displayed:
   - No test assets are displayed in the **Progress** page if there are no test assets that are running, scheduled to run, or have completed their run in the past hour.
   - Test assets in your project that are in any of the following states are listed:
     - Running
     - Scheduled
     - Completed
     - Stopped
     - Canceled
   - No test assets are displayed but the option **Show all** is enabled. You can click **Show all** to display the test assets that ran in the past hour. Test assets that were hidden from the display when **Hide inactive** was clicked are displayed.

   Note: Test assets are automatically removed from the display after *60 minutes* of being added to the **Progress** page. You can hide the test assets from the display by clicking **Hide inactive**, so that you can view only the test assets that were added to the **Progress** page in the past hour. You cannot use the **Progress** page to view the history of test assets that were run from the **Execution** page.

2. Identify the test asset by performing any of the following actions:

- Identify the test asset or resource by scrolling through the list.
- Search for the test asset by entering any text contained in the test asset name in the **Search** text box. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
- Create a filter query by using the **New filter** option by performing the following steps:
    a. Click **New filter**.
    b. Select the **Visibility** of the filter.

    Choose one of the following visibility levels:

    - **Private**: Makes the filter available only to the user who created it.
    - **Project**: Makes the filter available to all members of the current project. Only project owners can edit project filters.
    - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit team space filters.

    Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

    Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

    c. Select an operator, and then add a rule, or a group of rules.
    d. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

    You can select a parameter from the following list:

    - Type
    - Name
    - Started by
    - Started on
    - Finished on
    - Status
    - Verdict

    e. Apply the filter query to filter the assets based on the query.

    The test assets that match the filter criteria are displayed.

    f. Save the filter query by performing the following steps, if you want to reuse the filter query later:

    i. Click **Save**.
    ii. Enter a name for the filter query.
    iii. Click **Save**.

- Modify a saved filter query by performing the following steps:
  - Click the **Filter** icon  in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.
  - Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
  - Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.
- Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:

  Note: To open the filter query, you must have created and saved a filter query.

  a. Click the **Open filters** icon  .
  b. Select the saved filter in the **Filters** dialog.
  c. Click **Apply** to apply the filter.

  The test assets that match the filter criteria are displayed.
- Sort the test assets by the user who started the test by clicking the **Started by** column header, and find the test asset listed against your name.
- Sort the test assets by the type of the test assets by clicking the **Type** column header, and find the test asset type you ran or scheduled.
- Click **Hide inactive** to only display the test assets that are running or scheduled to run. This action removes all completed, canceled, stopped, or failed test assets from the display.

Note: The items in the rows are sorted in an ascending or descending order when the column header is clicked. The following icons show the status of sorting in a column:

-  items in the column are not sorted.
-  items in the column are sorted in the ascending order or with the earliest record on top.
-  items in the column are sorted in the descending order or with the latest record on top.

You can view the following details of the test assets on the **Progress** page:

Column header	Description
Type	Displays the icon for the type of the test asset.
Name	Displays the name of the test asset added to the project from the Git repository. Clicking the test name displays the **Details** panel. You can view the details of the test such as the description, the branch in the repository that contains the asset, the Git details, and the details of

Column header	Description
	the commits to the repository. You can also view the history of the test runs for the specific test under the **History** tab of the **Details** panel.
Started by	Displays the name of the user who started the run of the test asset.
Started on	Displays the date and time when the test run was started.
Finished on	Displays the date and time when the test run completed the run.
Status	Displays the state of the test asset progressively. For example, after the run is in the `Initiated` state, it moves on to the `Running` state and ends in its final state depending on the verdict of the test run as either `Completed`, `Failed`, or `Inconclusive`. If the test run is stopped or a scheduled run is canceled, the status displayed is `Stopped by User` for a stopped run or `Canceled` for a canceled test.

From the **Actions** column, you can perform the following operations on your test asset:

- Stop a run.
- Cancel a scheduled run.
- Cancel scheduled recurring test runs.
- View the test log.
- View the result of the test asset on the **Results** page.
- Query the test result data.

Results

You have viewed the progress of the test assets that you ran from the **Execution** page.

What to do next

You can choose to perform any of the following tasks from the **Progress** page:

- Stop test assets that are running. See Stopping test runs on page 932.
- Cancel scheduled runs. See Canceling scheduled test runs on page 934.
- View the results or the execution log of the test asset from the **Actions** column by clicking the **Open action menu** icon  . See Checking logs on page 930.
- View the results, reports, and logs of the completed test run from the **Results** page. See Test results on page 983.
- Query the test results data. See Test log queries on page 1022.

Checking logs

To verify how the test ran or to debug test run failure, you can check the `Test Log` and `Execution log`.

About this task

The `Test Log` displays the interaction between HCL DevOps Test Hub (Test Hub) and the application or system under test. After the test run completes, the verdict of the run can be *Pass*, *Fail*, or *Inconclusive*. If the verdict of the run is Pass, the `Test Log` is available in the **Results** page.



The `Execution log` displays the console messages of the run time process that runs the test. This log is useful in determining the cause of the failure if the verdict of the run is *Fail* or *Inconclusive*. You can view the `Execution log` from the **Progress** page.

Viewing the `Test Log`

1. Go to the **Results** page and identify the test that you ran.
2. Click the test so that the **Reports** panel is displayed.
3. Click the **Test Log** in the **Reports** panel.

   The `Test Log` is displayed in a browser window.

Viewing the `Execution` log

4. Go to the **Progress** page and identify the test that is in the *Running* state.
5. Click the **Open action menu** icon.
6. Click **Execution log**.

   The `Execution log` is displayed in a browser window.

Related information

Test run configurations on page 610

Resetting the configuration settings for a test run

When you are configuring a test run either as a first-run or when repeating the run, and you do not want to proceed with the settings configured or saved for the test run, you can reset the configuration settings. Resetting the configuration reverts all the settings to their default values for the test run.

Before you begin

You must be a member of the project with the *Owner* or *Tester* role to run the tests.

You must ensure that you have the **Execute test asset** dialog is displayed before you proceed with resetting the configuration settings for the test run.

About this task

You can reset the configuration settings when you are configuring a test run either for its first run or when you are repeating the test run.

1. Click **Reset** in the **Execute test asset** dialog at any point when you want to reset the configuration settings for the test run.

   Notes:

   - If you have configured some or all of the settings for the current test run and you do not want to continue with those settings, clicking **Reset** resets the settings to their default values.
   - If you are repeating a test run and do not want to use the saved settings from a previous run, clicking **Reset** reverts all the saved settings to their default values.

   The settings in the **Execute test asset** dialog are reset to their default values.
2. Use the following table to find the default values for each of the settings in the **Execute test asset** dialog:

Window or Tab	**Field or Setting**	**Default value is**
Execute test asset	Scheduling the test run	**Now** is selected.
Execute test asset	Label for settings	Null or empty field.
Advanced	**JVM Arguments**	Null or empty field.
	Program Arguments	Null or empty field.
	Environment Variables	Null or empty field.
ENVIRONMENT tab	API test environment	The environment configured in the test asset.
	Secrets collection	Null or empty field.
DATA SOURCES tab	**Override** list	Null or empty field if the asset has no dataset.

Window or Tab	Field or Setting	Default value is
		The dataset in the test asset if the asset has one dataset.
		The first dataset on the list of datasets, which are listed in increasing alphabetical order, if the asset has multiple datasets.
VARIABLES tab		Null or empty field.
LOCATION tab	Agents	Null or empty field if the asset has no agents configured.
		The first agent on the list of agents, which are listed in increasing alphabetical order, if the asset has multiple agents configured.

What to do next

You can complete configuring the settings that you want for the test run. See Test run configurations on page 610.

Stopping test runs

You might want to stop a test run or multiple tests that are running when you realized that you did not configure all the settings or you want to change a few settings for the test runs.

Before you begin

You must have initiated a run for one or multiple test assets from the **Execution** page.

About this task

You can stop either a running test or multiple running tests from the **Progress** page. You can stop the test if it is in either the `In transition` or `Running` state. You cannot stop the test run if the test has already completed its run.

1. Go to the **Progress** page and identify the tests in any of the following ways that you want to stop:

   You can select the tests that satisfy the following conditions:
   - The test must be either in the `In transition` or `Running` state.
   - The **Stop execution** icon  in the **Actions** column must be enabled for the test.
   - The checkbox must be enabled for the test.
2. Perform the actions in the following table:

If...	Action...
You want to stop any test that is running	Perform any of the following actions:

If...	Action...
	◦ Click the **Stop execution** icon in the **Actions** column of the selected test. ◦ Click the checkbox in the row of the selected test, and then click **Stop selected**. The **Stop execution** dialog is displayed.
You want to stop multiple tests that are running	Perform any of the following actions: ◦ Click the checkbox in the row of the selected tests, and then click **Stop selected**. ◦ Click the **Select all** checkbox in the header row, and then click **Stop selected**. **Note:** All running tests are selected when you click the **Select all** checkbox. The **Stop execution** dialog is displayed.

3. Perform the following steps in the **Stop execution** dialog that is displayed:

   a. Set the timeout period for stopping the test run. Enter a numeric value and select the unit from the options available as *Seconds*, *Minutes*, or *Hours*.

   The time out period is the time during which the test run is allowed to stop on its own and after the timeout period, the test is forced to stop abruptly.

   Important: If the timeout period is not set, a value of *30 seconds* is considered as the default timeout period.

   b. Perform any of the following actions:

   - Keep the **Capture results** option selected, if you want the results to be captured for the test.

     The captured results for the test are available to view from the **Results** page.
   - Clear this option if you do not want to capture the results.

   Note: The **Capture results** option is selected as the default option.

   Important: The results are always captured for API suites even if you clear the **Capture results** option.

   c. Perform any of the following actions:

- Keep the **Execute 'finally-block', if present** option selected, if you want the *finally-block* code in the test script, if present, to be run before the test run is stopped.
- Clear this option if you do not want to run the *finally-block* code in the test script.

Note: The option **Execute 'finally-block', if present** is selected as the default option.

Restriction: The *tear-down* steps equivalent to the **'finally-block'** code, if present in API suites are not controllable at test runtime. The **Execute 'finally-block', if present** option has no impact on API suites.

d. Click **Stop execution**.

A notification is displayed that the running tests are stopped.

Note: If the test run completes before you can configure the options for stopping the test run, a notification is displayed in the **Stop execution** dialog that the test asset has completed its run.

Results

You have successfully stopped a running test or stopped multiple running tests. The stopped test runs are displayed on the **Progress** page with the status as *Stopped by User*. Stopped tests are not considered in the count of tests executed that are displayed on the **Overview** page.

What to do next

You can re-initiate the test run from the **Execution** page or from the **Results** page by completing the configurations that you want for the test.

You can view the results or the execution log of the stopped test from the **Actions** column on the **Progress** page by clicking the **Open action menu** icon .

Related information

Test run configurations on page 610

Viewing the progress of running test assets on page 926

Canceling scheduled test runs on page 934

Canceling scheduled test runs

You might want to cancel a scheduled test run, scheduled recurring test runs, or multiple scheduled test runs when you realized that you did not configure all the settings for the runs, want to change a few settings, or do not want those tests to run.

Before you begin

You must have scheduled runs for one or multiple test assets from the **Execution** page.

About this task

You can cancel a scheduled test run, all instances of a scheduled recurring test runs, or multiple scheduled tests from the **Progress** page.

1. Preform any of the following actions:
   - Click **Scheduled** under the **Next Run** column on the **Execution** page.

     The test is displayed on the **Progress** page. If you selected a test that was scheduled as a recurring test, then all the tests that ran and the next scheduled test run are displayed.

     Go to Step 2 on page 935.
   - Go to the **Progress** page and identify the tests in any of the following ways that you want to cancel:

     You can select the tests that satisfy the following conditions:
     - The test must be in the `Scheduled` state.
     - The **Cancel** icon in the **Actions** column must be enabled for the test.
     - The checkbox must be enabled for the test.

     Go to Step 2 on page 935.
2. Perform the actions in the following table:

If...	Action...
You want to cancel any test run that is scheduled.	Perform any of the following actions: ◦ Click the **Cancel** icon in the **Actions** column of the test that displays *Scheduled* as the status. ◦ Click the checkbox in the row of the test that is scheduled, and then click **Stop selected**. The **Cancel scheduled execution** dialog is displayed. Go to Step 3 on page 936.
You want to cancel all scheduled instances of the recurring test runs.	
You want to stop multiple test runs that are scheduled	Perform any of the following actions:

If...	Action...
	◦ Click the checkbox in the row of the selected tests, and then click **Stop selected**. ◦ Click the **Select all** checkbox in the header row, and then click **Stop selected**. **Note:** All scheduled tests are selected when you click the **Select all** checkbox. The **Cancel scheduled execution** dialog is displayed. Go to Step 3 on page 936.

3. Click **Yes** in the **Cancel scheduled execution** dialog.
   Result
   .

   A notification is displayed that the scheduled test runs are canceled.

Results

You have successfully canceled a scheduled test, all instances of the scheduled recurring tests, or multiple scheduled tests. The canceled test runs are displayed on the **Progress** page with the status as *Canceled*. The canceled test runs are not displayed on the **Results** page and are not considered in the count of tests executed that are displayed on the **Overview** page.

What to do next

You can re-initiate the test run from the **Execution** page by completing the configurations that you want for the test.

Related information

Test run configurations on page 610

Viewing the progress of running test assets on page 926

Stopping test runs on page 932

Management of virtualized services

You can find information about the tasks that you can perform on and manage the virtualized services that run on HCL DevOps Test Hub (Test Hub). You can start or stop the virtual services that are available to Test Hub. You can view the routing rules and usage statistics of virtualized services, agents, or intercepts that are connected to Test Hub.

You can create stubs or virtual services in HCL DevOps Test Integrations and APIs (Test Integrations and APIs). You must commit the virtual service resources to a remote Git repository, and then add the Git repository to your project on Test Hub if you used Test Integrations and APIs to create virtual service resources.

You can start a running instance of the virtual services from Test Hub to run on the following locations:

- A registered API agent.
- The default Kubernetes cluster of Test Hub.
- A remote Kubernetes cluster.

You can run all virtual services on registered API agents except the virtual services that use CICS transport, DB2/Z transport, or Istio.

The default Kubernetes cluster or a remote Kubernetes cluster supports running of the following types of virtual services:

- Virtual services that utilize the WebSphere® MQ transport.
- Virtual services that utilize the HTTP transport.

If running a virtual service in a Kubernetes cluster, the following are the considerations that decide the selection of a container:

- Before a container is selected to run a virtual service, it is checked for a previous failure to start a virtual service or for a low heap size, and other conditions. If the container was not found suitable, then it is not selected for running the virtual service.
- Multiple virtual services can be run in the same container if the following conditions are satisfied:
  - Subsequent virtual services are started by the same project member.
  - All virtual services are created and contained in the same Test Integrations and APIs project.
  - The requests to run the virtual services refer to the same version, meaning that the git commit IDs on which they are based are the same. If you use the UI, this means that the **Version** field in the **Execute virtual service** dialog are the same.
  - The settings that you configure in the **Advanced** settings for the virtual services are the same.
  - The datasets selected for an override are the same, if datasets are used.

Working with virtual services

You can perform the following tasks on virtual services on Test Hub:

- Read about the considerations before you configure a run of the virtual services. See Prerequisites for running HTTP virtual services on page 938.
- Read about the considerations before you configure a run of the virtual service on an API agent. See Considerations for running virtual services on remote API agents on page 551.
- Set up the HTTP proxy, if you have configured stubs in the test assets to use an HTTP proxy to route requests. See Setting up intercepts on page 940.

- View all intercepts that are registered with Test Hub including those that you have not configured. See Viewing intercepts that are registered with a team space on DevOps Test Hub on page 942.
- View virtual service resources in the test assets that are in the repositories added to a project on Test Hub. See Viewing virtual service resources on page 943.
- Configure runs of virtual services. See Configuring a run of a virtual service on page 946.
- Configure runs of HTTP virtual services to run without using proxies. See Running HTTP virtual services in Kubernetes cluster without using proxies on page 961.
- View running instances of virtual services. See Viewing running instances of virtual services on page 966.
- View configurations of a running virtual service instance. See Viewing configurations of running instances of virtual services on page 971.
- Modify the configurations of a running virtual service instance. See Modifying configurations of running instances of virtual services on page 976.
- View details of the usage statistics of the virtual services. See Viewing usage statistics of virtual services on page 980.
- View routing rules of the intercepts. See Viewing routing rules of the virtual services on page 978.
- Stop a running virtual service instance. See Stopping virtual services on page 982.

Prerequisites for running HTTP virtual services

You can find information about the prerequisite tasks that you must complete before you configure a run of the HTTP virtual services in HCL DevOps Test Hub (Test Hub).

Prerequisites for running HTTP virtual services in a Kubernetes cluster

You can find information about the tasks that you must complete before you configure a run of the HTTP virtual services in the HCL DevOps Test Hub (Test Hub) cluster or a remote Kubernetes cluster.

Important: The prerequisites do not apply when you use an API agent to run the HTTP virtual services.

Conditions for running of HTTP virtual services

Before you run the HTTP virtual services in the Test Hub cluster or a remote Kubernetes cluster, you must check for the following conditions and perform the actions indicated:

If...	Then...
You want to route the HTTP traffic via the HTTP proxy either to the virtual service or live systems, based on routing rules.	You must set up the HTTP proxy. See Setting up intercepts on page 940.
You want to route the HTTP traffic to the virtual service via the HTTP proxy and the proxy is not configured to trust all certificates.	The trust store must be configured to include the certificate authority for the server certificate. See Certificate authority: Importing and extending lists on page 161.

If...	Then...
A client application is sending requests directly to the virtual service.	The client application must trust the certificate authority for the server certificate. See Certificate authority: Importing and extending lists on page 161.

You are working with Test Hub V10.1.3, which is installed on Ubuntu.	HTTP virtual services are exposed via hostnames are of the following form:
	`in-<unique_id>.<INGRESS_DOMAIN>`
	
	Note: The DNS used by a computer that runs a client application or the HTTP proxy that routes traffic to the virtual service needs to resolve such hostnames to the IP address of the ingress domain.
	
	You must perform the following actions that depend on the method you have used to set up the ingress domain:
	• If you used `nip.io` to form the ingress domain, then that virtual services host names are automatically resolved to the IP address of the ingress domain.
	For example, if you used `nip.io` to form the ingress domain as follows:
	`10.1.2.3.nip.io`
	The virtual services hostnames are resolved by `nip.io` to the IP address as `10.1.2.3`.
	• If your ingress domain does not use `nip.io` then you must ensure that the DNS in use by the client application or the HTTP proxy can resolve hostnames of the virtual services to the IP address of the ingress domain. For example, if the ingress domain is as follows:
	`myhost.mycom.com`
	The DNS must resolve `*.myhost.mycom.com` to the IP address of `myhost.mycom.com`.

HTTPS virtual service endpoints

When you want to run an HTTP virtual service where the HTTP transport settings have **Use SSL** enabled, the virtual service endpoint that is exposed for the virtual service instance expects an HTTPS request on port *443*. The exposed endpoint is

managed by the same ingress controller or the HAProxy that is used for the server endpoints. The SSL/TLS termination occurs at the endpoint before the request is routed to the virtual service instance and results in the following implications:

- The **Virtualization (Server) SSL** options that are specified in the HTTP transport have no effect.
- The certificate served by the exposed virtual service endpoint is the same as that of server that is signed by the server certificate authority.

Related information

Setting up intercepts on page 940

Running HTTP virtual services in Kubernetes cluster without using proxies on page 961

Setting up intercepts

When you want to run virtual services on HCL DevOps Test Hub (Test Hub), you can set up intercepts to route requests to the virtual service. You must configure the intercepts with the Test Hub URL, team space ID, and offline user token that is generated from Test Hub so that the intercepts can register with Test Hub.

Examples of intercepts include the HTTP proxy component of HCL DevOps Test Virtualization Control Panel (Test Virtualization Control Panel) and the HCL DevOps Test Integrations and APIs (Test Integrations and APIs) Java virtualization agent available from the Test Virtualization Control Panel tools page.

Before you begin

You must have completed the following tasks:

- Installed the intercept from Test Virtualization Control Panel of the same version as that of Test Hub. For example, installed the HTTP proxy.
- Read the following procedures:
  - Generating offline user tokens. See Access management for DevOps Test Hub on page 1194.
  - Viewing the *Team Space ID* as the URL alias of the team space. See Viewing the configuration of a team space on page 1204.

About this task

You must edit the `registration.xml` file for the intercept to allow it to register with Test Hub.

1. Perform the following steps in Test Hub:
   a. Log in to Test Hub.

      The **Team Space Dashboard** page is displayed.

   b. Generate the offline user token by clicking the **User** icon  , and then click **Create Token**.

   c. Copy the offline user token to a text file.

d. Click the **Settings** icon of the team space in the **My Team Spaces** page.

The **Team Space Configuration** page is displayed.

e. Copy the text that is displayed under **URL alias** as the *Team Space ID*.

2. Perform the following steps to edit the `registration.xml` file for the intercept. For most intercepts this is in the root directory of the extracted intercept compressed file. For the HTTP/TCT proxy it is in the `<DTVCP_install_directory>\httptcp` directory on the computer on which you have installed Test Virtualization Control Panel.

a. Open the `registration.xml` file by using a text editor.

b. Enter the values for the following attributes:

Note: If an entry exists, you must either comment out that entry and add a new one or replace the contents with the values indicated.

Attribute	Value
server base-url	URL of Test Hub
security-token	Offline user token generated from Test Hub
team-space-id	ID of the team space to which the proxies must register

c. Save and close the file.

d. Perform any of the following actions:

- Restart the HTTP proxy, if you updated the `registration.xml` file of the HTTP/TCP proxy.
- Restart the application that is configured to use the any of the other intercepts other than HTTP/TCP.

3. Perform the following steps in Test Hub to view the HTTP proxies that are registered in the team space:

a. Log in to Test Hub.

b. Click **Infrastructure > Agents and Intercepts** to view the registered intercepts.

The intercepts that you configured with your offline user token and *Team Space ID* are displayed.

Results

You have successfully configured intercepts and viewed that they are registered with the team space in Test Hub.

What to do next

You can perform the following tasks:

- Read for the prerequisite tasks that you must perform before you can run the HTTP virtual services. See Prerequisites for running HTTP virtual services in a Kubernetes cluster on page 938.
- Run the virtual service resource. See Configuring a run of a virtual service on page 946.
- View the routing rules for the intercept. See Viewing routing rules of the virtual services on page 978.
- View the routing rules for the intercept in a running instance of the virtual service. See Viewing configurations of running instances of virtual services on page 971.

Viewing intercepts that are registered with a team space on DevOps Test Hub

After you install the HTTP proxies and configure them with your offline user token, the proxies register with HCL DevOps Test Hub (Test Hub) in a team space as intercepts and you can view the registered intercepts on the **Agents and Intercepts** page of the team space.

Before you begin

You must have completed the following tasks:

- Completed the prerequisite tasks before you configure a run of the HTTP virtual services. See Prerequisites for running HTTP virtual services in a Kubernetes cluster on page 938.
- Installed the HTTP proxies of the same version as the version of Test Hub.
- Configured the HTTP proxy with your offline user token so that the proxy can register as an intercept with Test Hub. See Setting up intercepts on page 940.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

   The **Agents and Intercepts** page is displayed.

   You can view the API agents, intercepts, Kubernetes clusters, performance agents, or DevOps Test Runtime agents that are registered with Test Hub.
3. You can view the intercepts that you own in any of the following ways:
   - Search for the intercept by entering the name of the intercept in the **Search** field.

     Note: You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

   - Sort the **Type** column to show the items sorted with the intercepts in the rows at the top of the table.
   - Sort the **Agents** column and identify the intercepts by their names or by their owner.

You can view the following details about the intercepts that are registered with Test Hub:

- `Any project` is displayed in the **Projects** column indicating that the intercept can be used by virtual services in any project.
- The status of the intercept is displayed in the **Status** column.

Results

You have viewed the registered intercepts from the **Agents and Intercepts** page.

What to do next

You can start the virtual services that use the registered intercepts. See Configuring a run of a virtual service on page 946.

Viewing virtual service resources

When project repositories contain virtual service resources and you want to view the resources so that you can start them, you can view the virtual services from the **Resources** page in your team space on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created virtual services that use any of the supported transports for tests in Test Integrations and APIs, committed the test resources to the remote repository, and added the repository to your project.

About this task

After you have committed the test assets and resources to the remote repositories and added the repositories to projects in your team space on Test Hub, you can go to the **Resources** page and perform the following tasks:

- View the virtual service resources that are contained in any branch of the repositories that are added to your project.
- View the components in the System Model that are associated with the virtual service resources.
- Create or use saved filters for viewing virtual service resources.
- Run or start an instance of the virtual service resource.
- View the number of instances of the virtual service resources that are running.
- View all instances of a running virtual service on the **Instances** page by clicking the **Show in instances page** icon  .

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the virtual service resources in the test assets by clicking **My Projects > *project_name***.

   The **Overview** page is displayed.

3. Click **Virtualization > Resources** in the navigation pane.

   The **Resources** page is displayed.

4. Complete the steps for the task that you want to perform as listed in the following table:

Task	Action
Viewing the virtual service resources that are contained in a branch of any of the repositories that are added to your project.	Select the branch of the repository that contains the virtual services that you want to run from the list in the **Branch** field. All virtual services in the selected branch are displayed on the **Resources** page.
Viewing the components that are associated with the virtual service resource.	The components that are associated with the virtual service resource are displayed in the **Components** column. Click the component in the **Components** column to view the virtual resource on the **System Model** page.
Creating or using saved filters for viewing specific instances of virtual service resources.	To create a query, perform the following actions: a. Click **New filter**. b. Enter a name for the filter. c. Select the **Visibility** of the filter. Choose one of the following visibility levels: ▪ **Private**: Makes the filter available only to the user who created it. ▪ **Project**: Makes the filter available to all members of the current project. Only project owners can edit project filters. ▪ **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit team space filters. **Note:** The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled. d. Create a rule with an appropriate operator. e. Select criteria such as *Name*, *Path*, *Instance number*, *Instance state*, or *Components*. f. Select the condition, and then enter the value for the criteria. g. Apply the filter query.

Task	Action
	The virtual services that match the filter criteria are displayed. h. Save the filter query for retrieving it from the saved filters list. Modify a saved filter query by performing the following steps: ◦ Click the **Filter** icon  in the **ADVANCED** tab. The filter queries that you created and saved are displayed. ◦ Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**. ◦ Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**. To use a saved filter query, perform the following actions:  **Note:** To open the filter query, you must have created and saved a filter query. a. Click the **Open filters** icon  b. Select the saved filter. c. Apply the filter. The virtual services that match the filter criteria are displayed.
Running an instance of the virtual service resource.	After you identify the virtual service that you want to run, perform the following actions: a. Click the **Execute** icon  in the row of the identified virtual service. The **Execute virtual service** dialog is displayed. b. Run the virtual service with the settings that are configured in the resource by clicking **Execute** or modify any of the settings, and then click **Execute**. An instance of the virtual service starts to run. The virtual service is displayed with the state as `Running` along with the number of instances that are running in the **Active instances** column.
Viewing the number of instances of the virtual service resources that are running.	If any of the virtual services in the project repositories are started by any project member with the *Tester* or *Owner* role, you can view the number of instances that are running or active from the **Resources** page.

Task	Action
	The virtual services are displayed with the state as `Running` along with the number of instances that are running, in the **Active instances** column.
Viewing all instances of a running virtual service.	Click the **Show in instances page** icon  in the row of the virtual service on the **Resources** page. The **Instances** page is displayed with all the running instances of a particular virtual service. You can also view the number of requests that are received by the virtual service. The number of requests is displayed as the number of hits when you hover the cursor over the text in the **Activity** column.

Results

You have viewed virtual services from the **Resources** page on Test Hub.

Related information

Configuring a run of a virtual service on page 946

Configuring a run of a virtual service

When you have virtual service resources that are in the repositories added to your project on HCL DevOps Test Hub (Test Hub), you must configure a run of the virtual service to start an instance of the virtual service in the Test Hub cluster, a remote Kubernetes cluster, or an API agent.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created virtual services that use any of the supported transports for tests in Test Integrations and APIs, committed the test resources to the remote repository, and added the repository to your project.
- Read about the virtual services that are supported on Test Hub. See Supported virtual services on page 112.
- Completed the prerequisite tasks before you configure a run of the HTTP virtual services. See Prerequisites for running HTTP virtual services in a Kubernetes cluster on page 938.

- Read and completed the tasks mentioned in Considerations for using a remote Kubernetes cluster on page 573, if you want to use a remote Kubernetes cluster as a location to run the test.
- Read and completed the tasks mentioned in Considerations for running virtual services on remote API agents on page 551, if you want to use a remote API agent as the location to run the virtual services.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the virtual service resources in the test assets by clicking **My Projects >** ***project_name***.

   The **Overview** page is displayed.
3. Click **Virtualization > Resources** in the navigation pane.

   The **Resources** page is displayed.
4. Select the branch of the repository that contains the virtual services that you want to run from the list in the **Branch** field.

   All virtual services in the selected branch are displayed on the **Resources** page.
5. Identify the virtual service by completing any of the following steps:
   - Search for the virtual service by entering the name or the path of the virtual service in the repository in the **Search** field box.
   - Create a filter query by using the **New filter** option and complete the following steps:
     a. Create a rule with an appropriate operator.
     b. Select criteria such as *Name*, *Path*, *Instance number*, or *Instance state*. Select the condition and enter the value for the criteria. Apply the filter query.

        The virtual services that match the filter criteria are displayed.
     c. Save the filter query for retrieving it from the saved filters list.
   - Retrieve a saved filter by using the **Open filters** icon by completing the following steps:

     Note: To open the filter query, you must have created and saved a filter query.

     a. Select the saved filter.
     b. Apply the filter.

        The virtual services that match the filter criteria in the filter that is applied are displayed.

   You have identified the virtual service resources that you want to run.
6. Click the **Open action menu** icon in the row of the identified virtual service, and then click **Execute** icon.

   The **Execute virtual service** dialog is displayed.
7. Select the version of the virtual service that is in the repository that you want to start by performing any of the following actions:

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

  Use the following details about the version of the test resources that are displayed to identify the version that you want:
  - Commit message.
  - Tags labeled for the version committed.
  - The user who committed the version to the repository.
  - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

  The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version committed followed by the versions committed previously.
- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

  The version that matches the search criteria is displayed and it is selected for the test run.

8. Select the environment that was used to bind the physical and logical resource in the API project, in the **ENVIRONMENT** tab.

   Important: The configuration that you set for the run in the **Execute virtual service** dialog is preserved when you run the same virtual service again. The configurations that you set are not available to other members when they want to run the virtual service. For example, if you selected an environment, the same environment is selected when you run the virtual service again.

9. Enter a label, if required.

   A label that you enter for the test run that helps you to identify the virtual service instance on the **Instances** page. The label that you entered is displayed for the virtual service under the **Labels** column on the **Instances** page. After you have created a label, any member of the project can use that label.

10. Follow the instructions if you want to modify the configurations for the behavior of the virtual service:

    a. Click the **BEHAVIOR** tab, if it is not already open.

    b. Configure or change the settings for the following options:

       Note: The settings displayed are the settings that were configured for the virtual service when it was authored in Test Integrations and APIs.

Option	Description
Performance	The following settings are available for handling of requests by the virtual service:

<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>Optimize performance</td><td>If enabled, attempts are made to reduce the amount of processing between the time the virtual service receives a request and the time it sends a response. Specific optimizations depend on the message contents, as in the following examples:
When the virtual service receives requests, all validations are disabled, and for all XML payloads, the store and filter actions are converted to use XPath expressions.
When the virtualization sends responses, any store actions set on a message are disabled, and any XML content is collapsed when the virtual service is compiled instead of being collapsed every time that a response is sent.
The optimization is disabled as the default action for this setting and you can enable the performance optimization if you fully understand the implications of optimizing the performance.</td></tr>
<tr><td>Threads</td><td>The maximum number of threads used in processing requests received by the virtual service. The default number of threads is 10.</td></tr>
</table>
</td></tr>
<tr><td>Operation</td><td>Specifies an operation referenced by the virtual service.</td></tr>
<tr><td>Response time</td><td>Specifies the response time behavior for responses sent by the virtual service for the selected operation.

You can modify the value by selecting a value from the following options:
<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>No delay</td><td>Select this option for a response with no delay.</td></tr>
<tr><td>Minimum delay</td><td>Select this option for a delay in the response by entering the required delay in milliseconds (ms).</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>Uniform distribution</td><td>Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).</td></tr>
<tr><td>Gaussian distribution</td><td>Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).</td></tr>
</table>
</td></tr>
<tr><td>Passthrough</td><td>
Specifies the pass through behavior for the selected operation when requests are not handled within the virtual service.

You can modify the value by selecting a value from the following options:
<table>
<tr><td>Discard</td><td>This option stops the system under test from receiving the intercepted message. This option can disrupt the calling system. For example, the system might time out while it waits for a reply.</td></tr>
<tr><td>Pass Through</td><td>This option passes the intercepted message to the system under test, with an optional delay.</td></tr>
<tr><td>Simulate Error</td><td>This option returns an error to the calling system. The message is not passed to the system under test.</td></tr>
</table>
</td></tr>
</table>

11. Follow the instructions, if the virtual service references datasets.
    - You can use the dataset referenced in the asset.
    - You can choose to override the dataset with another data source. If alternative data sources are available, select from the set of overrides available.
12. Follow the instructions if the virtual service requires a variable that must be passed at run time.
    a. Click the **VARIABLES** tab, if it is not already open.
    b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon, enter the name, and value of the variable.
- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon and select the **Upload from local system** or **Browse from server** to select the variable file.

Note: You must have created a file with the variables before you can select the file.

13. Click **Advanced** to make the following advanced configurations:

    a. Enter any JVM arguments that must be passed at run time in the **JVM Arguments** field.

    Note: Each JVM argument should be separated with white space.

    For example, you can set a maximum Java heap size.

    b. Enter the environment variables that must be passed at run time in the **Environment Variables** field, if applicable.

    For example, enter the environment variables when the third-party libraries that are used in the run refer to the environment variables for configuration.

    c. Select the stub logging level for the virtual service from the following options in the **Logging** list:

Option	Description
None	Specifies that the virtual service does not write log messages.
Normal	Specifies that the virtual service writes informational messages.
Debug	Specifies that the virtual service writes informational and debugging messages.

    d. Enter other configuration options as parameters and their values in the **Additional Configuration Parameters** fields, if applicable.

    You can refer to the additional parameters that you can use for virtual services from the topic in the related links.

    For example, if you want to start the virtual service to run in a new container, you can specify the following parameter and its value:

Parameter name	Value
stub.dedicated.container	**true**

Note: Click **Add** to add additional parameters.

Note: You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for the fields for the advanced settings is null or an empty field.

Notes:

- If you have configured some or all of the settings for the current run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

14. Select the step for the location to run the virtual services based on the following types of virtual services:
    - If you want to run any of the following types of virtual services only in the default Kubernetes cluster or a remote Kubernetes cluster, then go to Step 15 on page 952.
    - If you want to run the virtual services on a remote API agent that you added to your project, then go to Step 16 on page 953.
15. Follow the instructions if you want to change the location of a Kubernetes cluster for running the test:
    a. Click the **LOCATION** tab, if it is not already open.

    The **Default Cluster** is the default location where the virtual service resource runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

    Important: You must have added the registered remote Kubernetes clusters to your project that are then displayed under the **Override** column:

    Notes:

- If the remote clusters are not added to your project, the option **No override options** is displayed as the default value and the virtual service resource runs in the Kubernetes cluster of Test Hub.
- If remote clusters are added to your project, the added clusters are displayed along with their availability status and ownership information.

b. Select the location where you want to run the virtual service resource from the following options:

- Select the **Default Cluster** when no remote clusters are available in your project.
- Select the remote cluster from the list when a remote cluster is available in your project.
- Select **No override options**, if you selected any remote cluster and want to revert to the **Default Cluster** to run the virtual service resource.

c. Click **Execute**.
The run is initiated.

16. Follow the instructions if you want to run the virtual services on the remote API agent that is connected to the default Kubernetes cluster:

a. Click the **LOCATION** tab, if it is not already open.

The API agents from the list of registered API agents that match with the attributes or tags configured for the virtual service resources in the project are listed under the **Host** column. The information about the availability of the agent is displayed.

Note: You must have added the API agents to your project from the **Agents and Intercepts** page for the API agents to be displayed under the **Override** column.

The default value for the API agents is null or an empty field if no agents were configured in the project resources. If the project resources contains API agents that match with the agent tags, then the default API agent is the first item to be displayed on the list of API agents listed in the increasing alphabetical order.

b. Select the API agent where you want to run the virtual service in the following scenarios:

If...	**Then...**	**Action**
The API agent that is specified in the project resources is available and no other agents are added to the project.	The API agent that is specified in the project resources is displayed in the **Override** column.	Select the API agent in the **Host** that is also displayed in the **Override** column as the location to run the test.
The API agent that is specified in the project resources is not available and no other API agents are added to the project.	There is no API agent displayed for override in the **Override** column.	You cannot run the test on the API agent.

If...	Then...	Action
		You must wait until the API agent is available to run the test on the API agent.
The API agent that is specified in the project resources is available and other API agents added to the project.	API agents that have capabilities that match with the API agent capabilities specified in the project resources are listed in the **Override** column as follows: ▪ API agents with capabilities that best match the capabilities of the API agent in the project resources are at the top of the list. ▪ API agents with capabilities that do not match with the capabilities of the API agent in the project resources are listed subsequently. You can find details of both API agents for the matching capabilities when you hover over the agent in the **Override** column.	Perform any of the following actions: ▪ Select the API agent in the **Host** that is also displayed in the **Override** column as the location to run the test. ▪ Select an API agent in the **Override** column as the location to run the test.

c. Click **Execute**.

The test run is initiated.

Results

You have started a virtual service from the **Resources** page on Test Hub.

What to do next

After you start a run of a virtual service, you can view the status of the virtual service in any of the following ways:

- On the **Resources** page, the virtual services are displayed with the state as `Running` along with the number of instances that are running in the **Active instances** column.
- On the **Resources** page, you must click the **Show in instances page** icon in the row of the virtual service to view all the running instances of a particular virtual service on the **Instances** page.

- Go to the **Instances** page by clicking **Virtualization > Instances**. You can view instances of all the virtual services that are running and are displayed with the state as `Running` under the **State** column. You can also view the requests that are received by the virtual service. The number of requests is displayed as number of hits when you hover the cursor over the text in the **Activity** column.

You can view the details, configuration settings, routing rules, usage statistics, or logs of a specific running instance of the virtual service. See Viewing configurations of running instances of virtual services on page 971.

You can modify the behavior or the logging level of a running instance of the virtual service that you started before running tests on them. See Modifying configurations of running instances of virtual services on page 976.

When you have completed testing the virtual service, you must stop the running virtual service. See Stopping virtual services on page 982.

Related information

Additional configuration parameters for virtual services on page 955

Additional configuration parameters for virtual services

You can find information about the additional configuration parameters that you can set when you want to run virtual services on Test Hub.

When you use the server UI to start the virtual service, some of the parameters are handled by fields in the **Execute virtual service** dialog.

You can enter the other parameters as a *name-value* pair in the **Additional configuration parameters** field that is displayed in the **Advanced** settings panel of the **Execute virtual service** dialog.

The following additional configuration parameters are supported when you configure a run for a virtual service.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
To specify a unique identifier for use in the generated host name. **Note:** This parameter applies only if the virtual service is run in a Kubernetes	`stub.ingress.host.identifier`	Any <custom_id> that you specify. **Note:** The **<custom_id>** must start and end with an alphanumeric character. The characters supported	*mystub*	See Running HTTP virtual services in Kubernetes cluster without using proxies on page 961, to know how the identifier you specified is used.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
cluster and not on an API agent.		include a-z, 0-9, and a hyphen. No other characters can be used in the **<custom_id>**. The **<custom_id>** must not be longer than 32 characters.		
To specify whether a virtual service instance should be run in a dedicated container. **Note:** This parameter applies only if the virtual service is run in a Kubernetes cluster and not on an API agent.	`stub.dedicated.container`	• *true* • *false* **Note:** The default value is *false*.	*true*	A dedicated container is used to run a single virtual service instance.
To specify whether performance optimization is enabled	`stub.performance.optimize`	• *true* • *false*	*true*	The performance optimization is enabled for the virtual service.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
for the virtual service instance. The value set overrides the setting from the stub definition. This option can be set in the **Behavior** tab from the UI.		**Note:** The default value is *false*.		
To specify the number of threads to be used for processing requests for the virtual service instance. The value set overrides the setting from the stub definition. This option can be set in the **Behavior** tab from the UI.	`stub.worker.thread.count`	Any number that you specify as number of threads to use.	*15*	15 threads are used for processing requests for the virtual service instance.
To specify the stub logging level for the virtual service instance. The value set overrides the setting from the stub definition. This option can be set in the **Advanced** settings from the UI.	`stub.logging.level`	• *none* • *normal* • *debug*	*debug*	The virtual service writes informational and debugging messages to the container log.
To specify the pass-through behaviour type of the virtual service instance. The value set overrides the	`stub.pass.through.behavior`	Valid values depend on the stub operation transport type but can be as follows:	*simulate_error*	This option returns an error to the calling system. The message is not passed to the system under test.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
setting from the stub definition. This option can be set in the **Behavior** tab from the UI.		• *simulate_error* • *discard* • *pass_through*		
To specify the delay in passing through from the virtual service instance. The value set overrides the setting from the stub definition.	`stub.pass.through.delay.period`	Any number of seconds in milliseconds.	*10000*	A delay of 10 seconds in passing through from the virtual service instance.
To specify the status code to use in the virtual service instance response when simulating an error. The value set overrides the setting from the stub definition. **Note:** Applies to HTTP stubs. This option can be set in the **Behavior** tab from the UI.	`stub.pass.through.status.code`	You must specify a valid HTTP status code as the value. For example, `503`.	*500*	The HTTP error code of `500` is sent in the response if the virtual service simulates an error when passing through.
To specify the reason phrase to use in the virtual service instance response when simulating an error. The value set overrides the setting from the stub definition.	`stub.pass.through.reason.phrase`	You must specify the text of the status message as the value.	*my custom error msg*	The text that you specify is set as the HTTP reason phrase in the response if the virtual service simulates an error when passing through.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
Note: Applies to HTTP stubs. This option can be set in the **Behavior** tab from the UI.				
To specify the type of response-time behavior that is required for the virtual service instance. The value set overrides the setting from the stub definition. This option can be set in the **Behavior** tab from the UI.	`stub.response.time.type`	• *no_delay* • *minimum_delay* • *gaussian* • *uniform* • *performance_profile* **Note:** *performance_profile* is only valid if the stub operation in the stub definition is defined to use a performance profile.	*no_delay*	The virtual service sends a response without any delay.
To specify the minimum time for the response time behaviour required for the virtual service instance. The value set overrides the setting from the stub definition.	`stub.response.time.minimum`	Any number of seconds in milliseconds.	*5000*	The virtual service sends a response after a minimum delay of 5 seconds.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
This option can be set in the **Behavior** tab from the UI.				
To specify the maximum time for the response time behaviour required for the virtual service instance. The value set overrides the setting from the stub definition. This option can be set in the **Behavior** tab from the UI.	`stub.response.time.maximum`	Any number of seconds in milliseconds.	*20000*	The virtual service must send a response after a maximum delay of 20 seconds.
To override the value of for an input tag used by the virtual service.	`stub.input.tag.`*`name`*, where *name* is the name of the input tag that you want the virtual service to use.	The value to set for the input tag.	You can specify a tag name as `stub.input.tag.mytag` with a value of *red*.	The input tag named *mytag* will have the value *red* when the virtual service is run.

Virtual services where the stub definition refers to multiple operations

When a stub definition has two or more events that refer to different operations and each operation has a different behavior, you must specify the *<operation_id>* suffixed to the configuration parameter name in the **Additional configuration parameters** field.

Note: The *<operation_id>* is the internal ID of the operation that is found in the **Documentation** tab when you open the operation in Test Integrations and APIs.

When the stub definition refers to multiple operations and you want to start a run of a virtual service, you must append the *<operation_id>* to the *<parameter_name>* in the **Additional configuration parameters** field.

The configuration parameter names that you must use for the virtual service, which is associated with a particular operation are as follows:

- `stub.pass.through.behavior.<operation_id>`
- `stub.pass.through.delay.period.<operation_id>`

- `stub.pass.through.status.code.<operation_id>`
- `stub.pass.through.reason.phrase.<operation_id>`
- `stub.response.time.type.<operation_id>`
- `stub.response.time.<operation_id>`
- `stub.response.time.maximum.<operation_id>`

For example, if the *<operation_id>* is *7281b750:162483a09f1:-7e68*, the parameter name for the `stub.response.time.type` option that you must specify is `stub.response.time.type.7281b750:162483a09f1:-7e68`.

Related information

Configuring a run of a virtual service on page 946

Running HTTP virtual services in Kubernetes cluster without using proxies

When you run HTTP virtual services in the Kubernetes cluster of HCL DevOps Test Hub (Test Hub) or a remote Kubernetes cluster, the endpoint on which the virtual service is exposed externally uses a host name that includes a generated ID. If a client application is required to call the virtual service directly via its external hostname and not via the HTTP proxy, then you might want to determine the hostname to configure the client first before you start the virtual service.

Important: This task does not apply when you use an API agent to run the HTTP virtual services.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in a team space on Test Hub and added the repository that contains the HTTP virtual services.
- Completed the prerequisite tasks before you start HTTP virtual services. See Prerequisites for running HTTP virtual services in a Kubernetes cluster on page 938.
- Selected the branch of the Git repository to view test resources on the **Execution** page.

About this task

When you installed Test Hub on Ubuntu, the default HTTP virtual service endpoints are of the following form:

```
in-<unique_id>.<ingress_domain>
```

For example, with an ingress domain as **10.1.2.3.nip.io**, a virtual service endpoint might be as follows:

```
in-cd3a755635574c6fa7e264911301821a.10.1.2.3.nip.io
```

When you installed Test Hub 10.2.1 or earlier on Red Hat OpenShift, the default HTTP virtual service endpoints are of the following form:

```
<ingress_domain_first_label>-<unique_id>.<remainder_of_ingress_domain>
```

For example, with an ingress domain as **dth.mycluster-123456-0000.region.containers.appdomain.cloud**, a virtual service endpoint might be as follows:

```
dth-cd3a755635574c6fa7e264911301821a.mycluster-123456-0000.region.containers.appdomain.cloud
```

The **<unique_id>** is generated when the virtual service is started. The full hostname can be viewed in the **Routing to** field of the routing rule created for the virtual service on the **Intercept Rules** page. When traffic is routed to the virtual service via the HTTP proxy, a client application is unaware of this hostname. If the client application is required to call the virtual services directly, it must be aware of the hostname of the virtual service.

To allow a hostname to be known before you start the virtual service, you can replace the generated ID with a user defined value that is specified as an additional configuration parameter when starting the stub. This value is referred to as **<custom_id>**.

Notes:

- The value of the **<custom_id>** that you specify is used as the **<unique_id>** for the virtual service.
- The **<custom_id>** must start and end with an alphanumeric character. The characters supported include a-z, 0-9, and a hyphen. No other characters can be used in the **<custom_id>**. The **<custom_id>** must not be longer than 32 characters.

Restriction: The **<custom_id>** must be unique to a virtual service that is running. You must not assign the same **<custom_id>** to another virtual service that is running simultaneously. Doing so might result in an undefined behavior of the virtual service.

For example, on Ubuntu, if you specify the value of the **<custom_id>** for the virtual service as `stub123-test` and Test Hub has an ingress domain as **10.1.2.3.nip.io**. Then the hostname of the virtual service endpoint is as follows:

```
in-stub123-test.10.1.2.3.nip.io
```

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the virtual service resources in the test assets by clicking **My Projects > *project_name***.

   The **Overview** page is displayed.
3. Click **Virtualization > Resources** in the navigation pane.

   The **Resources** page is displayed.

4. Select the branch of the repository that contains the virtual services that you want to run from the list in the **Branch** field.

   All virtual services in the selected branch are displayed on the **Resources** page.

5. Identify the virtual service that you want to run.
6. Click the **Open action menu** icon in the row of the identified virtual service, and then click **Execute** icon .

   The **Execute virtual service** dialog is displayed.

7. Select the version of the virtual service that is in the repository that you want to start by performing any of the following actions:

   - Expand the list in the **Version** field, find the version of the test resources, and then select the version.

     Use the following details about the version of the test resources that are displayed to identify the version that you want:

     - Commit message.
     - Tags labeled for the version committed.
     - The user who committed the version to the repository.
     - Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

     The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version committed followed by the versions committed previously.

   - Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

     The version that matches the search criteria is displayed and it is selected for the test run.

8. Select the environment that was used to bind the physical and logical resource in the API project, in the **ENVIRONMENT** tab.

   Important: The configuration that you set for the run in the **Execute virtual service** dialog is preserved when you run the same virtual service again. The configurations that you set are not available to other members when they want to run the virtual service. For example, if you selected an environment, the same environment is selected when you run the virtual service again.

9. Enter a label, if required.

   A label that you enter for the test run that helps you to identify the virtual service instance on the **Instances** page. The label that you entered is displayed for the virtual service under the **Labels** column on the **Instances** page. After you have created a label, any member of the project can use that label.

10. Follow the instructions if you want to modify the configurations for the behavior of the virtual service:

a. Click the **BEHAVIOR** tab, if it is not already open.

b. Configure or change the settings for the following options:

Note: The settings displayed are the settings that were configured for the virtual service when it was authored in Test Integrations and APIs.

<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>Performance</td><td>The following settings are available for handling of requests by the virtual service:
<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>Optimize performance</td><td>If enabled, attempts are made to reduce the amount of processing between the time the virtual service receives a request and the time it sends a response. Specific optimizations depend on the message contents, as in the following examples:
When the virtual service receives requests, all validations are disabled, and for all XML payloads, the store and filter actions are converted to use XPath expressions.
When the virtualization sends responses, any store actions set on a message are disabled, and any XML content is collapsed when the virtual service is compiled instead of being collapsed every time that a response is sent.
The optimization is disabled as the default action for this setting and you can enable the performance optimization if you fully understand the implications of optimizing the performance.</td></tr>
<tr><td>Threads</td><td>The maximum number of threads used in processing requests received by the virtual service. The default number of threads is 10.</td></tr>
</table></td></tr>
<tr><td>Operation</td><td>Specifies an operation referenced by the virtual service.</td></tr>
<tr><td>Response time</td><td>Specifies the response time behavior for responses sent by the virtual service for the selected operation.</td></tr>
</table>

<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td></td><td>You can modify the value by selecting a value from the following options:
<table><tr><th>Option</th><th>Description</th></tr><tr><td>No delay</td><td>Select this option for a response with no delay.</td></tr><tr><td>Minimum delay</td><td>Select this option for a delay in the response by entering the required delay in milliseconds (ms).</td></tr><tr><td>Uniform distribution</td><td>Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).</td></tr><tr><td>Gaussian distribution</td><td>Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).</td></tr></table></td></tr>
<tr><td>Passthrough</td><td>Specifies the pass through behavior for the selected operation when requests are not handled within the virtual service.

You can modify the value by selecting a value from the following options:
<table><tr><td>Discard</td><td>This option stops the system under test from receiving the intercepted message. This option can disrupt the calling system. For example, the system might time out while it waits for a reply.</td></tr><tr><td>Pass Through</td><td>This option passes the intercepted message to the system under test, with an optional delay.</td></tr><tr><td>Simulate Error</td><td>This option returns an error to the calling system. The message is not passed to the system under test.</td></tr></table></td></tr>
</table>

11. Click **Advanced** to make the following advanced configurations:

    a. Enter the following configuration options as parameters and their values in the **Additional Configuration Parameters** fields.

Parameter name	Value
stub.ingress.host.identifier	**<custom_id>**

12. Click **Execute**.

    The run is initiated.

Results

You have started an HTTP virtual service that does not use a proxy. The virtual service exposes the hostname that contains a custom ID that you specified.

What to do next

After you start a run of a virtual service, you can view the status of the virtual service in any of the following ways:

- On the **Resources** page, the virtual services are displayed with the state as `Running` along with the number of instances that are running in the **Active instances** column.
- On the **Resources** page, you must click the **Show in instances page** icon  in the row of the virtual service to view all the running instances of a particular virtual service on the **Instances** page.
- Go to the **Instances** page by clicking **Virtualization > Instances**. You can view instances of all the virtual services that are running and are displayed with the state as `Running` under the **State** column. You can also view the requests that are received by the virtual service. The number of requests is displayed as number of hits when you hover the cursor over the text in the **Activity** column.

Viewing running instances of virtual services

After you start virtual services from the **Resources** page, you can view the instances of the virtual services that are running from the **Instances** page on HCL DevOps Test Hub (Test Hub). You can view the details, configuration settings, routing rules, usage statistics, or logs of a specific running instance of the virtual service. You can also stop the running instance.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.

About this task

After you started the virtual services from the **Resources** page on Test Hub, you can go to the **Instances** page and you can perform the following tasks:

- View all running instances of the virtual services that are contained in the project repositories.
- View the components in the system model that are associated with the virtual service instances.
- Search for a specific instance of the virtual service for which you want to view the details.
- Create or use saved filters for viewing specific running instances of the virtual service.
- View the details, configuration settings, routing rules, usage statistics, or logs of a specific running instance of the virtual service.
- Change the behavior or logging level of a running instance of the virtual service before you run tests on the virtual service.

- Stop a specific running instance of the virtual service.
- Stop multiple instances or all instances of the virtual service that are running.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the virtual service resources in the test assets by clicking **My Projects >** ***project_name***.

   The **Overview** page is displayed.
3. Click **Virtualization > Instances** in the navigation pane.

   The **Instances** page is displayed.

   Any one of the following views is displayed:
   - No instances of virtual services are displayed if there are no instances that are running or were stopped in the past hour.
   - All running instances of virtual services in your project are displayed.
   - No running instances of virtual services are displayed but the option **Show all** is enabled. You can click **Show all** to display the instances that were stopped in the past hour. Instances that were hidden from the display when **Hide inactive** was clicked are displayed.

     Note: All instances that were stopped are automatically removed from the display after *60 minutes* of being stopped. You cannot use the **Instances** page to view the history of stopped instances. You can hide the stopped instances from the display by clicking **Hide inactive**, so that you can view only the running instances.

   You can find the following information about the running instances that are displayed on the **Instances** page.
   - The name and path of the virtual service in the project repository are displayed in the **Name** column.
   - The tags or labels added to the instance when the run was configured is displayed in the **Labels** column.
   - The name of the environment associated with the operation in which the virtual service was created in Test Integrations and APIs is displayed in the **Environment** column.
   - The state of the running instance is displayed as `Running` in the **State** column.
   - The activity of the running instance of the virtual service based on the number of requests received in the previous hour is displayed in the **Activity** column. The activities are classified and displayed as indicated in the following table:

Activity load	**Activity displayed as**
Received *0* requests	`None`
Received *1 - 10* requests	`Low`
Received *11 - 100* requests	`Mid`
Received *101 - 1000* requests	`High`

Activity load	Activity displayed as
Received more than *1000* requests	`Very High`

- The **Stop** icon is displayed in the **Actions** column and stops the running instance of the virtual service, when clicked.

4. Complete the steps for the task that you want to perform as listed in the following table:

Task	Action
Searching for a specific instance of the virtual service for which you want to view the details or want to stop the instance.	Enter either the complete text or part of the text in the name or the path of the virtual service that is in the repository in the **Search** field. The instances of the virtual services that match the text searched are displayed.
Viewing the component associated with the running instance of the virtual service.	You can view the name of the associated component that is displayed on the **Instances** page in the **Filter** row only if you clicked the icon on the **System Model** page to view all running instances of the virtual services .
Creating or using saved filters for viewing specific instances of virtual service resources.	To create a query, perform the following actions: a. Click **New filter**. b. Create a rule with an appropriate operator. c. Select criteria such as *Name*, *Path*, *Labels*, *Environment*, *State*, *Activity*, or *Location*. d. Select the condition, and then enter the value for the criteria. e. Apply the filter query. The virtual services that match the filter criteria are displayed. f. Save the filter query for retrieving it from the saved filters list. To use a saved filter query, perform the following actions: **Note:** To open the filter query, you must have created and saved a filter query. a. Click the **Open filters** icon. b. Select the saved filter. c. Apply the filter. The virtual services that match the filter criteria are displayed.
Viewing the details, configuration settings, routing rules, usage	After you identify the virtual service instance for which you want to view the details, perform the following actions:

Task	Action
statistics, or logs of a specific running instance of the virtual service.	a. Click the **Expand** icon  in the row of the identified instance of the virtual service. The **Details** panel of the instance is displayed. You can find the following details about the instance displayed: ▪ Name ▪ Path ▪ Labels ▪ API Environment ▪ Asset ID ▪ Git Repo ▪ Version ▪ Started at ▪ Started by ▪ Execution location ▪ Associated components ▪ Execution location b. Click **Configuration** to expand the panel and view the configuration details of the virtual service. You can find the following configuration details about the instance displayed: ▪ Dataset overrides ▪ Variables ▪ JVM args ▪ Environment variables ▪ Additional parameters c. Click **Behavior** to expand the panel and view the behavior that was configured for the virtual service. d. Click **Routing Rules** to expand the panel and view the routing rules that were configured in the virtual service. e. Click **Statistics** to expand the panel and view the graphical representation of the activities performed by the virtual service. f. Click **Diagnostics** to expand the panel and view the logging configuration of the virtual service and the links to the logs.
Changing the behavior or logging level of a running instance of the virtual service.	After you identify the virtual service instance for which you want to view the details, perform the following actions: a. Click the **Expand** icon in the row of the identified instance of the virtual service. The **Details** panel of the instance is displayed.

Task	Action
	Perform any of the following actions: ◦ Click **Behavior** to expand the panel. Perform the following steps to change the behavior configurations: a. Click the **Edit** icon  . b. Select a different operation for the virtual service, if available. c. Select a different response time and enter the values for the delay that you want to set. d. Click **apply**. ◦ Click **Diagnostics** to expand the panel. Perform the following steps to change the logging level configuration: a. Click the **Edit** icon  . b. Select a different logging level. c. Click **apply**.
Stopping a specific running instance of the virtual service.	a. Click the **Stop** icon  in the row of the virtual service instance. b. Click **Ok** in the **Stop virtual service instance** dialog that is displayed. The running virtual service instance is stopped.
Stopping multiple running instances or all the running instances of virtual services.	a. Perform any of the following actions: ▪ Select the checkbox in the row of the virtual service instances that you want to stop, and then click **Stop selected**. ▪ Select the **Select all** checkbox in the header row, and then click **Stop selected**. **Note:** All running instances are selected when you click the **Select all** checkbox. b. Click **Ok** in the **Stop all selected virtual service instances** dialog that is displayed. The running virtual service instances are stopped.

Results

You have viewed running instances of the virtual services from the **Instances** page on Test Hub.

What to do next

You can perform the following tasks:

- View the configurations of a running instance of the virtual service. See Viewing configurations of running instances of virtual services on page 971.
- Modify the behavior or logging level of a running instance of the virtual service before you run tests on the virtual service. See Modifying configurations of running instances of virtual services on page 976.
- Stop a running instance or all running instances of the virtual services. See Stopping virtual services on page 982.

Viewing configurations of running instances of virtual services

After you start virtual services from the **Resources** page, you can view the details, configuration settings, routing rules, usage statistics, or logs of a specific running virtual service instance from the **Instances** page on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.

About this task

After you have started the virtual services from the **Resources** page on Test Hub, you can go to the **Instances** page and view the information about a specific running virtual service instance.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the virtual service resources in the test assets by clicking **My Projects > *project_name***.

   The **Overview** page is displayed.
3. Click **Virtualization > Instances** in the navigation pane.

   The **Instances** page is displayed with all running instances of the virtual services that are in your project.
4. Identify the running instance of the virtual service of which you want to view the details, configuration settings, and other information.
5. Click the **Expand** icon  of the instance of the virtual service.

   The **Details** panel of the instance is displayed with the following details about the virtual service instance:

Note: The information displayed pertains to the configuration settings that were configured in the virtual service at the time it was created or run.

Option	Information displayed
Name	Displays the name of the virtual service.
Path	Displays the path of the virtual service resource in the project repository.
Labels	Displays the labels or tags that were entered for the virtual service when a run of the virtual service was configured.
API Environment	Displays the name of the environment associated with the operation in which the virtual service was created in Test Integrations and APIs or the environment selected for the run.
Asset ID	The ID of the virtual service resource in the project repository that was started as an instance.
Git Repo	The git repository added to the project that contains the virtual service resources.
Version	Displays the version of the virtual service resource that is in the project repository that was started as an instance.
Started at	Displays the date and time when the virtual service instance started.
Started by	Displays the name of the project member who started the virtual service instance.
Associated components	Displays the components in the system model that are associated with the virtual service.
Execution location	Displays the location of the Kubernetes cluster on which the virtual service instance ran. **Note:** The name of the remote Kubernetes cluster is displayed if the virtual service instance ran on the remote cluster or *Default Cluster* is displayed if the virtual service ran on Test Hub.

The other panels that you can view are as follows:

- Configuration
- Behavior
- Routing Rules
- Statistics
- Diagnostics

6. Expand each panel by clicking the panel to view the information in that panel.

   You can view the following information in the different panels:

<table>
<tr><th>Panel</th><th>Details</th></tr>
<tr><td>Configuration</td><td>
The Configuration panel of the instance is displayed when expanded with the following details about the virtual service instance:

Note: The information displayed pertains to the configuration settings that were configured in the virtual service at the time it was created or run.

<table>
<tr><th>Option</th><th>Information displayed</th></tr>
<tr><td>Dataset overrides</td><td>Displays the dataset overrides that were configured when executing the virtual service instance.</td></tr>
<tr><td>Variables</td><td>Displays the variables that were configured when executing the virtual service instance.</td></tr>
<tr><td>JVM args</td><td>Displays the JVM arguments that were configured when executing the virtual service instance.</td></tr>
<tr><td>Environment variables</td><td>Displays the environment variables that were configured when executing the virtual service instance.</td></tr>
<tr><td>Additional parameters</td><td>Displays the additional parameters that were configured when executing the virtual service instance.</td></tr>
</table>
</td></tr>
<tr><td>Behavior</td><td>
The Behavior panel of the instance is displayed when expanded with the following details about the virtual service instance:

Note: The information displayed pertains to the configuration settings that were configured in the virtual service at the time it was created or run.

<table>
<tr><th>Option</th><th>Information displayed</th></tr>
<tr><td>Performance</td><td>Displays the optimization setting and the maximum number of threads in use by the virtual service.</td></tr>
<tr><td>Operation</td><td>Displays the operations referenced by the virtual service. You can select each operation to view the related response time and the pass through settings.</td></tr>
<tr><td>Response time</td><td>Displays the response time option that is in use for the selected operation within the virtual service.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Panel</th><th>Details</th></tr>
<tr><td></td><td>
<table>
<tr><th>Option</th><th>Information displayed</th></tr>
<tr><td>Passthrough</td><td>Displays the pass through option that is in use for the selected operation within the virtual service.</td></tr>
</table>
</td></tr>
<tr><td>Routing Rules</td><td>
The Routing Rules panel of the instance is displayed when expanded.

The routing rules that relate to the virtual service are displayed.

<table>
<tr><th>Column</th><th>Description</th></tr>
<tr><td>Activity</td><td>Indicates the routing type.</td></tr>
<tr><td>Target</td><td>Displays the endpoint of the target service for which requests are being routed.</td></tr>
<tr><td>Recipient</td><td>Displays the endpoint of the virtual service that receives the routed requests</td></tr>
</table>

You can expand the panel for each intercept by clicking on the expand icon  . You can then find the following information:

<table>
<tr><th>Tab</th><th>Description</th></tr>
<tr><td>Details</td><td>
The Details panel provides the following information:

<table>
<tr><th>Option</th><th>Information displayed</th></tr>
<tr><td>Condition</td><td>Lists the condition that causes the traffic to be routed to a virtual service.

You can view the detailed condition of the rule defined for the intercept by clicking the View icon  .

The detailed condition of the rule is displayed in the Full condition window.

  Note: You can copy the detailed condition if you intend to use it elsewhere.</td></tr>
<tr><td>Routing to</td><td>Lists the virtual service to which traffic is routed.</td></tr>
<tr><td>Created by</td><td>Lists the user who created the rule.</td></tr>
</table>
</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Panel</th><th>Details</th></tr>
<tr><td></td><td>
<table>
<tr><th>Tab</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Option</th><th>Information displayed</th></tr>
<tr><td>Created at</td><td>Lists the date and time when the rule was created.</td></tr>
</table>
</td></tr>
<tr><td>Activity</td><td>The Activity panel lists the log of the activities performed in relation to the routing rule. This panel also displays the details of the activity, date and time, and message for that activity.</td></tr>
</table>
</td></tr>
<tr><td>Statistics</td><td>The Statistics panel of the instance is displayed when expanded.

The requests received by the instance of the virtual service in the previous hour are displayed as a graph. The total requests received is indicated in the Total hits field and the activity load in the previous hour are displayed in the Activity field.

The view can be refreshed by clicking the Refresh icon  .</td></tr>
<tr><td>Diagnostics</td><td>The Diagnostics panel of the instance is displayed when expanded with the following details:
<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>Logging</td><td>Displays the logging level that was configured for the virtual service.</td></tr>
<tr><td>Execution ID</td><td>Displays the ID of execution for the virtual service instance.</td></tr>
<tr><td>Stubs runtime ID</td><td>Displays the ID of the runtime where the virtual service instance runs. If executed on a cluster the value is that of the label execution-stubs-runtime-id on the pod that runs the virtual service instance. If executed on an API agent, the value is the ID of the engine on which the virtual service instance runs.</td></tr>
<tr><td>Init container log</td><td>Displays the link to the log of the Init container.</td></tr>
</table>
</td></tr>
</table>

Panel	Details	
	Option	**Description**
		Note: This logs contains messages that relate to configuring the container which is ready to run the virtual service.
	Main container log	Displays the link to the log of the main container. This log contains messages logged by the virtual service based on the selected logging level.
	Pod events log	Displays the link to the log of the pod events log. This log contains messages logged in the pod for the virtual service events.

Results

You have viewed the information about the configuration, routing rules, usage, or logs of the virtual service instance.

What to do next

You can modify the behavior or logging level configurations of the running virtual service instance before you run tests on the instance. See Modifying configurations of running instances of virtual services on page 976.

Related information

Configuring a run of a virtual service on page 946

Modifying configurations of running instances of virtual services

After you start virtual services from the **Resources** page and before you run tests on the virtual service, you might want to modify the configurations for the behavior or the logging level of the running instance.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.

About this task

After you have started the virtual services from the **Resources** page on HCL DevOps Test Hub (Test Hub), you can go to the **Instances** page and identify the running instance of the virtual service for which you want to modify the configurations. You can modify the configurations for the behavior or logging levels of the running instance of the virtual service.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the virtual service resources in the test assets by clicking **My Projects >** ***project_name***.

   The **Overview** page is displayed.
3. Click **Virtualization > Instances** in the navigation pane.

   The **Instances** page is displayed with all running instances of the virtual services that are in your project.
4. Identify the running instance of the virtual service for which you want to modify the configurations.
5. Click the **Expand** icon  of the instance that you want to modify the configurations.

   The **Details** panel of the instance is displayed.
6. Click **Behavior** to expand the panel and perform the following steps to change the configurations:

   a. Click the **Edit** icon  .

      The options that you can modify are enabled for editing.

   b. Select a different operation for the virtual service, if available, from the list for the **Operation** option.

   c. Select a different response time from the list for the **Response time** option, and then enter the values for the delay that you want to set.

      You can modify the value by selecting a value from the following options:

Option	Description
No delay	Select this option for a response with no delay.
Minimum delay	Select this option for a delay in the response by entering the required delay in milliseconds (ms).
Uniform distribution	Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).
Gaussian distribution	Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).

   d. Click **apply** to apply the modified settings.

7. Click **Diagnostics** to expand the panel and perform the following steps to change the logging level that is configured:
    a. Click the **Edit** icon  .

    The **Logging level** option is enabled for editing.

    b. Select a different logging level from the following options.

Option	Description
None	Specifies that the virtual service does not write log messages.
Normal	Specifies that the virtual service writes informational messages.
Debug	Specifies that the virtual service writes informational and debugging messages.

    c. Click **apply** to apply the modified settings.

Results

You have viewed and modified the configurations of a running virtual service instance from the **Instances** page on Test Hub.

What to do next

You can run tests that are in an API Suite in your project, which then run on the virtual service instance. See Configuring an API Suite or an API test run on page 652.

Viewing routing rules of the virtual services

When you run virtual services that are created to run on proxies or intercepts and have routing rules defined, you can view the proxies or intercepts and the routing rules from the **Intercept Rules** page.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.

About this task

After you run the virtual service resources that are in your project on HCL DevOps Test Hub (Test Hub), you can view the routing rules for the running instances of the virtual service resources, from the **Intercept Rules** page.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the virtual service resources in the test assets by clicking **My Projects > *project_name***.

The **Overview** page is displayed.

3. Click **Virtualization > Intercept Rules** in the navigation pane.

   The **Intercept Rules** page is displayed.

   The agents, proxies, or intercepts that are configured in the virtual service resources that are in your project are displayed.

   You can find the following information that is displayed on the **Intercept Rules** page.

Column	Description
Activity	Lists the agent or proxy.
Target	Lists a summary view of the endpoint that was recorded or virtualized.
Recipient	Lists the destination where messages sent to are captured.

4. Expand the proxy or intercept panel by clicking the **Expand** icon .

   The details of the proxy or intercept are displayed in the **Details** tab in the expanded panel.

5. Click each tab to view the following information:

<table>
<tr><th>Tab</th><th>Description</th></tr>
<tr><td>Details</td><td>The Details panel provides the following information:
<table>
<tr><th>Option</th><th>Description</th></tr>
<tr><td>Condition</td><td>Lists the condition that causes the traffic to be recorded or routed to a virtual service.

You can view the detailed condition of the rule defined for the proxy or intercept by clicking the View icon .

The detailed condition of the rule is displayed in the Full condition window.

Note: You can copy the detailed condition if you intend to use it elsewhere.</td></tr>
<tr><td>Routing to</td><td>Lists the virtual service to which traffic is routed.</td></tr>
<tr><td>Created by</td><td>Lists the user who created the rule.</td></tr>
<tr><td>Created at</td><td>Lists the date and time when the rule was created.</td></tr>
</table></td></tr>
<tr><td>Activity</td><td>The Activity panel lists the log of the activities performed by the proxy or intercept. This panel also displays the details of the activity, date and time, and message for that activity.</td></tr>
</table>

Results

You have viewed the routing rules for the virtual service, proxy, or intercept that are running or connected to Test Hub.

What to do next

You can perform any of the following tasks:

- You can view the usage statistics of the virtual services that are running. See Viewing usage statistics of virtual services on page 980.
- You can stop the running instances of the virtual services. See Stopping virtual services on page 982.

Related information

Management of virtualized services on page 936

Viewing usage statistics of virtual services

After you start virtual service resources that are in the project repositories, and then run tests on the virtual service, you can view the usage statistics of the virtual services from the **Usage** page on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Completed the prerequisite tasks before you configure a run of the HTTP virtual services. See Prerequisites for running HTTP virtual services in a Kubernetes cluster on page 938.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Verified the version of the supported assets and resources that you want to use. See Supported versions of assets and resources on page 109.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.
- Ran API Suites with tests that run on the virtual service resources that are configured.

About this task

After you run the virtual services that are in a project, you can view details of the usage statistics of the virtual services from the **Usage** page on Test Hub.

Restriction: The usage statistics of virtual services if they are run on Test Integrations and APIs agents, are collected only if the agents are of the same version as Test Hub.

Important: The settings such as the **Period**, or **From** and **To** that you configure to view the usage details are retained for you during your ongoing session and the settings are retained even when you perform the following activities:

- Log out of the server and log in again.
- Log in after a server restart.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the virtual service resources in the test assets by clicking **My Projects >** ***project_name***.

   The **Overview** page is displayed.
3. Click **Virtualization > Usage** in the navigation pane.

   The **Virtual Service Usage** page is displayed.

   The total requests that were received by all virtual services on the server are displayed as a graph.

   The default view displays the usage data of the virtual services that are gathered during the previous week and grouped in an hourly interval. The total requests received from when the virtual service was started are displayed.

   The metric data that is reported is for the number of requests that are received by a virtual service. The requests can be requests, messages, or other calls that are received by the virtual service.
4. Change the default settings to view the graph for the period for which you want to view the usage.

   You can change the default view by selecting an option from the following options:

Option	Description
During the last Hour	Displays the usage data of the virtual service that are gathered during the previous hour and grouped in a 20-second interval.
During the last 24 hours	Displays the usage data of the virtual service that are gathered during the previous day and grouped in an eight-minute interval.
During the last 7 days	Displays the usage data of the virtual service that are gathered during the previous week and grouped in an hourly interval. **Note:** This is the default view.
Custom Range	Displays the usage data of the virtual service that are gathered during the date range specified in the **From** and **To** fields. You can select the date and time for both the **From** and **To** fields. The data is dynamically grouped in intervals that depend on the range selected.

   Note: You can refresh the view at any time to present the graph for the latest data by clicking **Refresh**.
5. View details of the virtual service by hovering the mouse over any of the data-points to view the details at that point.

The following details about the virtual service are displayed when you hover the mouse over the data-point:

- The date and time of the request received.
- The number of requests received.

Results

You have viewed the usage details of the virtual services in your project that ran on Test Hub.

What to do next

You can perform any of the following tasks:

- You can view the routing rules defined in the virtual service, agent, proxy, or intercept. See Viewing routing rules of the virtual services on page 978.
- You can stop the virtual service instances that are running. See Stopping virtual services on page 982.

Related information

Management of virtualized services on page 936

Stopping virtual services

After you started instances of the virtual services from the **Resources** page on HCL DevOps Test Hub (Test Hub), you must stop the running instance from the **Instances** page when you do not want to use the instance. You can stop a single instance, multiple, or all running instances.

Before you begin

You must have started the virtual service from the **Resources** page. See Configuring a run of a virtual service on page 946.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Open the project that contains the virtual service resources in the test assets by clicking **My Projects > *project_name***.

   The **Overview** page is displayed.
3. Go to the **Instances** page and identify the instance of the virtual service that you want to stop.

   You can select the virtual service instances that satisfy the following conditions:
   - The virtual service instance must be either in the `In transition` or `Running` state.
   - The **Stop** icon  in the **Actions** column must be enabled for the virtual service instance.
   - The checkbox icon  must be enabled for selection for the virtual service instance.
4. Perform the actions indicated in the following table:

If...	Action...
You want to stop any virtual service instance that is running	Perform any of the following actions: ◦ Click the **Stop** icon  in the **Actions** column of the virtual service instance. The **Stop virtual service instance** dialog is displayed. ◦ Select the checkbox in the row of the virtual service instance, and then click **Stop selected**. The **Stop all selected virtual service instances** dialog is displayed.
You want to stop multiple virtual service instances that are running	Perform any of the following actions: ◦ Select the checkbox in the row of the virtual service instance, and then click **Stop selected**. ◦ Select the **Select all** checkbox in the header row, and then click **Stop selected**. **Note:** All running instances are selected when you select the **Select all** checkbox. The **Stop all selected virtual service instances** dialog is displayed.

5. Click **Ok**.

   A notification is displayed that the running virtual service instances are stopped.

Results

You have stopped a single instance, multiple, or all running virtual service instances. The stopped virtual service instance on the **Instances** page is displayed with the status as *Stopped by User* in the **State** column.

What to do next

You can restart the virtual service by completing the configurations that you want from the **Resources** page.

You can open and view the `Main container log` of the stopped virtual service.

Test results

After the tests, Suites, or schedules are run and completed, you can view the results and open the reports in HCL DevOps Test Hub (Test Hub) to analyze the verdict, the performance, and statistics. You can also re-execute tests and schedules from the **Results** page with the same commit id.

Summary

The **Results** page displays the verdict summary for all the tests run in the project.

In this page, you can perform any of the following tasks:

- Displaying the suites only, or the tests only, or tests and suites that are run with advanced settings.
- Searching for test results. You can filter test results by using any of the following ways:
  - By using the **Search** field to search for results by name. When you search by name, the number of rows that are matching with the search keyword are displayed next to the **Search** field.
  - By selecting predefined time interval from the **Selection interval** list.
  - By selecting **To** and **From date** from the **Date Interval** option.
  - By clicking verdict from the **Verdict summary** slider (*Pass*, *Fail*, *Inconclusive*, or *Error*).
  - By creating filter queries.
- Configuring the information displayed on the **Results** page from the **View Settings** window by clicking the **Settings** icon .
- Downloading the result list as a `.csv` file by clicking the **Download** icon .
- Adding labels if you have a tester or an owner role.
- Adding text in the **Notes** field. You can also group the notes similar to a conversation in a chat channel.
- Locking test results: Locked results can be unlocked by the project owner or the project member who locked the results.
- Deleting test or suite results.

  Note: When you delete the result of a suite run, the results associated with the tests and suites within that suite are also deleted. Test results that are part of a suite run cannot be selected individually. To delete such results, you must select and delete the parent test suite run.

- Comparing performance reports.
- Viewing trending reports.
- Viewing multiple reports depending on the test types and the parameters set when you execute the tests: Statistics reports, Functional Test reports, Test logs or Result data queries
- Re-executing tests with the same commit id.

  For more details, see the links in the next section of this page.

If you added any additional properties under the **Additional Configuration Parameters** section in the **Execute test asset** window, then these properties are displayed in the following sections after the test run:

- The **Properties** column on the **Results** page.
- The **Summary** section on the **Functional Report** page.
- The **Properties** section on the **Test log** page.

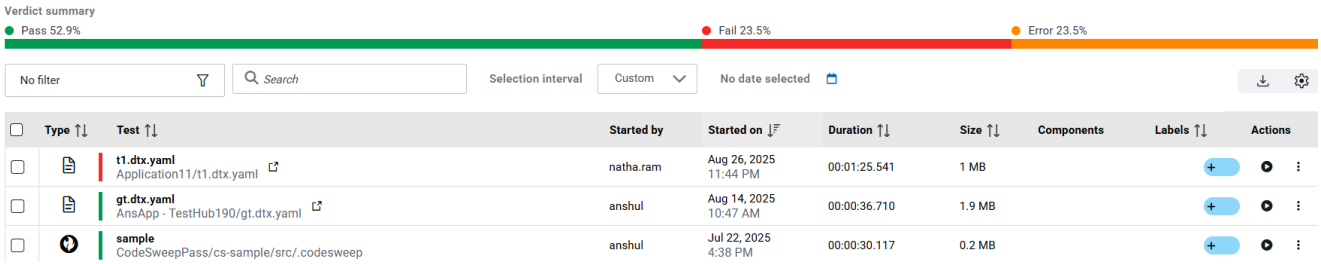
Test results and reports

HCL DevOps Test Hub (Test Hub) provides a single location for hosting the results and reports of all tests run on different desktop clients and for tests run from the server.

Test results list

When a test run is complete, you can see the results on the **Results** page that you open from **Analyze > Results** in the navigation pane.

The list of test results is organized based on the test type. You can display the suites only in the **Suites** tab, or the tests only in the **Tests** tab, or the tests and suites that are run with advanced settings in the **Advanced** tab.



Results, reports, and logs are generated for tests run from Test Hub, or from desktop clients such as Test Performance, Test UI, or Test Integrations and APIs.

You must have configured the desktop clients to publish reports of tests that are run from the desktop client to the Test Hub. For more information about the publishing procedure, refer to the links at the end of this page.

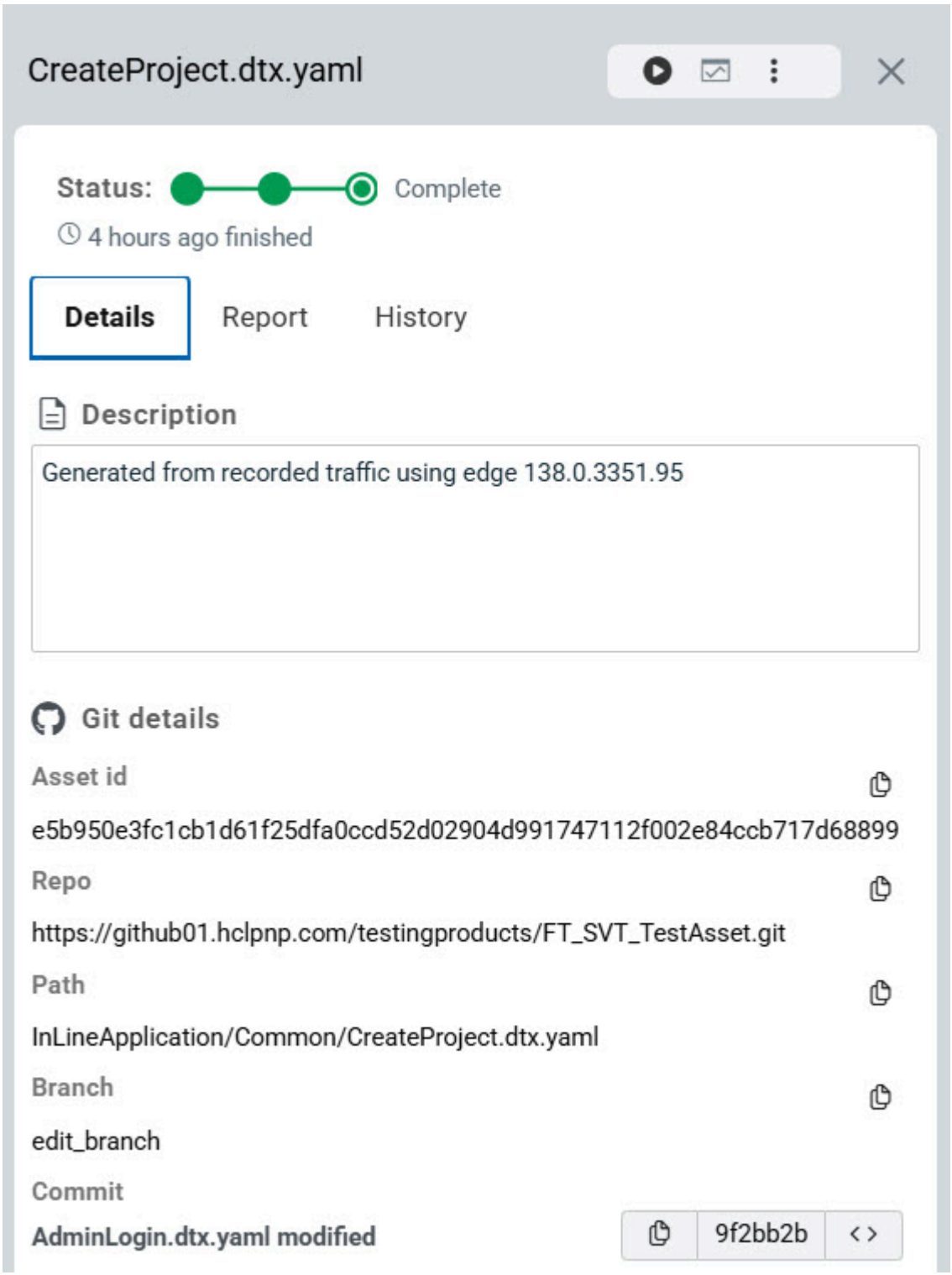
When you click the test result, a results panel is displayed with information in the following tabs:

- Details
- Reports
- History

Test result details

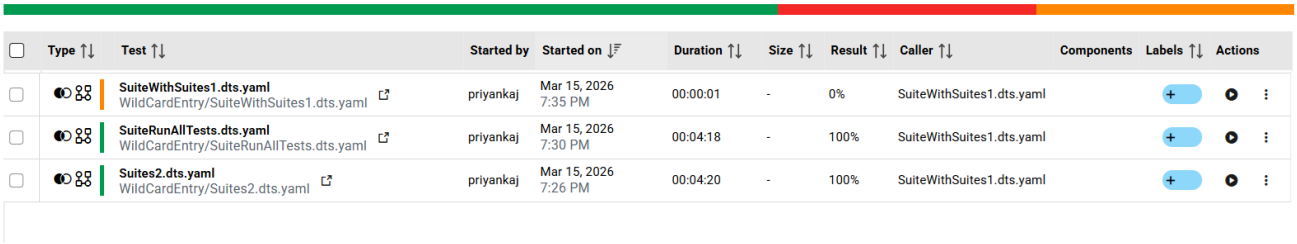
You can view the following details about the test assets used in the test run in the **Details** tab:

- The status of the test results, the date and time it was executed and completed.
- The details about the Git repository that contains the test assets.



You can configure the information displayed on the **Results** page from the **View Settings** window. You access this window by clicking the **Settings** icon .

Depending on the attribute you selected in the **View Settings** window, the selected attributes are displayed in the columns of the **Results** page.



The **Result** page supports selecting individual items from the list. When a row is selected, the corresponding high-level information is displayed in the top banner of the Test Editor.

The **Result** column displays the pass percentage for a test or suite. For a test, when you hover over the percentage, you can see a description indicating the number of steps that passed. For a test suite, the description indicates the number of tests that passed.

The **Caller** column displays the name of the suite that invoked the result.

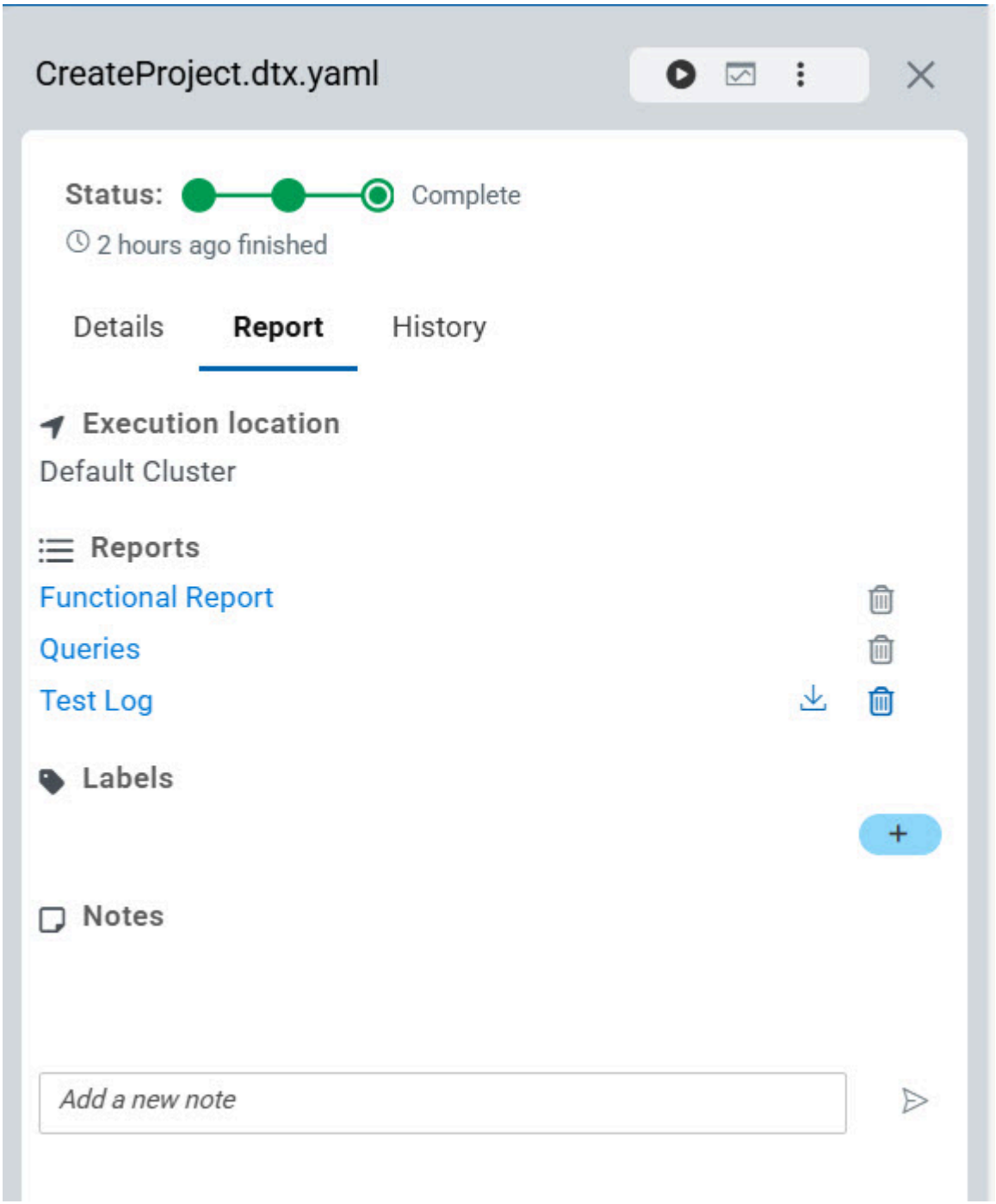
The icon in the **Type** column indicates that the test or suite was run as part of another suite. Hover over the icon to view the name of the suite in which the test or suite was run.

You can click the **Download** icon to download the list displayed on the **Results** page. The result list is downloaded into your local file system as a Comma Separated Values (CSV) file.

Reports

The **Report** tab provides the following information:

- The location where the test was run.
- The reports that are generated for the test run.
- The labels that you created for the test during its configuration for a run.
- Notes that you can add to the specific test run result or displays the notes already added by you or others.
- The size column displays the total disk usage of each test result. The size is rounded up and displayed in human-readable units. Hover over the size to view the exact disk usage. You can also sort by size.



You can perform the following actions on the reports and logs that are displayed on the **Reports** tab:

- View the report that opens in a web browser by clicking the report.
- Download the report by clicking the **Download** icon . You can enter a name and select the location on your computer where you want to save the report.

  The report is downloaded in any of the following formats:

- A compressed file and contains the HTML pages for the supported tests.
- A compressed file that contains the Surefire report.
- A PDF for the supported tests.

Note: For tests that contain more than 30 test steps, the details of the test steps are not included in the PDF.

- Delete the report by clicking the **Delete** icon .

Note: The deleted report is not available on the server. You might have to run the test again to generate the report.

As a default configuration, test logs are delivered in a traditional format for the executed Compound Tests and schedules.

The reports are generated for Compound Tests, Schedules, and Test Suites, or for other tests that are run from a desktop client or from Test Hub.

The following table lists the reports that are generated for each type of test and the links to help pages.

Table 30. Analyzing Reports

Report	Tests type	Product	More information
Functional Report	Server Test	Test Hub	Viewing a Functional Report on page 994
	Server Suite		
	API Test	Test Integrations and APIs	
	API Suite		
	Compound Test		
	Functional test	Test UI	
	AFT Suite		
	Compound Test		
Functional Test Report	API Suite	Test Integrations and APIs	Viewing reports
	Compound Test	Test UI	Unified Report
	AFT Suite		
	Functional test (For Web UI tests)		
Mobile and Web UI Report	Compound Test	Test UI and Test Performance	Mobile and web UI test result reports

Table 30. Analyzing Reports (continued)

Report	Tests type	Product	More information
	AFT Suite	Test UI	
Statistics Report	Compound Test	Test UI and Test Performance	Reports and counters. Service Performance report. From the Statistic reports, you can add counters. Refer to Adding additional counters on a separate page.. You can also display the statistics of counters on graphs. Refer to Displaying counter data in tables or as graphs..
	AFT Suite	Test UI	
	VU Schedule	Test Performance	
	Rate Schedule		
	Functional test (For Web UI tests)		
Test Log	Compound Test	Test UI and Test Performance	
	AFT Suite	Test UI	
	API Suite	Test Integrations and APIs	
	Functional Test (Web UI)	Test UI	
	VU Schedule	Test Performance	
	Rate Schedule		

Publishing test results to DevOps Test Hub

Publishing specific results to the server

Publishing test results to the server

Publishing functional reports to the server

Publishing specific results to the server

Publishing test results to DevOps Test Hub

Publishing existing test suite results to DevOps Test Hub

Creating search queries to filter test results

To filter the test and suite results list, you can create a search query that you apply as a filter in HCL DevOps Test Hub (Test Hub).

About this task

To filter the list of test and suite results, you can use rules and group of rules in your query. This procedure explains how to create a query that contains two rules and a group of rules as an example, and how to select other filters.

1. Click **New filter** on the **Result** page.
2. Select the **Visibility** of the filter.

   Choose one of the following visibility levels:

   - **Private**: Makes the filter available only to the owner who created it.
   - **Project**: Makes the filter available to all members of the current project. Only project owners can edit or delete project filters.
   - **Team space**: Makes the filter available to all members of the projects in the current team space. Only team space owners can edit the filters.

   Note: Saved filters are stored on the server, so they remain available after you clear browser data and can be shared with other users.

   Note: The **Project** and **Team space** visibility options are available only to project owners and team space owners. For other users, these options are disabled.

3. Select **Add a rule** in your query, and enter or select the required attribute in the fields to define the rule:

   a. Select the first attribute that you want to query in the drop-down list.
   Example
   Select **Verdict**.

   b. Select the condition as *equal to (=)* or *not equal to (!=)*, *Contain* or *Not contain*.
   Example
   Select **=**.

   c. Select a value for the first attribute you selected.
   The content of this field depends on the option you selected as first attribute. Enter a value for the **id**, **TypeTest**, **Version**, **Result**, **Labels**, **Branch**, **Test path**, and **Text** attributes. For the other attributes, select a value in the drop-down list.
   Example
   Select **Fail**.

d. Select either the **AND** or the **OR** operator in the first field to add a condition to compare the first rule with another rule.

e. Select **Add rule** in the drop-down list.

f. Select an attribute, a condition, and select or enter a value.
Example
Select **Test** as an attribute, select **=** as a condition operator, and enter *mytest* as a value.

Follow these steps to add a group in your query:

4. Select **Add group** in the drop-down list.

Note: A group is a collection of rules and can also contain other groups within a query. You can add a rule or a group within this group.

5. Select a condition operator.
Example
Select **Or**.
6. Select **Add rule** in the drop-down list to add a rule in the group.
7. Select the required an attribute, an operator, and enter a value for the rule.
Example
Select the **Type** attribute, **=** as an operator, and **Compound Test** as a value.
8. Select **Add rule** to add another rule in the group.
9. Select and enter the appropriate parameters as you did for the other rules.
Example
Select **Type** as an attribute, **=** as a condition, and **VU Schedule** as a value.
10. Click **Save**.
11. Perform the following steps to modify the filter:
    a. Click the **Filter** icon in the **ADVANCED** tab.

    The filter queries that you created and saved are displayed.

    b. Click **Edit** to modify the filter query if you have permission to edit the filter. Click **Save**.
    c. Click **Clone** if you do not have permission to edit the filter. A private copy of the filter is created. Modify the filter query, and then click **Save**.
12. Click the icon to select the filters you created from the list of saved filters.
13. Click a filter in the list of saved filters. If a filter does not display in the list, enter the name of a filter in the search field and select it.

Note: You can delete a filter from the list or disable filters by selecting **none** in the list.

14. Apply the selected filter.

    Result

    The filter is displayed as a new favorite filter in the **Results** page.

Comparing test reports

After you run performance tests, you can analyze the difference between two or multiple test results in HCL DevOps Test Hub (Test Hub). For example, you can compare the performance of an application at different time slots or different milestone builds between two test results.

About this task

You can compare the test runs for performance tests that are in the same project or in different projects.

1. Select results in the **Results** page by checking the boxes in the first column.
2. Click the  icon

Note: When comparing multiple runs, you cannot compare multiple time-ranges or stages.



Result

The two or multiple test results are displayed in a browser window in the same report.

Viewing trending reports

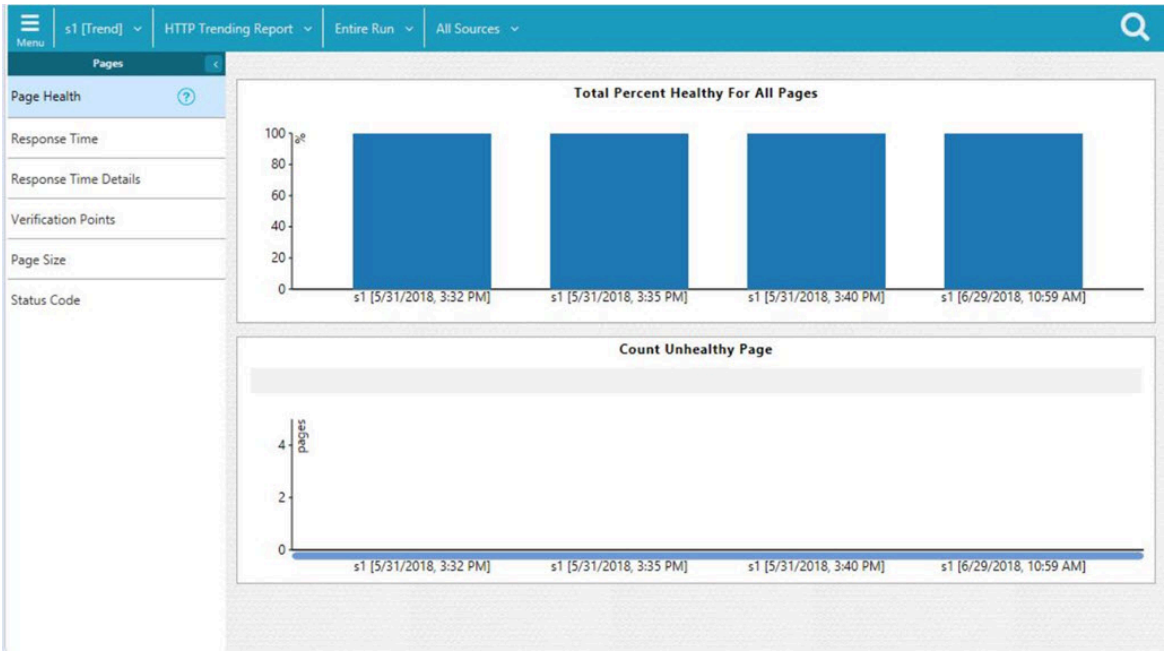
When the performance test runs are complete, you can view the trend of response time for an application over a period of time in HCL DevOps Test Hub (Test Hub) from trending reports. In addition to the response time, you can view the trend for the loops, transactions, and performance requirements for the application.

About this task

The trending reports are available for performance tests only.

1. Select results in the first column in the **Results** page.
2. Click the  icon to view the trending reports.

Note: When comparing multiple runs, you cannot compare multiple time-ranges or stages.



Results

You can analyze the trend information for multiple test results in the same report from a browser window.

Viewing a Functional Report

When the test run is complete, you can view the **Functional Report** in HCL DevOps Test Hub (Test Hub). The report contains a summary of the execution, the test environment, a detailed view of all tests, and the test steps that you executed.

1. Go to **Analyze > Results** in the navigation pane.

   The test assets that have completed their run are listed.
2. Select the tab based on the type of test assets that you ran as indicated in the following table:

If...	Then...
You ran AFT Suites, API Suites, Server Suites, or Compound Tests.	Click the **SUITES** tab.
You ran API Tests, Functional Tests, or Server Tests.	Click the **TESTS** tab.
You ran Suites or Tests and you want to find the test assets by using custom filters.	Click the **ADVANCED** tab.

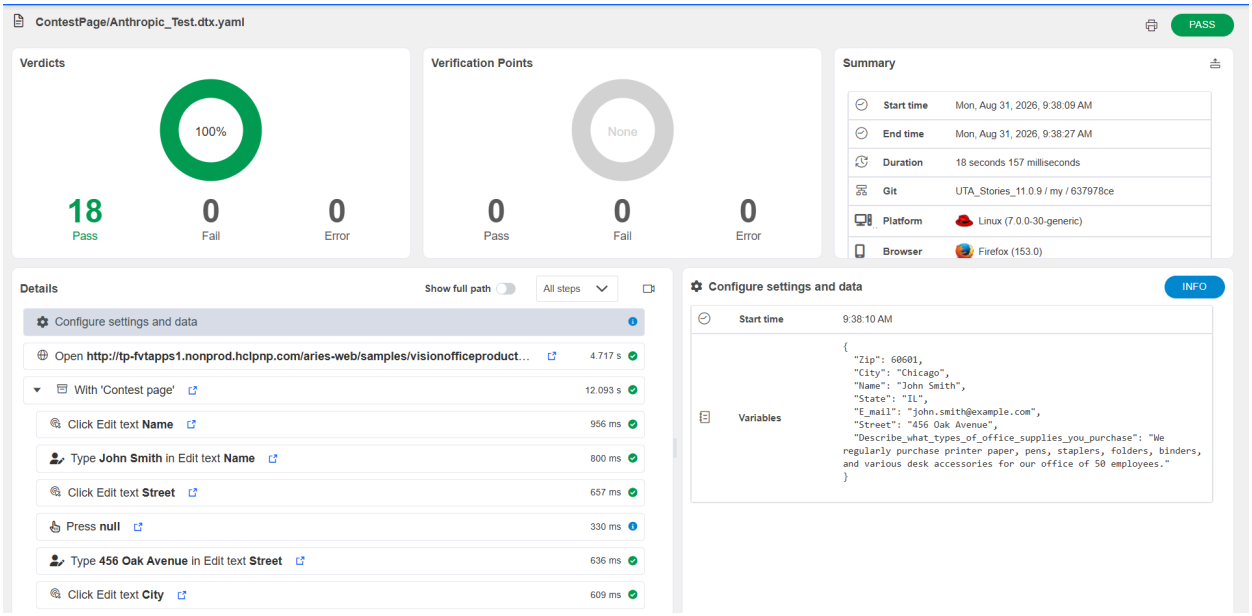
3. Click the test asset under the **Test** column.

   The results and reports panel is displayed.

4. Click the **Reports** tab, if it is not already open.
5. Click the **Functional Report** link.

   The **Functional Report** opens in a new tab.

   You can see the following information in the report:
   - Click the **Configure settings and data** node to view the initial variable values and dataset details used during test run. Variable details are now displayed directly in the parent activity viewer for better readability and navigation.
   - The top-level steps and first-level child steps are automatically expanded to provide better visibility.
   - A summary of the test run, including passed or failed status, count of passed or failed tests or test steps, and the verification points.
   - A summary of the test environment with the details of deployments, datasets, and components.
   - A detailed view of all tests that were executed in the case of an intelligent test suite, or in the case of a test with invoked tests.
   - A detailed view of all steps that were executed for each of the tests, including screenshots if any.



6. In the report, you can perform the following actions:

    a. Click the **Print** icon  to print the report or save it as a PDF.

    b. Click the **Browse Screenshots** icon  to view all the screen captures taken during the test run.

    c. Click the **View content** icon  to view the variables or stack trace, if available.

    d. Click the **Download attachment** icon  to download the variables or stack trace as a `.txt` file, if available.

    e. Click the **Previous failure** icon  and **Next failure** icon  to navigate between the failed steps, if there are failed steps in the report.

    f. Click the **Collapse** icon  to hide the test, verdict, and summary sections and provide more space for test and step details.

g. Click the **Expand** icon  to view the test, verdict, and summary sections.

Note: The Functional Report retains its last-used state. If it was last viewed in collapsed or expanded mode, it opens in the same mode the next time you access it.

Results

You have viewed the **Functional Report** and analyzed the summary of the tests that you ran.

Navigating to a test, suite, or step from the functional report

You can navigate to the related test, suite, or specific step from the functional report to investigate failures, review details, or make corrections. By using this feature, you can locate test resources faster and fix issues quickly.

Before you begin

You must have run a test or suite in Test Hub.

1. Go to **Analyze > Results** in the navigation pane.

   The test assets that have completed their run are listed.

2. Select the tab based on the type of test assets that you ran as indicated in the following table:

If...	Then...
You ran AFT Suites, API Suites, Server Suites, or Compound Tests.	Click the **SUITES** tab.
You ran API Tests, Functional Tests, or Server Tests.	Click the **TESTS** tab.
You ran suites or tests, and you want to find the test assets by using custom filters.	Click the **ADVANCED** tab.

3. Click the test asset under the **Test** column.

   The result and report panel is displayed.

4. Click the **Reports** tab, if it is not already open.
5. Click the **Functional Report** option.

   The functional report page opens in a new tab.

6. Click the expand icon  to view the result details in a functional report., then

   The individual test steps or suite results are displayed

7. In the **Details** panel, enable or disable the **Show full path** toggle to control the path displayed for each test.

   When **disabled**, only the name of the test asset is displayed.

   When **enabled**, the complete path of the test asset is displayed.

8. Click the **Open** icon to access the specific test, suite, or a step in the **Test Editor**.
   If the selected test, suite, or a step is on a different branch, the **Open Asset** dialog prompts you to switch to the appropriate branch before opening the asset in the **Test Editor** page.

   If the selected test, suite, or a step is on the same branch, it opens directly in the **Test Editor** in a new tab.

   Note: The **Open** icon is disabled if the test, suite or a step is deleted or renamed.

Results

You have directly navigated to a selected test, suite, or specific step from the functional report.

Re-executing tests from results

You can re-execute a test, a schedule, a collection of tests or schedules from HCL DevOps Test Hub (Test Hub) if you want to retain the same commit id.

Before you begin

You must have executed a test or multiple tests from the **Execution** page in Test Hub.

Re-executing a test from the **Results** view requires a Tester or an Owner role.

About this task

The following procedure describes how to re-execute a single test or multiple tests at a time.

Perform all the following steps from the Results page.

- Proceed as follows to re-execute one test at a time:

  1. Identify the test that you want to re-execute and click the **Re-execute** button in the Actions column.
     Result
     The **Execute test asset** dialog displays the same parameters as the ones that were set to execute the initial test from the **Execution** page.

  2. Select another **Version** of the test.

Note: If you select another version of the test, the settings configured in the dialog and in the **Environment**, **Data Sources** and **Locations** tabs might change.

3. Modify the **Schedule** parameters.

   Note: If you don't want to execute the test now, schedule the time when you want the test to be re-executed.

4. Click the **Environment** tab and select another **API test environment** if multiple environments were initially set for a test that is running an API Suite.
5. Click the **Data Sources** tab and select an override option if multiple data sources were defined for the initial test run.
6. Click the **Variables** tab and add a new variable if you want to re-execute a test asset with a different variable from the one configured in the asset.
7. Click the **Location** tab to override the default cluster location if the test asset has static agents configured.
8. Click **Advanced** to modify the advanced parameters that were set for the initial test run.
9. Enter a new instance of **Java arguments** if applicable to the test.
10. Enter the environment variables that must be passed to the test run at runtime in the **Environment Variables** field.
11. In the **Resource Monitoring** tab, click the  , and press the Ctrl + Space keys, or enter the initial letter of a label to select a label in the list and override the current sources in the schedule.

    Note: The **Resource Monitoring** tab is displayed for performance schedules and only if you have enabled the **Resource Monitoring from Service** option from Test Performance.

    The label field is grayed if you don't have any label entered in your current project. To collect data during the test execution, you must first enter labels in the Resource Monitoring sources page from Test Hub.

    For more details about controlling Resource Monitoring sources and labels, see the links at the end of this page.

12. Click the **Re-execute** icon to re-execute the test now or at a scheduled time.

- Proceed as follows to re-execute multiple tests at a time:

1. Click the checkboxes for the tests that you want to re-execute.
   You can select them one by one or select all of them.
   Result
   A toolbar is displayed.
2. Click the **Re-execute** icon  in the toolbar.
   Result
   A message indicates the number of selected tests and the number of tests to be re-executed. The tests are re-executed with the same initial commit ids and parameters.
3. Click **Execute**.

Result
You can view the results, logs, and reports of the re-executed tests in the **Results** page.

Related information

Creating a project on page 1213

Test results and reports on page 985

Controlling resource monitoring sources in a schedule on page 1079

Test run configurations on page 610

Test results data in Grafana

By using HCL DevOps Test Hub (Test Hub) Results Data and Grafana, you can retrieve and explore test log data that is generated in Test Hub. You must install and configure the Test Hub Results Data plugin in Grafana and create an instance of a data source that you use to explore your test log data.

Grafana is a visualization and analytics software. You can use Grafana to explore Test Hub metrics.

You must use Test Hub Results Data, which is a data source plugin, to make requests to Test Hub and represent data in a Grafana dashboard.

Installing and configuring the DevOps Test Hub Results Data plugin

You must install and configure the DevOps Test Hub Results Data plugin that is packaged along with HCL DevOps Test Hub (Test Hub) to view and analyze the log data in Grafana. You can view the test log data that is generated when you run a test from Test Hub in a Grafana dashboard.

Before you begin
You must have installed Grafana on your computer.

1. Perform the following steps to install the DevOps Test Hub Results Data plugin:
    a. Download the DevOps Test Hub Results Data plugin from the following URL:

    ```
    <your server address>/test/artifacts/grafana/devops-testhub-datasource.zip
    ```

    b. Install the DevOps Test Hub Results Data plugin by extracting the compressed file into your Grafana plugin directory.
    c. Restart the Grafana server.
2. Perform the following steps to configure the data source by using the DevOps Test Hub Results Data plugin:
    a. Enter the address of the local Grafana server in your web browser:

    Important: You must sign in with `http://localhost:3000` or `http://localhost:8080` URLs to use the plugin.

    b. Follow these steps to verify that the DevOps Test Hub Results Data plugin is available in Grafana:
        i. Click **Administration > Plugins and data > Plugins** in the left navigation panel of Grafana.
        ii. Enter **DevOps Test Hub Results Data** in the search bar in the **Plugins** tab.
        iii. Click **DevOps Test Hub Results Data** in the search results.

        The DevOps Test Hub Results Data page is displayed. You can find the links to Test Hub documentation and the description of the plugin on this page.

    c. Perform the actions indicated in the following table to create an instance of a data source in Grafana by using the DevOps Test Hub Results Data plugin on the **DevOps Test Hub Results Data** page:

    Note: Users with the role of a server administrator can add data sources.

Options	Actions
If you want to create an instance of the data source in Grafana by using the proxy connection	Perform the following steps: i. Click **Add new data source**. The data source configuration panel is displayed. ii. Click **Default** if you want to set the data source as the default data source in Grafana. iii. Enter a name to identify the data source. iv. Enter the URL of Test Hub in the **Server** field. v. Enter the token that you created in Test Hub. **Note:** The **Use proxy** is enabled by default. vi. Click **Save and test**.

Options	Actions
	Note: If the server certificate is self-signed, an error message is displayed. You must add the CA certificate to establish the connection with the server. vii. Perform the following steps to add the CA certificate: 1. Enable **With CA Cert** by using the toggle icon. 2. Add the CA certificate in the **CA Cert** field that you saved during the installation of Test Hub. 3. Click **Save and test**.
If you want to create an instance of the data source in Grafana with additional Helm parameters **Important:** Before you install Test Hub, you must use an additional Helm command parameter **router.allowedOrigin** during the installation. You can also run this command after installation. See the Additional Helm parameters topic in the related information section.	Perform the following steps: i. Click **Add new data source**. The data source configuration panel is displayed. ii. Click **Default** if you want to set the data source as the default data source in Grafana. iii. Enter a name to identify the data source. iv. Enter the URL of Test Hub in the **Server** field. v. Enter the token that you created in Test Hub. vi. Disable **Use proxy** by using the toggle icon. vii. Click **Save and test**.

Results

You have installed and configured the DevOps Test Hub Results Data plugin in Grafana.

What to do next

You can retrieve the test log data from Test Hub by using queries.

Related information

Additional Helm parameters on page 117

Download Grafana

Install Grafana plugins

Access management for DevOps Test Hub on page 1194

Add a data source

Explore

Viewing test results data in Grafana on page 1003

Viewing test results data in Grafana

After you install and configure DevOps Test Hub Results Data plugin in Grafana, you can retrieve the test log data that is generated in HCL DevOps Test Hub (Test Hub) and view the test results data in Grafana. You can use queries to display graphs, statistics, or tables. You can also view the details in a Grafana dashboard.

Before you begin

You must have completed the following tasks:

- Ran tests to get test logs results in Test Hub. For more information about the procedure, refer to Test logs on page 1007.
- Installed the DevOps Test Hub Results Data plugin and created an instance of data source for DevOps Test Hub Results Data. See Installing and configuring the DevOps Test Hub Results Data plugin on page 1000.
- Opened Grafana with one of these ports: `http://localhost:3000` or `http://localhost:8080`.

1. Click **Explore** in the **Toggle** menu in Grafana.

   The **Data query** page with the default data source is displayed.
2. Select **DevOps Test Hub Results Data** from the list of data sources.
3. Select the type of query by performing any of the following actions when you want to view statistics, events, or activities from the test data:

To view	Actions
The variation of a metric over time.	Perform the following steps: a. Click the time-series statistics tab icon  . b. Go to Step 4 on page 1004.

To view	Actions
The values of the metric computed over the entire time range.	Perform the following steps: a. Click the overall statistics tab icon  . b. Go to Step 4 on page 1004.
The events recorded in the test log.	Perform the following steps: a. Click the **Events** tab. b. Go to Step 4 on page 1004.
The activities recorded for all events in the test log.	Perform the following steps: a. Click the **Activities** tab. b. Go to Step 4 on page 1004.

4. Define the scope for the query by performing the actions indicated in the following table:

Options	Actions
If you want to use the data from the team space	**Note:** You can select the team space as the scope for your query only if you want to view the time-series statistics or overall statistics. Perform the following steps: a. Select the **Team Space** option. b. Select the team space from the list of team spaces that are listed. c. Select either **Service**, if you want to use Resource Monitoring data or **Stubs**, if you want to use the virtual service data in the query. d. Go to Step 5 on page 1005.
If you want to use the data from your project	**Note:** You can select the project as the scope for your query if you want to view any of the supported query types. Perform the following steps: a. Select the **Project** option. b. Select the project from the list of projects that are listed. c. Select the **Result** option. d. Select the result from the list of results that are displayed. Alternatively, you can also enter text for the name or ID of the result in the **Search** field, and then select the result from the listed items. e. Go to Step 5 on page 1005, if you selected the query type as time-series statistics or overall statistics, or go to Step 6 on page 1005, if you selected the query type as **Events** or **Activities**.

5. Perform the following steps to select metrics for your query:
    a. Select a metric from the list of metrics in the row of the **Metrics** option.
    b. Assign a name for the metric that helps you identify it, by entering text in the **Metric** field.
    c. Select the aggregation factor for the metric. For example, select **Sum** to add all values of the metric.
    d. Optionally, click the icon to add a separator.

    With separators, you can view details of a specific project or test result in comparison with the values of the metric across projects or results in the team space or project.

    e. Optionally, click the **Cumulated** toggle-button if the values for the metric must be added cumulatively over time. The default option is set to be non-cumulative.
    f. Add another metric by clicking the **Add metric** icon .

    Note: If you added multiple metrics and you want to remove a metric, you can click the **Remove metric** icon of the metric.

    g. Go to Step 7 on page 1005.
6. Perform the following steps for the **Events** or **Activities** query type:
    a. Enter the **Query Filters**.
    b. Enter the **Timing value** and **Value**.

    The timing values are sorted in ascending order. The value must be numeric so that it can be used in a graph.

    Tip: To find all the information needed on the **Explore** page of DevOps Test Hub Results Data, perform any of the following actions:
    - Select a result on the **Queries** page in Test Hub and copy the result number, query filters for the events or the activities, query filters, and timing values. You can find the project number in the URL of the **Queries** page.
    - Click **Copy curl command** on the **Queries** page in Test Hub and paste the curl command in a text editor. You can find all the information you need in the following command.



7. Click **Add query** if you want to combine queries from different projects, results ids, or with different properties in the query filters.
8. Click **Run query**.

Results

You configured a query to get the test log data from Test Hub, and you can view and analyze the log data displayed for your query.

What to do next

You can build a dashboard including graphs, charts, gauges, tables to get customized visual representations of your test result data and select options to display the test result values. See Dashboard overview for more details about dashboards in Grafana.

Viewing Surefire format reports

After you execute JMeter, JUnit, API, and Functional tests with advanced settings in HCL DevOps Test Hub (Test Hub), you can view the test results in a Surefire format report by using dedicated URLs.

Before you begin

You must have configured your tests with one of the following parameters:

- For JMeter, JUnit, Postman, and HCL® AppScan Suites, set the **history** parameter with **cisterna** value in the **Additional Configuration Parameters** field in the **Advanced** settings.
- For API Suites, AFT Suites, Compound Tests, Functional tests, and API tests, set the **-history cisterna** parameter in the **Program Arguments** field in the **Advanced** settings.

About this task

When a test run is complete, you can see the test result data in a report that is in a Surefire format. You can view this report by using dedicated URLs.

1. Click **Analyze > Results** in the navigation pane.
   Result
   The **Results** page is displayed.
2. Click a test result, and note its project identifier and test result identifier (ID) in the URL of the test log that is displayed in your browser. The URL is in the following format: `https://myProductionServer.com/test/#/Initial/project/<ProjectID>/analyze/test-log/<ResultID>`.
   The identifiers are required to view the Surefire test logs in the following steps.
3. Enter one of the following URLs with the appropriate information in a web browser or in an external tool that you use to view Surefire reports:
   - `/rest/projects/<ProjectID>/data/views/surefire/`

     Enter your REST API address and the project identifier to view the list of all test cases in the project with details about their status.

     Example: `https://myProductionServer.com/rest/projects/230200/views/surefire/`
   - `/rest/projects/<ProjectID>/results/<ResultID>/data/views/surefire/`

Enter your REST API address, the project identifier, and the test result identifier to view the results for this test result id in the project.

Example: `https://myProductionServer.com/rest/projects/230200/results/80250/data/views/surefire/`

Results

The test result reports are displayed in a Surefire format.

Related information

Test logs on page 1007

Test log queries on page 1022

Test logs

When you start a run of a test asset, HCL DevOps Test Hub (Test Hub) starts to log information about the test run in a test log. You can view the test log while the test runs and after the run is completed from the **Test Log** page.

Overview

Test Hub captures the test log for the supported tests that you can run on Test Hub. For the list of supported tests, see Supported suites and tests on page 110.

Test Hub uses a built-in PostgreSQL database to store information about the test while the test runs on the server. Test Hub captures and displays the test log as the default configuration and you do not have to configure other settings during or after installation of the server.

Supported formats of test logs

The formats in which you can view the test logs are as follows:

Table 31. Supported formats for test log display

Log format	Description	Action
Default format	Displays the test log in the built-in format on the **Test log** page. For example, the test log can be as follows:	You do not have to set any Program argument while you configure the test run.

Table 31. Supported formats for test log display (continued)

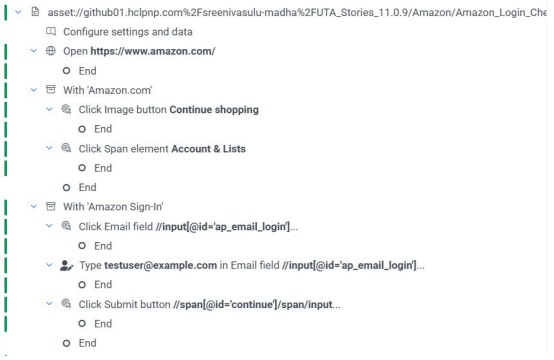
Log format	Description	Action
	 Click the **Configure settings and data** node to view the initial variable values and dataset details used during test run. Variable details are now displayed directly in the parent activity viewer for better readability and navigation. The top-level steps and first-level child steps are automatically expanded to provide better visibility.	
Text format	Displays the test log in a plain text format and is displayed in a different browser tab or window than the one that displays the **Test log** page. For example, the test log can be as follows:	You must set the **-history testlog** Program Argument in the **Advanced** settings dialog while you configure the test run.

Table 31. Supported formats for test log display (continued)

Log format	Description	Action
	---------- Name: autoResults Description: Test Version: null Verdict: fail Type: com.ibm.rational.test.lt.lttest ---------- ---------- Name: DefaultSchedule Description: Timestamp: May 25, 2022 at 9:24:10 AM GMT null Event type: com.ibm.rational.test.lt.scheduleStart Properties: Test Log Filter (ALL): 100 (ALL) Test Log Filter (WARNINGS): 100 (ALL) Test Log Filter (ERRORS): 100 (ALL) Type: start ---------- ---------- Name: DefaultUserGroup Description: Timestamp: May 25, 2022 at 9:24:10 AM GMT null Event type: com.ibm.rational.test.lt.usrGroupStart Properties: Type: start ---------- ----------	

Activities and events

The information about the tests that is collated by Test Hub can be classified into the following types:

Table 32. Types of information collated for test logs

Information type	Description
Activity	Specifies the events that are performed when the test runs as part of a test step. An activity comprises different events and is represented as a collection of events that were performed over time. For example, opening a specific web page by using a web browser in the test can be an activity that comprises the different events such as connecting to the web site, sending of a request, and receiving a reply. Each of these events might take a specific amount of time, which are captured in the test log.
Event	Specifies an action performed as part of a test step. For example, start connection can be an event in a test that requires a specific web page to be opened in a web browser, as detailed in the example for an activity. The exact time at which this event occurred when the test ran is captured in the test log.

Test log tasks

You can find information about the tasks that you can perform on the the test logs in the **Test log** page. The test logs for the tests that you ran on Test Hub are displayed in the **Test log** page.

The tasks that you can perform on the test logs are summarized in the following table:

Table 33. Test log tasks

Task	More information
Viewing test logs after you start a test run on the **Execution** page.	Viewing test logs after you start a test run on page 1010
Viewing test logs after the test run is completed.	Viewing test logs after the test run is completed on page 1011
Viewing details about the test log.	Test log details on page 1015

Viewing test logs after you start a test run

When you configure and start a test run, you can view the test log that is created progressively while the test runs.

Before you begin

You must have configured and started a test run. See Test run configurations on page 610.

Note: You must have opened the **Execution** page to start the test run.

1. Click **Show in progress page** icon in the **Actions** column of the test asset on the **Execution** page.

   The asset is displayed on the **Progress** page.
2. Click the **Open action menu** icon.
3. Click the **View test log** option.

   The **Test log** page is displayed with the test log collated for test run.

   Note: Because the test log is collected progressively as the test runs, you must click **Refresh** to view the latest or the updated test log.

Results

You have viewed the test log for the test asset that you ran.

What to do next

You can view the details of the test log and decide on the actions you might want to take on the test log. See Test log details on page 1015.

Viewing test logs after the test run is completed

After you configure a test run and the run is completed, you can view the test log that is created for the test.

Before you begin

You must have configured and started a test run, and noted that the test run has the status as *Complete* displayed on the **Progress** page.

About this task

You can view the test log that is created for a specific test run from the following pages:

- The **Progress** page. To view the test log from the **Progress** page, go to Step 1 on page 1011.
- The **Results** page. To view the test log from the **Results** page, go to Step 2 on page 1011.
- The **Test log** page. To view the test log from the **Test log** page, go to Step 3 on page 1014.

1. Perform the following steps after the test completes its run:
    a. Go to **Execute > Progress**, if you are not already on the **Progress** page.

       The test assets that are running or have completed their run are listed. If you are already on the **Progress** page, only the test asset that you selected on the **Execution** page is displayed with the status as *Complete*.

    b. Click the **Open action menu** icon ⋮ under the **Actions** column in the row of the test asset.

    c. Click the **View test log** option.

       The **Test log** page is displayed with the test log collated for test run.

       The top-level steps and first-level child steps are automatically expanded to provide better visibility.

       Click the **Configure settings and data** node to view the initial variable values and dataset details used during test run. Variable details are now displayed directly in the parent activity viewer for better readability and navigation.

    d. Go to Results on page 1014.

2. Perform the following steps after the test completes its run:
    a. Go to **Analyze > Results**.

       The test assets that have completed their run are listed.

    b. Select the tab based on the type of test assets that you ran as indicated in the following table:

If...	Then...
You want to run AFT Suites, API Suites, Compound Tests, Rate Schedules, VU Schedules, JMeter tests, JUnit tests, or Postman tests..	Click the **SUITES** tab.
You ran API tests, Functional tests, or Performance tests.	Click the **TESTS** tab.
You want to find Suites and tests by using custom filters, when you ran Suites and tests.	Click the **ADVANCED** tab.

c. Identify the test asset that you ran by performing any of the following actions:

- Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test asset by entering any text contained in the test asset name in the **Search** field.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab to filter the displayed assets based on the asset type.

  For example, select **API Suite** in the **SUITES** tab to display all API Suites or select **Functional Test** in the **TESTS** tab to display all Functional tests that are in the branch.
- Click the **Filter** icon in the **ADVANCED** tab, and then create a filter query by using the **New filter** option by performing the following steps:
  i. Click **New filter**.
  ii. Enter a name for the filter.
  iii. Select an operator, and then add a rule, or a group of rules.
  iv. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

  You can select a parameter from the following list:
  - Type
  - Test Asset Name
  - Test Asset Path
  - Last Result
  - Next Run
  - Components

v. Save the filter query to save and apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- Retrieve and apply a saved filter query, if you have saved filter queries previously by performing the following steps:

Note: The filter query applied previously is selected and the assets based on that filter query are displayed. To apply a different filter query, you must have created and saved the filter query.

i. Click the **Filter** icon  in the **ADVANCED** tab.

The filter queries that you created and saved are displayed.

ii. Click the filter that you want to apply.

The test assets that match the filter criteria are displayed.

You have identified the test asset that you ran.

d. Click the test asset under the **Test** column.

The results and reports panel is displayed.

e. Click the **Reports** tab, if it is not already open.

f. Click the **Test Log** link.

The test log created for the test run is displayed in the **Test log** page.

g. Go to Results on page 1014.

3. Perform the following steps after the test completes its run:

a. Go to **Analyze > Test Log**.

The **Test log** page is displayed. The test assets that were run are displayed.

b. Identify the test asset by its name or the type for which you want to view the test log.

c. Click the test asset under the **Test** column.

The test log created for the test run is displayed.

d. Go to Results on page 1014.

Results

You have viewed the test log for the test asset that you ran.

What to do next

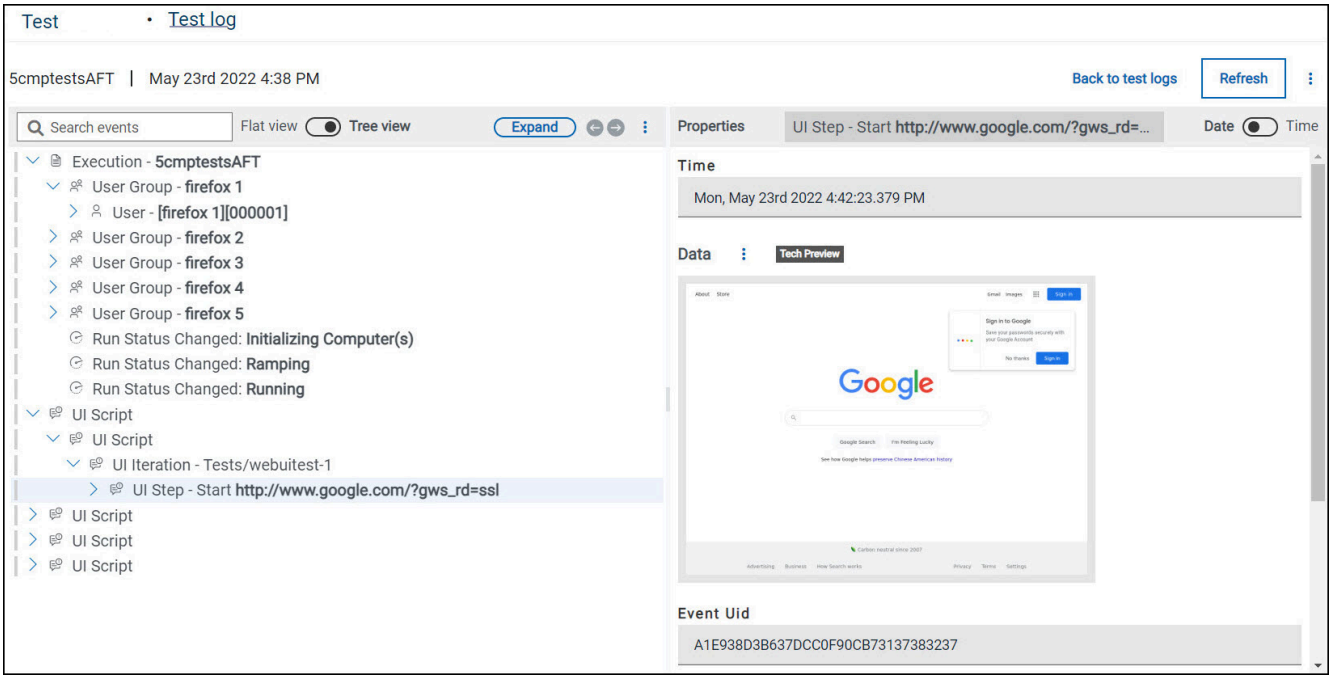
You can view the details of the test log and decide on the actions you might want to take on the test log. See Test log details on page 1015.

Test log details

You can find information about the details that are displayed for a test log on the **Test log** page. You can also find information to use the different options in the UI to change the view of the test log, find details of the events captured, navigate through the log, view or download attachments that are captured.

Before you can view the test log, you must have started a test run and opened the test log that is displayed on the **Test log** page.

An example of a test log that is displayed on the **Test log** page is as follows:



You can find the following details that are displayed on the **Test log** page:

Table 34. Test log details

Details about...	Examples as displayed in the Test log page...
The project name	An example of the project name that is displayed is as follows: Test
The name of the page	An example of the page name that is displayed is as follows:

Table 34. Test log details (continued)

Details about...	Examples as displayed in the Test log page...
	• Test log You can click the Test log link to view the test logs of all tests that ran in your project, if you are viewing the test log of a specific test run.
The name of the test asset	An example of the name of the asset that is displayed is as follows: 5cmptestsAFT
The date and time when the test run was started from the **Execution** page	An example of the date and time when the test run was started that is displayed is as follows: May 23rd 2022 4:38 PM
The activities and events captured in the test log are displayed in the **Tree view** in the **Test log** panel.	An example of the log in the tree view that is displayed is as follows: Execution - **5cmptestsAFT** User Group - **firefox 1** User - **[firefox 1][000001]** User Group - **firefox 2** User Group - **firefox 3** User Group - **firefox 4** User Group - **firefox 5** Run Status Changed: **Initializing Computer(s)** Run Status Changed: **Ramping** Run Status Changed: **Running** UI Script UI Script UI Iteration - Tests/webuitest-1 UI Step - Start **http://www.google.com/?gws_rd=ssl** UI Script UI Script UI Script
An event in the test log. Details of the event is displayed in the **Properties** panel.	An example of the details that is displayed in the **Properties** panel is as follows:

Table 34. Test log details (continued)

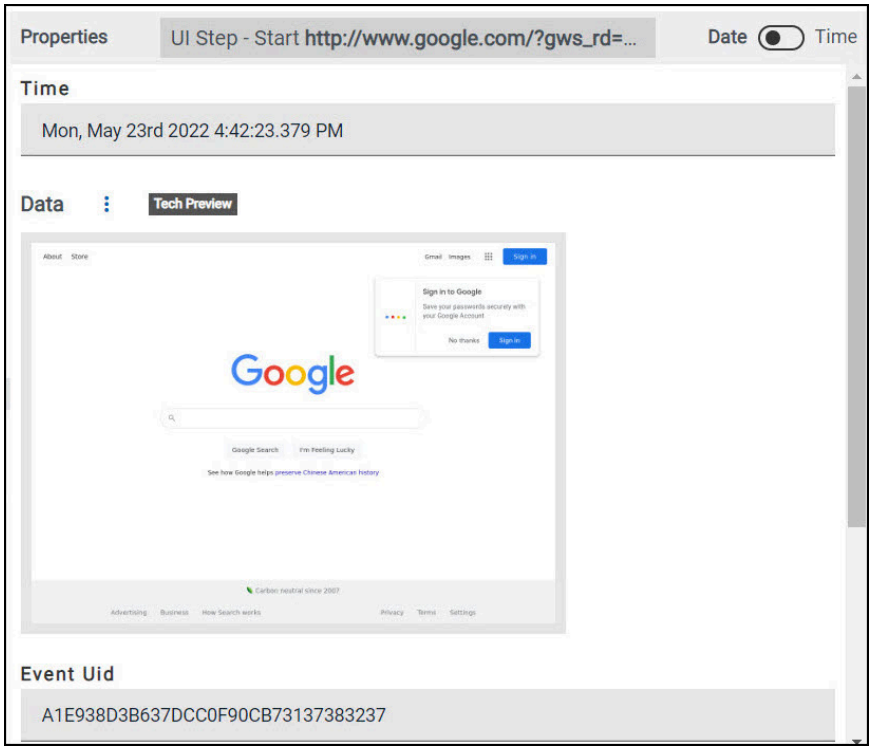
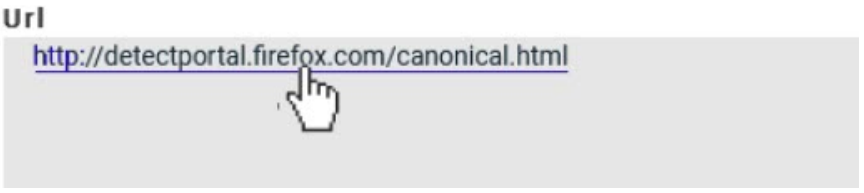
Details about...	Examples as displayed in the Test log page...
	
The link to a URL that is recorded in an event. The URL is displayed as a link. When you click this link, the specific URL opens in a web browser.	An example of a link recorded in the event. 
An attachment of an event in the test log. The details of the attachment is displayed in the **Properties** panel.	An example of a text extracted from the attachment. 

Table 34. Test log details (continued)

Details about...	Examples as displayed in the Test log page...
The reason of a Verification Point (VP) that is captured in an event in the test log is displayed in the **Properties** panel.	An example of the reason in a VP that is displayed in the **Properties** panel is as follows: 

You can perform the actions detailed in the following table to change the view of the test log, find details of the events captured, navigate through the log, view or download attachments that are captured.

Table 35. Tasks that you can perform on the Test log page

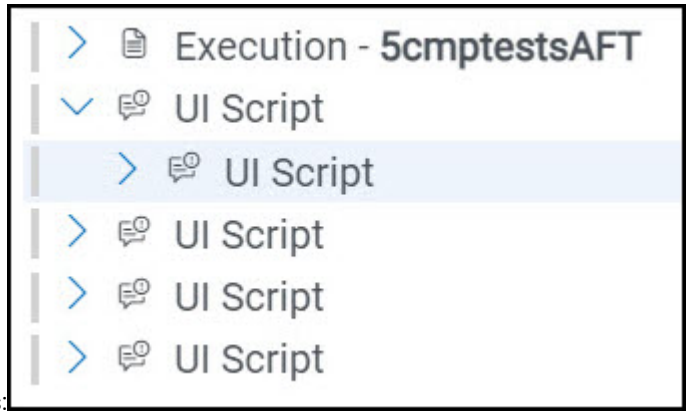
Task	Action
Changing the *Tree view* to a *Flat view* for the display of events in the test log panel. **Note:** The *Tree view* is the default view of the events in the test log.	Click  to toggle from the **Tree view** to the **Flat view**. **Note:** You can change the view from the one that is displayed by clicking  .
Expanding the display of events in the test log panel.	Click  to expand the events that are contained within the event selected in the test log panel. For example, the collapsed view in the test log panel can be as follows: 

Table 35. Tasks that you can perform on the Test log page (continued)

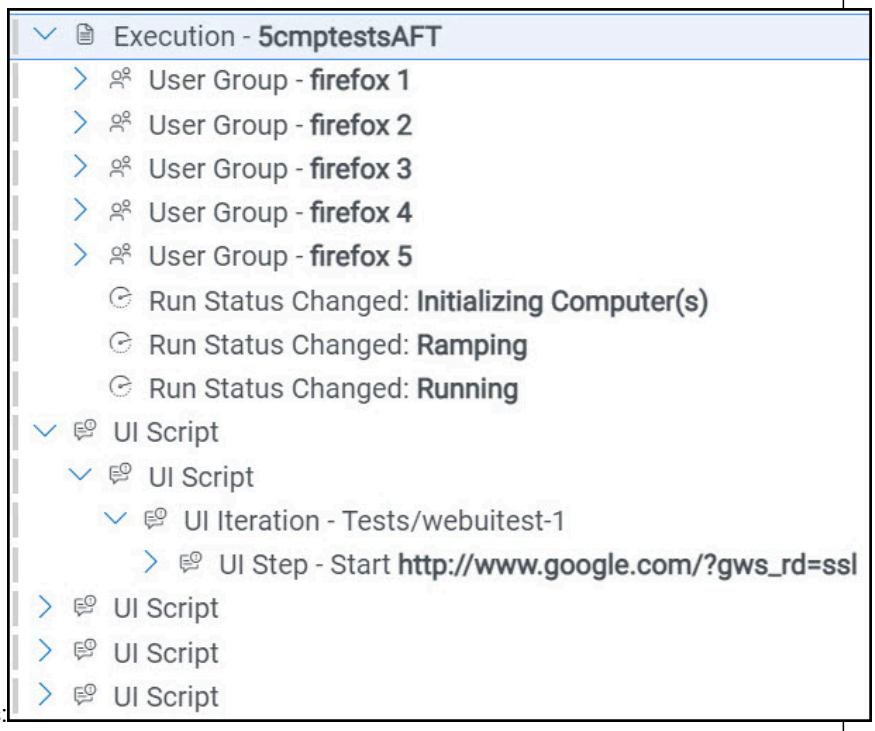
Task	Action
	When the event is expanded, the view in the test log panel can be as follo  ws:
Collapsing the display of events in the test log panel.	Click Collapse to collapse the events that are contained within the event selected in the test log panel.
Viewing primary activities when all activities are displayed.	Perform the following actions in the test log panel: 1. Click the **Open action menu** icon. 2. Click **Show only primary activities**. The primary activities captured in the test log are displayed.
Viewing all activities when only primary activities are displayed.	Perform the following actions in the test log panel: 1. Click the **Open action menu** icon. 2. Click **Show all activities**. All the activities captured in the test log are displayed.

Table 35. Tasks that you can perform on the Test log page (continued)

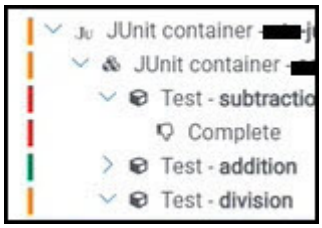
Task	Action
Viewing failures captured in the test log.	The test log displays events marked with a colored vertical bar as follows: • A green bar is marked for an event that passed in the test step. • An orange bar is marked for an event that had an error in the test step. • A red bar is marked for an event that failed in the test step. For example, the view can be as follows:  You can navigate up and down in the view from one error or failed event to another one by using arrows .
Searching for events in the test log.	In the search bar, you can enter text to find instances of events and properties that match with the text you entered. As a result of the search, the title of the nodes in the view and the names of the properties are highlighted. When the number of results found is displayed, you can press the **F3** key to navigate from one search result to another in the tree view or flat view. You can press **Ctrl** and **F3** to navigate from one result to another in the properties pane.
Viewing the properties related to the event that you select in the test log panel.	When you select an even in the test log panel, its properties are displayed in the properties panel. For example, the details that are displayed for an event can be as follows:

Table 35. Tasks that you can perform on the Test log page (continued)

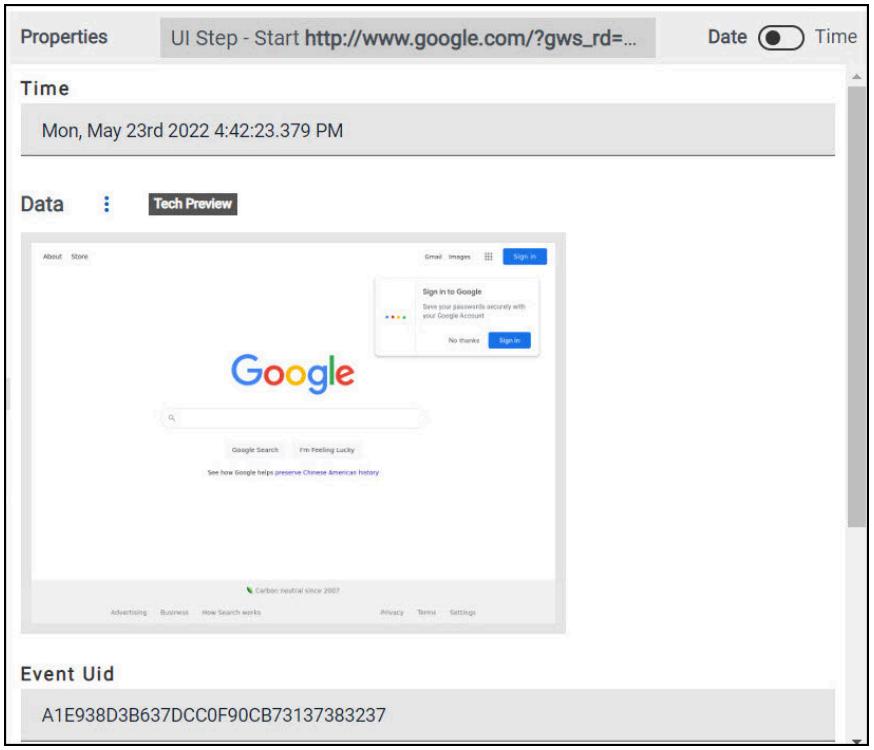
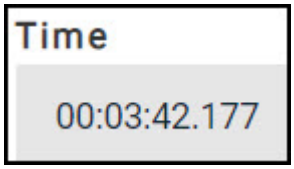
Task	Action
	 You can view the following information: • The variable in the event is displayed. • The date stamp showing the time at which the event was captured in the test log. You can toggle between the calendar time and the duration from the time the test started its run in the cluster by clicking the **Date** and **Time** toggle icon  . In the example, the date that is displayed in the **Time** field is as follows:  When the **Time** option is selected, the duration that is displayed in the **Time** field is as follows:

Table 35. Tasks that you can perform on the Test log page (continued)

Task	Action
	 This time is the time that has elapsed from the start of the test run to the event in the test step. • The name of the event. • The event ID. • Data that was captured as part of the test step. The items that are captured can be images, text files, PDF files, XML files, JSON files, and other supported formats. You can view the attachment that is in the **Data** field, if it is supported to be viewed in a web browser by clicking the attachment. The attachment displays in a browser tab other than the one in which the **Test log** page is displayed. You can also copy the URL of the attachment or download a copy to your computer to use the attachment elsewhere.
Viewing all test logs in the **Test log** page.	When you view the test log of a specific test asset and you want to return to the **Test log** page to view the test logs of the other test assets, you can click the **Back to test logs** option.

Test log queries

When a test run is complete in HCL DevOps Test Hub (Test Hub), if a test log is generated, you can run queries to extract and view test log data. The query is used to search for the events or activities in the test logs with a project or a result scope.

The query page contains a configuration section in which you enter and select search criteria.

The difference between an activity and an event are the following ones:

- An activity is characterized by a duration whereas an event is characterized by a particular time.
- An activity contains events.

When you define the search criteria and run a query, the returned test log data is displayed in a table. You can manage the data in the table by adding, moving, or deleting the columns.

Prerequisites

To get result log data queries, you must configure your tests with one of the following parameters to enable the test log and data query process:

- For JMeter, JUnit, and Postman Suites, set the **history cisterna** parameter in the **Additional Configuration Parameters** field in the **Advanced** settings 

.
- For API tests and Suites, AFT Suites and Functional test, and Compound Tests, set the **-history cisterna** parameter in the **Program Arguments** field in the **Advanced** settings.

Note: When you set this parameter, you can run your queries to collect test log data into Test Hub.

Query scope

A **Test result query** link is available in the test reports in the **Results** page when a test log is generated. If you click the link, it opens the query page in Test Hub where you can view all the events or all the activities for the current test result only. In the **Scope** field of the query page, you find the result id. You can run new queries with different properties as search criteria for this result.

You can also run new queries with different properties as search criteria for all the results in a project if you open the query page from **Analyze > Queries** in the navigation pane. In this case, the **Scope** field is at the project level.

In the **Queries** page, you can perform the following actions:

- Filter the events or activities.
- Sort your query results in an ascending or a descending order.
- Add, move, or delete fields in the columns of the table that displays the data of the query.
- Save a query, rename, reset or delete a query.
- Export a query.

  You can export the queries to Shell, Curl, or PowerShell scripts. You can select the script, the output format, and the number of results that are returned by the query.

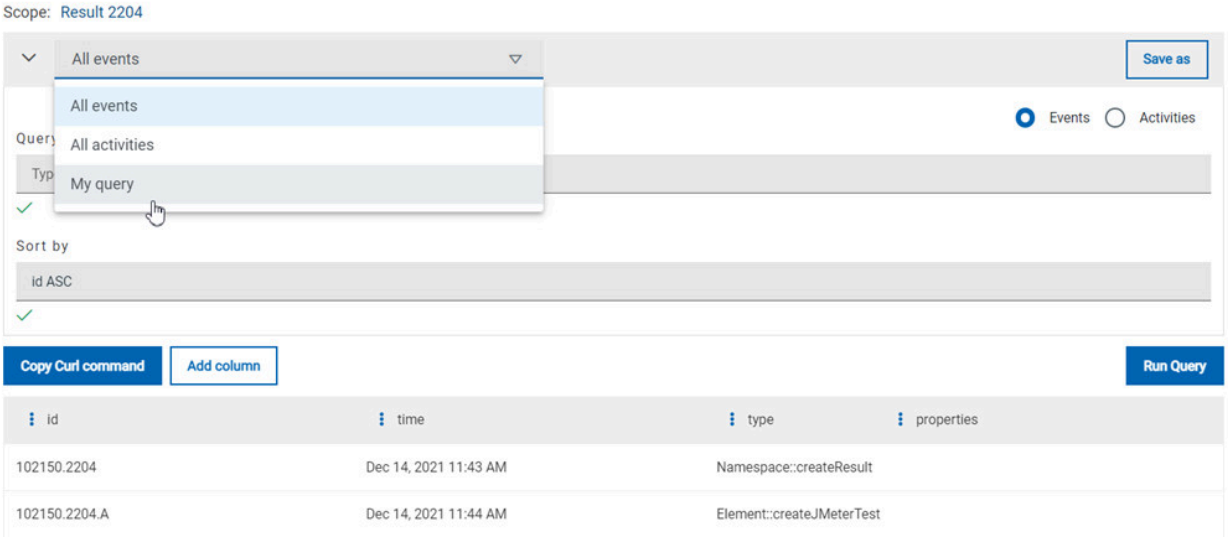
Configuring and running log queries

When a test run is complete and a test log created, you can configure and run queries to view test log data in HCL DevOps Test Hub (Test Hub) for one test log result or for all the test log results in a project.

Before you begin

You must have completed the following tasks:

- Configured and run a test. See Test log queries on page 1022.
- Opened the Queries page from the Results page or the navigation panel.



About this task

In the **Queries** page, you can configure a query and save it. You can customize the way the log data is displayed in the results table.

1. Click **Events** or **Activites**.
2. Enter properties and operators in the **Query filters** to filter the list of events or activities.

   Note: You can use the **Ctrl-Space** keyboard shortcut in this field to get the list of properties and operators that you can enter as filters to create your query.

   Example: Select **Activities** and enter 'type is JUnitTest' in **Query filters**.

3. Click **Run Query**.
4. Perform the following steps to save a query:

   a. Click **Save as**.
   b. Enter a name to identify the new query and click **Save**.

   Example: Enter 'My query'.

   Result

   The new query is displayed in the events and activities list.
5. Perform any of the following actions to configure the results table:

- Enter a property followed by the **ASC** operator or **DESC** in **Sort by** to sort the data in the table.

  Note: ASC is used to sort the data from the lowest value to the highest value and **DESC** to filter the data in a descending order. You can use the **Ctrl-Space** keyboard shortcut to assist you. If you enter a combination of sort criteria, separate them with commas.

  Example: Click the **Sort by** field, press **Ctrl-Space** and select in the list 'time' and enter 'ASC'.

- Click the action menu in a column of the table to change the column display, and then perform any of the following actions:
  - Click **Insert new column**, enter a field name in **Search new column** and click **Add**.
  - Click **Move** to move a column before or after another one in the table.
  - Click **Remove** to remove a column from the table.
- Follow these steps to add a column to the end of the table:
  a. Click **Add column**.

     Result: A window opens.

  b. Enter a field name in **Search new column**.
  c. Click **Add**.

6. Click **Save** to save the configuration of the results table.
7. Follow these procedures to manage your query after it is created:

   - Click the action menu that is next to the **Save** button and select **Reset** if you have just modified a query and you want to undo your modification.
   - Click the action menu and select **Delete** to delete the new query from the **Events and activities** list.
   - Click **Copy Curl command** if you want to export a query.

     Note: The curl command is copied to the clipboard. You can paste it in a command interface to use the query in an external tool.

Adding additional properties in test results

When you want to add additional properties to be included in test results after the test is run, you can do so by configuring a webhook to receive the customized results and defining the parameters in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Read and understood Configuration of a webhook on page 1248.
- Read and understood Configuring a webhook template on page 1248 and Configuring a server webhook on page 1256.

About this task

You can set up webhooks to send notifications to various messaging apps such as Microsoft Teams or Slack, or to any application that can receive an HTTP POST request, whenever specific events happen on Test Hub. For instance, a webhook can be triggered when a test run fails. After the event, the payload that is received by the application contains the details of the event, including the additional parameters that are added. If you want the result events to be correlated when using systems such as DevOps Velocity, you must add the properties that are present in the webhook template as additional parameters in the **Execute test asset** window.

1. Navigate to your project in Test Hub.
   The **Overview** page of the project is displayed.
2. Click **Manage > Webhooks**.
   The **Templates** tab is displayed.
3. Click **New Template**.
   The **New template** page is displayed.
4. Perform the following actions on the **DETAILS** tab.
   a. Enter a unique name for the template.
   b. Create a new channel type by clicking **Create new channel type ...** option.

      The **New channel type** dialog is displayed.

   c. Enter a unique name for the channel type, and then enter a description for the channel type.
5. Click **Apply**.
   The channel type is created and added to the **Channel Type** list.
6. Click **This template is suitable for events** and select the events from the **Execution Events** list.
7. Click the **Template** tab.
8. Click one of the following icons to select a predefined template sample in the **Template body**:
   - The **infinity** icon to insert a DevOps Velocity template
   - The **cloud** icon to insert a cloud event JSON template
   - The **Microsoft Teams** icon to replace a template with a Microsoft Teams template
   - The **Slack** icon to replace a template with a Slack template

   A message is displayed about the action that inserts a text in the template body.
9. Click **Ok**.
10. The text is inserted in the **Template body**.
11. Click **Apply**.
    The template is created and added to the **Templates** tab.
12. Click the **Webhooks** tab, and then click **New Webhook**.
    The **New Webhook** page is displayed.

13. Enter a name for the webhook in the **Name** field.
14. Enter the callback URL of the application or channel for which you are configuring a webhook in the **Webhook URL** field.
15. Click the **Channel Type**, and then select the channel type that you just created from the list.
16. Enter a description for the webhook in **Description**.
17. Select the template that you just created from the drop-down list.
18. Click **Apply**.
    The webhook is created and added to the **Webhooks** tab.
19. Click **Execution**, in the navigation pane.
20. Select the test that you want to run and click  in the **Actions** column to open the **Execute test asset** window.
21. Click **Advanced settings**.
22. Enter a parameter name that starts with result.property in the first edit field under **Additional Configuration Parameters**.
    For example, result.property.test_version.
23. Enter a value for the newly added parameter in the second edit field under **Additional Configuration Parameters**. For example, v1.5.14.
24. Click **Add** for the added parameter.
    You can add multiple parameters.
25. Click **Execute**.

Results

The additional parameter values are captured while your test is running. You can see the values in the response notification that you get for the webhook after the test run is complete.

What to do next

Navigate to the application or channel for which you have configured the webhook for receiving test result notifications, and then view the results.

Related information

Prerequisites to running tests on page 527

Test run configurations on page 610

Machine learning capabilities in DevOps Test Hub

HCL DevOps Test Hub (Test Hub) uses Machine learning (ML) algorithms to analyze tests that match certain requirements and criteria, and then presents the findings as insights and recommendations after the tests are run. You can use the insights and recommendations to interpret the test results and identify the problem areas in the system under test.

Test Hub has incorporated ML capabilities to provide an automated means to identify problems in the system under test.

Tests analyzed through ML

The following types of tests are supported for analysis by using ML capabilities in Test Hub:

- Performance tests
- Schedules

Parameters analyzed through ML

The following parameters in Performance tests or Schedules are supported for analysis by using ML capabilities in Test Hub:

- The **Response Time Lock-Step Pattern** parameter.
- The **Response Time Standard Deviation Pattern** parameter.
- The **Throughput Drop Pattern** parameter.
- The **Performance Agent Health** parameter.

Test Hub uses different analyzers for each of the parameters. For example, Test Hub uses the **Response Time Lock-Step Pattern** analyzer to analyze the tests for the **Response Time Lock-Step Pattern** parameter.

Criteria for analysis of the parameters

The following table lists the criteria used by the analyzers for analysis of the parameters in Test Hub:

Parameter analyzed	Criteria for analysis
Response Time Lock-Step Pattern	The analyzer identifies the **Response Time Lock-Step Pattern** parameter in the overall page response time observed against the user count based on the following criteria: • Tests have a minimum of 20 users. • Tests run for longer durations for a trend to be observed. • Tests contain a minimum of three different user groups and each group has an activity of more than two minutes.
Response Time Standard Deviation Pattern	The analyzer attempts to detect the response time of pages that are more than thrice the value of the standard deviation calculated for the page response time as the **Response Time Standard Deviation Pattern** parameter.
Throughput Drop Pattern	The analyzer attempts to detect sudden drops in network throughput as the **Throughput Drop Pattern** parameter that is based on the following criteria:

Parameter analyzed	Criteria for analysis
	Note: Sudden drops in throughput might be related to the performance tool itself, issues with network connectivity, or issues with signal-scalability of the system under test. • Tests have a minimum of 20 users. • Tests have stages with a reasonably high number of samples within a stage. • The length and intensity of the throughput drops that are less than 20% of the median observed within a stage are considered by the analyzer. When no throughput is received within 60 seconds, such drops are considered in the analysis. Short shark-tooth patterns in drops are not considered by the analyzer. • Parts of the stage with increasing users are not considered for analysis. For example, when the test adds one user every minute and the user range is between one user to 500 users, the **Throughput Drop Pattern** parameter is not analyzed.
Performance Agent Health	The analyzer inspects the health of agents that are participating in a performance execution and identifies potential configuration issues based on the following criteria: • **Minimum requirements** ◦ At least 10 users ◦ At least 10-minute run time ◦ Requires schedule with at least 1 static agent (Performance Agent Health insight is not generated if you are running the test on local host) • **Blocking threads** ◦ Considered as Blocking threads when 90% of available threads are used for 20% of the run time • **Pegged CPU** ◦ Considered as Pegged CPU when 90% of CPU is utilized for 10% of the run time • **Pegged CPU for interval** ◦ Considered as Pegged CPU for interval when there are intervals of high CPU utilization greater than 95% For example, above 95% of CPU utilized for 50% of a 10-minute interval or 98% of CPU utilized over a shorter interval. ◦ Considered as Pegged CPU for interval when there are spikes in CPU utilization during long test runs

Parameter analyzed	Criteria for analysis
	The agents that are not configured properly can distort performance assessments of the system under test. The analyzer can provide multiple insight types. The insight recommendation texts are parametrized and they provide specific recommendations such as identifying the agents with issues. You can view the following insights: • Pegged CPU for run insight • Pegged CPU for interval insight • Blocked thread insight

Analysis at the project level

When you as a *Project Owner* or *Tester* configure a run of a Performance Test or a Schedule from your project in Test Hub, you must ensure that the criteria required for ML analysis are met. During the test run, Test Hub uses its ML capabilities to analyze the supported parameters and presents its findings as insights on the **Insights** page.

Any member of your project can view the insights by clicking **Analyze > Insights** from the navigation in the project or from the **Insights** section on the **Overview** page in the project.

Project members after viewing the insights can react by either agreeing or disagreeing with the insights suggested by Test Hub.

Analysers in a team space

You can view the list of analysers and description about the analyzers available in a Team space by clicking **Infrastructure > Analyzers** from the navigation in the team space. Also, you can enable or disable the analyzer, view the count of the reviewed insights and the review history of the insights generated by the analyzers.

After viewing the recommendation for a specific parameter, members of the project can alter the default threshold value that specifies the level of confidence required in the analysis of the parameter and run the tests again to improve the accuracy of the recommendations and insights.

You can also improve the accuracy of the recommendations by running multiple tests with differing loads repeatedly.

Using Machine learning capabilities

You can find information in the following documentation about using Machine learning (ML) capabilities in Test Hub:

- To understand the requirements of Performance tests or Schedules that are analyzed by using ML capabilities in Test Hub, see Test run considerations for using Machine learning capabilities on page 534.
- To configure a run of a Performance Test, see Configuring a run of a Performance test on page 782.
- To configure a run of a Schedule, see Configuring a run of a Rate Schedule or VU Schedule on page 896.

- To view the ML insights into the Performance tests or Schedules in your project, which were analyzed after a run, see Viewing Machine learning insights on page 1031. You can also select your reaction to agree or disagree with the insights provided when you view the insight.
- To understand how to interpret the insights, see Interpretation of insights on page 1033.
- To view the ML analysis of the supported parameters when Performance tests or Schedules that are in different projects in your team space are run, see Viewing Machine learning analyzers on page 1037.
- To understand how to interpret the recommendations of the ML analysis of the supported parameters, see Interpretation of parameter analysis on page 1039.
- To view the reactions of project members to the ML insights, see Viewing reactions marked by members to Machine learning insights on page 1042.
- To modify the settings of the ML analyzer for a specific parameter, see Modifying Machine learning analyzer settings on page 1044.
- To delete the ML insights provided about the Performance tests or Schedules that were analyzed, see Deleting Machine learning insights on page 1046.
- To rerun the ML analysis without rerunning the test asset, see Rerunning Machine learning (ML) analysis on page 1047.

Viewing Machine learning insights

When you want to use Machine learning (ML) capabilities in HCL DevOps Test Hub (Test Hub), you need to run Performance Tests or Schedules that conform to certain criteria. After the test run is completed, you can view the ML insights on the **Insights** page in your project.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Ran the Performance Test or Schedule that matches the minimum requirements for ML analysis in Test Hub. See Configuring a run of a Compound Test that contains performance tests on page 743 or Configuring a run of a Rate Schedule or VU Schedule on page 896.

About this task

If the Performance Tests or Schedules match the requirements and criteria for ML analysis, Test Hub analyzes the test results for the relevant parameter and presents its findings as insights into the parameter.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Perform any of the following actions to view the insights:

- Scroll down the **Overview** page to see the **Insights** section.

  Go to Step 4 on page 1032 for the actions that you can perform to view the insights.

- Click **Analyze > Insights** in the navigation pane.

  The **Insights** page is displayed.

  Go to Step 5 on page 1032 for the actions that you can perform to view the insights.

4. Perform the following steps to view a specific insight from the **Overview** page:

   The **Insights** section on the **Overview** page displays a table of the insights for the supported parameters identified from the tests that were run in the project. The table lists the level of confidence with which the analysis of the specific parameter was done, the parameter analyzed, the name of the test asset, and the timestamp of the completed analysis.

   a. Click any item in the row of a specific insight to view the details of that insight.

      The **Insights** page is displayed with the details of the specific insight.

      Note: If you want to view the list of the other insights that are available in your project, you can click **Back to all insights**, and the list of available insights in the project is displayed in a tabular view.

   b. Go to Step 6 on page 1032.

5. Perform the following steps to view a specific insight from the **Insights** page:

   a. Identify the test asset or the parameter by performing any of the following actions:

      - Search for the test asset or parameter by entering any text contained in its name in the **Search** field.
      - Sort the items in the table of insights by clicking any of the column headers, and then identify the insight.

        For example, you can sort the items based on the name of the test asset in an ascending order or descending order, and then identify the test that you ran.

      You identified the insight.

   b. Click the **Expand** icon > of the insight.

      The details of the specific insight are displayed.

   c. Go to Step 6 on page 1032.

6. Perform the following steps when the details of the insight are displayed.

a. Read the details about the analysis and the issues that are identified by the analyzer.

b. Correlate the graph analyzed by the analyzer with your observations on the parameter.

c. Select your reaction to either agree or disagree with the insight by performing the following actions:

- Click the **Agree** icon, if you agree with the insights provided.

  After you agree, the icon changes to and displays the text `Agreed by username` when you hover over the icon.

- Click the **Disagree** icon, if you disagree with the insights provided.

  After you disagree, the icon changes to and displays the text `Disagreed by username` when you hover over the icon.

Note: A member of a project can only react once by either agreeing or disagreeing with the insight.

Results

You viewed the details of the insights into the parameters that are analyzed by Test Hub.

What to do next

You can perform any of the following tasks:

- Understand how to interpret the insights. See Interpretation of insights on page 1033.
- Delete an insight. See Deleting Machine learning insights on page 1046.
- Rerun the analysis so that the deleted instance of the insight is redisplayed. See Rerunning Machine learning (ML) analysis on page 1047.

Related information

Interpretation of insights on page 1033

Interpretation of insights

You can find information about interpreting the Machine learning (ML) insights into the supported parameters that are analyzed by HCL DevOps Test Hub (Test Hub) after certain tests are run.

Parameters analyzed through ML

The following parameters in Performance tests or Schedules are supported for analysis by using ML capabilities in Test Hub:

- The **Response Time Lock-Step Pattern** parameter.
- The **Response Time Standard Deviation Pattern** parameter.
- The **Throughput Drop Pattern** parameter.
- The **Performance Agent Health** parameter.

Test Hub analyzes the parameters based on certain criteria as detailed in Criteria for analysis of the parameters on page 1028 and then presents its findings as insights about the parameters.

Details of the insights

The insights about the parameters analyzed are displayed in the **Insights** section on the project **Overview** page or on the **Insights** page in a project.

The insights analyzed for the parameters from various tests that were run in the project are listed in a table. The table lists the following information:

- Name of the test asset in the **Test** column.
- The parameter analyzed in the **Insight** column.
- The level of confidence with which the analysis of the parameter was done in the **Confidence** column. The level of confidence displayed for the analysis can be as *Lowest*, *Low*, *Medium*, *High*, or *Highest*. You can view the confidence level as a number when you hover the cursor over the text in the **Confidence** column.
- The timestamp of the completed analysis in the **Identified on** column.
- The labels that you added for the specific test run in the **Labels** column.
- The components in the System Model that are associated with the test asset in the **Components** column.

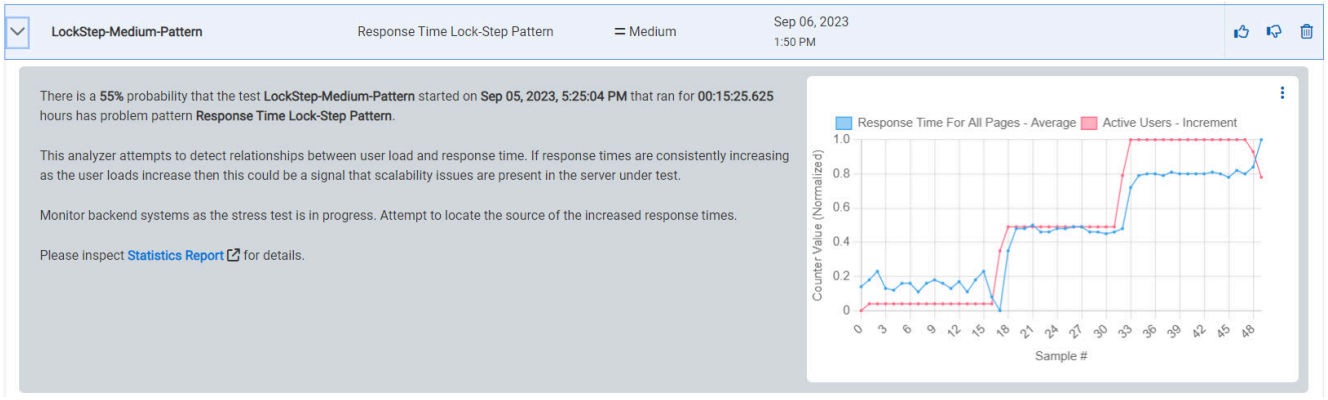
You can click the **Expand** icon > to view the details of the insight.

You can find information as an example about the following insights that help you to interpret the insight:

- Insights on the Response Time Lock-Step Pattern parameter on page 1034
- Insights on the Response Time Standard Deviation Pattern parameter on page 1035
- Insights on the Throughput Drop Pattern parameter on page 1036
- Insights on the Performance Agent Health parameter on page 1036

Insights on the Response Time Lock-Step Pattern parameter

When you expand the insight about the **Response Time Lock-Step Pattern** parameter, you can view the details displayed similar to the following sample:

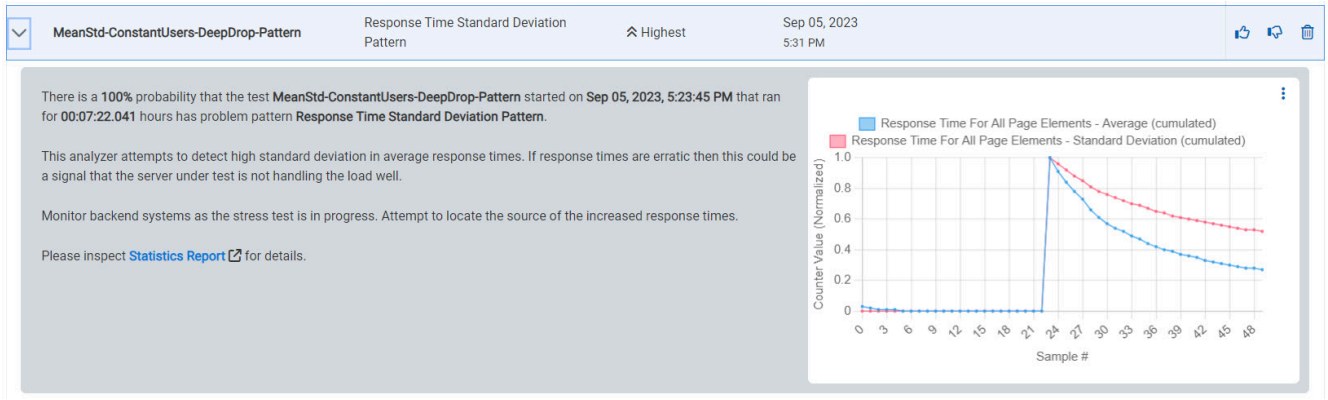


The following information is displayed as insights into the test on analyzing the **Response Time Lock-Step Pattern** parameter:

- The probability of finding a problem in the test.
- The description of how the ML analyzer has attempted to analyze the parameter in the test.
- The methods to control the problems are provided as suggestions.
- The graph of the observed data points.
- The link to the Statistics Report generated for the test.

Insights on the Response Time Standard Deviation Pattern parameter

When you expand the insight about the **Response Time Standard Deviation Pattern** parameter, you can view the details displayed similar to the following sample:

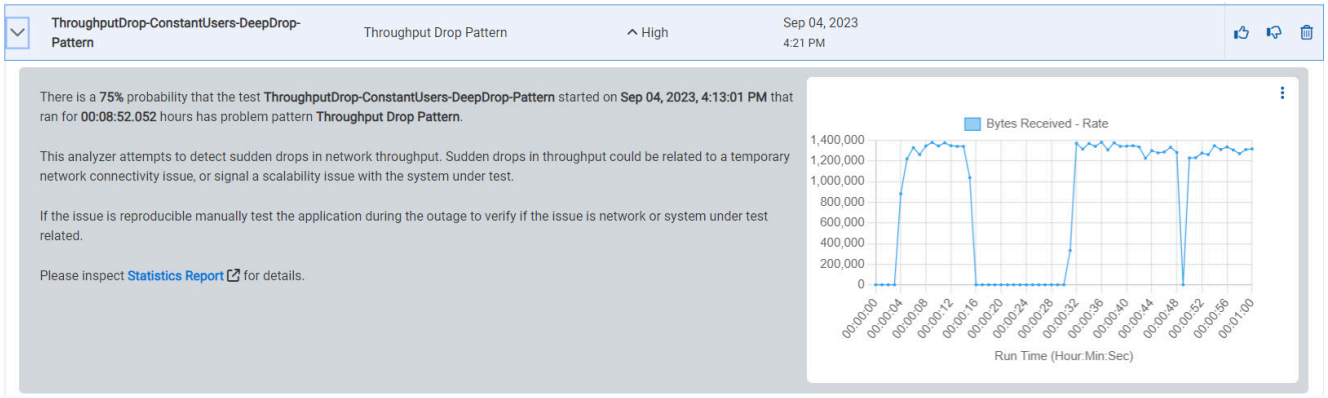


The following information is displayed as insights into the test on analyzing the **Response Time Standard Deviation Pattern** parameter:

- The probability of finding a problem in the test.
- The description of how the ML analyzer has attempted to analyze the parameter in the test.
- The methods to control the problems are provided as suggestions.
- The graph of the observed data points.
- The link to the Statistics Report generated for the test.

Insights on the Throughput Drop Pattern parameter

When you expand the insight about the **Throughput Drop Pattern** parameter, you can view the details displayed similar to the following sample:

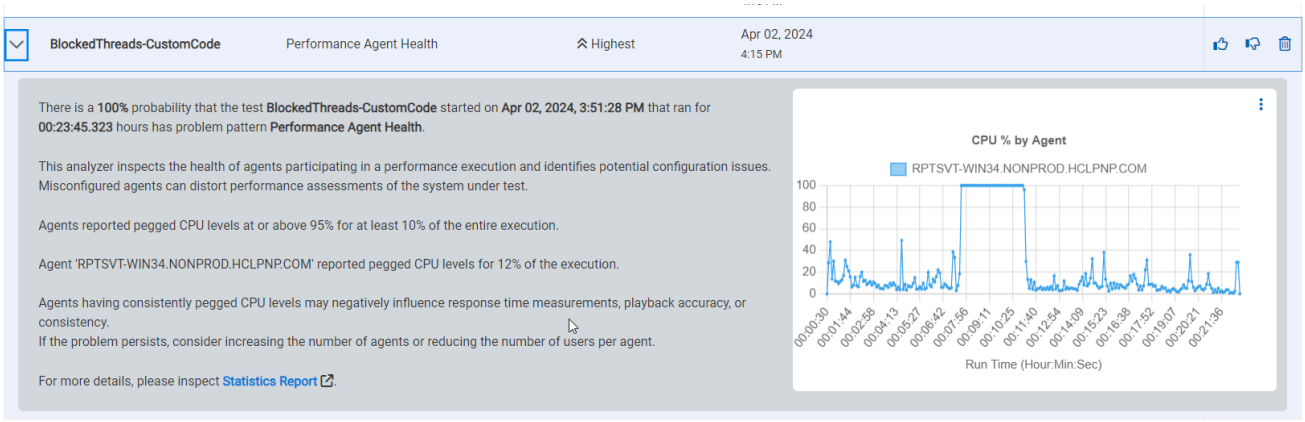


The following information is displayed as insights into the test on analyzing the **Throughput Drop Pattern** parameter:

- The probability of finding a problem in the test.
- The description of how the ML analyzer has attempted to analyze the parameter in the test.
- The methods to control the problems are provided as suggestions.
- The graph of the observed data points.
- The link to the Statistics Report generated for the test.

Insights on the Performance Agent Health parameter

When you expand the insight about the **Performance Agent Health** parameter, you can view the details displayed similar to the following sample:



The following information is displayed as insights into the test on analyzing the **Performance Agent Health** parameter:

- The probability of finding a problem in the test.
- The description of how the ML analyzer has attempted to analyze the parameter in the test.
- The methods to control the problems are provided as suggestions.

- The graph of the observed data points.
- The link to the Statistics Report generated for the test.

Related information

Viewing Machine learning insights on page 1031

Viewing Machine learning analyzers

HCL DevOps Test Hub (Test Hub) analyzes certain types of tests that are run in a team space by using its Machine learning (ML) capabilities. You can view the list of analysers and description about the analyzers available in a Team space.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Ran the Performance Test or Schedule that matches the minimum requirements for ML analysis in Test Hub. See Configuring a run of a Compound Test that contains performance tests on page 743 or Configuring a run of a Rate Schedule or VU Schedule on page 896.

About this task

If the Performance Tests or Schedules that are run from projects in a team space match the requirements and criteria for ML analysis, Test Hub analyzes the tests during their runs.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Analyzers** in the navigation pane.

   The **Analyzers** page is displayed.

   You can view the parameters that are supported for ML analysis listed in a tabular format. The information displayed is as follows:

Column header	Description
Status	Displays the status of the parameter analysis as active or inactive in the team space. The default value of the analysis is set as active or enabled.
Name	Displays the name of the parameters that are supported for ML analysis, which is as follows: ◦ **Response Time Lock-Step Pattern** ◦ **Response Time Standard Deviation Pattern** ◦ **Throughput Drop Pattern** ◦ **Performance Agent Health**

Column header	Description
Version	Displays the version of the ML analyzer.
Last Run	Displays the date when the tests were run for the analysis.
Unreviewed insights	Displays the number of insights into the specific parameter analyzed from different tests that are not reviewed by the project members.
Actions	Displays the following icons for the actions that you can perform on each parameter analyzed: **Icon** / **Description**  Displays the **Edit Analyzer** dialog when clicked. You can modify the analyzer settings for subsequent analysis.  Displays a dialog that contains the list of the reactions marked by different members on insights into the specific parameter when clicked. The agreement or disagreement reactions by members with the date and time of marking them are listed in chronological order in the dialog. You can view the insight on the **Insights** page in the project when you click  .

3. Click the **Expand** icon > of the parameter to view the details.

   The details of the parameter are displayed.

4. Read the details about the analysis and recommendations for troubleshooting or resolving the problems detected in the analysis.

   See the information in the related links to understand how to interpret the analysis of the parameters.

Results

You viewed the details of the parameters that are analyzed by Test Hub in tests that were run in the team space.

What to do next

You can perform any of the following tasks:

- Understand how to interpret the recommendations from the analysis of the parameters. See Interpretation of parameter analysis on page 1039.
- View reactions by members to an insight. See Viewing reactions marked by members to Machine learning insights on page 1042.
- Modify the ML analyzer settings. See Modifying Machine learning analyzer settings on page 1044.

Related information

Interpretation of parameter analysis on page 1039

Interpretation of parameter analysis

You can find information about interpreting the details of each of the Machine Learning (ML) analyzers that analyze the parameters after certain tests are run in HCL DevOps Test Hub (Test Hub).

Parameters analyzed through ML

The following parameters in Performance tests or Schedules are supported for analysis by using ML capabilities in Test Hub:

- The **Response Time Lock-Step Pattern** parameter.
- The **Response Time Standard Deviation Pattern** parameter.
- The **Throughput Drop Pattern** parameter.
- The **Performance Agent Health** parameter.

Test Hub analyzes the parameters based on certain criteria as detailed in Criteria for analysis of the parameters on page 1028. The ML analyzers analyze the results of tests run in all projects in a team space for each of the parameters. The analyzers present their findings on each of the parameters as an insight on the **Insights** page in the project.

Details of the parameters analyzed

The details about the parameters analyzed are displayed on the **Analyzers** page in the team space.

The parameters are listed in a table with the following information:

Column header	Description
Status	Displays the status of the parameter analysis as active or inactive in the team space. The default value of the analysis is set as active or enabled.
Name	Displays the name of the parameters that are supported for ML analysis, which is as follows: • **Response Time Lock-Step Pattern** • **Response Time Standard Deviation Pattern** • **Throughput Drop Pattern** • **Performance Agent Health**
Version	Displays the version of the ML analyzer.
Last Run	Displays the date when the tests were run for the analysis.
Unreviewed insights	Displays the number of insights into the specific parameter analyzed from different tests that are not reviewed by the project members.

<table>
<tr><th>Column header</th><th>Description</th></tr>
<tr><td>Actions</td><td>Displays the following icons for the actions that you can perform on each parameter analyzed:
<table>
<tr><th>Icon</th><th>Description</th></tr>
<tr><td>  </td><td>Displays the Edit Analyzer dialog when clicked. You can modify the analyzer settings for subsequent analysis.</td></tr>
<tr><td>  </td><td>Displays a dialog that contains the list of the reactions marked by different members on insights into the specific parameter when clicked. The agreement or disagreement reactions by members with the date and time of marking them are listed in chronological order in the dialog. You can view the insight on the Insights page in the project when you click  .</td></tr>
</table>
</td></tr>
</table>

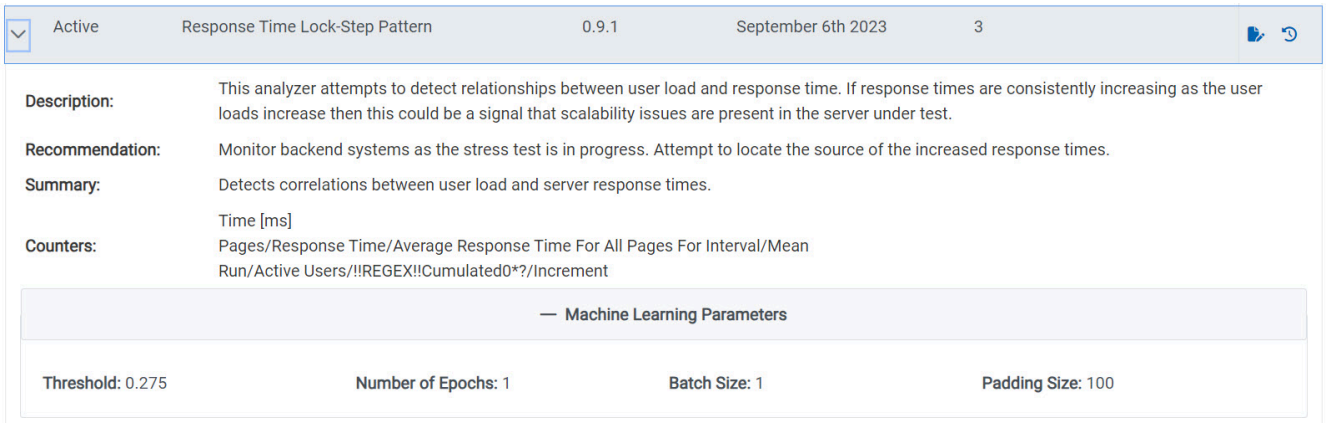
You can click the **Expand** icon to view the details of the parameter analyzed.

You can find information as an example about the following parameters that help you to interpret the parameter analysis:

- The Response Time Lock-Step Pattern parameter analyzer on page 1040
- The Response Time Standard Deviation Pattern parameter analyzer on page 1041
- The Throughput Drop Pattern parameter analyzer on page 1041

The Response Time Lock-Step Pattern parameter analyzer

When you expand the **Response Time Lock-Step Pattern** parameter, you can view the details displayed similar to the following sample:



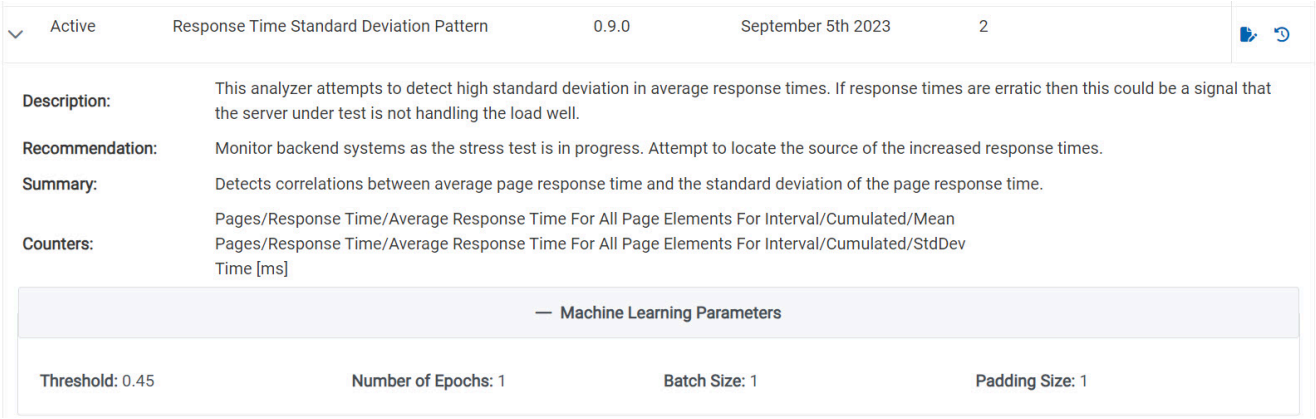
The following information is displayed for the **Response Time Lock-Step Pattern** parameter:

- The description of how the ML analyzer has attempted to analyze the parameter in the test.
- The methods to control the problems are provided as a recommendation.
- The summary of the analysis.

- The counters used in the analysis.
- The parameters used in the analysis.

The Response Time Standard Deviation Pattern parameter analyzer

When you expand the **Response Time Standard Deviation Pattern** parameter, you can view the details displayed similar to the following sample:

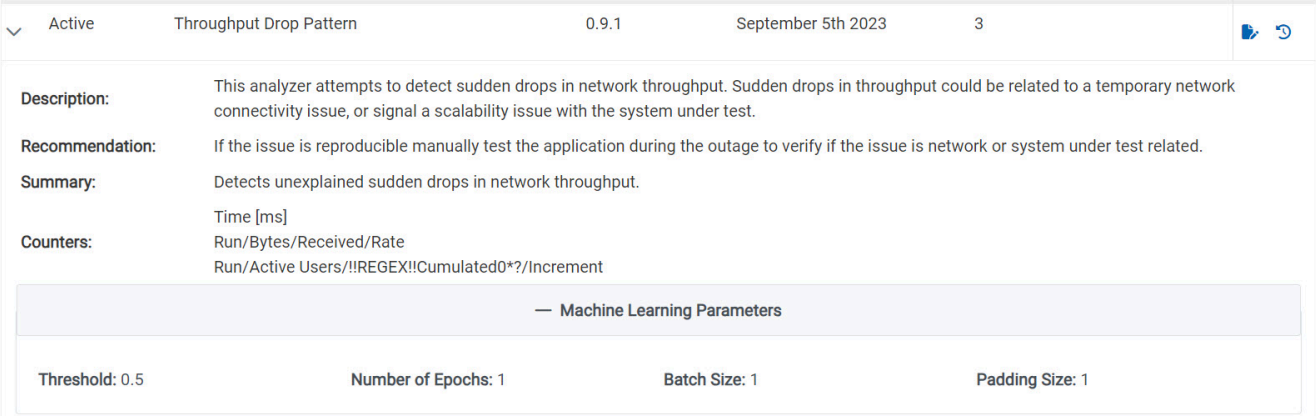


The following information is displayed for the **Response Time Standard Deviation Pattern** parameter:

- The description of how the ML analyzer has attempted to analyze the parameter in the test.
- The methods to control the problems are provided as a recommendation.
- The summary of the analysis.
- The counters used in the analysis.
- The parameters used in the analysis.

The Throughput Drop Pattern parameter analyzer

When you expand the **Throughput Drop Pattern** parameter, you can view the details displayed similar to the following sample:

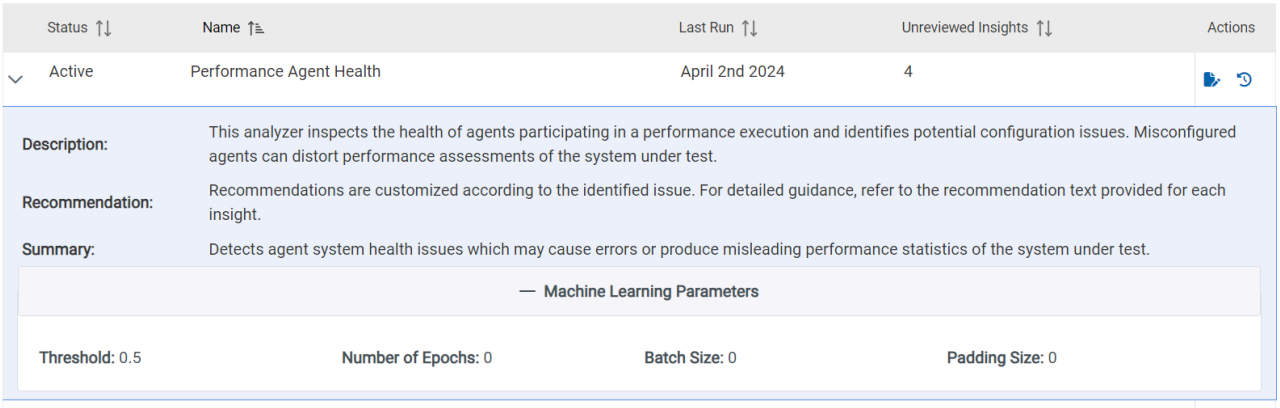


The following information is displayed for the **Throughput Drop Pattern** parameter:

- The description of how the ML analyzer has attempted to analyze the parameter in the test.
- The methods to control the problems are provided as a recommendation.
- The summary of the analysis.
- The counters used in the analysis.
- The parameters used in the analysis.

The Performance Agent Health parameter analyzer

When you expand the **Performance Agent Health** parameter, you can view the details displayed similar to the following sample:



The following information is displayed for the **Performance Agent Health** parameter:

- The description of how the ML analyzer has attempted to analyze the parameter in the test.
- The methods to control the problems are provided as a recommendation.
- The summary of the analysis.
- The parameters used in the analysis.

Related information

Viewing Machine learning analyzers on page 1037

Viewing reactions marked by members to Machine learning insights

You can view the reactions that members of a project mark to show their agreement or disagreement with the insights into the parameters on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Ran the Performance Test or Schedule that matches the minimum requirements for ML analysis in Test Hub. See Configuring a run of a Compound Test that contains performance tests on page 743 or Configuring a run of a Rate Schedule or VU Schedule on page 896.
- Viewed the ML insights and marked your reaction to an insight. See Viewing Machine learning insights on page 1031.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Analyze > Insights** in the navigation pane.

   The **Insights** page is displayed.
4. Identify the test asset or the parameter by performing any of the following actions:

   - Search for the test asset or parameter by entering any text contained in its name in the **Search** field.
   - Sort the items in the table of insights by clicking any of the column headers, and then identifying the insight.

     For example, you can sort the items based on the name of the test asset in an ascending order or descending order, and then identify the test that you ran.

   You identified the insight.
5. Use the information in the following table to interpret the reactions marked for the insight:

If the icon is displayed as	Interpretation of the reaction
	None of the members agreed with the insight.
	At least one member agreed with the insight.  **Note:** The name of the member who reacted is displayed as `Agreed by` *`username`* when you hover over the icon.
	None of the members disagreed with the insight.
	At least one member disagreed with the insight.  **Note:** The name of the member who reacted is displayed as `Disagreed by` *`username`* when you hover over the icon.

Results

You viewed the member reactions to an insight from the **Insights** page.

What to do next

You can perform any of the following tasks:

- Modify the threshold value of a specific parameter based on the number of disagreements marked by members so that you can improve the accuracy of the insight. See Modifying Machine learning analyzer settings on page 1044.
- Repeat the test runs with the updated thresholds to improve the Machine Learning insights for the specific parameter. See Test run configurations on page 610.

Modifying Machine learning analyzer settings

When you want to change the decision threshold set for a Machine learning (ML) analyzer or disable the analyzer, you can modify the settings from the **Analyzer** page.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Ran the Performance Test or Schedule that matches the minimum requirements for ML analysis in Test Hub. See Configuring a run of a Compound Test that contains performance tests on page 743 or Configuring a run of a Rate Schedule or VU Schedule on page 896.
- Viewed the insights into tests provided by ML analysis. See Viewing Machine learning insights on page 1031.
- Viewed the reactions from members of your project to the ML insight. See Viewing reactions marked by members to Machine learning insights on page 1042.
- Viewed the history of member reactions for a specific parameter analyzer from the **Analyzers** page. See Viewing Machine learning analyzers on page 1037.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Infrastructure > Analyzers** in the navigation pane.

   The **Analyzers** page is displayed. You can view the parameters that are supported for ML analysis listed in a tabular format.
3. Identify the parameter whose analyzer you want to modify, and then click the **Edit Analyzer** icon  in the **Actions** column in the row of the parameter.

   The **Edit Analyzer** dialog is displayed.
4. Perform any of the following actions to modify the default settings for the ML analysis of the specific parameter.

If you want to...	Action
Disable the analysis for the specific parameter.	Perform the following steps: a. Click **Is Enabled** to switch the analysis from being enabled or active to become disabled or inactive. **Note:** If you modify the default setting to be disabled, Test Hub does not perform ML analysis of the specific parameter when tests are run subsequently. b. Click **Submit**.
Enable the analysis for the specific parameter, if it is disabled.	Perform the following steps: a. Click **Is Enabled** to switch the analysis from being disabled or inactive to become enabled or active. b. Click **Submit**.
Increase or decrease the threshold level.	Perform the following steps: a. Edit the value in the **Decision Threshold** field and enter a value as required. The threshold value determines the level after which the ML analysis identifies an insight into the specific parameter when the test runs. **Note:** You can retain the default values for the decision threshold initially until a large number of tests are run that contribute to the insights for a specific parameter. You can change the value to a higher threshold when there are a large number of records learned by the ML analyzers. b. Click **Submit**.

Results

You either disabled or enabled the analyzer or changed the decision threshold set for a ML analyzer.

What to do next

You can perform any of the following tasks:

- Rerun the analysis of the test with the changed threshold values without running the tests again. See Rerunning Machine learning (ML) analysis on page 1047.
- Run tests if you changed the decision threshold values so that the ML analysis uses the changed values. See Test run configurations on page 610.

- View insights into the specific parameter whose decision threshold was modified after the test runs are completed. See Viewing Machine learning insights on page 1031.
- View the ML recommendations for the specific parameter whose decision threshold was modified. See Viewing Machine learning analyzers on page 1037.

Deleting Machine learning insights

After you view the Machine learning (ML) insights into the tests, you might want to delete the insight if many members of your project disagree with the insight.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.
- Viewed the insights into tests provided by ML analysis. See Viewing Machine learning insights on page 1031.
- Viewed the reactions from members of your project to the ML insight. See Viewing reactions marked by members to Machine learning insights on page 1042.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Perform any of the following actions to view and identify the insight:
   - Scroll down the **Overview** page to see the **Insights** section.

     The **Insights** section on the **Overview** page displays a table of the insights for the supported parameters identified from the tests that were run in the project. You can identify the insight and click any item in the row of the identified insight to display the insight on the **Insights** page.
   - Click **Analyze > Insights** in the navigation pane. The **Insights** page is displayed. You can identify the insight that you want to delete.
4. Click the **Delete** icon from the **Actions** column that is in the row of the identified insight.

   The **Delete Insight** dialog is displayed.
5. Click **Yes**.

   The insight is deleted and a confirmation message is displayed.

Results

You deleted an insight into the test arrived at by ML analysis of the test.

What to do next

You can perform any of the following tasks:

- Run tests to let the ML analyzers reanalyze and present the insights. See Test run configurations on page 610.
- Rerun the analysis without running the tests again so that the deleted instance of the insight is redisplayed. See Rerunning Machine learning (ML) analysis on page 1047.

Rerunning Machine learning (ML) analysis

If you change the decision threshold value set for a parameter analyzer, you can then rerun the analyzer to analyze the parameter for the changed threshold and present the insight without running the tests again.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Analyze > Results** from the navigation pane in your project.

   The **Results** page is displayed.
4. Click the **SUITES** tab, if you ran a Schedule or the **TESTS** tab, if you ran a Performance Test.
5. Identify the test result for which you want to rerun the analysis by performing any of the following actions:
   - Search for the test result by entering any text contained in the test asset name in the **Search** field.
   - Sort the items in the table of results by clicking the **Started by** column header, and then identifying the test runs that you started.
6. Click the **Open action menu** icon from the **Actions** column that is in the row of the identified test result.
7. Click **Rerun analysis** .

   The parameter is analyzed again with the changed threshold vale and the insight presented.

Results

You started the ML analysis of the previously analyzed test result so that the insight can be displayed based on the changed threshold value without running the test again.

Resource monitoring

When you apply load to a system under test, the amount of resources consumed by your system is increasing. If the capacity of the resources does not match the load, you can see performance issues in the results. Resource monitoring in HCL DevOps Test Hub (Test Hub) helps you monitoring the resources of a system and establish the performance metrics and usage data of the system. Thus, you can observe the health of these resources while a schedule is running.

You can use resource monitoring while running a schedule to capture data, such as processor or memory usage, or to monitor the availability of hosts and services by using counters. Test Hub provides a comprehensive view of a system under

test to help determine problems. Hosts and services can be servers, virtual machines, or any local host or network services. You can also monitor remote hosts and services with agents.

Monitoring a host or a server

You can use resource monitoring to monitor any host and network service. This method is used for host targets that the service can reach directly. With Test Hub, you can also monitor performance metrics that are collected and stored by a monitoring system that is monitoring a host. You can use a Prometheus server or a data collector through an exporter, OpenMetrics exporter, for example.

To monitor resources on a local system, you must add the source of the resource monitoring data from the resource monitoring page. Then you have to enter some connection settings and select the counters to monitor the sources. The metrics and usage data are exposed in a graph for each selected counter.

Monitoring a remote host with monitoring agents

You can monitor remote hosts and services with agents through wider sets of data collectors, computers, and networks. The purpose is to capture CPU load, disk space, memory, and the running process for example.

Agent-based monitoring is useful when remote services are not directly accessible from Test Hub through the network. Agents are closer to the target that you want to monitor. You can set up the agent on an authorized host, for example, when access to an Apache httpd or NGINX server status page or a JVM JMX port, a Prometheus server, or an OpenMetrics exporter is restricted to one or few client hosts only. The configuration task is simplified, and no security changes are required.

Agent-based monitoring is mandatory to collect data from a Windows or Linux Performance host.

To monitor a host across a monitoring agent, you must first set up a resource monitoring agent on the target host for which you want to collect the performance statistics and usage data. The agent establishes a connection with Test Hub. When the connection is set, the agent is showing up in the list of agents on the resource monitoring main page. Then, you can select the agent in the list of target sources in the dialog **Access target from** in the dialog that opens from **Add a Source**.

To learn more about configuring the resource monitoring sources, counters, and agents and collecting and viewing the data inTest Hub, go through the information in the following links.

Resource monitoring capabilities

With HCL DevOps Test Hub (Test Hub), you can monitor a host or a service that you add to a project as a source. You can select the statistics counters that are used to collect metrics and the resource usage data throughout the monitoring system. You can visualize metrics and data that are gathered during the monitoring process in Test Hub. You can add the resource monitoring sources and counters in a team space or in a project to collect metrics and resource usage data.

Requirements:

- You must be a *Team Space Owner* to create, modify and remove resource monitoring sources in a team space. Any team space *Member* (not owner) can only view resource monitoring sources.
- You must be a project *Owner* to create, modify and remove resource monitoring sources in a project. Any project *Member* (not the owner) can only view resource monitoring sources.

Supported sources

With Test Hub, you can monitor resources for the following sources:

- **Apache httpd server**: You can optionally use agents.
- **NGINX and NGINX Plus**: You can optionally use agents.
- **Java Virtual Machine**: You can optionally use agents.

  You can monitor JVM resources from a local or a remote system. Some parameters must be set in the command before running the Java Virtual Machine. For more details, see the link to the page about starting a Java Virtual Machine on this page.
- **Windows Performance host**:

  To monitor the performance of a Windows host, you must have installed an agent on the target Windows host

  or on a Windows host that is configured to access the performance data of the Windows host target.
- **Linux Performance host**:

  To monitor the performance of a Linux host, you must have installed an agent on the target Linux host.
- **Prometheus server**:

  You can monitor metrics of a host under test that are collected by a Prometheus server. Prometheus collects metrics from monitored targets by regularly requesting appropriate HTTP endpoints on these targets (called *scraping*).

  The metrics data are collected through data collectors sets (groups of monitoring counters) that are tracking the system performance. The performance data results are stored in the Prometheus database so that they can be consumed by external systems through a REST API.

  In Test Hub, you can query these performance metrics collected by Prometheus servers. 'PromQL' is the language for querying Prometheus metric data. Refer to Prometheus PromQL documentation for information about this query language.

  You can select default queries, or create additional ones.

  A Prometheus server is required. It must be configured to scrap a set of exporters, every 15 seconds by default. Pushgateway, service discovery, and Alertmanager are optional.

If Test Hub cannot directly reach the Prometheus server, you can consider setting up an agent on an authorized host or on the same host that is running Prometheus.

- **OpenMetrics exporter**:

  You can monitor the metrics of a host under test that are collected and exposed through an OpenMetrics format or a Prometheus exporter format.

  Test Hub can scrap metrics exposed by an OpenMetrics or Prometheus exporter through metric counters. No Prometheus server installation is required, you only need to set up one or multiple OpenMetrics, or Prometheus exporters that fit your software or host target.

  If Test Hub cannot directly reach the Prometheus server, you can consider setting up an agent on an authorized host or on the same host that is running Prometheus.

Monitoring an NGINX server

To monitor an NGINX server, you must add the NGINX server source to HCL DevOps Test Hub (Test Hub), enter connection settings and select the counters that are used to capture the monitoring usage data.

Before you begin

You must have logged in Test Hub and be the owner of an existing new project.

If you monitor a remote host, it must be connected with the computer that you use to access resource monitoring. See Resource monitoring agents on page 1065.

You can optionally use agents.

About this task

This task applies to NGINX and NGINX Plus.

If you use an agent to connect

1. Click **Infrastructure > Resource Monitoring**.
   The resource monitoring page opens.
2. In the resource monitoring page, click **Add a Source** and select **Add an NGINX server...**.
3. Fill in the following connection settings in the **New server** dialog:

   a. In **Target host**, enter the IP address or hostname and port number of the host where the server to monitor is installed.

   Note: For hosts that are already connected to Test Hub through an agent, you have only the **Access target from** field enabled. Move directly to step 4.

   b. In **Server edition**, select the appropriate server from the drop-down list.

c. In **Path to the status page of the server**, enter the name of the page to view the status of the server.

Note: If you select NGINX Plus (with API version 3) as a source, you must specify the name of the path to view the API root of the NGINX Plus server.

d. In **Security**, select the following options:

Secured with TLS/SSL

If the application server is secured with TLS/SSL.

Trust self-signed certificate

To accept the server certificate.

Do not verify hostname

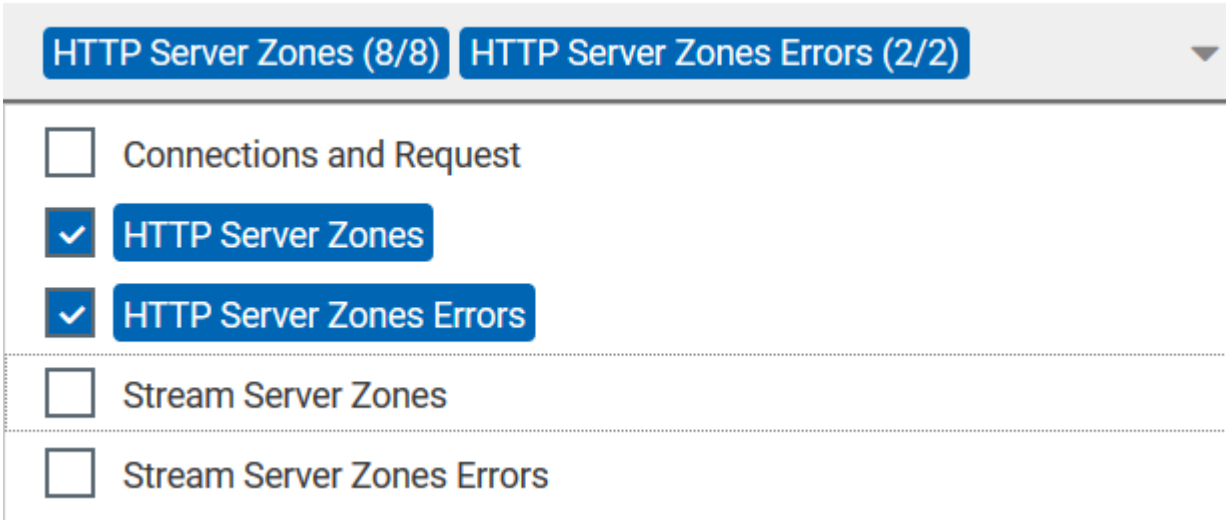
To ignore verification of the hostname in the certificate.

Require credentials

If the server requires log-in credentials, enter a **User name** and **Password**.

e. Click **Add**.
The **Choose Data Counters** dialog opens.

4. Select and save the statistics counters to monitor the source. You can select them from the list where they follow the server logical organization.
For a faster selection, select the counters from the built-in sets drop-down list where they are organized by theme and save your selection.



Note:

For Docker host, you can select different sets of counters.

- The first level of counters: Generic counters that are related to a Docker Image and all its running Docker Containers (in an exit, running or paused, created, restarted, removing, dead or transitive state).
- The second level of counters: Specific counters that are related to the existing Docker Containers.

Results

When the selected counters are saved, two tables are displayed in the resource monitoring main page. They contain the total number of sources you have added and the number of sources ordered by type.



What to do next

You can click the links in the tables to view the performance metrics of your monitored system.

Related information

Viewing resource monitoring usage data on page 1087

Monitoring an Apache httpd server

To monitor an Apache httpd server, you must add the Apache httpd server source to HCL DevOps Test Hub (Test Hub), enter connection settings and select the counters that are used to capture the monitoring usage data.

Before you begin

You must be logged in Test Hub and be the owner of an existing new project.

You can optionally use agents. See Resource monitoring agents on page 1065.

1. Click **Infrastructure > Resource Monitoring**.
   The resource monitoring page opens. See Resource monitoring agents on page 1065.

2. In the resource monitoring page, click **Add a Source** and select **Add an Apache httpd server...**.
3. Fill in the following connection settings in the **New server** dialog:

   Note: For hosts that are already connected to Test Hub through an agent, you have only the **Access target from** field enabled. Move directly to step 4.

   a. In **Target host**, enter the IP address or hostname and port number of the host where the server to monitor is installed.

   b. In **Server edition**, select the appropriate server from the drop-down list.

   c. In **Path to the status page of the server**, enter the name of the page to view the status of the server.

   d. In **Security**, select the following options:

   Secured with TLS/SSL

   If the application server is secured with TLS/SSL.

   Trust self-signed certificate

   To accept the server certificate.

   Do not verify hostname

   To ignore verification of the hostname in the certificate.

   Require credentials

   If the server requires log-in credentials, enter a **User name** and **Password**.

4. Click **Add**.
   The **Choose Data Counters** dialog opens.
5. Select and save the statistics counters to monitor the source. You can select them from the list where they follow the server logical organization.
   For a faster selection, select the counters from the built-in sets drop-down list where they are organized by theme and save your selection.

Results

When the selected counters are saved, two tables are displayed in the resource monitoring main page. They contain the total number of sources you have added and the number of sources ordered by type.

What to do next

You can click the links in the tables to view the performance metrics and resource monitoring usage data for the monitored server.

Related information

Resource monitoring on page 1047

Viewing resource monitoring usage data on page 1087

Monitoring metrics collected by a Prometheus server

You can monitor your system resources by using metrics data that are collected by a Prometheus server while executing a schedule.

Before you begin

You must have created a project in HCL DevOps Test Hub (Test Hub).

You must have installed and configured a Prometheus server.

Read the concept information about resource monitoring and monitoring Prometheus server source. See Resource monitoring on page 1047

About this task

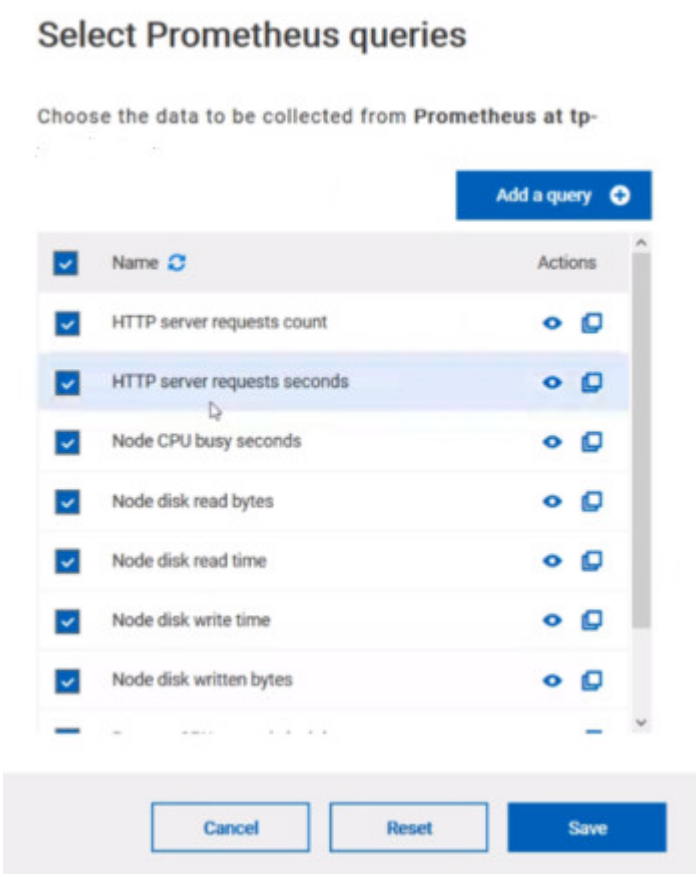
The following procedure describes how to select and create Prometheus queries in Test Hub. You must set a connection between Test Hub and the Prometheus server first to access to the Prometheus queries.

Enter the connection settings, then select queries.

1. Click **Add a source** and select **Add a Prometheus Server**.
2. Enter the connection settings in the **New server** dialog box,.
    a. In **Target host**, enter the IP address or host name and port number of the host where the Prometheus server is installed.
    b. Enter the **Path to the API root of Prometheus** to point to the Prometheus API where the monitoring data are saved.
    c. Enter the security parameters if required. For information about this section, see related links.
    d. Click **Add** to close the dialog.

    Result

    The Prometheus server source is created successfully in the background and a dialog opens to select the Prometheus data queries.
3. In the **Select Prometheus queries** dialog, select queries that are available in the list to collect Prometheus metrics data by clicking the checkboxes.
    You can reset the selection.



4. Follow these steps to create a query:

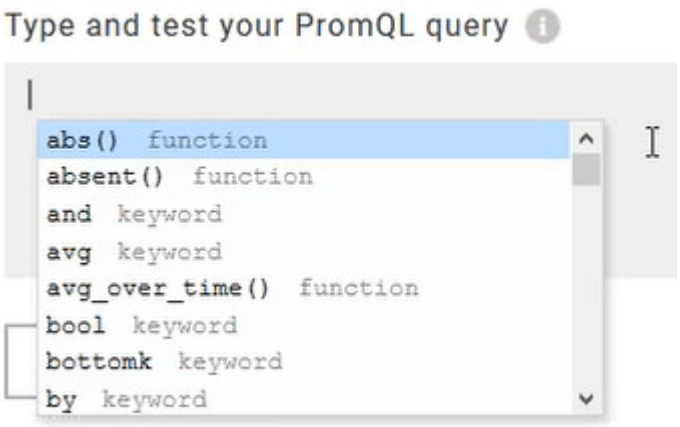
    a. Click **Add a query**.

    b. Enter a name for the new query in the **Create a Prometheus query** dialog.

    c. Enter a PromQL query.any keyword and press Ctrl+Space in the **Type and test your PromQL query.**

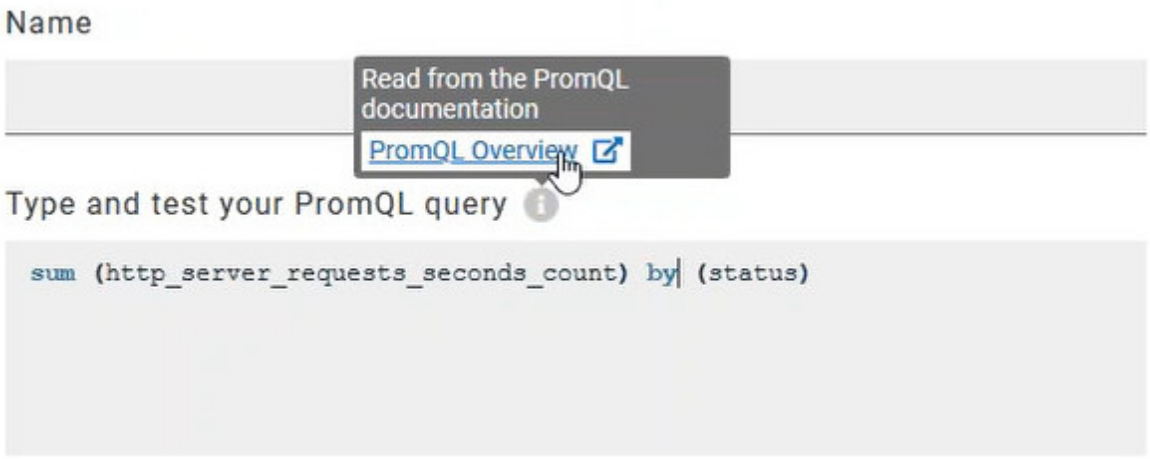
    Result

    All the possible combinations of keywords and metrics related to your searched keyword are listed.



d. Select a query in the list.

e. Click the link that points to the 'Querying Prometheus documentation' page in the information tool tip to find other examples of queries in a web page that you can use as samples.



f. Click **Test query.**
If the test is successful, a green check is displayed, otherwise you get a red check with an error message.

g. Click **Create**.

5. Proceed as follows to duplicate a query:

a. Click the **Duplicate** icon to duplicate a standard query.

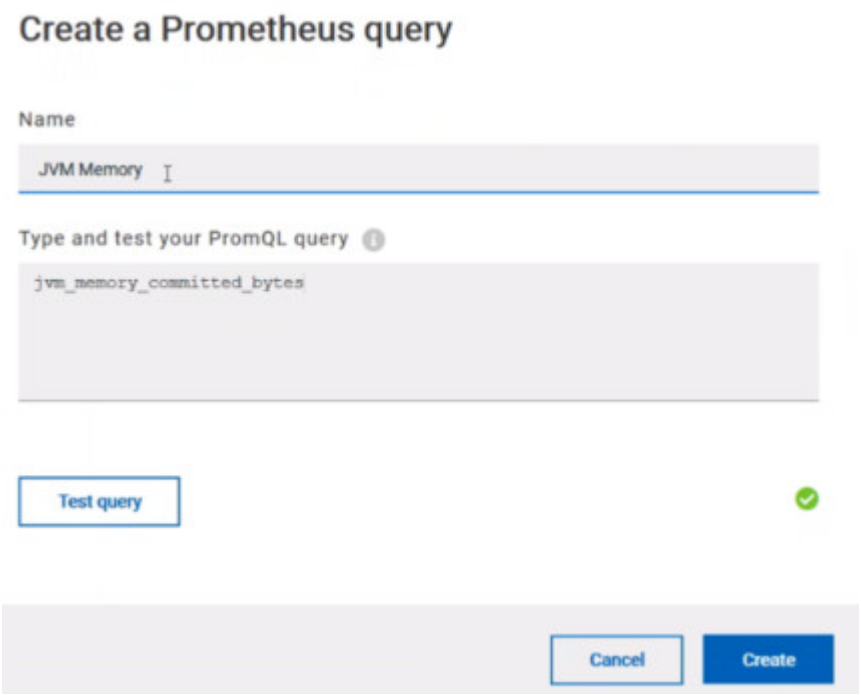
b. Rename the duplicated query in the **View the Prometheus query** dialog.
Make sure that the name you give to the query is not already used.

c. Modify the query type.

d. Test the query .

e. Click **Create**.

Note: You can click Create or Duplicate only if the test of the query is successful and the name of the query is not already in use.



6. Proceed as follows to modify a query:
   a. Click the **Modify** icon .
   b. Modify the name and type of the query.
   c. Test the query.
   d. Click **Update**.
   e. Delete duplicated or created queries.
   f. Click **Save** to close the **Select Prometheus queries** dialog.
7. Click Save and close the **Select Prometheus queries** dialog.

Results

The resource monitoring page displays the total number of sources you have added to your project and the number of sources ordered by type in cards.

What to do next

Click the links in the cards to go to the sources page and see the performance metrics for each source you added to your project. For more details, see Viewing resource monitoring usage data on page 1087

Related information

Resource monitoring on page 1047

Viewing resource monitoring usage data on page 1087

Monitoring metrics exposed by an OpenMetrics exporter

With HCL DevOps Test Hub (Test Hub), you can monitor your system resources by using metrics data that are exposed by an OpenMetrics or a Prometheus exporter through metric counters while executing a schedule.

Before you begin

You must have at least one exporter or software to expose monitoring metrics with an OpenMetrics or Prometheus format.

You must have created a project in Test Hub.

No Prometheus server installation is required, you only need to set up one or multiple OpenMetrics, or Prometheus exporters that fit your software or host target. Refer to the link at the end of this page to see the list of exporters.

About this task

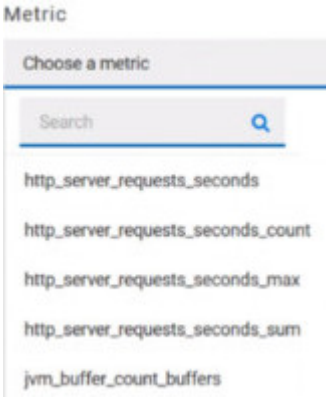
The following procedure describes how to select and create OpenMetrics counters in Test Hub. You must set a connection between Test Hub and the OpenMetrics exporter to select counters.

1. Click **Add a source** and select **Add an OpenMetrics exporter** .
2. In the **New OpenMetrics exporter** dialog that opens, enter the OpenMetrics exporter details to access its exposition endpoint as follows:
    a. In **Target host**, enter the IP address or host name and port number of the host where the server is installed.
    b. Enter the **Path to the exposition endpoint** to point to the application where the data performance results must be stored. As the path depends on the type of exporter used, refer to the exporter's documentation and configuration to indicate the right path.
    c. In **Security**, enter the security parameters if required.
    d. Click **Add** to close the dialog.
    Result
    The source is created successfully in the background. A dialog box opens to select the OpenMetrics counters.
3. Click the checkboxes to select counters in the **Select OpenMetrics counters** dialog.
4. Follow these steps to create a counter:

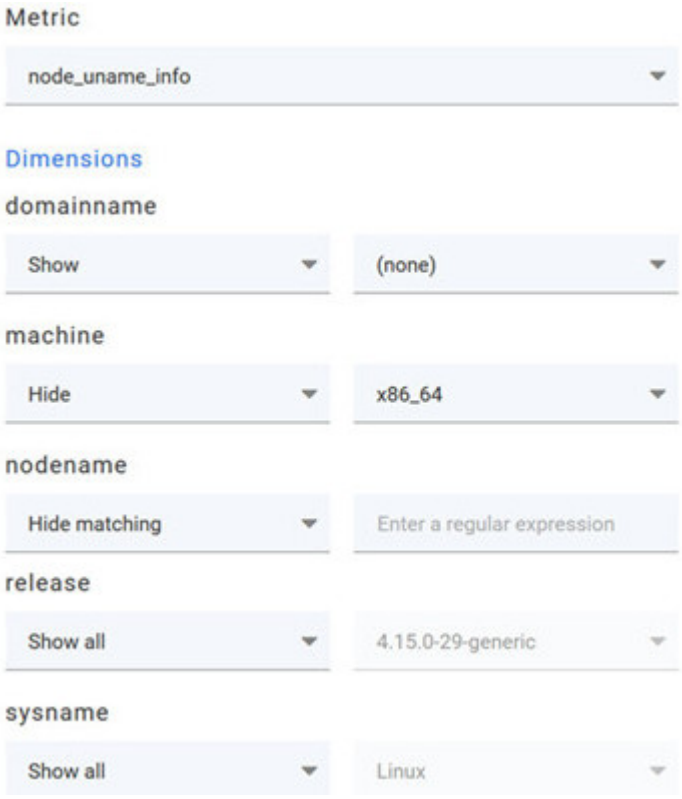
a. Click **Add a counter**.

b. Enter a name for the new counter, in the **OpenMetrics counter**.

c. Select metrics in the drop-down list or enter a keyword in the **Dimensions** field to enable the dynamic input help. You can also use the search field that appears in the list of metrics.

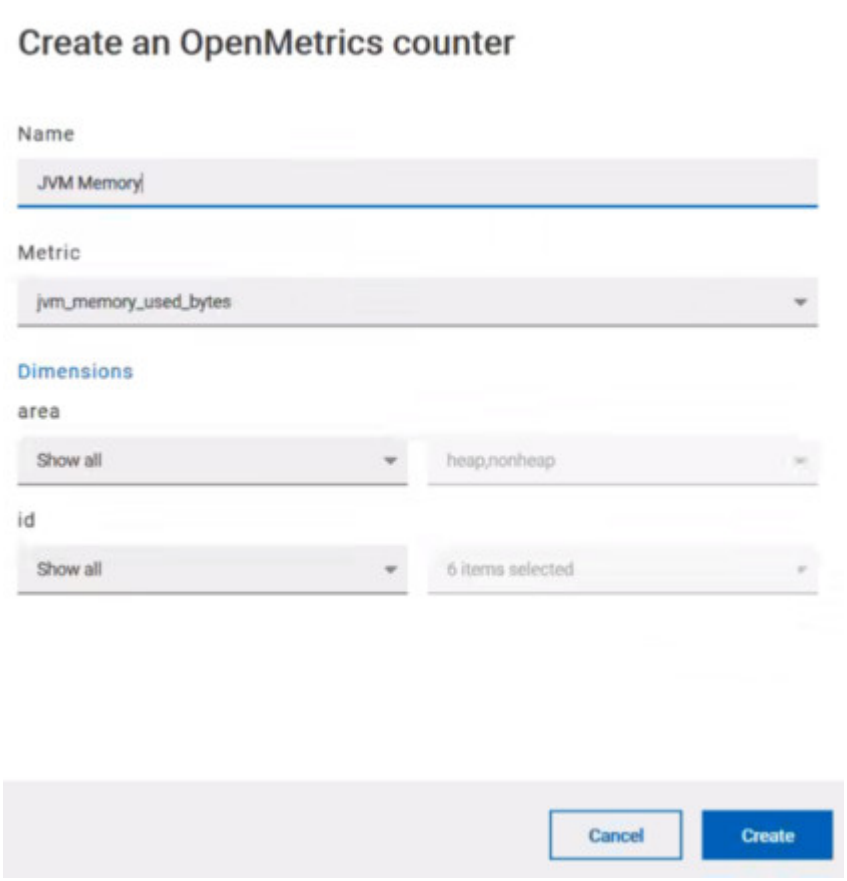


d. Select **Dimensions** from the dynamic fields when they are required.



e. Select the appropriate OpenMetrics counter that you are looking for.

f. Click **Create**.



5. Follow these steps to duplicate a counter:
    a. Select a counter in the list, and click the **Duplicate** icon .
    b. Modify the name of the counter and the Dimensions.
    c. Click **Create**.
6. Proceed as follows to modify a new counter:
    a. Click the **Modify** icon.
    b. Modify the name, Metrics, and Dimensions of the counter in the **Modify the OpenMetrics counter**.
    c. Click **Update**.

Note: You can create, duplicate, or update a counter only if the name of the counter is not already used.

7. Click **Save** and close the **Select OpenMetrics counters** dialog.

Note: You can delete counters, but only the ones that you created or duplicated.

Results

The resource monitoring page displays the total number of sources you have added to your project and the number of sources ordered by type in cards.

What to do next

Click the links in the cards to go to the sources page and see the performance metrics for each source you added to your project. For more details, see Viewing resource monitoring usage data on page 1087.

Related information

Exporters and Integrations

Resource monitoring on page 1047

Monitoring a Java Virtual Machine

To monitor a Java Virtual Machine, you must add the Java Virtual Machine source to the project in HCL DevOps Test Hub (Test Hub), and then select the counters that are used to capture the monitoring metrics.

Before you begin

You must have completed the following tasks:

- Configured and started the Java Virtual Machine from a command to connect it Test Hub.

  See Starting a Java Virtual Machine on page 1077.
- Logged in Test Hub. You must be the owner of an existing new project.
- You can optionally use agents. See Resource monitoring agents on page 1065.

About this task

The Java Virtual Machine source is added to the list of resource monitoring sources when the agent is installed.

1. Click **Infrastructure > Resource Monitoring**.
   The resource monitoring page opens.
2. In the resource monitoring page, click **Add a Source** and select **add a Java Virtual Machine**.
3. Fill in the following connection settings in the **New server** dialog:

   Note: For hosts that are already connected to Test Hub through an agent, you have only the **Access target from** field enabled. Move directly to step 4.

   a. In **Target host**, enter the IP address or hostname and port number of the host where the server to monitor is installed.

   b. In **Server edition**, select the appropriate server from the drop-down list.

   c. In **Path to the status page of the server**, enter the name of the page to view the status of the server.

   d. In **Security**, select the following options:

Secured with TLS/SSL

If the application server is secured with TLS/SSL.

Trust self-signed certificate

To accept the server certificate.

Do not verify hostname

To ignore verification of the hostname in the certificate.

Require credentials

If the server requires log-in credentials, enter a **User name** and **Password**.

4. Click **Add**.
5. Select and save the statistics counters to monitor the source. You can select them from the list where they follow the server logical organization.
   For a faster selection, select the counters from the built-in sets drop-down list where they are organized by theme and save your selection.
   Result

   When the selected counters are saved, two tables are displayed in the resource monitoring main page. They contain the total number of sources you have added and the number of sources ordered by type.



What to do next

You can click the links in the tables to view the performance metrics and usage data in a graph for the monitored Java Virtual Machine.

Related information

Resource monitoring on page 1047

Viewing resource monitoring usage data on page 1087

Monitoring a Windows Performance host

To monitor a Windows Performance host, you must add the Windows Performance host source to the project in HCL DevOps Test Hub (Test Hub), and then select the counters that are used to capture the monitoring usage data.

Before you begin

You must have completed the following tasks:

- Installed and started an agent. See Resource monitoring agents on page 1065.
- Logged in to Test Hub. You must be the owner of an existing new project.

About this task

The Windows Performance host source is added to the list of resource monitoring sources when the agent is installed.

1. Click **Infrastructure > Resource Monitoring**.
   The resource monitoring page opens.
2. In the resource monitoring page, click **Add a Source** and select **add a Windows Performance host**.
3. Select an address to connect to the target host in **Access target from**.
4. Click **Add**.
   The **Choose Data Counters** dialog opens.
5. Select and save the statistics counters to monitor the source. You can select them from the list where they follow the server logical organization.
   For a faster selection, select the counters from the built-in sets drop-down list where they are organized by theme and save your selection.
   Result

   When the selected counters are saved, two tables are displayed in the resource monitoring main page. They contain the total number of sources you have added and the number of sources ordered by type.



What to do next

You can click the links in the tables to view the performance metrics and usage data in a graph for the monitored Windows Performance host.

Related information

Resource monitoring on page 1047

Viewing resource monitoring usage data on page 1087

Monitoring a Linux Performance host

To monitor a Linux Performance host, you must add the Linux Performance host source to the project in HCL DevOps Test Hub (Test Hub), and then select the counters that are used to capture the monitoring usage data.

Before you begin

You must have completed the following tasks:

- Installed and started an agent. See Resource monitoring agents on page 1065.
- Logged in to Test Hub. You must be the owner of an existing new project.

About this task

The Linux Performance host source is added to the list of resource monitoring sources when the agent is installed.

1. Click **Infrastructure > Resource Monitoring**.
   The resource monitoring page opens.
2. In the resource monitoring page, click **Add a Source** and select **add a Linux Performance host**.
3. Select an address to connect to the target host in **Access target from**.
4. Click **Add**.
   The **Choose Data Counters** dialog opens.
5. Select and save the statistics counters to monitor the source. You can select them from the list where they follow the server logical organization.
   For a faster selection, select the counters from the built-in sets drop-down list where they are organized by theme and save your selection.
   Result

When the selected counters are saved, two tables are displayed in the resource monitoring main page. They contain the total number of sources you have added and the number of sources ordered by type.



What to do next

You can click the links in the tables to view the performance metrics and usage data in a graph for the monitored Linux Performance host.

Related information

Resource monitoring on page 1047

Viewing resource monitoring usage data on page 1087

Resource monitoring agents

You can connect agents to HCL DevOps Test Hub (Test Hub) from other hosts to monitor a larger set of data collectors, computers, servers, or systems, or if a server to monitor requires authorization access. To monitor these remote servers, you must configure and run the agents so that a connection is set between Test Hub resource monitoring service and the agents. You can configure the resource monitoring agents in a team space or in a project.

Requirements:

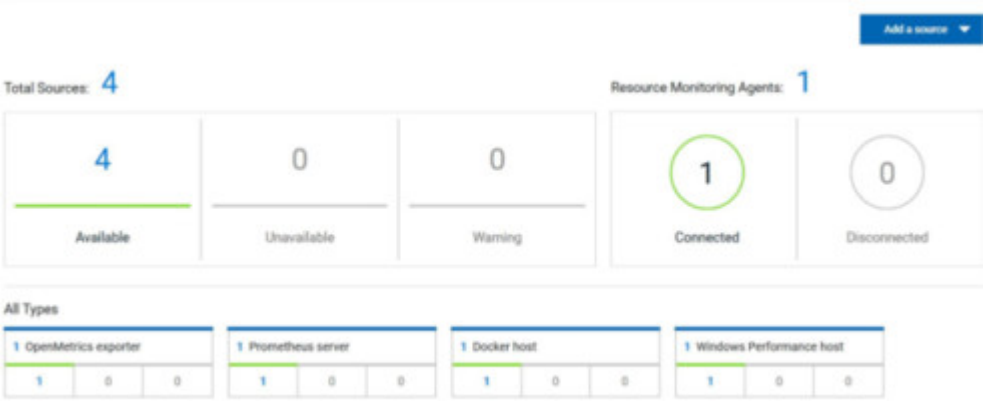
- To connect resource monitoring agents to Test Hub in a team space, you must be a *Team Space Owner*. Any other team space *Member* (not owner) can only view the resource monitoring agents.
- To connect resource monitoring agents to Test Hub in a project, you must be a project *Owner*. Any project *Member* (not the owner) can only view the resource monitoring agents.
- Agent-based monitoring is mandatory to collect data from a Windows or Linux Performance host.

For testing, you might need many hosts. For example, you might have one host with the application server, another host with the database server, and some hosts to apply the user load. Due to network or firewall issues, sometimes, it becomes difficult for multiple hosts to connect to each other. Resource monitoring agents are installed on the target hosts so that they can establish a connection with Test Hub to gather resource statistics of the target host.

The agents always try to connect with the server through the HTTPS protocol. You must install the agent and start it. From the resource monitoring page in Test Hub, you can copy the command lines to download files and run the agents. When you stop the agent, the monitoring sources that you have already added persist but the live data will not be available.

You can configure your resource monitoring agent as a service so that the agent automatically runs when the host is restarted. For details, see the documentation pages about starting resource monitoring agents on Windows or Linux.

When the agent is connected to Test Hub, it is added to the main page of the resource monitoring service. Then, you can select the agent to connect to a server from **Add a Source**.



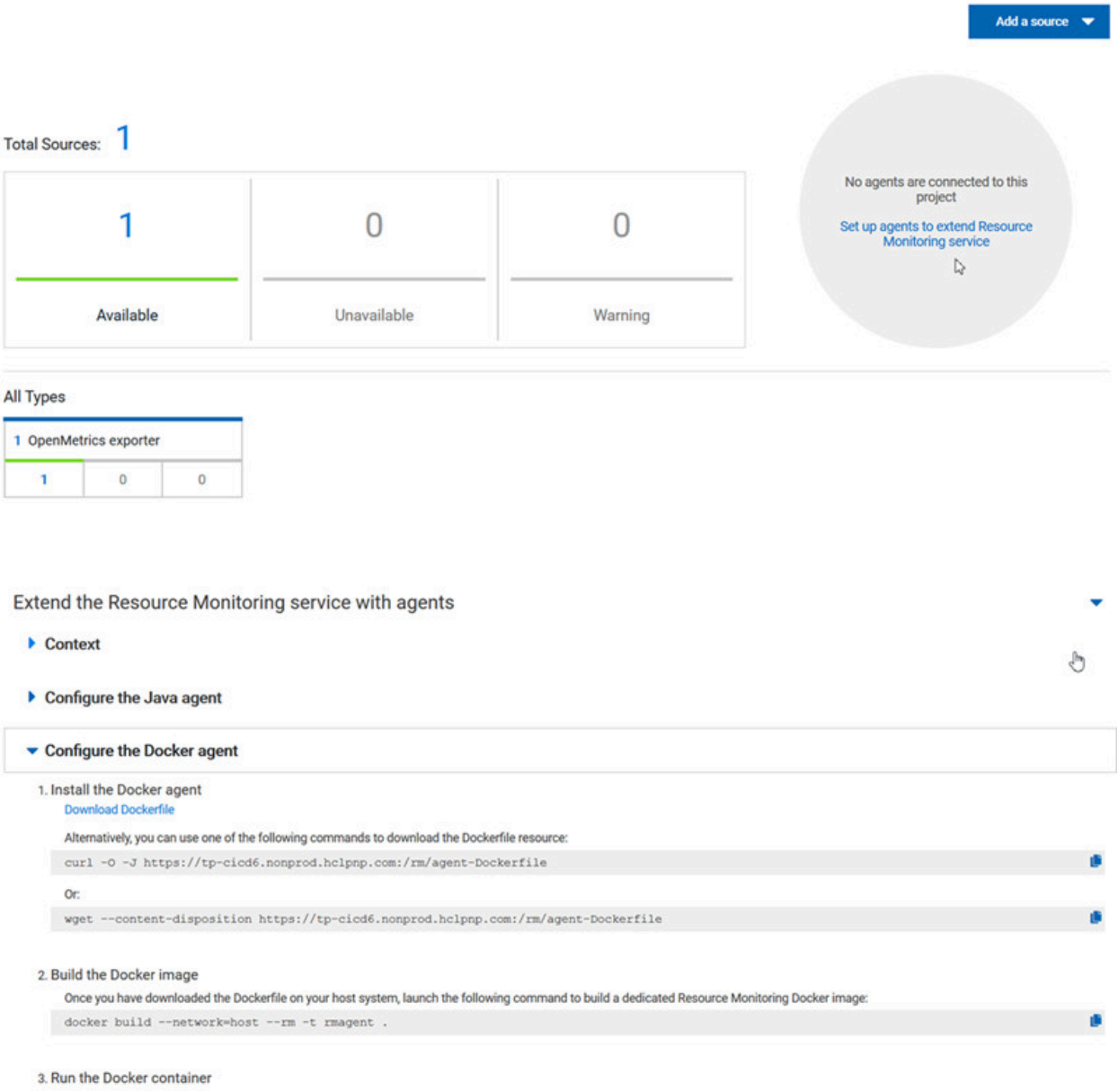
Installing and running a Docker agent on a remote host

You must install the resource monitoring Docker agents on the target host for which you want to monitor the resources and collect the metrics. You need to run an agent to establish a connection with HCL DevOps Test Hub (Test Hub).

Before you begin

- The resource monitoring service does not require access to the agent host but the agent must have reached the service host over HTTPS.
- The Docker agent must have been launched in a Docker container and requires Docker 19.03.
- Read and been familiar with the requirements to install and configure the resource monitoring agents on Test Hub. See Resource monitoring agents on page 1065.

1. Click **Set up Agents to extend Resource Monitoring service** link in Test Hub to access the agents page where you can find the instructions and commands that are to be used to install and run the agents.



2. Expand **Configure the Docker agent** and click **Download Dockerfile** to download a Docker agent.
   You can also use the *curl* or *wget* commands to download the agent without accessing the resource monitoring web UI.

   For more facility, use the code snippets to copy and fill in the commands with the valid offline token and the Docker file name.

   For Windows:

   ```
   curl -O -J https://hostName/rm/agent-Dockerfile
   ```

   For Linux:

```
wget --content-disposition hostName/rm/agent-Dockerfile
```

3. Copy the build command that is under **Build the Docker image**, paste it in your console and run it to build a resource monitoring Docker image.
   Before you start the a Docker agent, you must build a Docker image.

   ```
   docker build --network=host --rm -t rmagent
   ```

4. Follow these steps to run the agent:

   a. Click  to fill in the appropriate command for Windows or Linux with the valid offline token.

   b. Click  to copy the command to clipboard and paste the command in the console.

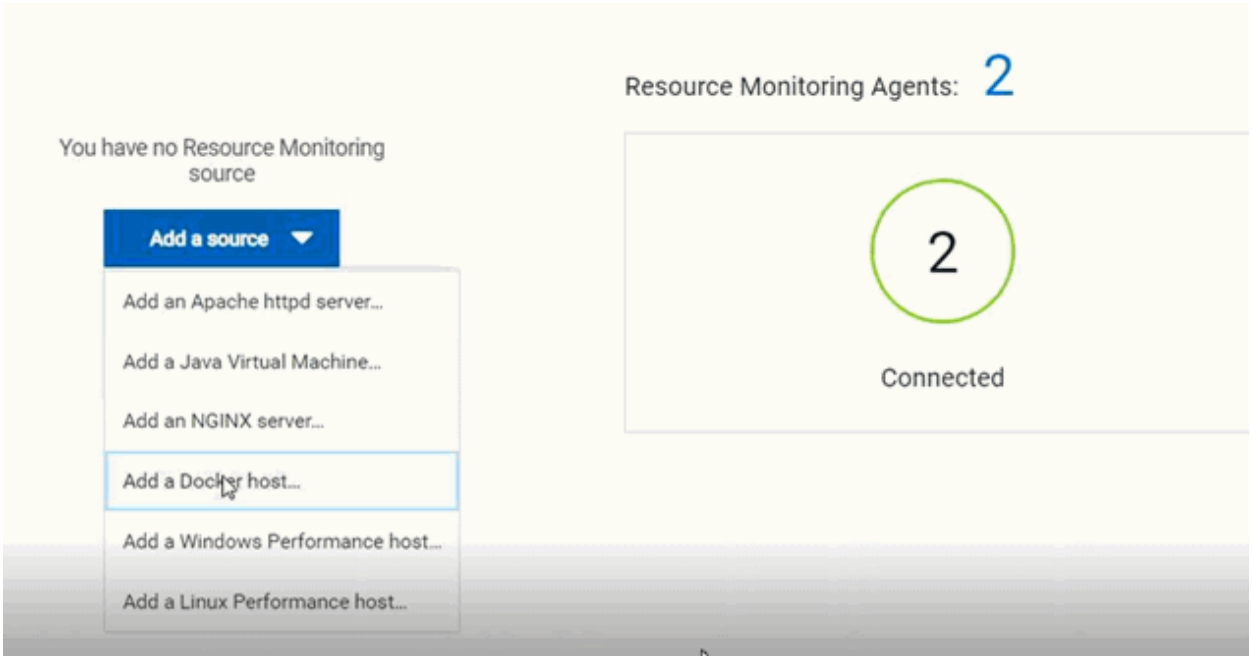
   - Command on Windows:

   ```
   docker run -it --rm --privileged -e PROJECT_ID=230200 -e HCL_ONETEST_OFFLINE_TOKEN=(Enter your
    offline token here) --network host -v //var/run/docker.sock:/var/run/docker.sock rmagent
   ```

   - Command on Linux:

   ```
   docker run -it --rm --privileged -e PROJECT_ID=230200 -e HCL_ONETEST_OFFLINE_TOKEN=(Enter your
    offline token here) --network host -v /var/run/docker.sock:/var/run/docker.sock -v /etc:/etc
    -v /proc:/proc rmagent
   ```

   c. Paste the command in the console and run the command.



Results

When the agent is started, the agent is connected to Test Hub. The agent is displayed in the list of connected resource monitoring agents with the hostname and status of the agent. The agent is also added to the main page of the resource monitoring service in Test Hub.

What to do next

You can add a new resource monitoring source. It can be collected from a service or from the named agents, depending on the capabilities supported by the environment.

Related information

Resource monitoring capabilities on page 1048

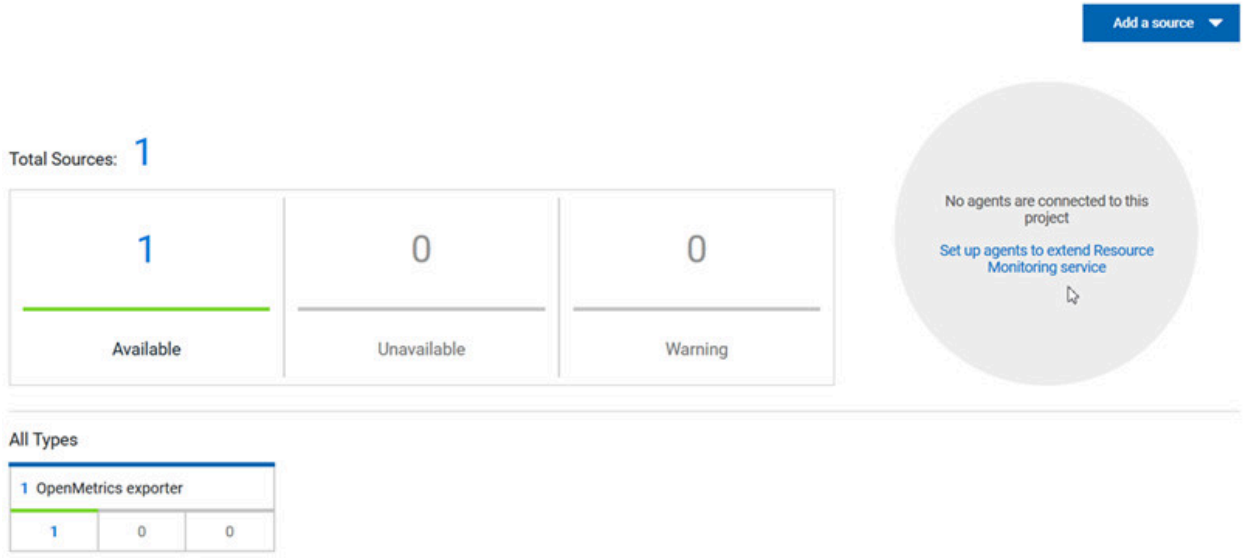
Installing and running a Java agent on a remote host

You must install the resource monitoring java agent on the target host for which you want to monitor the resources and collect the metrics of HCL DevOps Test Hub (Test Hub). You need to run the agent to establish a connection with Test Hub.

Before you begin

- The resource monitoring service does not require access to the agent host but the agent must have reached the service host over HTTPS.
- The Java agent must have been launched from a jar file and requires a Java 17 virtual machine.
- Read and been familiar with the requirements to install and configure the resource monitoring agents on Test Hub. See Resource monitoring agents on page 1065.

1. Click **Set up Agents to extend Resource Monitoring service** link in Test Hub to access the agents page where you can find the instructions and commands that you can use to install and run the agents.



2. Expand **Configure the Java agent** and click **Download jar file** to download the Java agent.
   You can also use the *curl* and *wget* commands to download the agent without accessing the resource monitoring web UI. For more facility, use the code snippets to copy and fill in the commands with the valid offline token and the jar file name.

With curl:

```
curl -O -J https://hostName/rm/Agent-jar
```

With wget:

```
wget --content-disposition https://hostName/rm/Agent-jar
```

3. Configure the command as follows to run your agent:
    a. Copy the appropriate command for Windows or Linux with the code snippets.
    b. Enter the path to the directory that contains the agent .jar file, and paste the command.

Example

To connect resource monitoring agents to Test Hub in a team space use the following commands:

On Windows:

```
set HCL_ONETEST_OFFLINE_TOKEN=(Enter your offline token here)
java -jar (Enter the name of the downloaded jar file here) --ServiceUrl=https://hostName/rm
 --spaceId=<team-space_id>
```

On Linux:

```
sudo HCL_ONETEST_OFFLINE_TOKEN=(Enter your offline token here) java -jar (Enter the name of the
 downloaded jar file here) --ServiceUrl=portNumber/rm --spaceId=<team-space_id>
```

Example

To connect resource monitoring agents to Test Hub in a project use the following commands:

On Windows:

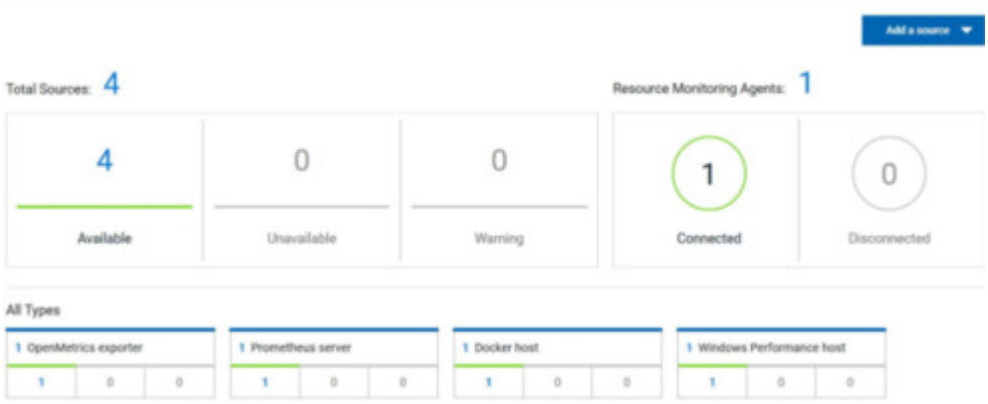
```
set HCL_ONETEST_OFFLINE_TOKEN=(Enter your offline token here)
java -jar (Enter the name of the downloaded jar file here) --ServiceUrl=https://hostName/rm
 --projectId=<project_id>
```

On Linux:

```
sudo HCL_ONETEST_OFFLINE_TOKEN=(Enter your offline token here) java -jar (Enter the name of the
 downloaded jar file here) --ServiceUrl=portNumber/rm --projectId=<project_id>
```

Results

When the agent is started, the agent is connected to Test Hub. The agent is displayed in the list of connected resource monitoring agents with the hostname and status of the agent. The agent is also added to the main page of the resource monitoring service in Test Hub.



What to do next

You can use another method to start an agent. See Starting a resource monitoring Java agent as a service on Windows on page 1071 or Starting a resource monitoring Java agent as a service on Linux on page 1074. You can then add a new resource monitoring source. Resource monitoring metrics can be collected from a service or from the named agents, depending on the capabilities supported by the environment.

Related information

Resource monitoring capabilities on page 1048

Starting a resource monitoring Java agent as a service on Windows

To ensure that the resource monitoring Java agent starts by itself when the host is running, you can set up the environment in such a way that the resource monitoring agent can be started as a service.

Before you begin

You must have completed the following tasks:

- Installed Java 17 on the host.
- Added HCL DevOps Test Hub (Test Hub) to the PATH environment variable.
- Created an offline token to connect the agent securely with appropriate permissions.

Note: You can create an offline token from the User menu of the resource monitoring page or you can re-use your active offline token. The token expires if it is not used for a month.

To allow remote access to the performance data on Windows 10, you must have considered the following information:

- The username and password are the same on the local and remote servers. Otherwise you must have provided remote server credentials.
- The remote user must be a member of the Performance Monitor Users group (start `lusrmgr.msc` and add the user to this group).
- The Remote Registry service must be running on the remote host (start `services.msc`) and must have verified the Remote Registry status).
- File and printer sharing must be enabled on the Network Interface of the remote host that is implied in the communication with the local host.
- You must have activated the following Windows Firewall rules so that the remote firewall does not block the access:
  - File and Printer Sharing (NB-Name-In)
  - File and Printer Sharing (NB-Session-In)
  - File and Printer Sharing (*)

1. Download the latest release of *winsw*.
   You can choose to download `WinSW.Net2.exe` or `WinSW.Net4.exe` depending on the version of .Net framework that you already have on the host Windows.
2. Create a folder on your local hard drive like `RMAgent-winservice`.
3. Copy the downloaded executable to this new folder and rename the file to `RMAgent-winservice.exe`.
4. Create a new text file in the same folder and name it `RMAgent-winservice.xml`.
5. Copy and adapt the following content to this new RMAgent-winservice.xml file to set up the offline token:

```
<service>
      <id>RMAgent-winservice</id>
      <name>Resource monitoring agent</name>
      <description>This service runs the resource monitoring agent.</description>
      <executable>java</executable>
      <env name="HCL_ONETEST_OFFLINE_TOKEN" value="(Enter your offline token here"/>

      <arguments>-jar %BASE%\RMAgent.jar --serviceUrl=https://<service-host>/rm --projectId=<projet-id>
 --autoUpgradeDownloadThen=execute:cmd,/c,start,%BASE%\auto-upgrade.bat</arguments>
      <onfailure action="restart" delay="10 sec"/>
      <logmode>rotate</logmode>
</service>
```

Notes:

- Replace *<service-host>* by the hostname of the host that runs Test Hub.
- Replace *<project-id>* which is the number that you find after /projects/ in the browser's URL when you browse to this project.

6. Create a new text file in the same folder and name it `auto-upgrade.bat`.
7. Copy the following content to this new file to upgrade the agent `.jar` file automatically:

```
@echo off
for /f "tokens=*" %%a in ('dir /b /od %BASE%\com.hcl.test.rm.agent-*.jar') do set newest=%%a
%BASE%\RMAgent-winservice.exe stop
del %BASE%\RMAgent.jar
mklink %BASE%\RMAgent.jar %BASE%\%newest%
%BASE%\RMAgent-winservice.exe start
```

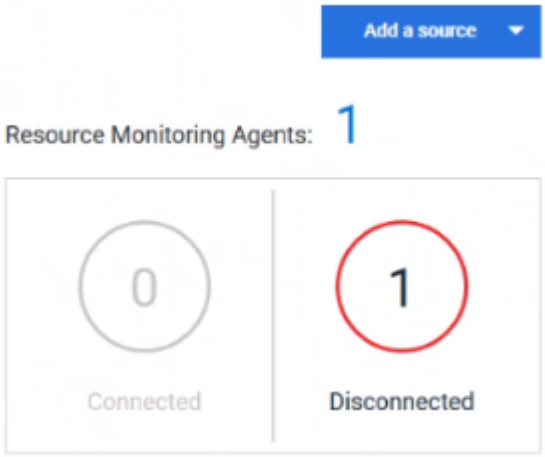
8. Start the command prompt as an administrator and change the directory to the newly created directory: `RMAgent-winservice`.
9. Download the agent .jar file that is available from the agents page in the resource monitoring service, under the **Extend the Resource Monitoring service with agents** section and save it to the same directory.
10. Enter the following command in the command prompt to create a symbolic link named `RMAgent.jar` to the agent jar file:

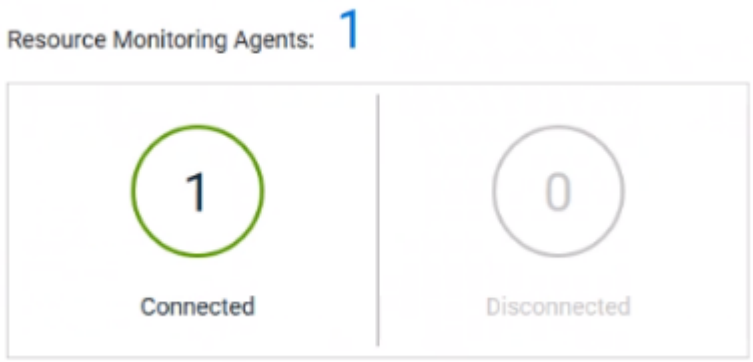
    ```
    mklink RMAgent.jar com.hcl.test.rm.agent-<version-and-datetime>.jar
    ```

11. Enter the following commands:

    ```
    RMAgent-winservice install
    ```

    ```
    RMAgent-winservice start
    ```





Notes:

In the Services Windows application, you can check whether the service is up and running. You can see in the resource monitoring page whether the resource monitoring agent is disconnected or connected.

The resource monitoring agents write the logs to the same folder in files named `RMAgent-winservice.out.log`, `RMAgent-winservice.err.log` and `RMAgent-winservice.wrapper.log`. If the Windows service for the agent is not started or if it is not connected to the resource monitoring service, verify the log files.

Results

The Java agent starts as a service on Windows. When the agent is connected to Test Hub, it is automatically added to the list of resource monitoring sources in Test Hub.

What to do next

You can monitor the Windows agent by selecting counters. See Monitoring a Windows Performance host on page 1063.

Related information

Other commands

winsw file

Resource monitoring capabilities on page 1048

Starting a resource monitoring Java agent as a service on Linux

To ensure that the resource monitoring Java agent starts by itself when the host is restarted, you can set up the environment in such a way that the resource monitoring agent can be started as a service.

Before you begin

You must have completed the following tasks:

- Installed Java 17 on the host.
- Added HCL DevOps Test Hub (Test Hub) to the PATH environment variable.
- Created an offline token to connect the agent securely with appropriate permissions.

Note: You can create an offline token from the User menu of the resource monitoring page or you can re-use your active offline token. The token expires if it is not used for a month.

About this task

This topic relies on systemd services that are the default on most modern Linux distributions. Other ways mi require adaptations of the instructions but the provided script will be a good basis in most cases.

1. Create a folder on your local hard drive like `/opt/RMAgent-linuxservice`.
2. Download the agent .jar file that is available from the Agents page in the resource monitoring service, below the **Extend the Resource Monitoring service with agents** section and save it to the same directory.
3. Create a new file `/etc/systemd/system/RMAgent-linuxservice.service`.

4. Add the following content to this new file:

```
[Unit]
Description = Resource monitoring agent
After = network.target

[Service]
Type = forking
ExecStart = /opt/RMAgent-linuxservice/RMAgent-linuxservice.sh start
ExecStop = /opt/RMAgent-linuxservice/RMAgent-linuxservice.sh stop
Restart = on-failure
RestartSec = 10

[Install]
WantedBy = multi-user.target
```

5. Create another file `/opt/RMAgent-linuxservice/RMAgent-linuxservice.sh`.

```
#!/bin/sh

#!/bin/sh

# Update the 3 following variables with the Server's hostname, project id and offline token:
SERVICE_URL=https://<hostname>/rm
PROJECT_ID=<project-id>
export HCL_ONETEST_OFFLINE_TOKEN=<offline-token>

ARGS="--serviceUrl=$SERVICE_URL --projectId=$PROJECT_ID --autoUpgradeDownloadThen=exitFailure"
SCRIPT=$(readlink -f "$0")
RMAGENT_HOME=$(dirname "$SCRIPT")

# Ensure we're using the latest downloaded jar file
PATH_TO_JAR=`ls -t $RMAGENT_HOME/com.hcl.test.rm.agent-*.jar | head -1`
if [ -z "$PATH_TO_JAR" ]
then
  cd $RMAGENT_HOME && { curl -k -O -J $SERVICE_URL/agent-jar; cd -; }
  PATH_TO_JAR=`ls -t $RMAGENT_HOME/com.hcl.test.rm.agent-*.jar | head -1`
  if [ -z "$PATH_TO_JAR" ]
  then
    echo "Start the server at $SERVICE_URL to allow download of the latest agent jar file"
    echo "Exiting..."
    exit 1
  fi
fi
SERVICE_NAME="Resource monitoring agent"
#Pid file will reside in this script’s folder
PATH_TO_PID=$RMAGENT_HOME/RMAgent-pid
#Log file will reside in this script’s folder
PATH_TO_LOG=$RMAGENT_HOME/RMAgent.log

case $1 in
    start)
        echo "Starting $SERVICE_NAME ..."
        if [ ! -f $PATH_TO_PID ]; then
            nohup java -jar $PATH_TO_JAR $ARGS >> $PATH_TO_LOG 2>&1 &
                        echo $! > $PATH_TO_PID
            echo "$SERVICE_NAME started ..."
```

```
        else
            echo "$SERVICE_NAME is already running ..."
        fi
    ;;
    stop)
        if [ -f $PATH_TO_PID ]; then
            PID=$(cat $PATH_TO_PID);
            echo "$SERVICE_NAME stopping ..."
            kill $PID;
            echo "$SERVICE_NAME stopped ..."
            rm $PATH_TO_PID
        else
            echo "$SERVICE_NAME is not running ..."
        fi
    ;;
    restart)
        if [ -f $PATH_TO_PID ]; then
            PID=$(cat $PATH_TO_PID);
            echo "$SERVICE_NAME stopping ...";
            kill $PID;
            echo "$SERVICE_NAME stopped ...";
            rm $PATH_TO_PID
            echo "$SERVICE_NAME starting ..."
            nohup java -jar $PATH_TO_JAR $ARGS >> $PATH_TO_LOG 2>&1 &
                        echo $! > $PATH_TO_PID
            echo "$SERVICE_NAME started ..."
        else
            echo "$SERVICE_NAME is not running ..."
        fi
    ;;
esac
```

Notes:

- Replace *<service-hostname>* by the hostname of the host that runs Test Hub.
- Replace *<project-id>* which is the number you'll find after `/projects/` in the browser's URL when browsing to this project.

Results

The resource monitoring Java agent starts automatically when the host restarts. The Linux agent is added to the main page of the resource monitoring service in Test Hub.

What to do next

You can monitor the Linux Performance host source by adding counters. See Monitoring a Linux Performance host on page 1064.

Related information

systemd services

Resource monitoring capabilities on page 1048

Starting a Java Virtual Machine

To monitor the sources from a Java Virtual Machine (JVM), you must first start the JVM. You must enter the IP address of the JVM, the IP port, and the security parameters in the command before running the Java Virtual Machine.

1. Enter The IP address of the JVM (local or remote host) and the IP port in the command that is used to run the Java Virtual Machine.
2. **Optional:** You can use the authentication security data to start the virtual machine. In this case, you must enter the name of the password file. You can also enter the name of an access file that might be needed if user privileges are required.

Example

Parameters used to launch a JVM without security:

```
java
-Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=9010
-Dcom.sun.management.jmxremote.local.only=false
-Dcom.sun.management.jmxremote.authenticate=false
-Dcom.sun.management.jmxremote.ssl=false
-jar MyapplicationFile.jar
```

Parameters used to launch a JVM with authentication security:

```
java
-Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=9010
-Dcom.sun.management.jmxremote.local.only=false
-Dcom.sun.management.jmxremote.authenticate=true
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.password.file=jmxremote.password
-Dcom.sun.management.jmxremote.access.file=jmxremote.access (this command is optional)
-jar MyapplicationFile.jar
```

Related information

Resource monitoring agents on page 1065

Monitoring a Java Virtual Machine on page 1061

Managing sources

After you add and connect monitoring sources to your project, you might want to modify the sources in you project and the configuration of the sources.

Before you begin

You must be a project owner or team space owner to add or delete sources.

About this task

In the page where you can view the resource monitoring usage data and metrics, you can also manage your sources. You can modify or delete a source, add or change statistic counters, add labels to identify the sources and use them when you execute a test to collect resource monitoring results.

1. Click **Infrastructure > Resource Monitoring** in the navigation pane.
   If you have sources and agents in your project, you can see the cards with the number of available sources and agents in your project.
2. Click a link in the card of available sources to open a list of sources.
3. Perform any of the following actions for a monitoring source:
   - Click the **Delete** icon to delete the monitoring source and click **Delete** to confirm the deletion in the dialog box.

     The source is removed from the list of sources.
   - Click the **Edit** icon to change the target host settings.

     You can change the target host address, the port, the path to the server status page and the security parameters.
   - Click the **Selection** icon to select additional counters or change the selected counters for other ones. The changes are reflected in the resource monitoring usage data graph.
   - Click to add a label to identify the source.
   - Click **Add a source** to add a source to your project from the list of sources.

Related information

Resource monitoring capabilities on page 1048

Viewing resource monitoring usage data on page 1087

Creating labels

After you connect a host or service to a project, you can add monitoring labels to identify the monitoring sources that are used when you execute a schedule from HCL DevOps Test Hub (Test Hub). You can also use labels in any test to collect resource monitoring usage data during a test run for a defined period of time.

Before you begin

You must have connected sources to your project in Test Hub. See Resource monitoring capabilities on page 1048.

About this task

You must be a project owner to add labels to the resource monitoring sources in a project and you must be a team space owner to add labels in a team space.

1. Click a source that is available in your project, either in the **Total Sources** cards or in the **All Types** cards from the resource monitoring page.
2. Click  under the **Labels** column of a resource monitoring source to add a label.
3. Enter a name for the label and press the Enter key.

Note: You can add one label or multiple labels for the same resource monitoring source. A label can contain a maximum of 200 characters.

Tip: You can create the same label for different sources that belong to the same logical group to monitor the whole group with only one label.

Results

The host source is identified by a label.

What to do next

You can use the created labels in tests to collect resource monitoring results or persistent usage data. See Controlling resource monitoring sources in a schedule on page 1079 and Collecting and viewing resource monitoring usage data on page 1085.

Controlling resource monitoring sources in a schedule

In HCL DevOps Test Hub (Test Hub), you can control the resource monitoring sources that are collected in a performance schedule and override them with the labeled resource monitoring sources that are available in your current Server project.

Before you begin

- You must have performed the following tasks in HCL DevOps Test Performance (Test Performance):
  - Created a performance schedule of **Rate** or **VU** type.
  - Added resource monitoring sources, or the labels of resource monitoring sources to a schedule. See Adding resource monitoring sources to a performance schedule by using labels.
  - Enabled Test Hub as resource monitoring service only if you want to replace resource monitoring labels set in Test Performance with the labels that you created in Test Hub.
- You must have created a project and added resource monitoring sources to your project in Test Hub. See Resource monitoring capabilities on page 1048.

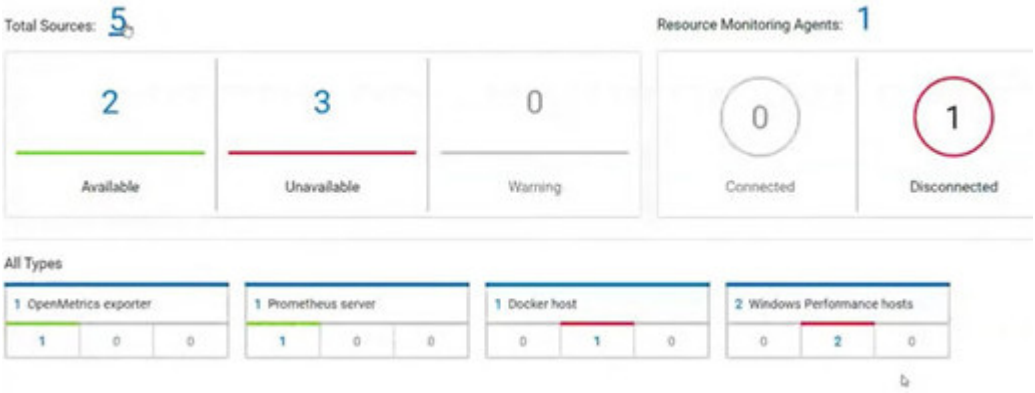
About this task

This procedure describes how to create resource monitoring labels in Test Hub and how to use them to override the source or label settings related to resource monitoring in a performance schedule. The procedure applies to the following use cases:

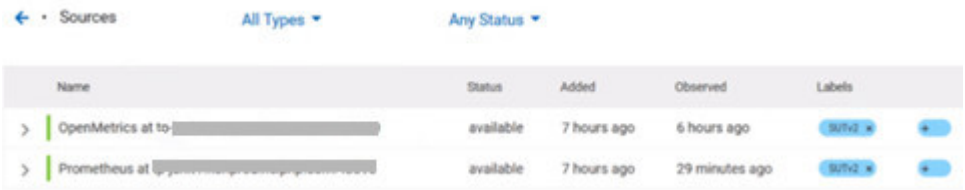
- To replace the resource monitoring labels that were initially set in a schedule from Test Performance with the labels that you create in Test Hub.
- To enable resource monitoring sources that were not enabled from Test Performance. In that case, you can enable resource monitoring sources from the **Execution** dialog in Test Hub.
- To ignore resource monitoring sources that were set in the schedule and to change them for labels that match sources that are available in the Test Hub project.

Notes: You must be a project owner to add labels to the resource monitoring sources in a project and you must be a team space owner to add labels in a team space. You must have a project owner or a tester role access to add labels to a performance schedule in a project.

1. Click a source that is available in your project, either in the **Total Sources** cards or in the **All Types** cards from the resource monitoring page.



2. Click + under the **Labels** column of a resource monitoring source to add a label.



3. Enter a name for the label and press the Enter key.

   Example: Enter 'SUTv2' in three different resource monitoring sources.

Note: You can add one label or multiple labels for the same resource monitoring source. A label can contain a maximum of 200 characters.

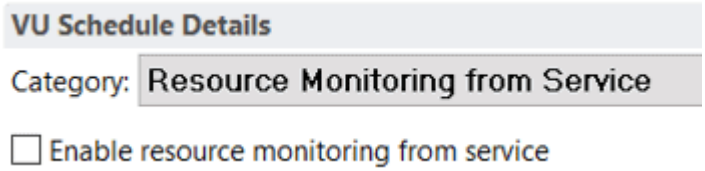
Tip: You can create the same label for different sources that belong to the same logical group to monitor the whole group with only one label.

4. Click **Execution** in the navigation pane and click on a performance schedule in the list of tests to execute a schedule.
5. Click the **Resource Monitoring** tab in the **Execute Test Asset** dialog.

Note: There are different use case scenarios:

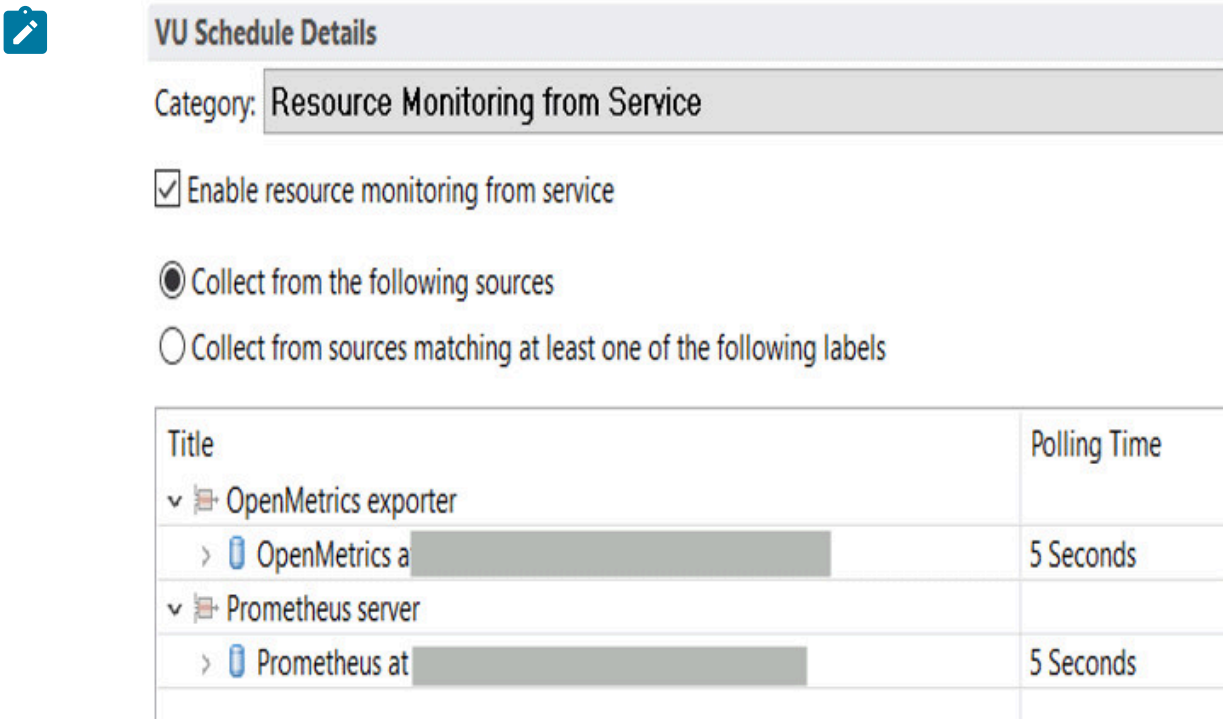
- If the **Resource Monitoring from Service** option was not enabled in Test Performance (see Figure 1), no source is available in Test Hub project.

  Figure 1:

  A message indicates in the **Resource Monitoring** tab that Resource Monitoring was not enabled. You can run the schedule with the resource monitoring sources that are created and labeled in Test Hub (steps 1 to 6 in this procedure) by selecting labels that are in your Test Hub project (step 6).
- If the **Resource Monitoring from Service** option was enabled in Test Performance, and data is collected from the sources you added to the performance schedule in Test Performance (see Figure 2), the **Resource Monitoring** tab displays the list of these sources that are available in your Test Hub project.

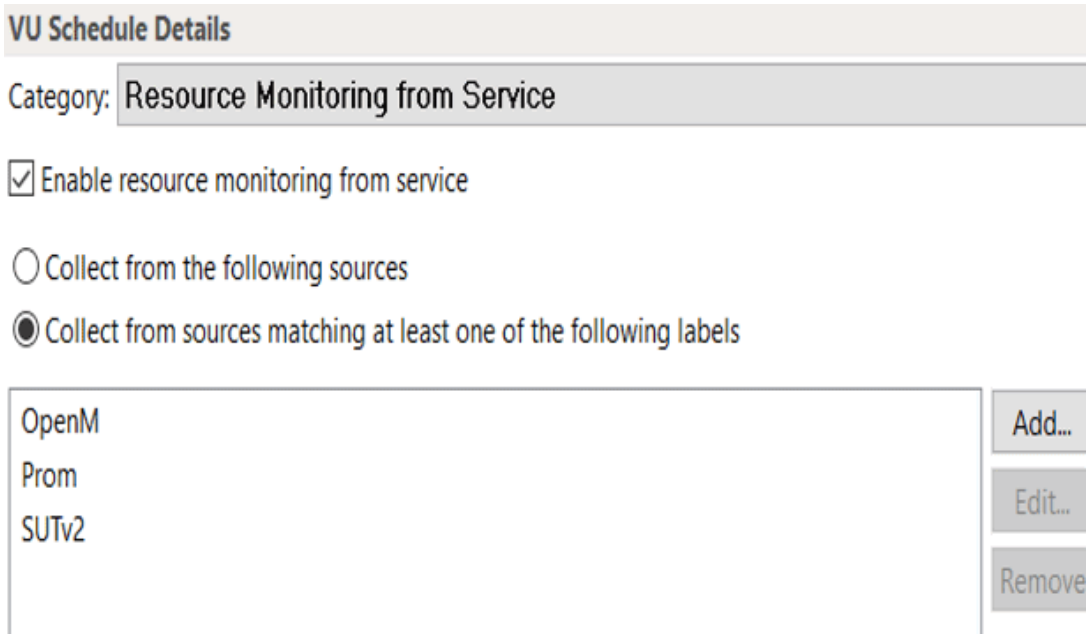
  Figure 2:



To override the initial sources with the sources that you created in Test Hub, select the labels that are in your Test Hub project and that match the Server sources (step 6 in this procedure).

- If you added sources to your performance schedule by using labels in Test Performance (see Figure 3), the **Resource Monitoring** tab displays the list of labels that are available in your Test Hub project.

Figure 3:



To select resource monitoring labels from your schedule, follow step 6 in this procedure.

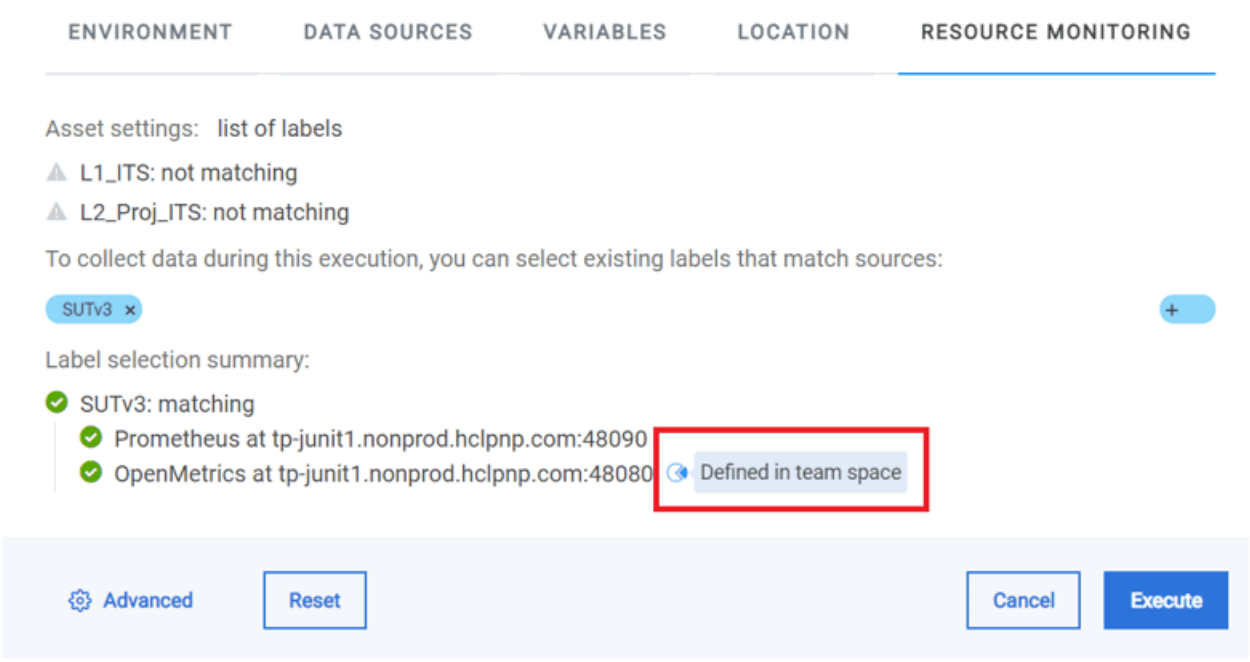
In the list of sources or labels that is displayed in the **Resource Monitoring** tab:

- Green icons identify either labels that match the sources or sources that are found in the current project.
- Red icons indicate that the sources are not running,
- Yellow icons identify either labels that do not match the sources, or sources that are not found in the current project.

  Data is collected for the sources that match labels only.

6. Click the + , and press the `Ctrl` + `Space` keys, or enter the initial letter of a label to select a label in the list.

   Example: Select **SUTv3**.

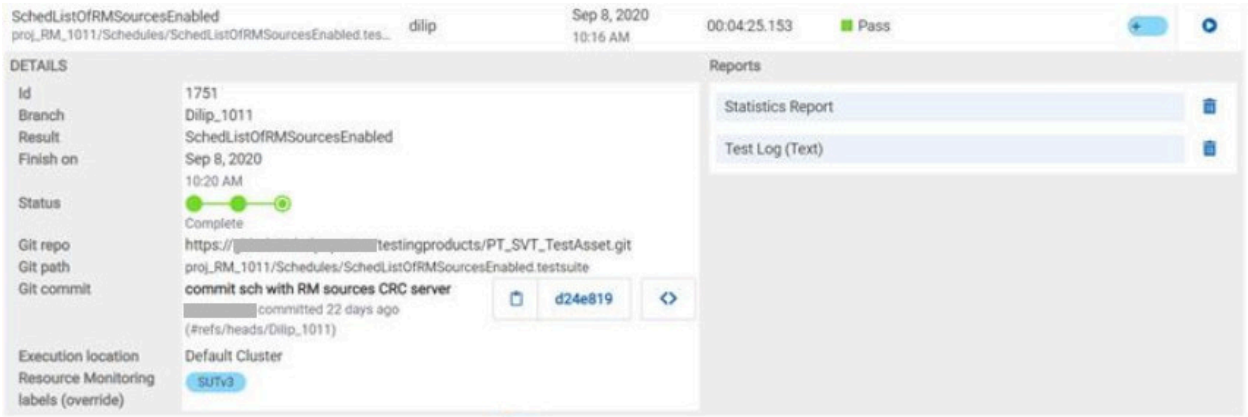


   You can see all the labels that you added to the resource monitoring sources in Test Hub.

Note: You can select labels matching resource monitoring sources that are created in a team space in addition to the ones that are created in a project.

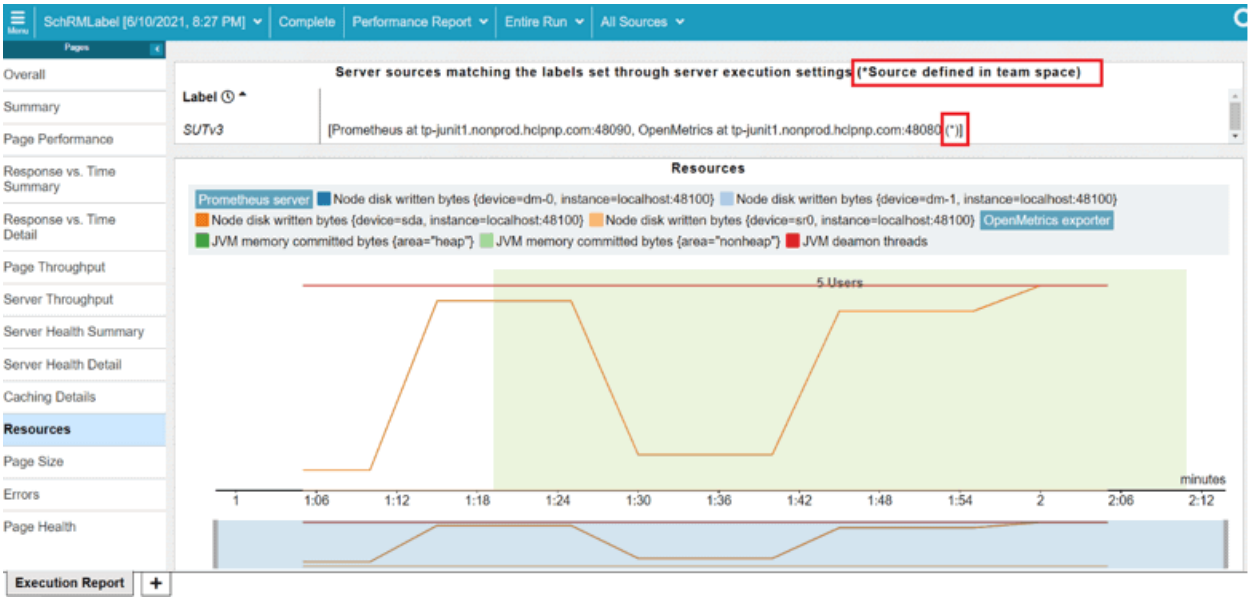
7. Select the labels that match the sources.
8. Click **Execute** to execute the performance schedule.
9. Click the **Result** view in the navigation bar of Test Hub when the schedule is complete.
10. Expand the performance schedule result to see the resource monitoring label name in the **Details** card.

The **Details** card displays the resource monitoring labels that are used in the schedule to override the initial labels and collect resource monitoring sources (in the example, the 'SUTv3' label). For other details about the **Results** page, see the link at the end of this page.



In the **Report** card, you find the link to the Statistics Report that is generated. In the **Resource** tab of the Statistics Report, you can view the information related to the Server sources matching the labels set in the schedule.

11. Click the **Statistics Report** link in the **Reports** card to see the performance metrics of the monitoring sources matching the selected labels.



Example: You can see the statistics for the sources tagged with the 'SUTv3' label.

Note: In the Resources report, each resource name followed by an asterisk identifies a resource monitoring source that is defined in a team space.

Related information

Test results and reports on page 985

Collecting and viewing resource monitoring usage data

With HCL DevOps Test Hub (Test Hub), you can collect resource monitoring data in a timeframe and view the resource monitoring usage data in a graph while the test is running. You can also create queries to retrieve the resource monitoring usage data.

About this task

While monitoring a system, you can view the current data in a graph but you might also want to view persistent data that is collected during a test run in a time frame. To view the persistent data, you can configure the test with additional parameters before the test execution. While the test is running, you can view the resource monitoring usage data in a graph. When the test is running or complete, you can create queries to retrieve the collected data results.

To collect and view resource monitoring data, perform these tasks:

1. Configure a test with two additional parameters in the advanced settings.
2. View the resource monitoring usage data in a graph.
3. Create queries to retrieve the resource monitoring usage data.

Related information

Configuring a test on page 1085

Viewing resource monitoring usage data on page 1087

Viewing resource monitoring usage data on Grafana on page 1089

Creating queries for resource monitoring usage data on page 1091

Configuring a test

You must configure a test with two parameters to collect resource monitoring usage data in a time frame when the test is running.

Before you begin

You must have completed the following tasks:

- Added and configured a source.
- Created resource monitoring labels for your source.

About this task

You can collect resource monitoring usage data during a test run for all types of monitoring sources, and all test types, except API stubs.

1. Click **Execution**, in the navigation pane.
2. Select the test that you want to run and click  in the **Actions** column to open the **Execute test asset** window.
3. Click **Advanced settings**.
4. Enter the following parameters and values in **Additional Configuration Parameters**:

Parameter name	Value
rm.log.labels	Enter a label name in the second edit field as the parameter value. If you enter multiple labels, enter a comma to separate them in the list. **Example:** For two labels with the following names: 'LabelA' and 'LabelB': rm.log.labels LabelA,LabelB If a label contains a comma in its name, you must add two commas in order to collect the resource monitoring usage data. **Example:** For a label named 'Label1A,Label1B', enter the following parameter and value: rm.log.labels Label1A,,Label1B
rm.log.period.seconds	Enter a sampling period in the second edit field as the parameter value. The minimum period is one second.

5. Click **Add** for each parameter.
6. Click **Execute**.

Results

The monitoring data is collected while your test is running. You can see the collected data in a graph of the resource monitoring page, even 3 minutes after the test is complete.

What to do next

You can view the resource usage data on the **Resource Monitoring** page in HCL DevOps Test Hub (Test Hub) and Grafana.

- Viewing resource monitoring usage data on page 1087
- Viewing resource monitoring usage data on Grafana on page 1089

Viewing resource monitoring usage data

After you add and configure monitoring sources in HCL DevOps Test Hub (Test Hub), you can view the resource monitoring usage data in a graph for each monitored source and for each selected counter.

Before you begin

You must have completed the following tasks:

- Added and connected a monitoring source, and an agent when required, to your project to be able to view the list of available sources and agents, and the resource monitoring usage data in your project.
- Configured a test with advanced parameters to view persistent usage data in a graph.

About this task

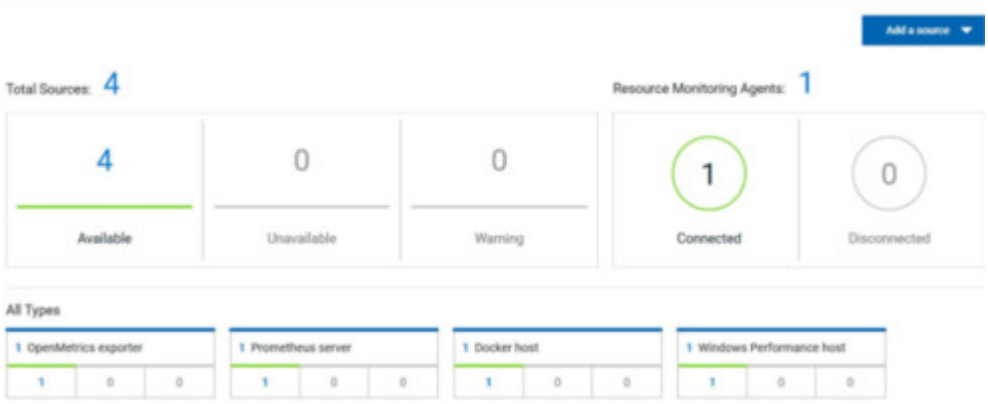
After you add a monitoring source to Test Hub, configure the source and select monitoring counters, you can view the current resource monitoring usage data for each selected counter in a graph. If you configure the test to collect data, you can also view the resource monitoring usage data that is collected for a defined timeframe. In this case, the data is still visible in the graph three minutes after the test run is complete. After this time, you can create queries to retrieve the data.



Note: For a Prometheus source, as data comes from the Prometheus database, it no longer shows up in the graph when the test run is complete. However, the data is collected and shows up in the queries if it matches a label.

1. Click a link either in the **Total Sources** cards or in the **All Types** cards from the resource monitoring page to access the resource monitoring usage data.
   The tables display the total number of sources in your project and the number of sources by type. You can also view the number of available and unavailable sources, and the number of sources that contain warnings. If you have added '4' sources to your project, for example, click '4' in the table to access the page where you can see the resource usage data for each source.

   If you have added agents to your project, you can also view the number of agents that are connected and disconnected to your project.

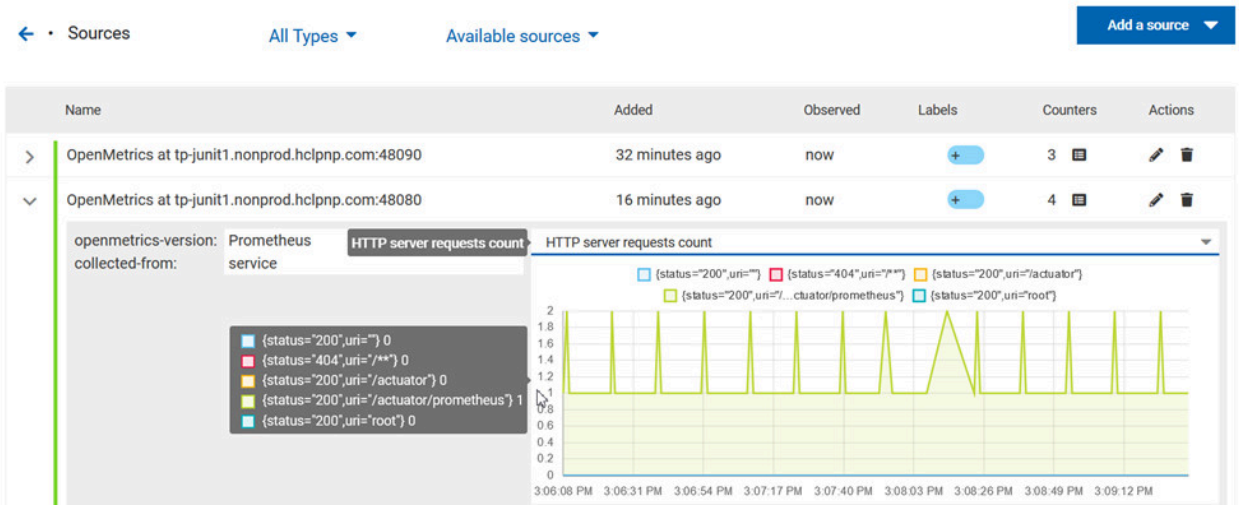


Result

The list of sources is displayed in a new page.

You can filter the list of sources by collector type or by status.

2. Expand a monitoring source under the **Name** column to see the current monitoring data usage in a graph.
3. Click the down arrow to select a counter in the list.
   A graph is displayed for each counter.



What to do next

From the page where you can view the resource monitoring usage data, you can also manage your sources. See Managing sources on page 1077.

Related information

Collecting and viewing resource monitoring usage data on page 1085

Creating queries for resource monitoring usage data on page 1091

Viewing resource monitoring usage data on Grafana

After you add and configure monitoring sources in HCL DevOps Test Hub (Test Hub), you can view the resource monitoring usage data in a graph for each monitored source and for each of the selected counters on Grafana.

Before you begin

You must have completed the following tasks:

- Added and connected a monitoring source, and an agent when required, to your project to view the list of available sources and agents, and the resource monitoring usage data in your project.
- Configured a test with advanced parameters to view persistent usage data in a graph. See Configuring a test on page 1085.
- Installed the DevOps Test Hub Results Data plug-in and created an instance of data source for DevOps Test Hub Results Data. See Installing and configuring the DevOps Test Hub Results Data plugin on page 1000.
- Opened Grafana with one of these ports: `http://localhost:3000` or `http://localhost:8080`.
- Started the test execution on Test Hub.

About this task

After you add a monitoring source to Test Hub, you must configure the source, and select monitoring counters. You can view the resource monitoring usage data for each of the selected counters in a graph on Grafana. You must configure the test to collect and view the resource monitoring usage data that is collected for a defined timeframe. During the test execution, the data can be seen with the default time range on Grafana. After the test execution is complete, you must carefully select the time range so that it corresponds to the test execution start and end time.

1. Click **Explore** in the **Toggle** menu in Grafana.

   The **Data query** page with the default data source is displayed.
2. Select **DevOps Test Hub Results Data** from the list of data sources.
3. Select the type of query by performing any of the following actions when you want to view statistics from the test data:

To view	Actions
The variation of a metric over time.	Perform the following steps: a. Click the time-series statistics tab icon  . b. Go to Step 4 on page 1089.
The values of the metric that are computed over the entire time range.	Perform the following steps: a. Click the overall statistics tab icon  . b. Go to Step 4 on page 1089.

4. Define the scope of the query by performing the actions indicated in the following table:

Options	Actions
If you want to use the data from the team space	Perform the following steps: a. Select the **Team Space** option. b. Select the team space from the list of team spaces that are listed. c. Select **Resource Monitoring** from the list in the **Service** option. d. Go to Step 5 on page 1090.
If you want to use the data from your project	**Note:** You can select the project as the scope for your query if you want to view any of the supported query types. Perform the following steps: a. Select the **Project** option. b. Select the project from the list of projects that are listed. c. Select the **Service** option. d. Select **Resource Monitoring** from the list e. Go to Step 5 on page 1090.

5. Optionally, enter a name for the metric that helps you identify it in the **Metric** field.
6. Select a metric from the list of metrics in the row of the **Metrics** option.
7. Select the aggregation factor for the metric. For example, select **Sum** to add all values of the metric.
8. Optionally, click the **Cumulated** toggle button to add the values for the metric cumulatively over time.

   Note: The **Cumulated** is set to off. This is the default selection.

9. Add another metric by clicking the **Add metric** icon  .

   Note: If you added multiple metrics and you want to remove a metric, you can click the **Remove metric** icon  of the metric.

10. Click **Add query** if you want to combine queries from different projects, results ids, or with different properties in the query filters.
11. Click **Run query**.

    Result

    A graph is displayed for the selected metrics.

Results

You viewed the resource monitoring usage data in a graph for each monitored source and for each of the selected counters on Grafana.

What to do next

You can build a dashboard including graphs, charts, gauges, and tables to get customized visual representations of your resource monitoring usage data. See Dashboard overview.

Creating queries for resource monitoring usage data

After you execute a test to collect resource monitoring usage data, you can create queries in HCL DevOps Test Hub (Test Hub) to retrieve the data.

Before you begin

You must have added resource monitoring sources to a project.

About this task

You can no longer view the monitoring usage data in graphs 3 minutes after completion of the test. However, you can create queries to retrieve the list of activities and events in your tests. You can use the following procedure as an example to create your own queries.

1. Follow these steps to filter the list of activities that are collected for a duration of less than 300 seconds:
    a. Click **Activities**.
    b. Enter `'Type is RMsource and duration < 300'` in the **Query filters** field.
    c. Enter `'time ASC'` in the **Sort by** field to sort data by ascending order.
    d. Click **Run Query**.
    e. Click **Save as**.
    f. Enter a name to identify the new query and click **Save**.

    The list of activities that are collected for a duration of less than 300 seconds is displayed in the query result table, and this list is sorted in an ascending order.
2. Follow these steps to filter the list of observations that are performed during a test execution and defined by the test result identifier:
    a. Click **Events**.
    b. Enter the following filters in the **Query filters** field:

    ```
    type is observation and anyParentActivity matches ( anyChildActivity matches (requesterNature = 'test-result' and requesterId=1202 ))
    ```

    c. Enter `'time ASC, parentActivity.description ASC, specifier ASC'` in the **Sort by** field to sort data by ascending order.

    Note: You can use the **Ctrl-Space** keyboard shortcut to enter the properties and operators.

    d. Click **Run Query**.
    e. Click **Save as**.
    f. Enter a name to identify the new query, and then click **Save**.

    The new query results are displayed in the table of activities and events.
3. Follow these steps to manage your query:

- Click the action menu that is next to the **Save** button and select **Reset** if you have just modified a query and you want to undo your modification.
- Click the action menu and select **Delete** to delete the new query from the **Events and activities** queries list.

4. Follow any of these steps to configure the query result table:
   - Click the action menu in a column of the table to change the column display, and then perform any of the following actions:
     - Click **Insert new column**, enter a field name in **Search new column** and click **Add**.
     - Click **Move** to move a column in the table.
     - Click **Remove** to remove a column from the table.
   - Follow these steps to add a column at the end of the table:
     a. Click **Add column**.

     Result: A window opens.

     b. Enter a field name in **Search new column**.
     c. Click **Add**.
5. Click **Save** to save the configuration of the results table.

Results

You have created two queries in your project to display:

- The list of activities that are collected for a duration of less than 300 seconds.
- The list of observations that are performed during a test execution and defined by the test result identifier.

Related information

Configuring and running log queries on page 1023

Configuration of a change management

When you use any application to create and track defects (bugs), issues, or other work items, you can configure the application as a change management on HCL DevOps Test Hub (Test Hub). You can then create defects (bugs), issues, or other work items without the need to open your application.

You use Test Hub to run tests for the software that you develop and view their results. When you discover that you might want to raise defects (bugs), issues, or other types of work items for these tests, you are required to open the application that you use. You can only then, create defects (bugs), issues, or other work items in your application.

When you configure your application as a change management on Test Hub, you can create defects (bugs), issues, or other work items without the need to open your application.

You can find details in the following table about the usage of the applications that you can configure as a change management on Test Hub.

Note: You can configure only one application as the change management on Test Hub.

Issue tracking application	Usage as a change management	Go to...
Atlassian Jira Software	You can create stories, defects, or tasks as issues in your project in Jira from Test Hub and also launch your project in Jira from Test Hub to track and manage these issues.	Configuration of Atlassian Jira as a change management on page 1093
Azure DevOps	You can create work items in your project in Azure DevOps from Test Hub and also launch your project in Azure DevOps from Test Hub to track and manage these work items.	Configuration of Azure DevOps as a change management on page 1105
DevOps Plan	You can use Plan as a change management in Test Hub. You can also launch Plan from Test Hub to track and manage change requests and other records.	Configuration of DevOps Plan as a change management on page 1114

Configuration of Atlassian Jira as a change management

You can use Atlassian Jira as a change management in HCL DevOps Test Hub (Test Hub) to create issues in the Jira project for the tests that ran in Test Hub. The links to the Jira issues are displayed on the **Results** page in Test Hub, and you can then view and track the Jira issues in Jira.

Prerequisites

Before you configure Jira as a change management on Test Hub, you must have completed the following tasks:

- Installed Jira Software. See Supported installations on page 1093.
- Ensured that you obtained the access and authentication details to Jira from the Jira administrator.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.

Supported installations

The following types of Jira Software installations are supported that can be configured as a change management on Test Hub:

- On-premise Jira Software installations.
- On-cloud Jira Software installations.

Task flow

You can follow the tasks that are listed in the following table when you want to use Jira as a change management on Test Hub. The table also provides you the links to the information about the tasks.

Task	Go to...
Configure Jira as a change management on Test Hub.	Configuring Jira as a change management on page 1094
Create defects or issues for tests that ran on Test Hub.	Creating Jira issues for the test results on page 1096
View details of the Jira issues that you created from Test Hub.	Viewing details of the Jira issues on page 1102

You can manage the Jira settings that you configured in Test Hub. You can edit the saved configuration or you can delete the configuration. For more information, see Managing the Jira configuration on page 1104.

Configuring Jira as a change management

You must configure Jira as a change management in your project in HCL DevOps Test Hub (Test Hub) so that can create issues in the Jira project for the tests that ran.

Before you begin

You must have completed the following tasks:

- Verified that you can access Jira.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.

About this task

With the introduction of OAuth 2.0 authentication for the connection between Test Hub and Jira, you must be familiar with the creation of Apps that are based on the OAuth 2.0 integration. There can be more than one method in the Atlassian Developer console that you can configure the created App. One such method is adopted in the configuration of an App in Jira.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Integration** in the navigation pane.

The **Integration** page is displayed.

4. Click **Add Change Management Integration** option within the **Change Management** section.
5. Perform the following steps in the **Add Jira configuration** dialog:

    a. Enter the URL of the Jira server or Jira cloud in the **Jira URL** field, and then click **Next**.

    Test Hub provides the *Callback URL* that you must use to generate the Client ID and Client secret for the App in Jira.

    b. Click the **Atlassian Developer console** link.

    The **Atlassian Developer console** page is displayed.

    Perform the following steps on the **Atlassian Developer console** page:

        i. Click **Create > OAuth 2.0 integration**.
        ii. Enter a name for the App in the **Name** field.
        iii. Click the **I agree to be bound** checkbox, and then click **Create.**

        The App is created in the Atlassian Developer Console and the details are displayed for the App.

        iv. Click **Authorization** in the navigation, and then click **Add** in the row of OAuth 2.0 (3LO).

        The OAuth 2.0 configuration page is displayed. You must paste the *Callback URL* provided by Test Hub.

        v. Copy the *Callback URL* from the **Add Jira configuration** dialog in Test Hub and then paste it into the **Callback URL** field.
        vi. Click **Save changes**.

        The App in Jira is configured to authenticate the requests from Test Hub by using the OAuth 2.0 protocols.

        vii. Click **Permissions** and then, click the **Configure** option in the row of **Jira API**.
        viii. Click **Edit Scopes** on the **Jira API** page, and then select the following scopes for your App that are required for a successful connection with Test Hub:
            - View Jira issue data
            - Manage Jira global settings
            - View user profiles
            - Create and manage issues
        ix. Click **Save** to save the App settings.
        x. Click **Settings** and scroll down the page to view the **Authentication details** panel.

        The **Client ID** and **Secret** must be copied from the **Settings** page to the corresponding fields on the **Add Jira configuration** dialog.

c. Click the **Copy** icon in the **Client ID** field to copy the key, and then paste the copied key in the **Client ID** field on the **Add Jira configuration** dialog.

d. Click the **Copy** icon in the **Secret** field to copy the key, and then paste the copied key in the **Secret** field on the **Add Jira configuration** dialog.

6. Click **Save** on the **Add Jira configuration** dialog.

   You added the URL of the Jira cloud and configured an App in Jira to authenticate the connection with Test Hub by using OAuth 2.0. You can view that the URL of the Jira server or cloud instance is displayed within the **Change Management** pane.

7. Click the **Configure** icon .

   A page in Atlassian is displayed in a browser that states that the App you created previously is requesting access to your Atlassian account.

8. Read the message that is displayed, and then click **Accept**.

   When the connection between Test Hub and the App in Jira is established successfully, the **Configure Jira project** dialog is displayed with the Jira URL that is configured as a change management system in Test Hub.

9. Select the project in Jira from the list under which you want to create defects.
10. Click **Save**.

Results

You configured Jira as a change management system in Test Hub.

What to do next

You can perform the following tasks:

- View the following configuration details of Jira on the **Change Management** about the Jira configuration:
  - The URL of the Jira server or cloud instance.
  - The project that you selected.
- Create issues for the test results that are available in your project in Test Hub. See Creating Jira issues for the test results on page 1096.

Creating Jira issues for the test results

You can create Jira issues for the tests that ran in HCL DevOps Test Hub (Test Hub). The issues are created in the project in Jira that you configured in Test Hub.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Configured Jira as a change management for your project in Test Hub. See Configuring Jira as a change management on page 1094.
- Configured a test run. See Test run configurations on page 610.
- Viewed the results of the completed test runs on the **Results** page. See Test results on page 983.

About this task

When you want to create a Jira issue in Test Hub, the issue types that display in the **Create an issue in Jira** dialog depend on the configuration done by the administrator of Jira.

The default fields that display on the **Create an issue in Jira** dialog are as follows:

- **Issue Type**
- **Summary**
- **Description**
- **Attach Reports**

If you want to view any additional fields in the **Create an issue in Jira** dialog from Test Hub, then you must inform your Jira administrator to configure those additional fields as the required fields in the project in Jira. You can then use those fields in the **Create an issue in Jira** dialog when you create a defect from Test Hub.

You can refer to the **Related information** section to know more about Jira field configuration.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Analyze > Results**.

   The **Results** page is displayed and all the test assets that were run earlier are displayed in tabs, as listed in the following table:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

Note: You can click the relevant tab to view the tests that were run earlier.

4. Perform any of the following actions to identify the test asset:
    - Scroll through the list.

Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test by entering any text that is contained in the test asset name in the **Search** field.
- Select the time interval within which the test was run.

  For example, you can select the *Last 24 hours* option to view the tests that were run in the previous 24 hours.
- Select the range of dates within which the test was run.

  You can click the calendar icon to select the date in the first image as the start date, and then select the date in the second image as the end date.

  Note: You can select the range of dates only when you select the time interval as *Custom*.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab, and then select the test type.

  All the tests based on the selected test type are displayed.

  For example, you can select **API Suite** in the **SUITES** tab to display all the API suites or you can select **Functional Test** in the **TESTS** tab to display all the functional tests.
- Click the **Filter** icon in the **ADVANCED** tab, and then perform the following actions to create a filter query:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select an operator, and then select **Add rule** to add a rule or **Add group** to add a group of rules.
  d. Select or enter the relevant parameters, and then either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:
     - ID
     - Type
     - Test
     - Version
     - Result
     - Started on

- Finish on
- Duration
- Status
- Verdict
- Labels
- Locked
- Repository
- Branch
- Text

e. Click **Save**.

The filter query is saved and all the tests that match the filter criteria are displayed.

- Perform the following actions in the **ADVANCED** tab to retrieve and apply a previously saved filter query:

a. Click the **Filter** icon  .

All the filter queries that you created and saved earlier are displayed.

b. Click the filter query that you want to apply.

All the tests that match the filter criteria are displayed.

You have identified the test that you want to use.

5. Select the step that depends on the role assigned to you in the project on Test Hub:

- If you are a member with a role assigned as *Owner*, go to Step 6 on page 1100.
- If you are a member with a role assigned as *Tester*, and if you did not complete the one-time authorization process, go to Step 7 on page 1100.
- If you are a member with a role assigned as *Tester* and completed the one-time authorization process, go to Step 8 on page 1101.

6. Perform the following steps if you are assigned the role of a project *Owner*:

a. Click the **Jira: Create a Jira issue** icon  that is under the **Actions** column in the row of the test.

Alternatively, you can click the test under the **Test** column to view the results panel, and then click the **Jira: Create a Jira issue** icon  that is displayed in the panel.

The **Create an issue in Jira** dialog is displayed.

b. Go to Step 9 on page 1101.

7. Perform the following steps if you are assigned the role of a *Tester* and if you have not yet completed the one-time authorization process:

Note: You must complete the one-time authorization process if you are not a project owner.

a. Click the **Jira: Create a Jira issue** icon that is under the **Actions** column in the row of the test.

   Alternatively, you can click the test under the **Test** column to view the results panel, and then click the **Jira: Create a Jira issue** icon that is displayed in the panel.

   The **Configure Jira server to authorize the application** dialog is displayed.

b. Click the URL of the Jira server.

c. Log in to the Jira server account with your valid credentials.

   Note: The **Login** page is displayed only if you are not already logged in to Jira.

d. Read the message that is displayed, and then click **Allow** to authorize the Jira account.

   After the authorization is successful, the **Access Approved** page is displayed with a verification code.

e. Copy the verification code, and then enter the verification code in the field on the **Configure Jira server to authorize the application** dialog of Test Hub.

f. Click **Save**.

g. Go to Step 8 on page 1101.

8. Perform the following steps if you are assigned the role of a *Tester* and you have completed the one-time authorization process:

   a. Click the **Jira: Create a Jira issue** icon that is under the **Actions** column in the row of the test.

      Alternatively, you can click the test under the **Test** column to view the results panel, and then click the **Jira: Create a Jira issue** icon that is displayed in the panel.

      The **Create an issue in Jira** dialog is displayed.

   b. Go to Step 9 on page 1101.

9. Perform the following steps in the **Create an issue in Jira** dialog:

   Note: The fields that are displayed might depend on the fields configured in the Jira project.

   a. Select the issue type that you want to create.

   b. Enter text for the title of the Jira issue. The name of the test is displayed as the default text in the **Summary** field.

You can view the details of the test run that is displayed in the **Description** field. These details are appended to the Jira issue under the **Description** field, after you create the issue.

c. Select the existing labels in your Jira project or enter a new name for the label in the **Label** field, if required.

Note:

If the **Label** field is set to "Required" in your Jira project, you must select an existing label or create a new label by providing a name in the **Label** field. If you don't select an existing label or create a new label, an error message is displayed that indicates the **Label** field is a required field.

d. Select the test reports or test logs of the test run that you want to attach to the Jira issue.

Note: You can select any report, test log, or all the reports and the test log as attachments.

e. Click **Create**.

The issue is created in the project on the Jira server and the link to the issue is displayed in the **Results** panel of the test on Test Hub.

Note: You can create multiple issues for the same test result, if you want.

Results

You created issues in the Jira project for the test results from Test Hub.

The Jira issues are created in the Jira project and the links to the issues are displayed in the **Details** panel of the test result on the **Results** page.

What to do next

You can perform any of the following tasks:

- View details of the Jira issues that you created in Jira for the test. See Viewing details of the Jira issues on page 1102.
- Modify the configuration that you set for Jira on Test Hub. See Managing the Jira configuration on page 1104.

Related information

Specifying Jira fields

Viewing details of the Jira issues

After you create Jira issues in HCL DevOps Test Hub (Test Hub), you can view and track the details of the Jira issues in Jira.

Before you begin

You must have completed the following tasks:

- Completed a test run and the test results are available on the **Results** page.
- Created Jira issues against the test results. See Creating Jira issues for the test results on page 1096.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Analyze > Results**.

   The **Results** page is displayed and all the test assets that were run earlier are displayed in tabs, as listed in the following table:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

   Note: You can click the relevant tab to view the tests that were run earlier.

4. Click the test that is listed under the **Test** column.
5. Click the **Report** tab in the **Results** panel, if it is not already displayed.
6. Click the Jira issue link under **Defect tracking**.

   The Jira log-in page is displayed, if you are not already logged into Jira.
7. Enter your username and password.
8. Click **Login**.

   The issue details are displayed in Jira.

   Note: The **Description** field of the Jira issue in Azure DevOps provides additional details of the test that ran in Test Hub such as the name of the test, the person who ran the test, the verdict of the test result, the date and

time of the test run, the links to the test report that is generated in Test Hub, and the link to the query that is created. You can also view the reports that you attached as a compressed file under the **Attachments** field in the Jira issue.

Results

You have viewed the details in Jira of the Jira issue that you created in Test Hub.

Managing the Jira configuration

You can manage the Jira configuration that you set in HCL DevOps Test Hub (Test Hub) when you want to modify the project configured in Jira or when you want to delete the existing Jira configuration.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned the role of a *Project Owner* in the project. See Managing access to the server projects on page 1220.
- Ensured that you obtained the access and authentication details to Jira from the Jira administrator.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Integration** in the navigation pane.
   The **Integration** page is displayed.
4. Go to the **Configuration details of Jira** within the **Change Management** section, the configured details of the Jira integration are displayed.
5. Perform any of the following actions when you want to update or delete the Jira server configuration:
   - To update the project in Jira that was set previously in the Jira configuration. Go to Step 6 on page 1104.
   - To delete the Jira configuration. Go to Step 7 on page 1104.
6. Perform the following steps to modify the project in Jira:

   a. Click the **Configure** icon  of the Jira integration that you want to manage.
      The **Select project in Jira** dialog is displayed.

   b. Select the project that you want from the list displayed in the **Project** field.

   c. Click **Update** to save the updated project in the Jira server configuration.

7. Perform the following steps to delete the existing Jira server configuration:

a. Click the **Delete** icon  .

The **Delete Jira configuration** dialog is displayed.

b. Click **Confirm**.

The Jira configuration is deleted and a confirmation message is displayed.

Note: The links to the issues that you created for the tests that ran are not deleted from the results panel in Test Hub even after you delete the Jira configuration.

Results

You achieved the following results:

- Updated the project in Jira and saved the Jira configuration.
- Deleted the Jira configuration.

Configuration of Azure DevOps as a change management

You can use Azure DevOps as a change management in HCL DevOps Test Hub (Test Hub) to create work items in the Azure DevOps project for the tests that ran in Test Hub. Links to the Azure DevOps work items are displayed on the **Results** page in Test Hub, and you can then view and track the work items in the Azure DevOps server.

Prerequisites

Before you configure Azure DevOps as a change management in Test Hub, you must have completed the following tasks:

- Ensured that you obtained the access and authentication details to Azure DevOps from the Azure DevOps administrator.
- Ensured that you are assigned the role of a *Project Owner* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.

Supported installations

The following types of Azure DevOps installations are supported that can be configured as a change management system on Test Hub:

- On-premise Azure DevOps installations.
- On-cloud Azure DevOps installations.

Task flow

You can follow the tasks that are listed in the following table when you want to use Azure DevOps as a change management in Test Hub. The table also provides you the links to the information about the tasks.

Task	Go to...
Configure Azure DevOps as a change management.	Configuring Azure DevOps as a change management on page 1106
Create Azure DevOps work items for the test asset.	Creating Azure DevOps work items on page 1107
Track and view the details of the created Azure DevOps work items.	Viewing details of the Azure DevOps work items on page 1112

You can edit the saved configuration, or you can delete the configuration and create a new configuration. For more information, see Managing the Azure DevOps configuration on page 1113.

Configuring Azure DevOps as a change management

You must configure Azure DevOps as a change management in your project in HCL DevOps Test Hub (Test Hub) before you can create work items in the Azure DevOps project for the tests that ran.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Ensured that you obtained the access and authentication details to Azure DevOps from the Azure DevOps administrator.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Integration** in the navigation pane.
   The **Integration** page is displayed.
4. Click the **Add Change Management Integration** option, and then select **Azure DevOps** option from the list.

   The **Configure Azure DevOps** dialog is displayed.
5. You can perform the following actions:

   a. Enter the URL of **Azure DevOps**.

   Note: You must enter the URL of the Azure DevOps in one of the following formats:

- https://dev.azure.com/{Organization}
- https://{Organization}.visualstudio.com
- https://{Hostname}:{Port}/{Collection}

  The URL contains information such as the server name (hostname, port, or domain name) and organization name.

b. Click **Next**, and then paste the copied personal access token.

Note: If you do not have your personal access token, you can click **Generate** to generate your personal access token to authorize full access or access to the work items.

c. Click **Next**, and then select an **Azure DevOps** project from the drop-down list.

d. Click **Add**.
The configuration is added and saved within the **Change Management** section under **Configure details of Azure DevOps.**

Results

You configured Azure DevOps as a change management in Test Hub.

What to do next

You can create work items for the test results that are available in your project in Test Hub. See Creating Azure DevOps work items on page 1107.

Creating Azure DevOps work items

You can create Azure DevOps work items for the tests that ran in HCL DevOps Test Hub (Test Hub). The issues are created in the project in Azure DevOps that you configured in Test Hub.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Configured Azure DevOps as a change management system in Test Hub for your project. See Configuring Azure DevOps as a change management on page 1106.
- Extended the CA list. See Extending the trusted certificate authority list on Linux on page 164 or Extending the trusted certificate authority list on Windows on page 164.
- Configured a test run. See Test run configurations on page 610.
- Viewed the results of the completed test runs on the **Results** page. See Test results on page 983.

About this task

When you want to create a Azure DevOps work item on Test Hub, the fields that display in the **Create a work item** dialog depends on the configuration done by the Azure DevOps administrator.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Analyze > Results**.

   The **Results** page is displayed and all the test assets that were run earlier are displayed in tabs, as listed in the following table:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

   Note: You can click the relevant tab to view the tests that were run earlier.

4. Perform any of the following actions to identify the test asset:
   - Scroll through the list.

     Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

     Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test			

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test by entering any text that is contained in the test asset name in the **Search** field.
- Select the time interval within which the test was run.

  For example, you can select the *Last 24 hours* option to view the tests that were run in the previous 24 hours.
- Select the range of dates within which the test was run.

  You can click the calendar icon to select the date in the first image as the start date, and then select the date in the second image as the end date.

Note: You can select the range of dates only when you select the time interval as *Custom*.

- Click the **Filter** icon in the **SUITES** or **TESTS** tab, and then select the test type.

  All the tests based on the selected test type are displayed.

  For example, you can select **API Suite** in the **SUITES** tab to display all the API suites or you can select **Functional Test** in the **TESTS** tab to display all the functional tests.

- Click the **Filter** icon in the **ADVANCED** tab, and then perform the following actions to create a filter query:
  a. Click **New filter**.
  b. Enter a name for the filter.
  c. Select an operator, and then select **Add rule** to add a rule or **Add group** to add a group of rules.
  d. Select or enter the relevant parameters, and then either select or enter the condition and the criteria for the condition.

     You can select a parameter from the following list:
     - ID
     - Type
     - Test
     - Version
     - Result
     - Started on
     - Finish on
     - Duration
     - Status
     - Verdict
     - Labels
     - Locked
     - Repository
     - Branch
     - Text

  e. Click **Save**.

     The filter query is saved and all the tests that match the filter criteria are displayed.

- Perform the following actions in the **ADVANCED** tab to retrieve and apply a previously saved filter query:
  a. Click the **Filter** icon .

     All the filter queries that you created and saved earlier are displayed.

b. Click the filter query that you want to apply.

All the tests that match the filter criteria are displayed.

You have identified the test that you want to use.

5. Perform the following actions on the test that was run to create a work item:

a. Click the **Azure: Create work item** icon that is under the **Actions** column in the row of the test.

Alternatively, you can click the test under the **Test** column to view the results panel, and then click the **Azure: Create work item** icon that is displayed in the panel.

The **Create a work item** dialog is displayed.

b. Perform the actions indicated in the following table for each of the fields:

Field	Description	Action
Project	The name of the Azure DevOps project.	Select the Azure DevOps project from the drop-down list.
Work item type	The type of work item.	Select the work item you want to assign for the test asset.
Title	A title that identifies the work item.	Specify a title for the work item.
Description	A detailed description of the work item.	Enter a description of the work item.

Note: The other fields selected as **Required** fields on the **Azure DevOps** project are displayed in the **Create a work item** dialog.

c. Enter the information for all the required fields, and then click **Create**.

Note: You can create multiple work items for the same test result, if required.

Results

You created work items in the Azure DevOps project for the test results from Test Hub.

The work items are created in the Azure DevOps project and the links to the work items are displayed in the **Details** panel of the test result on the **Results** page.

What to do next

You can perform any of the following tasks:

- View details of the Azure DevOps work items that you created in Azure DevOps for the test. See Viewing details of the Azure DevOps work items on page 1112.
- Modify the configuration that you set for Azure DevOps on Test Hub. See Managing the Azure DevOps configuration on page 1113.

Viewing details of the Azure DevOps work items

After you create a work item in HCL DevOps Test Hub (Test Hub), you can view and track the details of the Azure DevOps work item in Azure DevOps.

Before you begin

- Created Azure DevOps work item against the test results. See Creating Azure DevOps work items on page 1107.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Analyze > Results**.

   The **Results** page is displayed and all the test assets that were run earlier are displayed in tabs, as listed in the following table:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

   Note: You can click the relevant tab to view the tests that were run earlier.

4. Click the test that is listed under the **Test** column.
5. Click the **Report** tab in the **Results** panel, if it is not already displayed.
6. Click the Azure DevOps work item link under **Defect tracking**.

   The Azure DevOps log-in page is displayed.
7. Enter your username and password.
8. Click **Login**.

   The work item details are displayed in Azure DevOps.

Note: The **Description** field of the work item in Azure DevOps provides additional details of the test that ran in Test Hub such as the name of the test, the person who ran the test, the verdict of the test result, the date and time of the test run, the links to the test report that is generated in Test Hub, and the link to the query that is created.

Results

You have viewed the details in Azure DevOps of the Azure DevOps work item that you created in Test Hub.

Managing the Azure DevOps configuration

You can edit or modify the Azure DevOps configuration that you added and saved to the Change management section on HCL DevOps Test Hub (Test Hub). If you do not want to use the existing configuration and want to set up a new Change management configuration, then you can delete the configuration.

Before you begin

You must have completed the following task:

- Been assigned a project *Owner* role.

1. Open the project that you created on Test Hub.
2. Go to **Manage**, and then click **Integration** in the navigation pane.
   The **Integration** page is displayed.
3. Go to the **Configuration details of Azure Devops** within the **Change Management** section, the configured details of the Azure DevOps integration are displayed.
4. Perform any of the following actions:
   - To edit the Azure DevOps configuration. Go to Step 5 on page 1113.
   - To delete the Azure DevOps configuration. Go to Step 6 on page 1114.
5. Follow the step to edit the Azure DevOps configuration:

   a. Click **Edit** icon of the Azure DevOps integration that you want to manage.
      The **Configure Azure DevOps** dialog is displayed.

   b. Edit the URL by performing the following steps:
      i. Enter the URL of **Azure DevOps**.
      ii. Click **Next**.
      iii. Paste the copied personal access token or click **Generate** to generate your personal access token.
      iv. Click **Add**.

      The URL is updated and saved.

   c. Edit the project by performing the following steps:

i. Select an **Azure DevOps** project from the drop-down list.
ii. Click **Add**.

The project is updated and saved.

6. Follow these steps to delete the Azure DevOps configuration:

a. Click the **Delete** icon .
The **Delete Azure DevOps configuration** dialog is displayed.

b. Click **Delete**.
The Azure DevOps configuration is deleted.

Results

You achieved the following results:

- Edited the Azure DevOps configuration and saved the configuration.
- Deleted the Azure DevOps configuration.

Configuration of DevOps Plan as a change management

You can use HCL DevOps Plan (Plan) as a change management to create work items after you run tests on HCL DevOps Test Hub (Test Hub). You can then view and track the work items in Plan.

Prerequisites

Before you configure Plan on Test Hub, you must have completed the following tasks:

- Ensured that you obtained the access details and authentication details of Plan from the Plan administrator.
- Ensured that you are assigned the role of a *Project Owner* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.

Task flow

You can follow the tasks that are listed in the following table when you want to use Plan as a change management on Test Hub. The table also provides links to the information about the tasks.

	Task	Go to...
1	Configure Plan as a change management.	Configuring DevOps Plan as a change management on page 1115
2	Create work items in Plan for the tests that ran on Test Hub.	Creating work items on page 1116

	Task	Go to...
3	View and track the details of the created work items.	Viewing details of the work items on page 1121

You can manage the Plan settings that you configured in Test Hub. You can edit the saved configuration or you can delete the configuration. For more information, see Managing the DevOps Plan configuration on page 1122.

Configuring DevOps Plan as a change management

You can configure HCL DevOps Plan (Plan) as a change management system in your project to create work items from Test Hub in Plan.

Before you begin

You must have completed the following tasks:

- Read about the prerequisite tasks in Prerequisites on page 1114.
- Ensured that you are assigned the role of a *Project Owner* in the project. See Managing access to the server projects on page 1220.
- Ensured that you obtained the access details and authentication details of Plan administrator.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Integration** in the navigation pane.
   The **Integration** page is displayed.
4. Click **Add Change Management Integration** option within the **Change Management** section.
5. Select the **DevOps Plan** option from the list.

   The **Configure DevOps Plan application** dialog is displayed.
6. Configure the Plan application by performing the following steps:

   a. Enter the URL of the Plan server.

   Note: You can enter the URL of Plan in any of the following formats:
   - https://*IP_address*:*port*/
   - https://*hostname*/

Where	Is
IP_address	The IP address of the server on which Plan is installed.

Where	Is
port	The port that is used to communicate with Plan .
hostname	The hostname of the computer on which Plan is installed.

b. Enter the ID for the Plan Teamspace.

c. Enter the name of the Plan application.

d. Click **Add**.

The configuration is added and saved within the **Change Management** section under the **Configuration details of DevOps Plan**.

Results

You have configured Plan as a change management in your project on Test Hub.

What to do next

You can perform any of the following tasks:

- Create work items in Plan for the tests that you ran on Test Hub. See Creating work items on page 1116.
- Modify the configuration that you set for Plan on Test Hub. See Managing the DevOps Plan configuration on page 1122.

Creating work items

You can create work items in HCL DevOps Plan (Plan) for the tests that ran on HCL DevOps Test Hub (Test Hub). For example, you might want to create a work item in Plan when the test results differ from the expected results.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Configured Plan as a change management in Test Hub. See Configuring DevOps Plan as a change management on page 1115.
- Configured applications in Plan.
- Configured a test run. See Test run configurations on page 610.
- Viewed the results of the completed test runs on the **Results** page. See Test results on page 983.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

The **Overview** page of the project is displayed.

3. Click **Analyze > Results**.

   The **Results** page is displayed and all the test assets that were run earlier are displayed in tabs, as listed in the following table:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

   Note: You can click the relevant tab to view the tests that were run earlier.

4. Perform any of the following actions to identify the test asset:

   - Scroll through the list.

     Tip: You can hover over the icon in the **Type** column to know the type of the test asset.

     Note: You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	API test		✓	✓
	Functional test		✓	✓
	Performance test		✓	✓
	AFT Suite	✓		✓
	API Suite	✓		✓

Icon	Represents	Listed in the SUITES tab	Listed in the TESTS tab	Listed in the ADVANCED tab
	Compound Test	✓		✓
	JMeter Test	✓		✓
	JUnit Test	✓		✓
	Postman test	✓		✓
	Rate Schedule	✓		✓
	Selenium test	✓		✓
	Server Suite asset	✓		✓
	Server Test asset		✓	✓
	VU Schedule	✓		✓

- Search for the test by entering any text that is contained in the test asset name in the **Search** field.
- Select the time interval within which the test was run.

  For example, you can select the *Last 24 hours* option to view the tests that were run in the previous 24 hours.
- Select the range of dates within which the test was run.

  You can click the calendar icon to select the date in the first image as the start date, and then select the date in the second image as the end date.

  Note: You can select the range of dates only when you select the time interval as *Custom*.
- Click the **Filter** icon in the **SUITES** or **TESTS** tab, and then select the test type.

  All the tests based on the selected test type are displayed.

  For example, you can select **API Suite** in the **SUITES** tab to display all the API suites or you can select **Functional Test** in the **TESTS** tab to display all the functional tests.

- Click the **Filter** icon  in the **ADVANCED** tab, and then perform the following actions to create a filter query:
    a. Click **New filter**.
    b. Enter a name for the filter.
    c. Select an operator, and then select **Add rule** to add a rule or **Add group** to add a group of rules.
    d. Select or enter the relevant parameters, and then either select or enter the condition and the criteria for the condition.

       You can select a parameter from the following list:
       - ID
       - Type
       - Test
       - Version
       - Result
       - Started on
       - Finish on
       - Duration
       - Status
       - Verdict
       - Labels
       - Locked
       - Repository
       - Branch
       - Text

    e. Click **Save**.

       The filter query is saved and all the tests that match the filter criteria are displayed.

- Perform the following actions in the **ADVANCED** tab to retrieve and apply a previously saved filter query:
    a. Click the **Filter** icon  .

       All the filter queries that you created and saved earlier are displayed.

    b. Click the filter query that you want to apply.

       All the tests that match the filter criteria are displayed.

You have identified the test that you want to use.

5. Perform the following actions on the identified test to create a work item in Plan:

    a. Click the **DevOps Plan: Create a work item** icon  that is under the **Actions** column in the row of the test.

    Alternatively, you can click the test under the **Test** column to view the results panel, and then click the **DevOps Plan: Create a work item** icon  that is displayed in the panel.

The **Create a work item in DevOps Plan** dialog is displayed.

b. Select the application from the list in the **Application** field.

Remember: The application that you select must be the application in Plan for which you ran the test.

c. Enter the username in the **Username** field.

Note: The username is authorized to access the Plan server.

d. Enter the password in the **Password** field to authenticate the username.

Note: After the credentials that you provided in the **Username** and **Password** fields are authorized by the Plan server, you need not enter these values for subsequent connections to create a work item.

e. Click **Next**.

The dialog displays the URL of the Plan server and the **Application** that you selected.

f. Select the work item type as *WorkItem*.

g. Click **Next**.

h. Select the project from the list in the **Project** field.

i. Select the type of the work item from the list in the **Type** field.

j. Enter text that is to be used as the title of the work item in the **Title** field. The text entered is displayed as the title in the work item in Plan.

You can view the details of the test run in the **Description** field, which is appended to the work item that is created for the test run in Plan.

k. Click **Create**.

A work item is created in Plan and a link to the work item is created in the **Report** panel of the **Results** page.

Notes:

- You cannot delete the work item that is created in Plan from Test Hub.
- You can create multiple work items for the same test result, if required.

Results

You have created a work item in Plan from the **Results** page for the test that ran on Test Hub.

What to do next

You can perform any of the following tasks:

- View details of the work items that you created in Plan for the test. See Viewing details of the work items on page 1121.
- Modify the configuration that you set for Plan on Test Hub. See Managing the DevOps Plan configuration on page 1122.

Viewing details of the work items

After you create a work item in HCL DevOps Plan (Plan) from HCL DevOps Test Hub (Test Hub), you can view and track the details of the work item in Plan.

Before you begin

You must have completed the following tasks:

- Completed a test run and the test results are available on the **Results** page.
- Created a work item in Plan against the test results. See Creating work items on page 1116.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Analyze > Results**.

   The **Results** page is displayed and all the test assets that were run earlier are displayed in tabs, as listed in the following table:

Tab	Description
SUITES	Lists all suites, Compound Tests, JMeter tests, JUnit tests, Postman tests, Rate Schedules, Selenium tests, or VU Schedules that are in the selected branch of the repository.
TESTS	Lists all API tests, functional tests, performance tests, or server test assets that are in the selected branch of the repository.
ADVANCED	Lists all assets that are displayed when custom filters are applied for assets in the selected branch of the repository.

   Note: You can click the relevant tab to view the tests that were run earlier.

4. Click the test that is listed under the **Test** column.

   The **Results** panel for the test is displayed.

5. Click the **Report** tab in the **Results** panel, if it is not already displayed.
6. Click the work item link under **Defect tracking**.

   The Plan log-in page is displayed.
7. Enter your username and password.
8. Click **Sign In**.

   The work item details are displayed in Plan.

Note: The **Description** field of the work item provides additional details of the test that ran in Test Hub such as the name of the test, the person who ran the test, the verdict of the test result, the date and time of the test run, the links to the test report that is generated in Test Hub, and the link to the query that is created.

Results

You have viewed the details in Plan of the work item that you created in Test Hub.

Managing the DevOps Plan configuration

You can manage the HCL DevOps Plan (Plan) configuration that you set in HCL DevOps Test Hub (Test Hub) when you want to edit the configuration or when you want to delete the existing configuration.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned the role of a *Project Owner* in the project. See Managing access to the server projects on page 1220.
- Ensured that you obtained the access details and authentication details of Plan from the Plan administrator.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Integration** in the navigation pane.
   The **Integration** page is displayed.
4. Click **Add Change Management Integration** option within the **Change Management** section.
5. Perform any of the following actions when you want to edit or delete the Plan configuration:
   - To edit the Plan configuration. Go to Step 6 on page 1122.
   - To delete the Plan configuration. Go to Step 7 on page 1123.
6. Perform the following steps to edit the Plan application that is configured:

   a. Click **Edit** icon .
      The **Modify DevOps Plan configuration** dialog is displayed.

b. Edit the URL or change it to the one that you want to use in the **URL** field, and then enter the application that you want to use.

c. Click **Update**.

The Plan configuration is updated and a confirmation message is displayed.

d. Go to Results on page 1123.

7. Perform the following steps to delete the existing Plan configuration:

a. Click the **Delete** icon .

The **Delete DevOps Plan configuration** dialog is displayed.

b. Click **Delete**.

The Plan configuration is deleted and a confirmation message is displayed.

c. Go to Results on page 1123.

Note: The links to the records that you created for the tests that you ran are not deleted from the results panel in Test Hub even after you delete the Plan configuration.

Results

You achieved the following results:

- Edited and saved the Plan configuration.
- Deleted the Plan configuration.

Integrations with other applications

You can integrate certain applications with HCL DevOps Test Hub (Test Hub) to run tests, view the test results, and create defects.

Integration plugin compatibility matrix

You can find information about the versions of the integration plugin that are compatible with HCL DevOps Test Hub (Test Hub).

The following table lists the versions of the integration plugin that are required to integrate Test Hub with DevOps Deploy, IBM® Engineering Test Management, and Jenkins.

Note: You must download the required version of the integration plugin from My HCLSoftware (MHS) based on the existing version of Test Hub. You can then integrate Test Hub with HCL DevOps Deploy, IBM® Engineering Test Management, and Jenkins.

Test Hub	Engineering Test Management plugin	Jenkins plugin	HCL DevOps Deploy plugin
12.0.0	HCL_DTHub_v1200_ETM_ADAPTER	DTHub-Jenkins-HCL-7.11	DTHub-DD-HCL-5.2 DTHub-DD-HCL-5.2
11.0.9	HCL_DTHub_v1109_ETM_ADAPTER	DTHub-Jenkins-HCL-7.8	DTHub-DD-HCL-5.1 DTHub-DD-HCL-5.1
11.0.8	HCL_DTHub_v1108_ETM_ADAPTER	DTHub-Jenkins-HCL-7.8	DTHub-DD-HCL-5.0 DTHub-DD-HCL-5.0
11.0.7	HCL_DTHub_v1107_ETM_ADAPTER	DTHub-Jenkins-HCL-7.6	DTHub-DD-HCL-5.0 DTHub-DD-HCL-5.0
11.0.6	HCL_DT_HUB_1106_ETM	HCL_DT_HUB_1106_Jenkins	HCL_DT_HUB_1106_Launch HCL_DT_HUB_1106_UCD
11.0.5	HCL_DT_HUB_1105_ETM	HCL_DT_HUB_1105_Jenkins	HCL_DT_HUB_1105_Launch HCL_DT_HUB_1105_UCD
11.0.4	HCL_DT_HUB_1104_ETM	HCL_DT_HUB_1104_Jenkins	HCL_DT_HUB_1104_Launch HCL_DT_HUB_1104_UCD
11.0.3	HCL_DT_HUB_1103_ETM	HCL_DT_HUB_1103_Jenkins	HCL_DT_HUB_1103_Launch HCL_DT_HUB_1103_UCD
11.0.2	HCL_DT_HUB_1102_ETM	HCL_DT_HUB_1102_Jenkins	HCL_DT_HUB_1102_Launch HCL_DT_HUB_1102_UCD

Test Hub	Engineering Test Management plugin	Jenkins plugin	HCL DevOps Deploy plugin
11.0.1	HCL_DT_HUB_1101_ETM	HCL_DT_HUB_1101_Jenkins	HCL_DT_HUB_1101_Launch HCL_DT_HUB_1101_UCD
11.0.0	HCL_DT_HUB_110_ETM	HCL_DT_HUB_110_Jenkins	HCL_DT_HUB_110_Launch HCL_DT_HUB_1101_UCD
10.5.4	HCL_OT_SERVER_1054_ETM	HCL_OT_SERVER_1054_Jenkins	HCL_OT_SERVER_1054_Launch HCL_OT_SERVER_1054_UCD
10.5.3	HCL_OT_SERVER_1053_ETM	HCL_OT_SERVER_1053_Jenkins	HCL_OT_SERVER_1053_Launch HCL_OT_SERVER_1053_UCD
10.5.2	HCL_OT_SERVER_1053_ETM	HCL_OT_SERVER_1052_Jenkins	HCL_OT_SERVER_1052_Launch HCL_OT_SERVER_1052_UCD
10.5.1	HCL_OT_SERVER_1051_ETM	HCL_OT_SERVER_1051_Jenkins	HCL_OT_SERVER_1051_Launch HCL_OT_SERVER_1051_UCD
10.5.0	HCL_OT_SERVER_1050_ETM	HOT-SERVER-Jenkins-6.3	HOT-SERVER-UCD-3.3
10.2.3	HCL_OT_SERVER_1023_ETM	HOT-SERVER-Jenkins-6.1	HOT-SERVER-UCD-3.2
10.2.2	HCL_OT_SERVER_1022_ETM	HOT-SERVER-Jenkins-6.0	HOT-SERVER-UCD-3.0
10.2.1	HCL_OT_SERVER_1021_ETM	HOT-SERVER-Jenkins-5.0	HOT-SERVER-UCD-3.0
10.2.0	NA	HOT-SERVER-Jenkins-5.0	HOT-SERVER-UCD-3.0

Test Hub	Engineering Test Management plugin	Jenkins plugin	HCL DevOps Deploy plugin
10.1.3	NA	HOT-SERVER-Jenkins-4.0	HOT-SERVER-UCD-2.0
10.1.2	NA	HOT-SERVER-Jenkins-4.0	HOT-SERVER-UCD-2.0
10.1.1	NA	HOT-SERVER-Jenkins-3.0	HOT-SERVER-UCD-1.0
10.1.0	NA	HOT-SERVER-Jenkins-2.0	HOT-SERVER-UCD-1.0

Integration with Azure DevOps

When you use Azure DevOps for continuous integration and continuous development of your application, you can run tests created for your application and available in a project on HCL DevOps Test Hub (Test Hub), in Azure DevOps pipelines by using the HCL DevOps Test Hub extension.

Overview

You can use the HCL DevOps Test Hub extension to integrate Test Hub with Azure DevOps.

The HCL DevOps Test Hub extension enables you to select any type of test available for your project in Test Hub that you can add to your task for the job in the Azure DevOps pipelines.

You can now follow the tasks listed in the task flow table to integrate Test Hub with Azure DevOps. See Task flow for integrating Azure DevOps on page 1126.

Task flow for integrating DevOps Test Hub with Azure DevOps

The table shows the task flow for integrating HCL DevOps Test Hub (Test Hub) with Azure DevOps by using the HCL DevOps Test Hub extension. You must perform these tasks in sequence as listed in the following table. The table also provides you the links to the information about the tasks.

S.N.	Tasks	More information
1	Install Test Hub.	Installation of DevOps Test Hub on page 115
2	Create tests in the desktop clients for the application that you want to test.	• DevOps Test Performance documentation • DevOps Test UI documentation • DevOps Test Integrations and APIs documentation

S.N.	Tasks	More information
3	Create a project on Test Hub and add the test assets that are created in the desktop clients to your project.	Managing DevOps Test Hub projects on page 1209
4	Create an organization and a project in Azure DevOps for running jobs in Azure DevOps pipelines.	Creating an organization
5	Access the **Visual Studio Marketplace** portal and search for the latest version of the HCL DevOps Test Hub extension.	Visual Studio Marketplace
6	Install the latest version of the HCL DevOps Test Hub extension.	Installing the HCL DevOps Test Hub extension on page 1127
7	Extend the trust certificate if you have used an internal CA certificate.	Certificate authority: Importing and extending lists on page 161
8	Run tests in an Azure DevOps pipeline.	Running tests in an Azure DevOps Pipeline on page 1128

Related information

Integration with Azure DevOps on page 1126

Installing the HCL DevOps Test Hub extension

You must install the latest version of the HCL DevOps Test Hub extension in your Azure DevOps organization before you can use the extension for running your application tests in an Azure DevOps pipeline.

Before you begin

You must have access to the **Visual Studio Marketplace** portal to install the latest version of the HCL DevOps Test Hub extension.

About this task

After you install the latest version of the HCL DevOps Test Hub extension from the **Visual Studio Marketplace** portal in your Azure DevOps organization, you can select the tests that you want to run for your application in an Azure DevOps pipeline by using the HCL DevOps Test Hub extension.

1. Log in to the **Visual Studio Marketplace** portal, if you are not already logged in.
2. Click the **Azure DevOps** tab.
3. Search for the **HCL DevOps Test Hub extension**.
4. Click **HCL DevOps Test Hub extension**.
5. Click **Get it free**.
   Result

   The **Visual Studio Marketplace** portal for the HCL DevOps Test Hub extension is displayed.
6. Select the organization where you want to run your test from the **Select an Azure DevOps Organization** list.
7. Click **Install**.
   Result

   The installation is completed.
8. Click **Proceed to organization**.
   Result

   The **Organization** page in Azure DevOps is displayed.
9. Click **Organization settings > Extensions**.
   Result

   The HCL DevOps Test Hub extension is displayed as an installed extension.

Results

You have installed the HCL DevOps Test Hub extension in your Azure DevOps organization.

What to do next

You can run tests that are available in Test Hub as a job in an Azure DevOps pipeline. See Running tests in an Azure DevOps Pipeline on page 1128.

Running DevOps Test Hub tests in an Azure DevOps Pipeline

After you install the HCL DevOps Test Hub extension in your organization, you can run tests that are available in HCL DevOps Test Hub (Test Hub) as a job in Azure DevOps pipelines.

Before you begin

You must have completed the following tasks:

- Added the tests that you created in the desktop clients for your application to the project on Test Hub.
- Installed the latest version of the HCL DevOps Test Hub extension in your organization. See Installing the HCL DevOps Test Hub extension on page 1127.
- Installed an agent in your pipeline. See Azure Pipelines agents.
- Extended the CA list. See Extending the trusted certificate authority list on Linux on page 164 or Extending the trusted certificate authority list on Windows on page 164.
- Generated an offline user token from Test Hub. See Generating an offline token on page 1194.

- Created test cases under test plans if you want to view results of the test runs on the Test Plans dashboard.
- Created encrypted secrets or environment variables for your personal access token (PAT), if you want to use an encrypted PAT.

About this task

After you add the HCL DevOps Test Hub extension in your Azure DevOps organization, you can use an existing pipeline or create a new one to add Test Hub test tasks. You can install an agent or use the one that you installed in your default agent pool. You can add Test Hub tests as tasks to your agent job, configure the task, and then run the task in the Azure DevOps pipeline.

If you have created test cases under test plans in a project on Azure DevOps, you must provide the details of the Azure DevOps URL, test plan, test case, and your PAT while you configure the test job in a pipeline so that you can view the results of the test run on your Test Plan dashboard.

1. Open your **Organization** page in Azure DevOps and perform the following steps:
    a. Click the project you want to use.
    b. Initialize the repository by performing the following steps:
        i. Click **Repos** from the left pane.
        ii. Click **Initialize** from the **Initialize with a README or gitignore** section.

        Note: Select the **Add a README** checkbox if it is not selected.

    c. Click **Pipelines** from the left pane.
    d. Click **Create Pipeline**.
    e. Click **Use the classic editor** to create a pipeline without YAML.
    f. Verify the project, repository, and branch for manual and scheduled builds, and then click **Continue**.
    g. Click **Empty job**.
2. Select **Pipeline** and complete the following steps:
    a. Change the name for the build pipeline if required.
    b. Select the **Agent pool** for your build pipeline.

    You can use the agent from the default agent pool or use the one you have installed.

    c. Select the **Agent Specification** for the agent if required.
3. Add a task to the agent job by completing the following steps:

a. Click the **Add Task** icon + for the agent job.
Result

The **Add tasks** pane is displayed.

b. Search for the HCL tasks defined in the HCL DevOps Test Hub extension.
Result

The tasks that you can select are displayed.

c. Select the **HCL DevOps Test Hub Task**, and then click **Add** to add the task to the agent job.
Result

The selected task is added to the agent job and it is displayed with a warning that some settings require attention. You must configure the settings mentioned in Step 4 on page 1130.

You can also remove the tasks that are not required in your job. Select the tasks in the list that you want to remove. You can then right-click the tasks, and click **Remove selected task(s)** to remove them.

4. Configure the settings by performing the following steps:

a. Select the task version from the list if required.

b. Follow the action for the task by referring to the following table:

Note: All mandatory fields are marked with an asterisk (*) in the UI.

Field	Description	Action
Display name	The default name of the task. The default name that is displayed is **HCL DevOps Test Hub Task**.	Enter a name for the task that helps you to identify the task by changing the default name.
HCL DevOps Test Hub Task service	The details of Test Hub to set up a connection with Azure DevOps.	Select the service connection from the drop-down list. If you are selecting the Test Hub service connection for the first time, you must click **New** to add the following details to add Test Hub service connection, and then save the connection details:

Field	Description	Action
connection		▪ **Server URL** - Enter the URL of Test Hub. The format of the URL is as follows: `https://hostname.` ▪ **Offline Token** - Enter the offline token that you generated from Test Hub. ▪ **Service connection name** - Enter a service connection name. ▪ **Description (optional)** - Enter the details of the service connection if required. ▪ Optionally, select the **Grant access permission to all pipelines** checkbox if required. You can save the service connection details. The service connection is available for selection from the Test Hub service connection drop-down list. **Note:** You can edit or delete the service connection that you added if required. Click **Manage**, and then select the service connection from the **Service connections** list to open the service connection. You can click **Edit** to edit the service connection details. If you want to delete an existing service connection details, click the **Vertical ellipsis** icon, and then click **Delete** to delete the service connection details. Optionally, you can also create, edit, and delete a service connection from your Azure DevOps project dashboard. Open your project, click **Project settings > Service connections**, and then perform any of the following tasks: ▪ Create a new service connection. ▪ Edit or delete an existing service connection that you already added.
Team Space Name	The name of the team space on Test Hub that contains your project with the test assets.	Enter the name of the team space that contains the project. **Note:** The license on the team space must be configured and you must be a member of that team space.
Project	The name of the project containing the test assets	Enter the name of the project from the available names in the team space.

Field	Description	Action
	in your team space on Test Hub.	**Note:** **Note:** The **Project** drop-down list displays the projects where you are an *Owner* or a *member* of Test Hub. You must be an *Owner* or a *Tester* of the project that is available in the team spaces to run the tests from the Jenkins server. Azure DevOps pipeline.
Branch	The name of the branch of the repository in your project on Test Hub that contains the test assets.	Enter the branch where the test assets are stored.
Repository Link	The URL or link to the repository in your project that contains the test assets.	Enter the repository path that is configured for the project that contains the test to run.
File Path	The path of the test in the repository of your project.	Enter the path of the test in Test Hub that you want to run.
API Test Environment	The environment defined in the API Suite or test to use in the test run.	Enter the name of the environment that is configured in the test asset for the test created in Test Integrations and APIs. **Note:** This field is applicable only if you are running an API suite test.
Variables	The variable name and its value.	Enter the name of the variable and its value if your test requires variables during the test run time. You must enter the variables in the following format: `name_of_the_variable=value_of_the_varibale` You can add multiple variables to the test run separated by a semicolon. For example, *varname1=value1;varname2=value2*

Field	Description	Action
Datasets	The name with path to the dataset that must be used during the test run.	Enter the path to the dataset if you want to replace the values of the dataset during a test run. You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the **Datasets** field, you must also include the path to the dataset. The path must be in the following format: `/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv` **Note:** You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon.
Labels	The option to add labels to the test results when the test run is complete.	Enter the labels to add to test results when the test run is complete. You can add multiple labels to a test result separated by a comma. For example, `label1, label2` After the test run is complete, then the values provided in the **Labels** field are displayed on the **Results** page of Test Hub.
Result Properties	The option to add additional details to the test results when the test run is complete.	Enter a key-value pair of the property to be added to test results. You can add multiple properties separated by a semicolon (;). For example, `buildID=DTHub202609070223-2f3d81f80f;executedFrom=AzurePlugin;` After the test run is complete, the values added here are displayed in the **Properties** column on the **Results** page of Test Hub. Additionally, for server test assets, the properties are also displayed on the **Functional Report** page and **Test Log** page.
Security Group	The name of the security group that contains the	Enter the name of the security group that contains the secrets to be used in the API Suite or test.

Field	Description	Action
	secrets to be used in certain API Suites or tests.	**Note:** This field is applicable only if you are running an API suite test.
Azure DevOps Project URL	The URL of the test project in the organization on the Azure DevOps server. **Note:** You must enter the details for this option if you have created test cases under test plans in your Azure project and want to view the test results on the Test Plans Dashboard.	Enter the URL of the test project in the organization on the Azure DevOps server in the following format: https://<host>/<orgname>/<projectName> You must use this option with the following options: ▪ Azure DevOps PAT ▪ Test Plan Name ▪ Test Case Name
Azure DevOps PAT	The personal access token (PAT) of the Azure server where the test project that contains the test plan is hosted. You can also enter the variable for the encrypted PAT that you create in Azure DevOps. **Note:** The token must have read and write access.	Enter your personal access token. You must use this option with the following options: ▪ Azure DevOps Project URL ▪ Test Plan Name ▪ Test Case Name
Test Plan Name	The name of the test plan in your Azure project.	Enter the name of the test plan. You must use this option with the following options:

Field	Description	Action
		▪ Azure DevOps Project URL ▪ Azure DevOps PAT ▪ Test Case Name
Test Case Name	The name of the test case under the test plan in your Azure project.	Enter the name of the test case. You must use this option with the following options: ▪ Azure DevOps Project URL ▪ Azure DevOps PAT ▪ Test Plan Name
Fail the job if the test or any of the containing tests fail	The flag to indicate whether the job should be marked as failed when the test or any child test within the test fails.	Select the checkbox.
Export JUnit Report	The location to store the JUnit report of the test.	Enter the path to store the JUnit report that is generated for the test in the XML format. You must specify the complete path to the directory including the file name of the report. For example, `C:/Users/Documents/Reports/junit_test_report.xml`

c. Expand **Control Options** and configure the settings for your task if required.

d. Expand **Output Variables** and configure the settings for your task if required.

5. Select from the following options:

a. Click **Save** to save the configured settings for the task.



Note: The task is not queued for a run.

You can save the task to a build pipeline and opt to run the build at a later time.

b. Click **Save & queue** to save the configurations and queue the run in the pipeline.
Result

The **Run pipeline** dialog is displayed.

6. Complete the following steps:
    a. Enter a comment for the test in the **Save comment** field.
    b. Select the agent that you configured for the test from the **Agent pool** list.
    c. Select the agent specification from the **Agent Specification** list if required.
    d. Select the branch from the **Branch/tag** list.
    e. Add the variables and demands for the task run from the **Advanced Options** pane if required.
    f. Select the **Enable system diagnostics** checkbox for a detailed log view.
    g. Click **Save and run**.
    Result

    The `pipeline summary` page displays the progress of the job run.

Results

You have run the tests for the application that you are testing in the Azure DevOps pipeline.

What to do next

You can perform any of the following actions in Azure DevOps:

- View the test results on the **Task** page by clicking the task.

  The **Reports information** section on the **Task** page displays the names of the report along with its corresponding URLs. The report URLs are the Test Hub URLs where the reports are stored. You can access the report URLs to view the test run information at any point of time.
- View the task logs from the **pipeline summary** page for the job you ran.
- View the test results on the Test Plans Dashboard if you created test cases under test plans in your Azure project.

You can perform any of the following actions in Test Hub:

- View the test results of the test that you ran from Azure DevOps on the **Results** page.
- View the logs and reports of the test that you ran from Azure DevOps by clicking the result panel of the test asset or resource, and then click the log or the report you want to view.

Integration with HCL DevOps Deploy

You can integrate HCL DevOps Test Hub (Test Hub) with HCL DevOps Deploy (Deploy) to run test assets as a **Process** from Deploy.

When you use Deploy for automating the deployments of your applications to an environment during the development cycle, you can create test assets for your application either in Test Hub or desktop clients. After you configure the project in Test Hub, you can then run those tests assets from Deploy.

You must use the Test Hub Deploy plugin to integrate Test Hub with Deploy. Integrating Test Hub with Deploy automates the process of test runs. If you have many tests to run at regular intervals, you can use Deploy to initiate test runs automatically.

The following table lists the tasks that you must perform to run performance test assets from Deploy:

Tasks	Go to...
Create any or all the following types of test assets in desktop clients to test your application: • API test assets • Functional test assets • Performance test assets	• DevOps Test Performance documentation • DevOps Test UI documentation • DevOps Test Integrations and APIs documentation
Create a project, and then configure the repository in Test Hub.	Working with projects in a team space on page 1210
Install Deploy with at least one agent.	HCL® Launch Documentation. System Requirements on page 99 for information about specific versions of Deploy requirements.
Install the DevOps Test Hub Deploy plugin on Deploy.	Installing the plugin on HCL DevOps Deploy on page 1137.
Create a **Component** in Deploy.	Creating a component in HCL DevOps Deploy on page 1140.
Create a **Process** for the component in Deploy.	Creating a process in HCL DevOps Deploy on page 1140.
Configure the **Process** that you created for the component in Deploy.	Configuring the process in HCL DevOps Deploy on page 1141.
Create a **Resource** in Deploy.	Creating a resource in HCL DevOps Deploy on page 1146.
Configure the **Resource** in Deploy.	Configuring the resource in HCL DevOps Deploy on page 1146.
Create an **Application** in Deploy.	Creating an application in HCL DevOps Deploy on page 1148.
Configure the **Application** that you use to run performance test assets in Deploy.	Configuring the application HCL DevOps Deploy on page 1148.
Run the test assets from Deploy.	Running tests from HCL DevOps Deploy on page 1150.

Installing the plugin on HCL DevOps Deploy

You must install the DevOps Test Hub Deploy plugin to run test assets from HCL DevOps Deploy (Deploy).

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Installed Deploy. For more information about the installation, refer to the HCL DevOps Deploy documentation.
  - Been granted access to Deploy.
  - Downloaded the DevOps Test Hub Deploy plugin from the My HCLSoftware (MHS).

    For more information about specific versions of plug-in, see Integration plugin compatibility matrix on page 1123.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Settings**.
3. Click **Installed Plugins** from the **Automation** pane.
   If you are using Deploy 8.1.1 or older version, then you must click **Automation Plugins** from the **Automation** pane.
4. Click **Load plugin**.
5. Click **Choose File** to locate and open the compressed DevOps Test Hub Deploy plugin file.

   Remember: You must not extract the DevOps Test Hub Deploy plugin compressed file contents.

6. Click **Submit**.
   Result

   The installed plugin is displayed on the **Automation Plugins** tab.

Results

You have installed the DevOps Test Hub Deploy plugin on Deploy.

What to do next

You must create a **Component** in Deploy. See Creating a component in HCL DevOps Deploy on page 1140.

Related information

Installing plug-ins from the Web UI of Deploy

Using a custom trust store for HCL DevOps Deploy

You can use a custom trust store in the DevOps Test Hub Deploy plugin file to establish a trusted and secure connection between the HCL DevOps Deploy (Deploy) server and HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

You must have completed the following tasks:

- Saved the Certificate Authority (CA) assigned to Test Hub.

  Note: You can contact the administrator of Test Hub if you do not have a copy of the CA.

- Installed Java on the computer from where you access the Test Hub UI.

About this task

If the SSL certificate assigned to Test Hub is signed by an internal Certified Authority (CA), then you must download and import the CA certificate to a custom trust store. You can then use the custom trust store in the Test Hub test process step to establish a trusted and secure connection between the Deploy server and Test Hub.

Note: If the internal CA certificate is already imported to the default trust store that is used by the Deploy server, you need not use a custom trust store.

1. Locate the default trust store file (`cacerts` file) from the JRE directory.
   For example: `C:\Program Files\Java\jdk-18.0.1.1\lib\security`
2. Copy the `cacerts` file to a location of your choice on your computer.
3. Place the CA assigned to Test Hub in the same location where you saved the `cacerts` file.
4. Run the following command from the command-line interface or terminal to import the CA of Test Hub to custom trust store:

   ```
   keytool -import -trustcacerts -file {path to the downloaded CA with the file extension} -alias {label
    for the certificate} -keystore {path to the trust store cacerts file}
   ```

   For example,

   ```
   keytool -import -trustcacerts -file C:\Users\Desktop\cert\ca-master-hcl.crt -alias "otsca" -keystore C:
   \Users\Desktop\cert\cacerts
   ```

   Note: The default password of the trust store is *changeit*. It remains the same for the custom trust store. If you want to change the password, you can run the following command, and then enter the new password:

   ```
   keytool -storepasswd -keystore {path to the trust store cacerts file}
   ```

   For example, `keytool -storepasswd -keystore C:\Users\Desktop\cert\cacerts`

Results

You have successfully imported the downloaded CA to the custom trust store.

What to do next

You can then run the tests that are available in your Test Hub project from the Deploy server.

Creating a component in HCL DevOps Deploy

You must create a component to include artifacts and processes. The artifacts include runnable files, images, databases, configuration instructions. Whereas the processes define the activities that components can perform.

Before you begin

- You must be familiar with working with Deploy.
- You must have been granted access to Deploy.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Components**, and then click **Create Component**.
3. Enter a name for the component in the **Name** field.
4. Enter the details in the other optional fields based on your requirement, and then click **Save**.
   Result
   The component that you created is displayed.

Results

You have created the component in Deploy.

What to do next

You must create a process for the component in Deploy. See Creating a process in HCL DevOps Deploy on page 1140.

Related information

HCL DevOps Deploy Documentation

Creating a process in HCL DevOps Deploy

You must create a process for the component to include step properties for the test that you want to run from HCL DevOps Deploy (Deploy).

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Been granted access to Deploy.
  - Created a component in Deploy. See Creating a component in HCL DevOps Deploy on page 1140.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Components**.
   Result
   A list of components that are available in Deploy is displayed.
3. Select the component from the list for which you want to create a process.
4. Click the **Processes** tab, and then click **Create Process**.
   Result
   The **Create Process** dialog is displayed.
5. Enter a name for the process in the **Name** field.
6. Select **Operational (No Version Needed)** from **Process Type** drop-down list.
7. Verify the **Default Working Directory** field.

   The **Default Working Directory** field defines the location that the agent uses to run the process. The default value is `${p:resource/work.dir}/${p:component.name}`.

   Where `${p:resource/work.dir}` is the default working directory for the agent and `${p:component.name}` is the name of the component.
8. Click **Save**.
   Result
   The process that you created is listed in the **Processes** tab and the **Design** tab for the process is displayed.

Results

You have created the process for the component in Deploy.

What to do next

You must configure the process in Deploy. See Configuring the process in HCL DevOps Deploy on page 1141.

Related information

HCL DevOps Deploy Documentation

Configuring the process in HCL DevOps Deploy

You must configure the process that you created for the component to organize the steps in the process, specify the properties of the steps, and connect them.

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:

- Been granted access to Deploy.
- Created a component in Deploy. See Creating a component in HCL DevOps Deploy on page 1140.
- Created a process for the component in Deploy. See Creating a process in HCL DevOps Deploy on page 1140.

About this task

When you open any process to configure, the process is displayed in the process editor. The process editor lists the plugins and steps. The required **Start** and **Finish** steps represent the beginning and the end of the process and steps are automatically placed on the design area.

You must provide the values for certain fields in the properties for the selected test step to run tests from Deploy.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Components**.
   Result
   A list of components that are available in Deploy is displayed.
3. Select the component from the list in which you created the process.
4. Click the **Processes** tab.
   Result
   A list of processes that are available for the component is displayed.
5. Select the process from the list that you want to configure.
   Result
   The **Design** tab for the process is displayed.
6. Click **HCL DevOps Test Workbench**, and then **HCL DevOps Test Hub** from the left menu.
7. Drag the **Run HCL DevOps Test Hub test** step, and then drop it into the design area.

   Note: The selected test must be placed between **Start** and **Finish** steps.

8. Specify the properties for the selected test by performing the following steps:
   a. Click the **Edit** icon.
      Result
      The **Edit Properties for Run HCL DevOps Test Hub test** dialog is displayed.
   b. Specify the properties for the selected test step by referring to the following table:

      The following table lists the required fields that you must provide to run the test from Deploy:

Fields	Action
Name	Enter the name for the test step.
HCL DevOps Test Hub URL	Enter the URL of Test Hub. If you do not have the details of the custom trust store, you must append #insecure-skip-tls-verify to the URL. For example, https://10.134-.56.236.nip.io/test#insecure-skip-tls-verify
Offline Token	Enter the offline token that you generated from Test Hub.
Team Space Name	Enter the name of the team space that contains the project. **Note:** The license for the team space must be configured and you must be a member of that team space.
Project Name	Enter the name of the project that are available in the team space.
Repository Link	Enter the URL of the Git repository that you added to your project in Test Hub.
File Path	Enter the path of the test assets that you want to run. You can find the path of the test assets from the **Execution** page in Test Hub

The following table lists the optional fields that you can provide to run the test from Deploy:

Fields	Description
Branch Name	Use this field to enter the name of the branch in the Git repository where you stored test assets.
Datasets	Use this field to enter the path to the dataset if you want to replace the values of the dataset during a test run. You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the **Datasets** field, you must also include the path to the dataset. The path must be in the following format:

Fields	Description
	`/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv` **Note:** You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon.
Environment	Use this field to enter the environment details for the following types of test assets: ▪ APISUITE ▪ APITEST ▪ APISTUB
Labels	Use this field to add labels to test results when the test run is complete. You can add multiple labels to a test result separated by a comma. For example, *label1, label2* After the test run is complete, then the values provided in the **Labels** field are displayed on the **Results** page of Test Hub.
Result Properties	Use this field to enter a key-value pair of the property to be added to test results. You can add multiple properties separated by a semicolon (;). For example, `buildID=DTHub202609070223-2f3d81f80f;executedFrom=AzurePlugin;` After the test run is complete, the values added here are displayed in the **Properties** column on the **Results** page of Test Hub. Additionally, for server test assets, the properties are also displayed on the **Functional Report** page and **Test Log** page.
Security Group	Use this field to enter the name of the security group if you created secrets for your project and to use secrets at test run time. This field is mandatory only if you want to run the following types of test assets: ▪ APISUITE ▪ APITEST ▪ APISTUB
Show Hidden Properties	Select this checkbox to display the additional fields that are hidden. You can configure the following fields if you have the details of a custom trust store:

Fields	Description
	▪ **Custom Trust Store Password**: Use this field to enter the trust store password if you have modified the password while creating the custom trust store. **Note:** If you have not modified the trust store password, you can retain the **Custom Trust Store Password** field blank. ▪ **Custom Trust Store Path**: Use this field to enter the file path of your trust store followed by the file name if Test Hub uses an internal CA certificate and you have imported the certificate to a custom trust store. Also, you can configure the HTTP Proxy to connect to Test Hub by using the following fields: ▪ **HTTP Proxy Host**: Use this field to enter the hostname of the HTTP proxy that you want to connect to Test Hub. ▪ **HTTP Proxy Port**: Use this field to enter the port number where the HTTP proxy is used to listen to both HTTP and HTTPS traffic. ▪ **HTTP Proxy User name**: Use this field to provide the username for the HTTP proxy. ▪ **HTTP Proxy Password**: Use this field to enter a valid password for the user that you specified in the **HTTP Proxy User name** field. **Note:** You can use the HTTP-related fields only when you want to run the following types of test assets: ▪ APISUITE ▪ APITEST ▪ APISTUB
Variables	Use this field to enter the name of the variable and its value if your test requires variables during the test run time. You must enter the variables in the following format: `name_of_the_variable=value_of_the_varibale` You can add multiple variables to the test run separated by a semicolon. For example, *varname1=value1;varname2=value2*

Note: In addition to optional fields, you can also use the following fields from Deploy to configure your test run:

- **Working Directory**
- **Precondition**
- **Post Processing Script**
- **Use Impersonation**
- **Auth Token Restriction**

You can accept the default values or change the values based on your requirements. For more information about these fields, see the related links.

c. Click **OK** to save the properties for the test.

9. Click **Save** in the design area.

Results

You have configured the process for the component in Deploy.

What to do next

You must create a resource in Deploy. See Creating a resource in HCL DevOps Deploy on page 1146.

Creating a resource in HCL DevOps Deploy

You must create a resource to associate agents with components that you created in HCL DevOps Deploy (Deploy).

Before you begin

- You must be familiar with working with Deploy.
- You must have been granted access to Deploy.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Resources**, and then click **Create Top-Level Group**.
   Result
   The **Create Resource** dialog is displayed.
3. Enter a name for the resource in the **Name** field.
4. Click **Save**.

Results

You have created the resource in Deploy.

What to do next

You must configure the resource. See Configuring the resource in HCL DevOps Deploy on page 1146.

Configuring the resource in HCL DevOps Deploy

You must configure the resource to add an agent and associate the agent with the component.

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Been granted access to Deploy.
  - Created a component in Deploy. See Creating a component in HCL DevOps Deploy on page 1140.
  - Created a resource in Deploy. See Creating a resource in HCL DevOps Deploy on page 1146.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Resources**.
   Result
   A list of resources that are available in Deploy is displayed.
3. Perform the following steps to add an agent to the resource:
   a. Click the resource from the list for which you want to add an agent.
   b. Click the **Actions** icon from the last column, and then click **Add Agent**.
   c. Select the agent from the drop-down list.

      Note: The **Name** field is auto populated with the name of the agent.

   d. Click **Save**.
      Result
      The selected agent is added to the resource and you can view the status of the agent in the **Status** column.
4. Perform the following steps to add a component to the agent:
   a. Click the agent from the list for which you want to add a component.
   b. Click the **Actions** icon from the last column, and then click **Add Component**.
   c. Select the component from the drop-down list.

      Note: The **Name** field is auto populated with the name of the component.

   d. Click **Save**.
      Result
      The selected component is added to the agent.

Results

You have configured the resource in Deploy.

What to do next

You must create an application. See Creating an application in HCL DevOps Deploy on page 1148.

Creating an application in HCL DevOps Deploy

You must create an application to fetch all the components together that you want to deploy.

Before you begin

- You must be familiar with working with Deploy.
- You must have been granted access to Deploy.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Applications**.
3. Click **Create Applications**, and then **New Applications**.
4. Enter a name for the application in the **Name** field.
   Result
   The **Environments** page for the application that you created is displayed.

Results

You have created the application in Deploy.

What to do next

You must configure the application. See Configuring the application HCL DevOps Deploy on page 1148.

Configuring the application HCL DevOps Deploy

You must configure the application to associate resources with environments and define processes to run test assets.

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Created a process for the component in Deploy. See Creating a process in HCL DevOps Deploy on page 1140.
  - Created a component in Deploy. See Creating a component in HCL DevOps Deploy on page 1140.
  - Created a process for the component in Deploy. See Creating a process in HCL DevOps Deploy on page 1140.

- Configure the process for the component in Deploy. See Configuring the process in HCL DevOps Deploy on page 1141.
- Created a resource in Deploy. See Creating a resource in HCL DevOps Deploy on page 1146.
- Configured the resource in Deploy. See Configuring the resource in HCL DevOps Deploy on page 1146.
- Created an application in Deploy. See Creating an application in HCL DevOps Deploy on page 1148.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Applications**.
   Result
   A list of applications that are available in Deploy is displayed.
3. Click the application that you want to configure from the **Name** column.
   Result
   The **Environments** page for the selected application is displayed.
4. Perform the following steps to create an environment for the application that you selected:
   a. Click **Create Environment**.
   b. Enter a name for the environment in the **Name** field.
   c. Click **Save**.
5. Perform the following steps to configure resources to the environment:
   a. Click the environment that you created.
   b. Click **Add Base Resources**.
      Result
      A list of resources that are available in Deploy is displayed.
   c. Select the checkbox to add resources to the environment.
   d. Click **Save**.
      Result
      You can view the corresponding agent and the component that you added for the resource by using the **Expand** icon.
6. Perform the following steps to add the component to the application:
   a. Click **Applications**, and then select your application from the list.
   b. Click the **Components** tab, and then **Add Components**.
   c. Select the checkbox from the drop-down list to add components to the application.
   d. Click **Save**.
7. Perform the following steps to create a process for the application:

a. Click the **Processes** tab, and then **Create Process**.

b. Enter a name for the process in the **Name** field.

c. Click **Save**.

Result

The **Design** tab for the process that you created is displayed.

8. Drag the component process listed under the **Component Process Steps** option from the left navigation pane and drop it into the design area.
9. Select the component process from the drop-down list in the **Operational (No Version Needed) Process** field.
10. Click **Save**.
11. Click the **Edit** icon, and then change the name of the properties.
12. Click **OK**, and then click **Save**.

Results

You have configured the application to run test assets from Deploy.

What to do next

You can run test assets from Deploy. See Running tests from HCL DevOps Deploy on page 1150.

Related information

HCL DevOps Deploy Documentation

Running tests from HCL DevOps Deploy

You can run test assets from HCL DevOps Deploy (Deploy) as a **Process** to test your application under test.

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Been granted access to Deploy.
  - Created a component in Deploy. See Creating a component in HCL DevOps Deploy on page 1140.
  - Created a process for the component in Deploy. See Creating a process in HCL DevOps Deploy on page 1140.
  - Configure the process for the component in Deploy. See Configuring the process in HCL DevOps Deploy on page 1141.
  - Created a resource in Deploy. See Creating a resource in HCL DevOps Deploy on page 1146.

- Configured the resource in Deploy. See Configuring the resource in HCL DevOps Deploy on page 1146.
- Created an application in Deploy. See Creating an application in HCL DevOps Deploy on page 1148.
- Configured the application in Deploy. See Configuring the application HCL DevOps Deploy on page 1148.
- Started the Deploy agent. For more information, refer to the HCL Launch documentation.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Applications**.
   Result
   A list of applications that are available in Deploy is displayed.
3. Click the application that you configured for the test run.
4. Click the **Request Process** icon.
   Result
   The **Create Deployment** page is displayed.
5. Select the process that you created for the application which contains the test assets from the **Process** drop-down list.
6. Click **Next**, and then select the specific version of component, if any.

   Note: If you have multiple versions for your component, then only you can choose the required version by clicking the **Add** option.

7. Click **Next**, and then verify the properties for the test run.
8. Click **Next**, and then verify that the status of the **Run Now** option is enabled.
   You can disable the **Run Now** option, and then use the **Date**, **Time**, and **Recurrence Pattern** fields to run tests as **Process** at the scheduled time.
9. Click **Next**, and then verify the details about the **Process**.
10. Click **Submit Deployment** to initiate the test run.

Results

You have run the performance test assets as **Process** from Deploy.

What to do next

You can view the status of the process as follows:

- **Success**: When the test run is successful
- **Failed**: When the test run fails

You can view the details of the test run in the **Execution Log** section. The **Reports information** section on the Log file displays the names of the report along with its corresponding URLs.

Integration with IBM® DevOps Deploy

You can integrate HCL DevOps Test Hub (Test Hub) with IBM® DevOps Deploy (Deploy) to run test assets as a **Process** from Deploy.

When you use Deploy for automating the deployments of your applications to an environment during the development cycle, you can create test assets for your application in desktop clients. After you configure the project in Test Hub, you can then run those tests assets from Deploy.

You must use the DevOps Test Hub Deploy plugin to integrate Test Hub with Deploy. Integrating Test Hub with Deploy automates the process of test runs. If you have many tests to run at regular intervals, you can use Deploy to initiate test runs automatically.

The following table lists the tasks that you must perform to run performance test assets from Deploy:

Tasks	Go to...
Create any or all the following types of test assets in desktop clients to test your application: • API test assets • Functional test assets • Performance test assets	• DevOps Test Performance documentation • DevOps Test UI documentation • DevOps Test Integrations and APIs documentation
Create a project, and then configure the repository in Test Hub.	Working with projects in a team space on page 1210
Install Deploy with at least one agent.	Installing the server System Requirements on page 99 for information about specific versions of Deploy requirements.
Install the DevOps Test Hub Deploy plugin on Deploy.	Installing the DevOps Test Hub Deploy plugin on page 1153.
Create a **Component** in Deploy.	Creating a component in IBM DevOps Deploy on page 1155.
Create a **Process** for the component in Deploy.	Creating a process in IBM DevOps Deploy on page 1155.
Configure the **Process** that you created for the component in Deploy.	Configuring the process in IBM DevOps Deploy on page 1156.
Create a **Resource** in Deploy.	Creating a resource in IBM DevOps Deploy on page 1161.
Configure the **Resource**.	Configuring the resource in IBM DevOps Deploy on page 1162.
Create an **Application** in Deploy.	Creating an application in IBM DevOps Deploy on page 1163.

Tasks	Go to...
Configure the **Application** that you want to use to run test assets from Deploy.	Configuring the application in IBM DevOps Deploy on page 1163.
Run the test assets from Deploy.	Running tests from IBM DevOps Deploy on page 1165.

Installing the DevOps Test Hub Deploy plugin

You must install the DevOps Test Hub Deploy plugin to run test assets from IBM® DevOps Deploy (Deploy).

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Installed Deploy. For more information about the installation, refer to the IBM Urban Code documentation.
  - Been granted access to Deploy.
  - Downloaded the DevOps Test Hub Deploy plugin from the My HCLSoftware (MHS).

    For more information about specific versions of plug-in, see Integration plugin compatibility matrix on page 1123.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Settings**.
3. Click **Installed Plugins** from the **Automation** pane.
   If you are using Deploy 8.1.1 or older version, then you must click **Automation Plugins** from the **Automation** pane.
4. Click **Load plugin**.
5. Click **Choose File** to locate and open the compressed DevOps Test Hub Deploy plugin file.

   Remember: You must not extract the DevOps Test Hub Deploy plugin compressed file contents.

6. Click **Submit**.
   Result

   The installed plugin is displayed on the **Automation Plugins** tab.

Results

You have installed the DevOps Test Hub Deploy plugin on Deploy.

What to do next

You must create a **Component** in Deploy. See Creating a component in IBM DevOps Deploy on page 1155.

Related information

Installing plug-ins from the Web UI of Deploy

Using a custom trust store for IBM® DevOps Deploy

You can use a custom trust store in the DevOps Test Hub Deploy plugin file to establish a trusted and secure connection between the IBM® DevOps Deploy (Deploy) server and HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Saved the Certificate Authority (CA) assigned to Test Hub.

  Note: You can contact the administrator of Test Hub if you do not have a copy of the CA.

- Installed Java on the computer from where you access the Test Hub UI.

About this task

If the SSL certificate assigned to Test Hub is signed by an internal Certified Authority (CA), then you must download and import the CA certificate to a custom trust store. You can then use the custom trust store in the **Run HCL DevOps Test Hub test** process step to establish a trusted and secure connection between the Deploy server and Test Hub.

Note: If the internal CA certificate is already imported to the default trust store that is used by the Deploy server, you need not use a custom trust store.

1. Locate the default trust store file (`cacerts` file) from the JRE directory.
   For example: `C:\Program Files\Java\jdk-18.0.1.1\lib\security`
2. Copy the `cacerts` file to a location of your choice on your computer.
3. Place the CA assigned to Test Hub in the same location where you saved the `cacerts` file.
4. Run the following command from the command-line interface or terminal to import the CA of Test Hub to custom trust store:

   ```
   keytool -import -trustcacerts -file {path to the downloaded CA with the file extension} -alias {label
    for the certificate} -keystore {path to the trust store cacerts file}
   ```

   For example,

   ```
   keytool -import -trustcacerts -file C:\Users\Desktop\cert\ca-master-hcl.crt -alias "otsca" -keystore C:
   \Users\Desktop\cert\cacerts
   ```

   Note: The default password of the trust store is *changeit*. It remains the same for the custom trust store. If you want to change the password, you can run the following command, and then enter the new password:



```
keytool -storepasswd -keystore {path to the trust store cacerts file}
```

For example, `keytool -storepasswd -keystore C:\Users\Desktop\cert\cacerts`

Results

You have successfully imported the downloaded CA to the custom trust store.

What to do next

You can then run the tests that are available in your Test Hub project from the Deploy server.

Creating a component in IBM® DevOps Deploy

You must create a component to include artifacts and processes in IBM® DevOps Deploy (Deploy). The artifacts include runnable files, images, databases, configuration instructions. Whereas the processes define the activities that components can perform.

Before you begin

- You must be familiar with working with Deploy.
- You must have been granted access to Deploy.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Components**, and then click **Create Component**.
3. Enter a name for the component in the **Name** field.
4. Enter the details in the other optional fields based on your requirement, and then click **Save**.
   Result
   The component that you created is displayed.

Results

You have created the component in Deploy.

What to do next

You must create a process for the component in Deploy. See Creating a process in IBM DevOps Deploy on page 1155.

Creating a process in IBM® DevOps Deploy

You must create a process for the component to include step properties for the test that you want to run from IBM® DevOps Deploy (Deploy).

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Been granted access to Deploy.
  - Created a component in Deploy. See Creating a component in IBM DevOps Deploy on page 1155.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Components**.
   Result
   A list of components that are available in Deploy is displayed.
3. Select the component from the list for which you want to create a process.
4. Click the **Processes** tab, and then click **Create Process**.
   Result
   The **Create Process** dialog is displayed.
5. Enter a name for the process in the **Name** field.
6. Select **Operational (No Version Needed)** from **Process Type** drop-down list.
7. Verify the **Default Working Directory** field.

   The **Default Working Directory** field defines the location that the agent uses to run the process. The default value is `${p:resource/work.dir}/${p:component.name}`.

   Where `${p:resource/work.dir}` is the default working directory for the agent and `${p:component.name}` is the name of the component.
8. Click **Save**.
   Result
   The process that you created is listed in the **Processes** tab and the **Design** tab for the process is displayed.

Results

You have created the process for the component in Deploy.

What to do next

You must configure the process in Deploy. See Configuring the process in IBM DevOps Deploy on page 1156.

Configuring the process in IBM® DevOps Deploy

You must configure the process that you created for the component to organize the steps in the process, specify the properties of the steps, and connect them in IBM® DevOps Deploy (Deploy).

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Been granted access to Deploy.
  - Created a component in Deploy. See Creating a component in IBM DevOps Deploy on page 1155.
  - Created a process for the component in Deploy. See Creating a process in IBM DevOps Deploy on page 1155.
  - Generated an offline user token from Test Hub. See Generating an offline token on page 1194.
  - Added the tests assets to the project on Test Hub.

About this task

When you open any process to configure, the process is displayed in the process editor. The process editor lists the plugins and steps. The required **Start** and **Finish** steps represent the beginning and the end of the process and steps are automatically placed on the design area.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Components**.
   Result
   A list of components that are available in Deploy is displayed.
3. Select the component from the list in which you created the process.
4. Click the **Processes** tab.
   Result
   A list of processes that are available for the component is displayed.
5. Select the process from the list that you want to configure.
   Result
   The **Design** tab for the process is displayed.
6. Click **HCL DevOps Test Workbench**, and then **HCL DevOps Test Hub** from the left menu.
7. Drag the **Run HCL DevOps Test Hub test** step, and then drop it into the design area.

   Note: The selected test must be placed between **Start** and **Finish** steps.

8. Specify the properties for the selected test by performing the following steps:
   a. Click the **Edit** icon.
      Result
      The **Edit Properties for Run HCL DevOps Test Hub test** dialog is displayed.
   b. Specify the properties for the selected test step by referring to the following tables:

The following table lists the required fields that you must provide to run the test from Deploy:

Fields	Action
Name	Enter the name for the test step.
HCL DevOps Test Hub URL	Enter the URL of Test Hub. If you do not have the details of the custom trust store, you must append #insecure-skip-tls-verify to the URL. For example, https://10.134-.56.236.nip.io/test#insecure-skip-tls-verify
Offline Token	Enter the offline token that you generated from Test Hub.
Team Space Name	Enter the name of the team space that contains the project. **Note:** The license for the team space must be configured and you must be a member of that team space.
Project Name	Enter the name of the project that are available in the team space.
Repository Link	Enter the URL of the Git repository that you added to your project in Test Hub.
File Path	Enter the path of the test assets that you want to run. You can find the path of the test assets from the **Execution** page in Test Hub

The following table lists the optional fields that you can provide to run the test from Deploy:

Fields	Description
Branch Name	Use this field to enter the name of the branch in the Git repository where you stored test assets.
Datasets	Use this field to enter the path to the dataset if you want to replace the values of the dataset during a test run.

Fields	Description
	You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the **Datasets** field, you must also include the path to the dataset. The path must be in the following format: `/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv` **Note:** You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon.
Environment	Use this field to enter the environment details for the following types of test assets: ▪ APISUITE ▪ APITEST ▪ APISTUB
Labels	Use this field to add labels to test results when the test run is complete. You can add multiple labels to a test result separated by a comma. For example, *label1, label2* After the test run is complete, then the values provided in the **Labels** field are displayed on the **Results** page of Test Hub.
Result Properties	Use this field to enter a key-value pair of the property to be added to test results. You can add multiple properties separated by a semicolon (;). For example, `buildID=DTHub202609070223-2f3d81f80f;executedFrom=AzurePlugin;` After the test run is complete, the values added here are displayed in the **Properties** column on the **Results** page of Test Hub. Additionally, for server test assets, the properties are also displayed on the **Functional Report** page and **Test Log** page.
Security Group	Use this field to enter the name of the security group if you created secrets for your project and to use secrets at test run time. This field is mandatory only if you want to run the following types of test assets: ▪ APISUITE ▪ APITEST ▪ APISTUB

Fields	Description
Show Hidden Properties	Select this checkbox to display the additional fields that are hidden. You can configure the following fields if you have the details of a custom trust store: ▪ **Custom Trust Store Password**: Use this field to enter the trust store password if you have modified the password while creating the custom trust store. **Note:** If you have not modified the trust store password, you can retain the **Custom Trust Store Password** field blank. ▪ **Custom Trust Store Path**: Use this field to enter the file path of your trust store followed by the file name if Test Hub uses an internal CA certificate and you have imported the certificate to a custom trust store. Also, you can configure the HTTP Proxy to connect to Test Hub by using the following fields: ▪ **HTTP Proxy Host**: Use this field to enter the hostname of the HTTP proxy that you want to connect to Test Hub. ▪ **HTTP Proxy Port**: Use this field to enter the port number where the HTTP proxy is used to listen to both HTTP and HTTPS traffic. ▪ **HTTP Proxy User name**: Use this field to provide the username for the HTTP proxy. ▪ **HTTP Proxy Password**: Use this field to enter a valid password for the user that you specified in the **HTTP Proxy User name** field. **Note:** You can use the HTTP-related fields only when you want to run the following types of test assets: ▪ APISUITE ▪ APITEST ▪ APISTUB
Variables	Use this field to enter the name of the variable and its value if your test requires variables during the test run time. You must enter the variables in the following format: `name_of_the_variable=value_of_the_varibale` You can add multiple variables to the test run separated by a semicolon. For example, *varname1=value1;varname2=value2*

Note: In addition to optional fields, you can also use the following fields from Deploy to configure your test run:

- **Working Directory**
- **Precondition**
- **Post Processing Script**
- **Use Impersonation**
- **Auth Token Restriction**

You can accept the default values or change the values based on your requirements. For more information about these fields, see the related links.

c. Click **OK** to save the properties for the test.

9. Click **Save** in the design area.

Results

You have configured the process for the component in Deploy.

What to do next

You must create a resource in Deploy. See Creating a resource in IBM DevOps Deploy on page 1161.

Creating a resource in IBM® DevOps Deploy

You must create a resource to associate agents with components that you created in IBM® DevOps Deploy (Deploy).

Before you begin

- You must be familiar with working with Deploy.
- You must have been granted access to Deploy.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Resources**, and then click **Create Top-Level Group**.
   Result
   The **Create Resource** dialog is displayed.
3. Enter a name for the resource in the **Name** field.
4. Click **Save**.

Results

You have created the resource in Deploy.

What to do next

You must configure the resource. See Configuring the resource in IBM DevOps Deploy on page 1162.

Configuring the resource in IBM® DevOps Deploy

You must configure the resource to add an agent and associate the agent with the component in IBM® DevOps Deploy (Deploy).

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Been granted access to Deploy.
  - Created a component in Deploy. See Creating a component in IBM DevOps Deploy on page 1155.
  - Created a resource in Deploy. See Creating a resource in IBM DevOps Deploy on page 1161.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Resources**.
   Result
   A list of resources that are available in Deploy is displayed.
3. Perform the following steps to add an agent to the resource:
   a. Click the resource from the list for which you want to add an agent.
   b. Click the **Actions** icon from the last column, and then click **Add Agent**.
   c. Select the agent from the drop-down list.

      Note: The **Name** field is auto populated with the name of the agent.

   d. Click **Save**.
      Result
      The selected agent is added to the resource and you can view the status of the agent in the **Status** column.
4. Perform the following steps to add a component to the agent:
   a. Click the agent from the list for which you want to add a component.
   b. Click the **Actions** icon from the last column, and then click **Add Component**.
   c. Select the component from the drop-down list.

Note: The **Name** field is auto populated with the name of the component.

d. Click **Save**.

Result

The selected component is added to the agent.

Results

You have configured the resource in Deploy.

What to do next

You must create an application. See Creating an application in IBM DevOps Deploy on page 1163.

Creating an application in IBM® DevOps Deploy

You must create an application in IBM® DevOps Deploy (Deploy) to fetch all the components together that you want to deploy.

Before you begin

- You must be familiar with working with Deploy.
- You must have been granted access to Deploy.

1. Log in to Deploy, if you are not already logged in.

   Result

   The Deploy dashboard is displayed.
2. Click **Applications**.
3. Click **Create Applications**, and then **New Applications**.
4. Enter a name for the application in the **Name** field.

   Result

   The **Environments** page for the application that you created is displayed.

Results

You have created the application in Deploy.

What to do next

You must configure the application. See Configuring the application in IBM DevOps Deploy on page 1163.

Configuring the application in IBM® DevOps Deploy

You must configure the application in IBM® DevOps Deploy (Deploy) to associate resources with environments and define processes to run test assets.

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
  - Been granted access to Deploy.
  - Created a component in Deploy. See Creating a component in IBM DevOps Deploy on page 1155.
  - Created a process for the component in Deploy. See Creating a process in IBM DevOps Deploy on page 1155.
  - Configure the process for the component in Deploy. See Configuring the process in IBM DevOps Deploy on page 1156.
  - Created a resource in Deploy. See Creating a resource in IBM DevOps Deploy on page 1161.
  - Configured the resource in Deploy. See Configuring the resource in IBM DevOps Deploy on page 1162.
  - Created an application in Deploy. See Creating an application in IBM DevOps Deploy on page 1163.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Applications**.
   Result
   A list of applications that are available in Deploy is displayed.
3. Click the application that you want to configure from the **Name** column.
   Result
   The **Environments** page for the selected application is displayed.
4. Perform the following steps to create an environment for the application that you selected:
   a. Click **Create Environment**.
   b. Enter a name for the environment in the **Name** field.
   c. Click **Save**.
5. Perform the following steps to configure resources to the environment:
   a. Click the environment that you created.
   b. Click **Add Base Resources**.
      Result
      A list of resources that are available in Deploy is displayed.
   c. Select the checkbox to add resources to the environment.
   d. Click **Save**.
      Result
      You can view the corresponding agent and the component that you added for the resource by using the **Expand** icon.

6. Perform the following steps to add the component to the application:
    a. Click **Applications**, and then select your application from the list.
    b. Click the **Components** tab, and then **Add Components**.
    c. Select the checkbox from the drop-down list to add components to the application.
    d. Click **Save**.
7. Perform the following steps to create a process for the application:
    a. Click the **Processes** tab, and then **Create Process**.
    b. Enter a name for the process in the **Name** field.
    c. Click **Save**.
    Result
    The **Design** tab for the process that you created is displayed.
8. Drag the component process listed under the **Component Process Steps** option from the left navigation pane and drop it into the design area.
9. Select the component process from the drop-down list in the **Operational (No Version Needed) Process** field.
10. Click **Save**.
11. Click the **Edit** icon, and then change the name of the properties.
12. Click **OK**, and then click **Save**.

Results

You have configured the application to run test assets from Deploy.

What to do next

You can run test assets from Deploy. See Running tests from IBM DevOps Deploy on page 1165.

Running tests from IBM® DevOps Deploy

You can run test assets from IBM® DevOps Deploy (Deploy) as a **Process** to test your application under test.

Before you begin

- You must be familiar with working with Deploy.
- You must have performed the following tasks:
    - Been granted access to Deploy.
    - Created a component in Deploy. See Creating a component in IBM DevOps Deploy on page 1155.
    - Created a process for the component in Deploy. See Creating a process in IBM DevOps Deploy on page 1155.
    - Configure the process for the component in Deploy. See Configuring the process in IBM DevOps Deploy on page 1156.

- Created a resource in Deploy. See Creating a resource in IBM DevOps Deploy on page 1161.
- Configured the resource in Deploy. See Configuring the resource in IBM DevOps Deploy on page 1162.
- Created an application in Deploy. See Creating an application in IBM DevOps Deploy on page 1163.
- Configured the application in Deploy. See Configuring the application in IBM DevOps Deploy on page 1163.
- Started the Deploy agent. For more information, refer Starting agents.

1. Log in to Deploy, if you are not already logged in.
   Result
   The Deploy dashboard is displayed.
2. Click **Applications**.
   Result
   A list of applications that are available in Deploy is displayed.
3. Click the application that you configured for the test run.
4. Click the **Request Process** icon.
   Result
   The **Run Application Process** page is displayed.
5. Select the process that you created for the application which contains the test assets from the **Process** drop-down list.
6. Click **Submit**.
   Result
   The Deploy dashboard shows the progress of the process request.

Results

You have run the performance test assets as **Process** from Deploy.

What to do next

After the process completes, you can view the status of the process displayed as follows:

- **Success**: When the test run is successful
- **Failed**: When the test run is failed

You can view the details of the test run in the **Execution Log** section. The **Reports information** section on the Log file displays the names of the report along with its corresponding URLs.

Integration with IBM® Engineering Test Management

As a *Project Owner*, you can integrate Engineering Test Management as a quality management solution with HCL DevOps Test Hub (Test Hub). You can run test assets that are available in a project from Engineering Test Management.

The following topics provides more information about the integration of Test Hub with Engineering Test Management.

Task flow for integration of IBM® Engineering Test Management

You must perform the following tasks in sequence to integrate HCL DevOps Test Hub (Test Hub) with Engineering Test Management. For more information about the tasks, refer to the links provided in the table.

S/N	Tasks	More information
1	Create a project in Test Hub.	Creating a project on page 1213
2	Install the required version of Engineering Test Management on your computer.	Engineering Lifecycle Management Documentation
3	Download the Engineering Test Management adapter, and then extract the contents of a compressed file. For more information about specific versions of plugin, see Integration plugin compatibility matrix on page 1123.	My HCLSoftware (MHS)
4	Add the adapter files that you downloaded to the installation directory of Engineering Test Management.	Adding the adapter files to IBM Engineering Test Management on page 1167
5	Configure the details of Engineering Test Management on Test Hub.	Configuring IBM Engineering Test Management on DevOps Test Hub on page 1168
6	Create a test script and a test case on Engineering Test Management to run the test assets.	Preparing test cases in IBM Engineering Test Management on page 1170
7	Run the test assets from Engineering Test Management.	Running tests from IBM Engineering Test Management on page 1171

Adding the adapter files to IBM® Engineering Test Management

You must add the adapter files to the installation directory to run test assets of HCL DevOps Test Hub (Test Hub) from Engineering Test Management.

Before you begin

You must have completed the following tasks:

- Installed the latest version of Engineering Test Management from the Jazz portal on your computer. See System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94.
- Downloaded the adapter (a compressed file) for Engineering Test Management from the My HCLSoftware (MHS).

  For more information about specific versions of plugin, see Integration plugin compatibility matrix on page 1123.

About this task

You must add the adapter files to the location where you installed Engineering Test Management to run test assets of Test Hub.

1. Extract the compressed file that you downloaded.
2. Copy the `etm-adapter-updatesite` folder.
3. Go to the location where you installed Engineering Test Management on your computer.
4. Place the `etm-adapter-updatesite` folder in the `sites` folder of the installation directory of Engineering Test Management.
   For example, `D:\ELM\JazzTeamServer\server\conf\qm\sites`.
5. Copy the `dthub-profile.ini` file from the `provision_profiles` folder of the extracted package and place it in the `provision_profiles` folder of the Engineering Test Management installation directory.
   For example, `D:\ELM\JazzTeamServer\server\conf\qm\provision_profiles`.

Results

You have added the adapter files to Engineering Test Management.

What to do next

You must perform the following tasks:

- You must reset the Engineering Test Management server from the web console by using the **Request Server Reset** option.

  If you have administrator privileges, then you can access the **Request Server Reset** option from the following URL:

  ```
  https://<hostname>:<portnumber>/qm/admin?
  internal#action=com.ibm.team.repository.admin.serverReset
  ```

  Note: You must replace the following values with the actual values in the URL:

  - `<hostname>` with the hostname that you used when you installed Engineering Test Management.
  - `<portnumber>` with the port number that is associated with an IP address of the hostname.

- Restart the Engineering Test Management server.
- Configure the details of Engineering Test Management on Test Hub to establish a connection between the two products. See Configuring IBM Engineering Test Management on DevOps Test Hub on page 1168.

Configuring IBM® Engineering Test Management on DevOps Test Hub

You must configure Engineering Test Management on HCL DevOps Test Hub (Test Hub) to establish a connection between the two products.

Before you begin

You must have completed the following tasks:

- Granted sufficient privileges to access **Project area** on Engineering Test Management.
- Created a project area in Engineering Test Management.
- Created a project in Test Hub.

About this task

If you have multiple instances of Engineering Test Management, then you can configure all those instances on Test Hub. If you have only one instance of Engineering Test Management, you can configure the same instance multiple times. However, you must choose a different **Project area** during the configuration of Engineering Test Management.

1. Log in to Test Hub, if you are not already logged in.
2. Open a project from the team space.
3. Click **Manage > Integration**.
   The **Integration** page is displayed.
4. Click **Add IBM ETM Adapter** option within the **IBM Engineering Test Management (ETM)** section.
   The **Add IBM ETM Adapter** dialog is displayed.
5. Enter the following information in the **Add IBM ETM Adapter** dialog:

Fields	Actions
Server URL	Enter the URL of Engineering Test Management. For example, `https://<hostname>:<portnumber>/qm`
Username	Enter the username associated with Engineering Test Management.
Password	Enter the password associated with the username of Engineering Test Management.

6. Click **Validate**.
7. Select the project name that you created in Engineering Test Management from the **Project area** list.
8. Enter a name for the Engineering Test Management adapter in the **Adapter name** field.

   Note: You cannot provide an adapter name that already exists for the **Project area**.

9. Click **Save**.

Results

You have configured the details of Engineering Test Management on Test Hub.

What to do next

You must create a test script and test case on Engineering Test Management to run the test assets of Test Hub. See Preparing test cases in IBM Engineering Test Management on page 1170.

Preparing test cases in IBM® Engineering Test Management

You must create a test script and a test case on Engineering Test Management to run the test assets.

Before you begin

You must have completed the following tasks:

- Added the adapter files to the installation directory of Engineering Test Management. See Adding the adapter files to IBM Engineering Test Management on page 1167.
- Configured the details of Engineering Test Management on HCL DevOps Test Hub (Test Hub). See Configuring IBM Engineering Test Management on DevOps Test Hub on page 1168.
- Generated an offline user token from Test Hub. See Generating an offline token on page 1194.
- Copied the following values to be used during the preparation of a test script and a test case:
  - Branch name of the Git repository
  - Asset ID of the test that you want to run
  - **Optional**: Name of the environment if you want to run the API test suite.

1. Log in to Engineering Test Management, if you are not already logged in.
2. Open your project area from where you want to run the test assets.
3. Click **Construction**, and then click **Test Script** from the drop-down list.
4. Enter a name for the test script in the **Test Script** field.
5. Select **HCL DevOps Test Hub** from the **Type** drop-down list.
6. Enter the following test information:

Fields	Actions
Offline Token	Enter the offline token that you generated from Test Hub.
Branch	Enter the name of the branch where your test assets are stored.
Asset ID	Enter an asset ID of the test asset that is available on the **Execution** page of Test Hub.  **Tip:** You can hover over the name of the test asset to view and copy the asset ID.
Environment	Enter a name of the API test environment.

Fields	Actions
	Remember: You must use this field only if you want to run the API test suites.

7. Click **Save**.
8. Perform the following steps to add a test script to a test case:

    a. Click **Construction > Test Case**.
    Result
    The **Test Cases** page is displayed.

    b. Enter a name for the test case in the **Test Case** field.

    c. Click **Test Scripts** from the left navigation pane.

    d. Click the **Plus** icon from the **Test Scripts** panel.
    Result
    The **Add Test Scripts** dialog is displayed.

    e. Select the checkbox to add the test script that you created to the test case, and then click **Add and Close**.

    f. Click **Save**.

Results

You have prepared Engineering Test Management.

What to do next

You can run the test assets of Test Hub from Engineering Test Management. See Running tests from IBM Engineering Test Management on page 1171.

Running tests from IBM® Engineering Test Management

After you create a test case in Engineering Test Management, you can run the test. The test case includes the test scripts that contains test assets of HCL DevOps Test Hub (Test Hub).

Before you begin

You must have performed the following tasks:

- Prepared Engineering Test Management to run test assets of Test Hub. See Preparing test cases in IBM Engineering Test Management on page 1170.
- Configured the details of Engineering Test Management on Test Hub. See Configuring IBM Engineering Test Management on DevOps Test Hub on page 1168.

1. Log in to Engineering Test Management.
2. Open the test case that you created in Engineering Test Management.

3. Click the **Play** icon  , and then click **Run**.
   Result
   The **Run Test Case** dialog is displayed.
4. Click the **ChangeMachine** icon  next to the **Machine** field to select the adapter that you configured on Test Hub.
5. Select the required adapter name from the list, and then click **OK**.
6. Click **Finish** to start the test run.
   The Engineering Test Management dashboard displays the progress of the test case run.

   Note: If the dashboard does not display the progress of the test run correctly, then you must refresh the browser to view the correct status.

Results

You have run the test assets of Test Hub from Engineering Test Management.

What to do next

You can perform the following tasks:

- Click **Show Result** to view information about the test results from Engineering Test Management when the test run is complete.
- Click **Link to Result** from **Results details** to view the test results from the **Results** page of Test Hub. See Test results and reports on page 985.

Deleting an IBM® Engineering Test Management adapter

You might want to delete an IBM® Engineering Test Management adapter when it is no longer needed.

Before you begin
You must have completed the following tasks:

- Been assigned the role of *Owner* of a project or *Owner* of the project in HCL DevOps Test Hub (Test Hub).
- Configured the details of Engineering Test Management on Test Hub. See Configuring IBM Engineering Test Management on DevOps Test Hub on page 1168.

1. Log in to Test Hub and open the team space that contains your project.
2. Click **Manage > Integration**.
   The **Integration** page is displayed.
3. Go to the **IBM Engineering Test Management (ETM)** and within the **Configuration details of IBM ETM Adapters** section.

   You can view the Engineering Test Management adapters that are configured with the project on Test Hub.

4. Identify the Engineering Test Management adapter that you want to delete, and then click the **Delete** icon .

   Note: If any of the test runs are in progress on Engineering Test Management, then those runs are aborted.

5. Click **Confirm** to delete the Engineering Test Management adapter from the project.

Results

You have deleted the Engineering Test Management adapter from the project on Test Hub.

Integration with Jenkins

You can use the HCL DevOps Test Hub (Test Hub) plugin to integrate Test Hub with Jenkins. After you complete the integration, you can run tests assets that are available in a project of Test Hub from a Jenkins server.

You must have installed Jenkins on your computer to install the Test Hub Jenkins plugin. You can run the following command to start the Jenkins server to support UTF-8 character sets:

java -Dfile.encoding=UTF8 -jar jenkins.war

You can use either the Jenkins **Freestyle** project or the **Pipeline** project to run test assets from Jenkins. With **Freestyle** project, you can create build step from the Jenkins UI to run the test assets. Whereas the **Pipeline** project uses a simple text scripts based on the Groovy programming language. You can define pipeline scripts during the configuration of the **Pipeline** project by using any of the following methods:

- **Pipeline script**: In this method, you can directly enter a script in the field provided within the Jenkins UI or you can select a sample **Pipeline script** from the drop-down list.
- **Pipeline script from SCM**: In this method, you can create a script with a Groovy editor and then commit the script file into the Git repository.

For more information about the Jenkins Pipeline, refer to the Jenkins documentation.

Refer to the following topics to learn more about integrating Jenkins with Test Hub.

Task flows for running test assets from Jenkins

You can perform certain tasks to run test assets from the Jenkins **Freestyle** project or the **Pipeline** project.

The following table lists the task flows for running test assets from the Jenkins **Freestyle** project:

Tasks	Go to...
Install the Test Hub Jenkins plugin.	Installing the Test Hub Jenkins plugin on page 1174
Import the CA certificate to your custom trust store.	Using a custom trust store on page 1175

Tasks	Go to...
Configure the **Freestyle** project.	Configuring the Freestyle project on page 1177
Run Test Hub test assets on Jenkins.	Running tests from Jenkins on page 1187

You can perform the following tasks to run test assets from the Jenkins **Pipeline** project by using the sample script:

Tasks	Go to...
Install the Test Hub Jenkins plugin.	Installing the Test Hub Jenkins plugin on page 1174
Import the CA certificate to your custom trust store.	Using a custom trust store on page 1175
Create a pipeline script from the Jenkins UI.	Creating a pipeline script from Jenkins on page 1181
Configure the **Pipeline** project.	Configuring the Pipeline project by using the sample script on page 1185
Run Test Hub test assets on Jenkins.	Running tests from Jenkins on page 1187

You can perform the following tasks to run test assets from the Jenkins **Pipeline** project by using the script from Source Code Management (SCM):

Tasks	Go to...
Install the Test Hub Jenkins plugin.	Installing the Test Hub Jenkins plugin on page 1174
Import the CA certificate to your custom trust store.	Using a custom trust store on page 1175
Create a pipeline script and commit the script to the Git repository.	For more information, refer to the Jenkins documentation.
Configure the **Pipeline** project.	Configuring the Pipeline project by using the script from SCM on page 1186
Run Test Hub test assets on Jenkins.	Running tests from Jenkins on page 1187

Installing the Test Hub Jenkins plugin

Before you can run the tests that are available in your HCL DevOps Test Hub (Test Hub) project on the Jenkins server, you must install the Test Hub Jenkins plugin on the Jenkins server.

Before you begin

You must have completed the following tasks:

- Installed Jenkins on your computer. For more information about installing Jenkins, refer to the Jenkins documentation.
- Downloaded the Test Hub Jenkins plugin from the My HCLSoftware (MHS).

  For more information about specific versions of plugin, see Integration plugin compatibility matrix on page 1123.

1. Log in to the Jenkins server.
   Result
   The Jenkins dashboard is displayed.
2. Click **Manage Jenkins > Manage Plugins**.
3. Click the **Advanced** tab.
4. Click **Choose File** from the **Upload plugin** section to locate and **Open** the Test Hub Jenkins plugin.
5. Click **Upload**.
   Result

   The Test Hub Jenkins plugin is displayed in the **Installed** tab.

Results

You have installed the Test Hub Jenkins plugin on the Jenkins server.

What to do next

You must import the CA certificate to your custom trust store to establish a trusted and secure connection between the Jenkins server and Test Hub. See Using a custom trust store on page 1175.

Using a custom trust store

You must use a custom trust store in the Jenkins build to establish a trusted and secure connection between the Jenkins server and HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Saved the Certificate Authority (CA) assigned to Test Hub.

  Note: You can contact the administrator of Test Hub if you do not have a copy of the CA.

- Installed Java on the computer from where you access the Test Hub UI.

About this task

If the SSL certificate assigned to Test Hub is signed by an internal CA, then you must download and import the CA to a custom trust store. You can then use the custom trust store in the Jenkins build step to establish a trusted and secure connection between the Jenkins server and Test Hub.

Note: If the internal CA is already imported to the default trust store that is used by the Jenkins server, you need not use a custom trust store.

Restriction:

When you use Red Hat Enterprise Linux (RHEL) operating systems for Jenkins, you must run the Jenkins service with a user who has access to the custom trust store path to utilize the custom trust store feature. To change the Jenkins user, you must open the `/etc/sysconfig/jenkins` file and set the JENKINS_USER to the user who has access to the custom trust store path.

You can then run the following commands to set JENKINS_USER to another user who has access to the path of the custom trust store:

```
$JENKINS_USER= <username>
```

For example, `$JENKINS_USER= <user1>`

Note: You must ensure that the user account is available in the `/etc/passwd` file.

You can then run the following commands to change the ownership of the Jenkins folder:

```
chown -R username:username /var/lib/jenkins
```

```
chown -R username:username /var/cache/jenkins
```

```
chown -R username:username /var/log/jenkins
```

For example,

```
chown -R user1:user1 /var/lib/jenkins
```

```
chown -R user1:user1 /var/cache/jenkins
```

```
chown -R user1:user1 /var/log/jenkins
```

After the change of ownership is complete, you can run the following command to restart the Jenkins server:

```
/etc/init.d/jenkins restart
```

1. Locate the default trust store file (`cacerts` file) from the JRE directory.
   For example: `C:\Program Files\Java\jdk-18.0.1.1\lib\security`
2. Copy the `cacerts` file to a location of your choice on your computer.
3. Place the CA assigned to Test Hub in the same location where you saved the `cacerts` file.

4. Run the following command from the command-line interface or terminal to import the CA of Test Hub to custom trust store:

   ```
   keytool -import -trustcacerts -file {path to the downloaded CA with the file extension} -alias {label
    for the certificate} -keystore {path to the trust store cacerts file}
   ```

   For example,

   ```
   keytool -import -trustcacerts -file C:\Users\Desktop\cert\ca-master-hcl.crt -alias "otsca" -keystore C:
   \Users\Desktop\cert\cacerts
   ```

   Note: The default password of the trust store is *changeit*. It remains the same for the custom trust store. If you want to change the password, you can run the following command, and then enter the new password:

   ```
   keytool -storepasswd -keystore {path to the trust store cacerts file}
   ```

   For example, `keytool -storepasswd -keystore C:\Users\Desktop\cert\cacerts`

Results

You have successfully imported the downloaded CA to the custom trust store.

What to do next

You can run test assets either from **Freestyle** or **Pipeline** project from the Jenkins server.

Configuring the Freestyle project

You must configure a **Freestyle** project to add a build step, and then run tests that are available in your HCL DevOps Test Hub (Test Hub) project from Jenkins.

Before you begin

You must have completed the following tasks:

- Installed the Test Hub Jenkins plugin on the Jenkins server. See Installing the Test Hub Jenkins plugin on page 1174.
- Created a Jenkins **Freestyle** project.
- Generated an offline user token from Test Hub. See Access management for DevOps Test Hub on page 1194.

1. Open your Jenkins **Freestyle** project from the list, and then click **Configure**.
2. Click the **Build** tab, and then click **Add build step**.
3. Select the **HCL DevOps Test Hub test** option from the drop-down list.
4. Provide the details about the test run for the fields in the following table:

   The following table lists the required fields that you must provide to run the test from Jenkins:

Fields	Action
Name	Enter a name for the Jenkins build step.
Server URL	Enter the URL of Test Hub. The format of the URL is as follows: *https://hostname*.
Offline Token	Enter the offline user token that you generated from Test Hub.

The following table lists the optional fields that you can provide to run the test from Jenkins:

Fields	Descriptions
Asset Type	Use this field to select the test asset type that you want to run from the **Asset Type** drop-down list.
Branch	Use this field to select the branch of the Git repository from the drop-down list. The **Branch** drop-down list displays the branches that are available in the corresponding project of Test Hub. **Note:** After you select the branch from the **Branch** drop-down list, if you want to change the URL of Test Hub, offline user token, or the project in the **Build** step, then you must close the build step. Then you must click **Add build step**, select **Run HCL DevOps Test Hub test**, and then enter the details.
Datasets	Use this field to enter the path to the dataset if you want to replace the values of the dataset during a test run. You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the **Datasets** field, you must also include the path to the dataset. The path must be in the following format: `/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv` **Note:** You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon.
Variables	Use this field to enter the name of the variable and its value if your test requires variables during the test run time.

Fields	Descriptions
	You must enter the variables in the following format: `name_of_the_variable=value_of_the_varibale` You can add multiple variables to the test run separated by a semicolon. For example, *varname1=value1;varname2=value2*
Result Properties	Use this field to enter a key-value pair of the property to be added to test results. You can add multiple properties separated by a semicolon (;). For example, `buildID=DTHub202609070223-2f3d81f80f;executedFrom=AzurePlugin;` After the test run is complete, the values added here are displayed in the **Properties** column on the **Results** page of Test Hub. Additionally, for  server test assets, the properties are also displayed on the **Functional Report** page and **Test Log** page.
Labels	Use this field to add labels to test results when the test run is complete. You can add multiple labels to a test result separated by a comma. For example, *label1, label2* After the test run is complete, then the values provided in the **Labels** field are displayed on the **Results** page of Test Hub.
Project	Use this field to select a project from the drop-down list. The **Project** drop-down list displays the projects that are available in the corresponding team space of Test Hub. **Note:** The **Project** drop-down list displays the projects where you are an *Owner* or a *member* of Test Hub. You must be an *Owner* or a *Tester* of the project that is available in the team spaces to run the tests from the Jenkins server.
Secrets	Use this field to select the Secret from the **Security Group** drop-down list if you created secrets for your project to use Secrets at test run time. The **Secrets** field is mandatory only if you select one of the following options in the **Asset Type** field: ◦ **APISUITE** ◦ **APITEST**

Fields	Descriptions
	◦ **APISTUB** ◦ **POSTMAN** The following message is displayed when you select any other asset type than preceding list: `You can select Secrets only if you are running tests of type APISUITE, APITEST, APISTUB, or POSTMAN.`
Team Space	Use this field to select the name of the team space from the drop-down list. The **Team Space** drop-down list displays the names of the team spaces if you are an *Owner* or a *member* of the team space.
Test	Use this field to select the required test that you want to run from the drop-down list. The **Tests** drop-down list displays the available test assets from the corresponding branch in the following format: `Project_name [Path: the path of the test assets] [Repo: URL of the Git repository that the test belongs to]` **Note:** After you select the test details from the **Tests** drop-down list if you want to change the URL of Test Hub, offline user token, or the project in the Jenkins **Build step**, then you must delete the build step and add a new build step.
Test Environment	Use this field to select an environment from the drop-down list. The **Test Environment** field displays the available test environments from Test Hub. The **Test Environment** field is mandatory only if you select one of the following options in the **Asset Type** field: ◦ **APISUITE** ◦ **APITEST** ◦ **APISTUB** ◦ **POSTMAN** The following message is displayed when you select any other asset type than preceding list: `You can select Test Environment only if you are running tests of type APISUITE, APITEST, APISTUB, or POSTMAN.`

Fields	Descriptions
Use Custom Trust Store	Select this checkbox if you used an internal CA certificate and imported the certificate to the custom trust store. You can then enter the file path of your trust store followed by the file name in the **Custom Trust Store Path** field. Select the **Use Custom Password for the Trust Store** checkbox if you modified the trust store password, and then enter the new password in the **Custom Trust Store Password** field. **Note:** If you did not modify the trust store password, you must clear the **Use Custom Password for the Trust Store** checkbox.
Fail the job if the test or any of the containing tests fail	Select the checkbox to mark the job as failed when the test or any child test within the test fails.

Note: You can run multiple tests sequentially in the same job by adding multiple build steps and providing details for the tests that you want to run.

5. Click **Save** to save the build step.

Results

You have configured the **Freestyle** project by adding the build step.

What to do next

You can run test assets from the Jenkins server. See Running tests from Jenkins on page 1187.

Creating a pipeline script from Jenkins

You must create a pipeline script to run test assets that are available in the HCL DevOps Test Hub (Test Hub) project from the Jenkins **Pipeline** project.

Before you begin

You must have completed the following tasks:

- Installed the Test Hub Jenkins plugin on the Jenkins server. See Installing the Test Hub Jenkins plugin on page 1174.
- Generated an offline user token from Test Hub. See Access management for DevOps Test Hub on page 1194.
- Created a Jenkins **Pipeline** project.

1. Open your Jenkins **Pipeline** project from the list.
2. Click **Configure**, and then select the **Pipeline** tab.
3. Click **Pipeline Syntax** to generate a pipeline script.
4. Select the **step: General Build Step** option from the drop-down list in the **Sample Step** field.
5. Select the **Run HCL DevOps Test Hub test** option from the drop-down list in the **Build Step** field.
6. Provide the details about the test run for the fields in the following table:

   The following table lists the required fields that you must provide to run the test from Jenkins:

Fields	Action
Name	Enter a name for the Jenkins build step.
Server URL	Enter the URL of Test Hub. The format of the URL is as follows: *https://hostname*.
Offline Token	Enter the offline user token that you generated from Test Hub.

Field	Description
Asset Type	Use this field to select the test asset type that you want to run from the **Asset Type** drop-down list.
Branch	Select the branch from the drop-down list. The **Branch** drop-down list displays the branches that are available in the corresponding project of Test Hub.
Datasets	Use this field to enter the path to the dataset if you want to replace the values of the dataset during a test run. You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the **Datasets** field, you must also include the path to the dataset. The path must be in the following format: `/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv` **Note:** You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon.
Labels	Use this field to add labels to test results when the test run is complete.

Field	Description
	You can add multiple labels to a test result separated by a comma. For example, *label1, label2* After the test run is complete, then the values provided in the **Labels** field are displayed on the **Results** page of Test Hub.
Project	Use this field to select a project from the drop-down list. The **Project** drop-down list displays the projects that are available in the corresponding team space of Test Hub. **Note:** The **Project** drop-down list displays the projects where you are an *Owner* or a *member* of Test Hub. You must be an *Owner* or a *Tester* of the project that is available in the team spaces to run the tests from the Jenkins server.
Secrets	Use this field to select the Secret from the **Security Group** drop-down list if you created secrets for your project to use Secrets at test run time. The **Secrets** field is mandatory only if you select one of the following options in the **Asset Type** field: ◦ **APISUITE** ◦ **APITEST** ◦ **APISTUB** ◦ **POSTMAN** The following message is displayed when you select any other asset type than preceding list: `You can select Secrets only if you are running tests of type APISUITE, APITEST, APISTUB, or POSTMAN.`
Team Space	Use this field to select the name of the team space from the drop-down list. The **Team Space** drop-down list displays the names of the team spaces if you are an *Owner* or a *member* of the team space.
Test	Use this field to select the required test that you want to run from the drop-down list. The **Tests** drop-down list displays the available test assets from the corresponding branch in the following format: `Project_name [Path: the path of the test assets] [Repo: URL of the Git repository that the test belongs to]`

Field	Description
	Note: After you select the test details from the **Tests** drop-down list if you want to change the URL of Test Hub, offline user token, or the project in the Jenkins **Build step**, then you must delete the build step and add a new build step.
Test Environment	Use this field to select an environment from the drop-down list. The **Test Environment** field displays the available test environments from Test Hub. The **Test Environment** field is mandatory only if you select one of the following options in the **Asset Type** field: ◦ **APISUITE** ◦ **APITEST** ◦ **APISTUB** ◦ **POSTMAN** The following message is displayed when you select any other asset type than preceding list: `You can select Test Environment only if you are running tests of type APISUITE, APITEST, APISTUB, or POSTMAN.`
Use Custom Trust Store	Select this checkbox if you used an internal CA certificate and imported the certificate to the custom trust store. You can then enter the file path of your trust store followed by the file name in the **Custom Trust Store Path** field. Select the **Use Custom Password for the Trust Store** checkbox if you modified the trust store password, and then enter the new password in the **Custom Trust Store Password** field. **Note:** If you did not modify the trust store password, you must clear the **Use Custom Password for the Trust Store** checkbox.
Variables	Use this field to enter the name of the variable and its value if your test requires variables during the test run time. You must enter the variables in the following format: `name_of_the_variable=value_of_the_varibale` You can add multiple variables to the test run separated by a semicolon. For example, *varname1=value1;varname2=value2*

Field	Description
Fail the job if the test or any of the containing tests fail	Select the checkbox to mark the job as failed when the test or any child test within the test fails.

7. Click **Generate Pipeline Script** and copy the script that you created for the build step to use it in the sample script.

Results

You have created the pipeline script.

What to do next

You can configure the **Pipeline** project by using the **Pipeline script** option. See Configuring the Pipeline project by using the sample script on page 1185.

Configuring the Pipeline project by using the sample script

You can configure a **Pipeline** project directly from the Jenkins UI by using the sample pipeline scripts to run test assets from Jenkins.

Before you begin

You must have completed the following tasks:

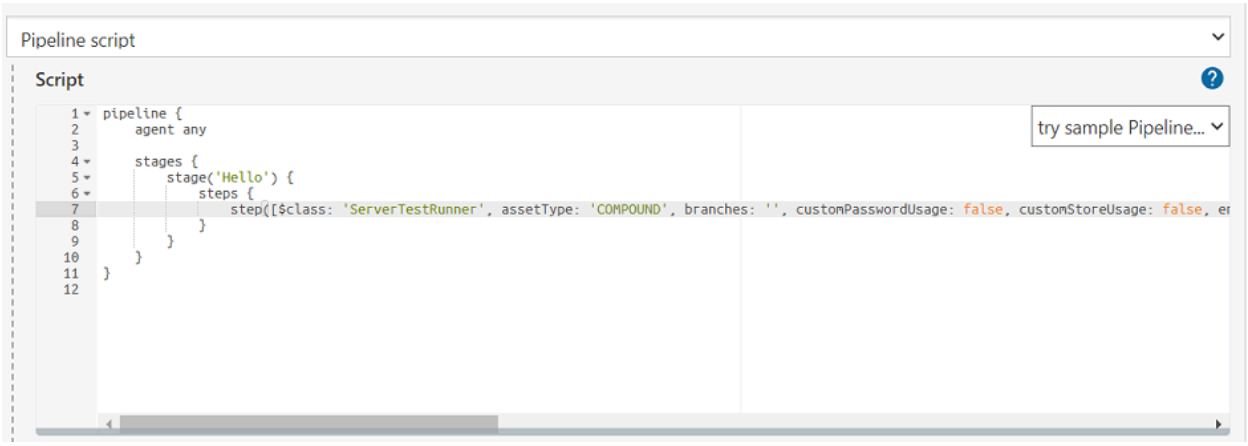
- Installed the Test Hub Jenkins plugin on the Jenkins server. See Installing the Test Hub Jenkins plugin on page 1174.
- Created a pipeline script. See Creating a pipeline script from Jenkins on page 1181.

1. Open your Jenkins **Pipeline** project from the list.
2. Click **Configure**, and then select the **Pipeline** tab.
3. Select the **Pipeline script** option in the **Definition** field.
4. Select the sample script from the drop-down list.

   For example, if you select `Hello Word` from the drop-down list, then the sample script is displayed as follows:



5. Specify the script that you created in place of *"echo 'Hello Word'"*.



Note: You can provide a different name for the `stage` in the script.

Results

You have configured the **Pipeline** project by using the **Pipeline script** option.

What to do next

You can run test assets from the Jenkins server. See Running tests from Jenkins on page 1187.

Configuring the Pipeline project by using the script from SCM

You can configure the **Pipeline** project to use the pipeline script from the Source Code Management (SCM), and then run test assets that are available in the HCL DevOps Test Hub (Test Hub) project from the Jenkins **Pipeline** project.

Before you begin

You must have completed the following tasks:

- Installed the Test Hub Jenkins plugin on the Jenkins server. See Installing the Test Hub Jenkins plugin on page 1174.
- Created a pipeline script and committed the script to the Git repository. For more information, refer to the Jenkins documentation.
- Been granted administrator permission of the Jenkins server to add credentials into Jenkins.
- Added your GitHub credentials into Jenkins. For more information about adding global credentials, refer to the Jenkins documentation.
- Created a Jenkins **Pipeline** project.

About this task

If your pipeline scripts are complex, you can then write the script outside of Jenkins UI. You must commit that script file into your Git repository. During the **Pipeline** build process, Jenkins checks out the script file from the Git repository and runs your **Pipeline** project.

1. Open your Jenkins **Pipeline** project from the list.
2. Click **Configure**, and then select the **Pipeline** tab.
3. Select the **Pipeline script from SCM** option in the **Definition** field.
4. Select the **Git** option from the drop-down list in the **SCM** field.
5. Enter the URL of the Git repository in the **Repository URL** field where you stored your pipeline script.
6. Select the credentials of the Git repository from the drop-down list.
7. **Optional:** Enter the branch name of the Git repository in the **Branch Specifier** field.
8. Enter the path of the script that you stored in the Git repository in the **Script path** field.
9. Click **Save**.

Results

You have configured the **Pipeline** project by using the **Pipeline script from the SCM** option.

What to do next

You can run test assets from the Jenkins server. See Running tests from Jenkins on page 1187.

Running tests from Jenkins

You can run test assets either from the Jenkins **Freestyle** or the **Pipeline** project on the Jenkins server to test an application under test.

Before you begin

You must have completed the following tasks:

- Configured the **Freestyle** project, if you want to run test assets from the **Freestyle** project. See Configuring the Freestyle project on page 1177.
- Configured the **Pipeline** project either by using the **Pipeline script** or **Pipeline script from SCM**, if you want to run test assets from the **Pipeline script** project. See Configuring the Pipeline project by using the sample script on page 1185 or Configuring the Pipeline project by using the script from SCM on page 1186.
- Downloaded the latest **Test Results Analyzer** plugin from the Jenkinsci repository. For more information about the **Test Results Analyzer** plugin, refer to the Jenkins documentation.
- Installed the **Test Results Analyzer** plugin on the Jenkins server. For more information about installing the plugin, refer to the Jenkins documentation.

About this task

After the test run is complete, you can view the test results in a tabular format when you use the Jenkins **Test Results Analyzer** plugin. With a TRA-based report, you can filter the results based on the status of the build. The status of the builds can be PASSED, FAILED, and SKIPPED. You can also identify the failed build from the TRA-based report and take the required action to resolve the build errors. The TRA report also represents the build information in **Line**, **Bar**, and **Pie** graphs so that you can easily read the data.

1. Log in to the Jenkins server.
   Result
   The Jenkins dashboard is displayed.
2. Open your Jenkins **Freestyle** or **Pipeline** project from the list.
3. Click **Build Now** to run the test assets from Jenkins.

Results

You have run the test from the Jenkins server.

What to do next

You can view the build logs by clicking the build number from the **Build History** pane, and then selecting the **Console Output** option. You can also view the history of test results in a tabular format by clicking the **Test Results Analyzer** option.

You can access the report URLs to view the test run information at any point in time. You can also view the test reports and logs of the test that was run on the Jenkins server from the **Results** page on Test Hub. See Test results and reports on page 985.

Integration with Xray test management for Jira

When you use Jira as a change management system on HCL DevOps Test Hub (Test Hub) and you use Xray as the test management application in Jira, you can integrate Xray with Test Hub. With this integration, you can export the test results of the tests that run on Test Hub to your Jira project. You can then view and use the test results in your project in Jira.

Prerequisites

Before you integrate Xray with Test Hub, you must have completed the following tasks:

- Configured Jira as a change management system on Test Hub. See Configuration of Atlassian Jira as a change management on page 1093.
- Installed the Xray test management app in Jira.

  Note: The Xray integration is supported only when the Xray test management application is installed on a Jira cloud.

- Ensured that you obtained the access and authentication details to Jira from the Jira administrator.
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.

Task flow

You can follow the tasks that are listed in the following table when you want to export test results to your Xray test project in Jira. The table also provides you the links to the information about the tasks.

Task	Go to...
Generate the Xray API keys for your username on Jira.	Generating API keys for the Xray app in Jira on page 1189
Use the generated Xray API keys to configure the Xray integration in Test Hub.	Configuring the Xray integration on page 1190
Select an issue in your Xray project in Jira as a location to export the test results of a test run. You must select the issue while you configure a test run on Test Hub.	Selecting a Jira Xray issue to export test results on page 1191

You can manage the Xray settings that you configured in Test Hub. You can edit the saved configuration or you can delete the configuration. For more information, see Managing the Xray configuration on page 1193.

Generating API keys for the Xray app in Jira

You must generate API keys for the Xray app in Jira and use them to configure the Xray integration in HCL DevOps Test Hub (Test Hub). The Xray API keys are used by Jira to authenticate the requests from Test Hub.

Before you begin

You must have completed the following tasks:

- Read about the prerequisite tasks in Prerequisites on page 1189.
- Verified that you can access Jira.

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.

1. Log in to Jira.
2. Click **Settings > Apps**.

   The apps that are installed in Jira are listed in the navigation pane.
3. Find the **Xray App** in the navigation pane, and then click the **API Keys** option.

   The **API Keys** page is displayed.
4. Click **Create API Key**.

   The **Create API Key** dialog is displayed.
5. Enter your username in the **User** field, and then select the username that is displayed.
6. Click **Generate**.

   The API keys are generated and displayed under the **Client ID** and **Client Secret** fields.
7. Click the **Copy** icon in the **Client ID** field to copy the key, and then paste the copied key into a text file.
8. Click the **Copy** icon in the **Client Secret** field to copy the key, and then paste the copied key into a text file.

Results

You have generated the Xray API keys in Jira.

What to do next

You must use the Xray API keys to configure the Xray integration in Test Hub. See Configuring the Xray integration on page 1190.

Configuring the Xray integration

You can set up the Xray integration when you want to export test results from HCL DevOps Test Hub (Test Hub) to an issue in your Xray project in Jira. You must configure the Xray integration in your project in Test Hub before you select the Xray project in Jira as the location to export the test results after the test runs in Test Hub.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Ensured that you must have set up Jira as the Change Management system to add Jira Xray Integration.
- Generated the Xray API keys in Jira and saved the keys in a text file. See Generating API keys for the Xray app in Jira on page 1189.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Integration** in the navigation pane.
   The **Integration** page is displayed.
4. Click **Add Jira Xray Integration** option within **Jira Xray Test Management** section.

   The **Add Xray Integration** dialog is displayed.
5. Perform the following steps:
   a. Enter the Xray API key that was generated for the **Client ID** in the **Client ID** field.
   b. Enter the Xray API key that was generated for the **Client Secret** in the **Client Secret** field.
   c. Click **Save**.

      The Xray configuration is saved and displayed under the **Jira Xray Test Management** pane.

Results

You configured the Xray integration on Test Hub.

What to do next

You can perform the following tasks:

- View the following information on the **Jira Xray Test Management** that you used to configure the Xray integration:
  - The URL of the Jira server on which the Xray test management app is installed.
  - The project that you selected in Jira.
  - The Xray API key that is used as the Client ID.
- Select an issue in Jira as the location to which you can export the test results of a test run while you configure a run of the test on Test Hub. See Selecting a Jira Xray issue to export test results on page 1191.

Selecting a Jira Xray issue to export test results

After you configure the Xray integration on HCL DevOps Test Hub (Test Hub), you can select a Jira Xray issue in Jira or create a new issue in Jira to export the test results. The test results are exported to the selected issue after the tests run on Test Hub. You can then view and track the test results in your Jira project.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Configured the Xray integration by using the Xray API keys that are generated for your username. See Configuring the Xray integration on page 1190.
- Ensured that the **Execute test asset** dialog is displayed and you have configured the other settings for a test run in Test Hub.

About this task

When you configure a test run in Test Hub, you can select a Jira issue as the location where you want the test results to be exported. After the test is run, the results are exported to the Jira issue. You can then view and track the test results in the Jira issue from your project in Jira.

1. Click the **RESULTS** tab in the **Execute test asset** dialog.

   The **RESULTS** tab is displayed.

   Note: The **Export report options** field is enabled only if you configured the Xray integration with the Xray API keys that are generated for your username in Jira.

2. Perform the actions listed in the following table to select an option to export the test results:

Export options	Results in...	Actions
Jira Xray	The test results of the test that you configure for a run are exported to the selected issue after the test run is completed. The results are exported to the selected issue that exists in your Xray project in Jira.	Perform the following steps: a. Select the **Jira Xray** option. The **Select issue** field is displayed. b. Click the field to view the list of issues in the Xray project in Jira. Alternatively, you can enter the issue key in the Jira Xray project. c. Select the issue to which you want to export the test results. d. Continue with the next step.
New Jira item	The test results of the test that you configure for a run are exported to a new Jira issue. The issue is created in your project in Jira after the test run is completed.	Perform the following steps: a. Select the **New Jira item** option. b. Continue with the next step.

Export options	Results in...	Actions
None	The test results are not exported. **Note:** This export option is the default option.	Perform the following steps when: ◦ You do not want to export the test results. ◦ You selected a Jira issue in a previous run of the same test, and you do not want to export the results of the current run. a. Select the **None** option. b. Continue with the next step.

Results

You selected an issue in your Xray project or created an issue in Jira as a location to export the test results of a test run on Test Hub.

What to do next

After the test run is completed, you can go to the **Results** page and expand the test result to view the Xray issue key to which the test results were exported. You can then click the Xray issue key link to open the issue and view the details of the issue in Jira.

Managing the Xray configuration

You can manage the configuration of the Xray integration that you set in HCL DevOps Test Hub (Test Hub) when you want to modify the API keys configured or when you want to delete the existing Xray configuration.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See Managing access to the server projects on page 1220.
- Ensured that you obtained the access and authentication details to Jira from the Jira administrator.
- Configured the Xray API keys for the Xray integration on Test Hub. See Configuring the Xray integration on page 1190.
- Regenerated the Xray API keys in Jira by following the steps to generate the Xray API keys in Generating API keys for the Xray app in Jira on page 1189.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Integration** in the navigation pane.
   The **Integration** page is displayed.

4. Go to the **Jira Xray Test Management** and within the **Configuration details of Jira Xray** section the details about the Xray integration is configured are displayed.
5. Perform any of the following actions when you want to update or delete the Xray configuration:
    - To update the Xray API keys that were set previously in the Xray configuration, go to Step 6 on page 1194.
    - To delete the Xray configuration, go to Step 7 on page 1194.
6. Perform the following steps to update the Xray API keys:

    a. Click the **Update Credentials** icon  of the Xray integration that you want to manage.
    The **Update Xray Integration** dialog is displayed.

    b. Enter the Xray API key that was generated for the **Client ID** in the **Client ID** field.

    c. Enter the Xray API key that was generated for the **Client Secret** in the **Client Secret** field.

    d. Click **Update** to save the updated Xray API keys in the Configuration details of Jira Xray.

    The Xray configuration is updated and a confirmation message is displayed.
7. Perform the following steps to delete the existing Xray configuration:

    a. Click the **Delete** icon  .
    The **Delete Xray configuration** dialog is displayed.

    b. Click **Confirm**.

    The Xray configuration is deleted and a confirmation message is displayed.

 **Note:** The links to the issues that you created to export the test results are not deleted from the results panel in Test Hub even after you delete the Xray configuration.

Results

You achieved the following results:

- Updated the Xray configuration.
- Deleted the Xray configuration.

Access management for DevOps Test Hub

Desktop clients and third-party integrations use offline user tokens to connect to HCL DevOps Test Hub (Test Hub).

You can configure desktop clients and third-party integrations to access Test Hub by using the offline user tokens generated from the server.

Note: Offline tokens remain valid indefinitely. However, if a token is unused for 30 days, it expires. When you use an offline token, it uses the permissions that are currently assigned to your user account.

You can use offline user tokens to perform the following tasks:

- Retrieve secrets configured on the server to be used in certain tests in HCL DevOps Test Integrations and APIs (Test Integrations and APIs).
- Enable publishing of test results and reports from desktop products that include Test Integrations and APIs, HCL DevOps Test Performance (Test Performance), or HCL DevOps Test UI (Test UI) to Test Hub.
- Enable the Resource Monitoring Service in Test Performance to monitor a data source during a run on Test Hub.
- Enable agents to connect with Test Hub.
- Enable HTTP proxies to connect with Test Hub.
- Enable third-party integrations.

See the following table for a list of integrations for which you can use the tokens.

Integrations	More information
Test Integrations and APIs with Test Hub	See Test Integrations and APIs and Test Hub.
Test Performance with Test Hub	See Publishing test results to the server. See Enablement of Resource Monitoring services for a schedule.
Test UI with Test Hub	See Publishing test results to the server.
Third-party integrations	See Integrations with other applications on page 1123.

Creating an offline user token in DevOps Test Hub

You can create an offline user token, copy it, and then use that token to enable connections to be authenticated by HCL DevOps Test Hub (Test Hub).

Before you begin

You must have logged in to the Test Hub with an active user account.

1. Click the **User** icon  from the menu bar, and then select **Create Token**.

   The **Create Offline User Token** dialog is displayed with a token.

   Note: After you copy the token, it is no longer accessible because the dialog closes.

2. Click **Copy** to copy the token.

The dialog closes.

3. Paste the copied token in a text file and save it for reference.
4. Use the token in the UI of the desktop products or in the application where this token is required.

Results

You have created an offline user token.

Deleting an offline user token in DevOps Test Hub

You can delete all the offline tokens when you want to revoke server access or when the existing tokens are no longer required.

Before you begin

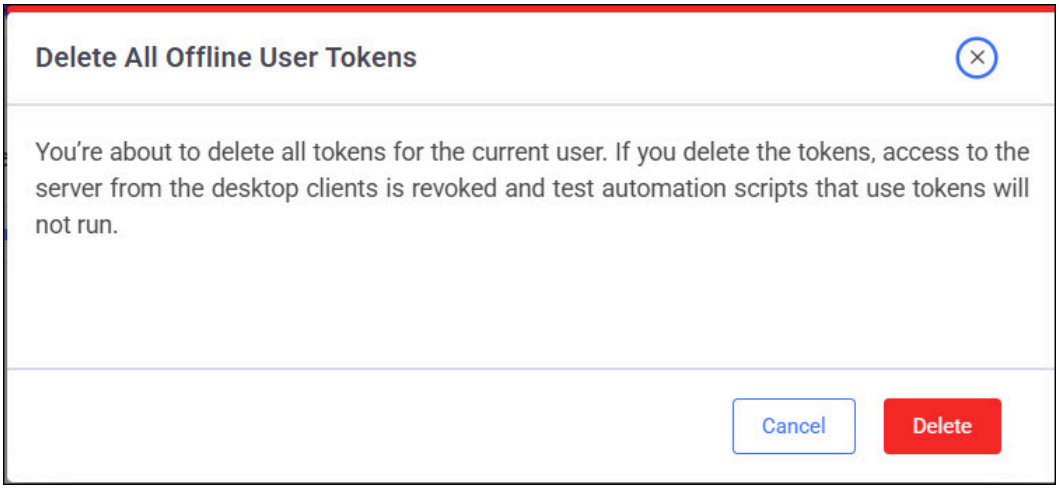
You must have logged in to Test Hub with an active user account.

About this task

Offline tokens provide access for desktop clients and third-party integrations to connect to the server.

Warning: Deleting offline tokens invalidates all existing offline tokens. Any desktop client, automated agent, proxy, or third-party integration that currently uses these tokens will lose connection to the server and fail to authenticate until a new token is configured.

1. Click the **User** icon from the menu bar, and then select **Delete Tokens**.
   The **Delete All Offline User Tokens** dialog is displayed with a warning.



2. Click **Delete** to confirm deletion.
3. Perform any of the following necessary actions:
   - Inform the affected desktop client users that the existing tokens were deleted.
   - Update any test automation scripts that used the deleted tokens to prevent authentication failures.

What to do next

If you need to restore connections for desktop applications, agents, or integrations, you must complete the task to generate a new offline user token. See Creating an offline user token in DevOps Test Hub on page 1195.

Working with DevOps Test Hub REST services

When you want to use the REST APIs of HCL DevOps Test Hub (Test Hub) in your test automation scripts, you can view the REST APIs that are exposed by Test Hub. You can also run the APIs from within your project in Test Hub.

All REST services are accessible from the base URL of Test Hub and are defined by using Swagger. A navigable user interface lists the available set of APIs at `<Test_Hub_URL>/test/#/Initial/Admin/docs`. This Swagger file describes the individual APIs along with details of the parameters and response bodies.

You must be assigned the role of an administrator to be able to view or run the REST APIs.

After you log in to Test Hub with your credentials that has administrator privileges, you must go to the following URL: `<Test_Hub_URL>/test/#/Initial/Admin/docs`

The Swagger docs that are available can be viewed from the drop down list.

You can select the Swagger documentation for the REST APIs that you want to view or use. For example, you can select the **Execution API (Public)** to view the APIs for the execution micro-service.

Note: Only the Execution API is public, and other APIs are internal and subject to change. The Swagger UI indicates whether an API is considered as public.

You can perform the following steps to view or run the API:

1. Expand the section to view the APIs that are available.
2. Expand each of the APIs to view the details such as the description, examples, parameters, and responses.
3. Click **Try it out**.
4. Enter the values for the required parameters.
5. Click **Execute** to run the API.

You can view the response after the API runs.

You can view the API doc for the execution micro-service that is extracted in this PDF document.

Related information

Running APIs in command line on page 1197

Running APIs in command line

You can run the APIs in the command line by adding a bearer access token to authorization header in every API call.

About this task

To generate the bearer access token, you must create an offline token.

1. Click the **User** icon  , and select **Create Token** to obtain an offline token.

   Result

   The token is automatically generated and displayed in the **Create Offline User Token** dialog.
2. Click **Copy** to copy the token and save it in a secure location.

   You may assign the token to variable `OFFLINE_TOKEN` for using in the following steps.
3. Generate an access token by sending a POST request to `<baseUrl>/test/rest/tokens/` with the post data below:

   `refresh_token=$OFFLINE_TOKEN`

   Result

   The response JSON displays an 'access_token' field.
4. Add HTTP header Authorization with the access token in every API call:

   Authorization: `Bearer $access_token`

   The following example uses a bash command to start the execution:

```
TEST_HUB_URL=UNSET_URL
PROJECT_ID=UNSET_PROJECT_ID
BRANCH=UNSET_BRANCH
ASSET_PATH=UNSET_ASSET_PATH
if [ -n "$OFFLINE_TOKEN" ] && [[ "$PROJECT_ID" != *"UNSET"* ]] && [[ "$TEST_HUB_URL" != *"UNSET"* ]] &&
 [[ "$ASSET_PATH" != *"UNSET"* ]]; then \
  CURL_CMD="curl -k"; \
  echo "Getting access token from $TEST_HUB_URL/test/rest/tokens/"; \
  access_token=$($CURL_CMD -H "Accept: application/json" \
  -d refresh_token=$OFFLINE_TOKEN $TEST_HUB_URL/test/rest/tokens/ | grep -oP
 '"access_token"\s*:\s*"\K[^"]*'); \
  POST='{"testAsset": {"revision": "'$BRANCH'", "name": "'$ASSET_PATH'"},"offlineToken":
 "'$OFFLINE_TOKEN'"}'; \
  echo "Posting execution to $TEST_HUB_URL/test/rest/projects/$PROJECT_ID/executions/"; \
  execution_rsc=$($CURL_CMD -H "Content-Type: application/json" -H "Authorization: Bearer $access_token"
 \
        -d "$POST" $TEST_HUB_URL/test/rest/projects/$PROJECT_ID/executions/); \
  echo $execution_rsc; \
else \
  echo "OFFLINE_TOKEN env variable is not set (UI->Account->Create Token) or PROJECT_ID/ASSET_PATH is
 not set"; \
fi
```

Results

You ran APIs outside of the Swagger UI by using an offline token.

Chapter 7. Test Manager Guide

This guide describes how to manage and track your overall test effort. This guide is intended for test managers.

Related information

Managing access to the server projects on page 1220

Managing team spaces

You might want to become a team space member, add members to your team space, change a user role in a team space, modify the team space, or delete a team space.

Each team space is associated with at least one *Team Space Owner*. You can assign multiple owners to your team space. Any user who becomes a member of a team space and can be assigned the role of a *Member*, *Project Creator*, *Architect*, or *Team Space Owner* of a team space, can access the team space.

As an owner of a team space, after you create a team space, you must configure the team space license. You can then add members to the team space and can assign different roles to each member of the team space. You can update or delete a team space. You can also add a repository to a team space to store a system model and edit the repository settings. Apart from adding a repository, you can add, update, or delete the Resource Monitoring agent if no longer required.

As an *Architect* of a team space, you can manage a team space repository and design a system model.

As a *Project Creator* of a team space, you can create a project in the team space.

You can view the configured agents or intercepts of your team space.

Viewing team spaces

After you log in to HCL DevOps Test Hub (Test Hub), you can view all the existing team spaces on the **Team Space Dashboard** page.

About this task

As a member of a team space, you can view your team spaces under **My Team Spaces**. You can also view the team spaces that are created by other users as the public team space and the Permissive team spaces under **Other Team Spaces**.

If the initial team space does not exist, then you can view the **Team Space Dashboard** without any navigation pane.

You as a member of a team space, when you want to use that team space immediately after you log in to Test Hub, you can suffix the alias in the application URL.

For example, if the alias of the team space is `first`, then you must use the following Test Hub URL in the address bar:

```
<server url>/#/first
```

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane, and then click **Join Team Space**.
   Result
   The **Team Space Dashboard** is displayed.

   You can view the team spaces in the following panels:
   - **My Team Spaces**: You can view all the team spaces that you created. You can also view the team spaces in which you are a member. You can create a new team space by clicking **New Team Space** option.
   - **Other Team Spaces**: You can view all the team spaces that are created by others.

   Note: You cannot view the Restrictive private type team spaces.

Results

You have viewed the team spaces that are created by you and other users on the **Team Space Dashboard**.

What to do next

You can select a team space and can send a request to become a member of that team space. See Becoming a team space member on page 1205.

Adding members to a team space

When you create a team space to manage projects on HCL DevOps Test Hub (Test Hub), you must add users to your team space so that they can create and access the projects.

Before you begin

- As an administrator, you are assigned the default role of a *Team Space Owner*. If you log in to Test Hub as another user, then you must have been assigned the role of a *Team Space Owner*.
- As an owner of a team space, you must have completed the following tasks:
  - Created a team space. See Creating a team space on page 175.
  - Configured a license for the team space. See Configuring licenses for team spaces on page 184.

You must have been assigned a role as a *Team Space Owner*. If you are a server administrator, then you are assigned with the default role of a *Team Space Owner*.

About this task

As an owner of a team space, you can add multiple users to your **Permissive** or **Restrictive** team space.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.

2. Click the **Switch Team Space** icon  from the left navigation pane.

   Note: If the initial team space does not exist, then you can view the **Team Space Dashboard** page without any navigation pane.

3. Identify the team space for which you want to add a member.
   You can either use the same team space where you choose to work or switch to any other team space.
4. Click **Manage > Configuration**.
5. Click **Add member** to add members in the initial team space.
   Result
   The **Add Member** page is displayed.
6. Enter a username in the **Add Members** dialog to add users.
   Result
   The list of users is displayed.
7. Select the user.
8. Assign a role to the selected user.

   Note: All members are assigned the default roles of a *Project Creator* and *Architect*.

   You can choose the role for the user based on its associated actions:

Roles	Responsibilities
Team Space Owner	◦ Create a team space. ◦ Update and delete a team space. ◦ Add members to a team space. ◦ Remove members from the team space. ◦ Assign the roles to the members. ◦ Modify the roles of the members. ◦ Create and modify a project. ◦ Add or update the system model. ◦ Add a repository to store a system model. ◦ Update and delete a repository in a team space. ◦ Configure or update agent and intercepts. ◦ Register, update, delete, and list Resource Monitoring sources in a team space.
Project Creator	◦ Create, update, and view a project ◦ Configure or update agent and intercepts.
Architect	◦ Create, update, delete, and view a repository in a team space. ◦ Create, update, and delete components in the system model. ◦ Configure or update agent and intercepts.

Roles	Responsibilities
Member	◦ View the list of team spaces in the **Team Space Dashboard.** ◦ View the configuration of the team space. ◦ Configure or update agent and intercepts. ◦ View the registered agents, or intercepts. ◦ View the Resource Monitoring agents in a team space. ◦ View the registered Resource Monitoring agents. ◦ View components in the system model.

Notes:

- The members with the assigned role of a *Team Space Owner*, *Project Creator*, *Architect*, or *Member* of a team space can be assigned with any of the following project roles:
  - *Project Owner*
  - *Tester*
  - *Viewer*
- You can add or remove the role of a member by using the **Edit roles for members** icon next to each user in the **Members** list.

9. Repeat the applicable steps to add another user.

Results

You have added members with assigned roles to your team space. You can view the list of all added users on the **Team Space Configuration** page.

What to do next

You can reassign or remove the roles of users to one or more team spaces that you own. See Managing members and their roles in a team space on page 1207.

All users can request to be a member of another team space. See Becoming a team space member on page 1205.

Users added as members to the team space can view the team space in **My Team Spaces**.

Modifying the configuration of a team space

After you create a team space on HCL DevOps Test Hub (Test Hub), if you want to modify the details of the team space, add a repository to your team space, add or remove members of a team space or delete a team space, then you can perform these tasks from the **Team Space Configuration** page.

Before you begin

You must have been assigned the role of a *Team Space Owner*.

About this task

The **Team Space Configuration** page displays details of the team space.

As an owner of a team space, you can modify details of the team space, add members to your team space, and assign roles to the members of the team space on the **Team Space Configuration** page.

The *Team Space Owner* and *Architect* can add or remove a repository from the team space.

If the team space is no longer required, then the *Team Space Owner* can also delete that team space.

As a *Member*, *Project Creator*, or *Architect*, you can only view the configuration details of a team space on the **Details** tab of the **Team Space Configuration** page.

As a *Member*, and *Project Creator* you can only view the configuration details of a team space on the **Repository** tab of the **Team Space Configuration** page.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon from the left navigation pane.
3. Identify the team space for which you want to modify the configuration.
   You can either use the same team space where you are working or switch to any other team space.
4. Click **Manage > Configuration**.
   Result
   The **Configuration** page is displayed.
5. Perform the modifications on the following details of the **Configuration** page of the team space:

Fields	More information
Name	Displays the name of the team space.
URL alias	Displays an alias of the team space. You must suffix this alias in the application URL to access a specific team space after you log in to the application.
Member Access	Displays two options for Member Access:: ◦ Approval required to join the team space. ◦ Team space can be discovered by potential members.:
Description	Brief description of the team space.
Members	List of members in a team space.
Repositories	Displays the details of the repository that host the system model.

Results

You have modified the configuration details of the selected team space.

What to do next

You can add members and assign roles to each member of your team space.

Related information

Adding members to a team space on page 1200

Managing members and their roles in a team space on page 1207

Adding repositories to a server project on page 1214

Viewing the configuration of a team space

You can view the configuration of a team space as a member of any team space on HCL DevOps Test Hub (Test Hub). You can view the configuration details on the **Team Space Configuration** page.

Before you begin

You must be a member of a team space.

About this task

The **Team Space Configuration** page displays the configuration details of the team space.

As a member of a team space, if you want to access your team space immediately after logging into Test Hub, then you must suffix the alias in the application URL. You can find details of the alias on the **Team Space Configuration** page. You can also view the list of members and their roles in that team space on the **Team Space Configuration** page.

1. Log in to Test Hub.
2. Click the **Switch Team Space** icon  from the left navigation pane.
3. Identify the team space for which you want to view the configuration.
   You can either use the same team space where you are working or switch to any other team space.
4. Click **Configuration** under **Manage** tab.
   Result
   The **Configuration** page is displayed.

   You can view the following details about the team space:

Fields	Tabs	More information
Name	**Details**	Displays the name of the team space.
URL alias	**Details**	Displays an alias of the team space. You must suffix this alias in the application URL to access a specific team

Fields	Tabs	More information
		space after you log in to the application.
Member Access	**Details**	Displays two options for Member Access:: ◦ Approval required to join the team space. ◦ Team space can be discovered by potential members.:
Description	**Details**	Displays a brief description of the team space.
Members	**Details**	Displays a list of members of the team space.
Repositories	**Details**	Displays the details of the repository that host the system model.

Results

You have viewed the configuration details of the selected team space.

Becoming a team space member

After logging into HCL DevOps Test Hub (Test Hub), the licensed users must be a member of any team space to perform test activities in a project. As a licensed user, if you are not assigned the role of a member in any team space, then you can request the *Team Space Owner*to become a member of any existing team space.

Before you begin

You must be a licensed user.

About this task

If you want to become a member of a **Restrictive** team space, then only the owner of that team space can grant you the role of a member of a team space. If you request to become a member of a **Permissive** team space, then you automatically become a member of that team space.

Note: The server administrator can join any team space automatically without any approval and is assigned with a default role of *Team Space Owner*.

1. Log in to Test Hub.

   Result

   The **Projects** page of the initial team space is displayed.

2. Click the **Switch Team Space** icon from the left navigation pane, and then **Join Team Space**.
   Result
   The **Team Space Dashboard** is displayed.

   You can view the team spaces in the following panels:
   - **My Team Spaces**: You can view all the team spaces that you created. You can also view the team spaces in which you are a member.
   - **Other Team Spaces**: You can view all the team spaces that are created by others.

   Note:

   You can view only the public type of **Restrictive** team spaces and the **Permissive** team spaces. You cannot view the **Restrictive** private type team spaces.

3. Select the team space that you want to join from the list of other team spaces in the **Team Space Dashboard**.
4. Click the **Key** icon of the selected team space.
5. Confirm your request when prompted.

   If you agree, then the *Team Space Owner* sees a notification about the request.

   After opening the notification, the *Team Space Owner* can view the user who is requesting the access and can accept or decline the request. If the *Team Space Owner* accepts your request, you are added and assigned a *Member* role.

   Notes:
   - You can become a member of any team space only when the license of the selected team space is configured.
   - If you are a server administrator, then the *Team Space Owner* role is the default role assigned.
   - If you are a licensed user, then the *Member* role is the default role assigned and is restricted to specific actions.

   You can then see the team space under **My Team Spaces** on the **Team Space Dashboard** page.

   You might want to follow up with the *Team Space Owner* if your request is pending for some time.

Results

You have become a member of another team space.

What to do next

You can view the actions that you can perform based on your assigned role. See Adding members to a team space on page 1200.

Managing members and their roles in a team space

After you, as a *Team Space Owner* assign roles to the members of a team space onHCL DevOps Test Hub (Test Hub), you might want to remove a member from the team space or reassign the role to the existing members of your team space.

Before you begin

You must have been assigned the role of a *Team Space Owner*.

About this task

As a licensed user, you are assigned a role of a *Member* in any team space after the approval of the *Team Space Owner*.

As an owner of a team space, you can remove or reassign any member of your team space with the roles of a *Team Space Owner*, *Project Creator*, or *Architect*. You can also remove any member from the list of members of the team space.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon from the left navigation pane.
3. Select the team space for which you want to reassign the roles of the members or remove a member from the list of members.
   You can either use the same team space where you are working or switch to any other team space.
4. Identify the member from the list of **Members**.
5. Perform any one of the following operations to the identified member of the team space:
   Choose from:
   - Click **Remove member** to remove the identified member from the team space.
   - Click **Edit roles for member** and select the role that you want to assign or remove the assigned role for the identified member.

     You can choose multiple roles for the members based on the associated actions:

Roles	Actions
Team Space Owner	▪ Create a team space. ▪ Update and delete a team space. ▪ Add members to a team space. ▪ Remove members from the team space. ▪ Assign the roles to the members. ▪ Modify the roles of the members. ▪ Create and modify a project. ▪ Add or update the system model. ▪ Add a repository to store a system model. ▪ Update and delete a repository in a team space.

Roles	Actions
	▪ Configure or update agent and intercepts. ▪ Register, update, delete, and list Resource Monitoring sources in a team space.
Project Creator	▪ Create, update, and view a project. ▪ Configure or update agent and intercepts.
Architect	▪ Create, update, delete, and view a repository in a team space. ▪ Create, update, and delete components in the system model. ▪ Configure or update agent and intercepts.
Member	▪ View the list of team spaces in the **Team Space Dashboard.** ▪ View the configuration of the team space. ▪ Configure or update agent and intercepts. ▪ View the registered agents, or intercepts. ▪ View the Resource Monitoring agents in a team space. ▪ View the registered Resource Monitoring agents. ▪ View components in the system model.

Note: The *Team Space Owner*, *Project Creator*, *Architect*, or *Member* of a team space can be assigned with any of the following project roles:

- *Project Owner*
- *Tester*
- *Viewer*

Similarly, you can remove the assigned role of any member of your team space.

Results

You have completed any of the following tasks in your team space:

- Reassigned the role of the members.
- Removed the assigned role of the members.
- Removed the members.

Related information

Adding members to a team space on page 1200

Deleting a team space

You might want to delete a team space on when it is no longer needed.

Before you begin

You must have been assigned the role of a *Team Space Owner* of HCL DevOps Test Hub (Test Hub).

About this task

As a *Team Space Owner*, you can delete a team space that you do not need. When you delete a team space, it is a permanent deletion. You cannot undo the delete action on a team space.

If you delete the initial team space, then you can only view the **Team Space Dashboard** page without any navigation pane.

After you delete the team space, you can use the name of the team space again.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **Switch Team Space** icon  from the left navigation pane.
3. Identify the team space that you want to delete.
   You can either use the same team space where you are working or switch to any other team space.
4. Click **Manage > Configuration**.
   Result
   The **Configuration** page is displayed.
5. Click the **Delete Team Space** icon  .
6. Confirm that you want to delete the team space when prompted.

   If you proceed, the team space is deleted including all of its projects and members.

Results

You have deleted the selected team space.

What to do next

You can view the available team spaces on the **Team Space Dashboard** page. See Viewing team spaces on page 1199.

After you delete a team space, the license that is used for that team space is released and is available to configure the other team spaces, if required. See Configuring licenses for team spaces on page 184.

Managing DevOps Test Hub projects

HCL DevOps Test Hub (Test Hub) projects manage access to your test assets, which are stored in a Git repository. Projects are maintained in a team space. Projects are either public by default or private. Private projects are not discoverable by other users. You can either add your own project or you can request to be a member of another public project.

All the licensed users are assigned with a default role of a *Project Creator*. As a *Project Creator*, when you create a project immediately after logging in the application, the project is created in the initial team space. If you want to create a project in any specific team space, then you must become a member of that team space. When you add your own project, you can configure it now or later. Configuring the project includes adding details about the project, adding one or more Git repositories, optionally adding secrets, classifying encrypted datasets, and adding a change management system. To run test assets that are associated with an encrypted dataset, you must categorize the encrypted datasets by creating a security group. You can also add users to your project to access your test assets only if the users are the members of that team space. To use Jira to create defects for the tests available in your project, you must configure the Jira server for your project on Test Hub.

Related information

Team space overview on page 172

Becoming a team space member on page 1205

Becoming a project member on page 1220

Managing access to the server projects on page 1220

Managing repositories on page 1226

Protecting test assets by using secrets on page 1235

Managing an encrypted dataset on page 447

Default user administration on page 142

Configuring Jira as a change management on page 1094

Working with projects in a team space

After you log in to HCL DevOps Test Hub (Test Hub), you must have a project to perform the test activities. To work with projects, you must create and configure projects.

Before you begin

You must have logged into Test Hub.

About this task

You must be assigned with the role of an *Owner*, *Tester*, or *Viewer* of a project.

1. Go to the **Projects** page from the **Initial Team Space** Home page. See Viewing active projects on page 1218.
2. Add a project. See Creating a project on page 1213.
3. Add a repository. See Adding repositories to a server project on page 1214.
4. Add secrets in a security group. See Secrets configuration on page 1217.
5. Encrypt the data set resources. See Managing an encrypted dataset on page 447.
6. Configure a change management system. See Configuration of a change management on page 1092.
7. Add members to a project and configure their roles. See Adding users to a project on page 1217.

8. Select a global branch in a project to view the test assets or resources maintained in the repositories. See Selecting the global branch in a project on page 1233
9. Perform the following tasks based on the type of test assets or resources that you want to run:
    a. Add agents to your project, if you are using remote agents as a location to run certain tests. See Adding a performance agent to a project on page 546.
    b. Read the considerations that you must take into account before you configure a run or start a virtual service. See Prerequisites for running HTTP virtual services on page 938.
    c. Read the considerations that you must take into account before you configure a test run. See Prerequisites to running tests on page 527.
10. Configure a run of the test asset or resource. See Test run configurations on page 610.
11. View the progress of a test run. See Viewing the progress of running test assets on page 926.
12. Manage a running test. See Management of running tests on page 922.
13. View the results of a test run after the run is completed. See Test results on page 983.
14. Review the test results and create defects for test runs from the Results page. See Creating Jira issues for the test results on page 1096.

Results

You have performed all the tasks with the projects.

Repository considerations for a server project

To collaborate with other project stakeholders, you can open test assets from a local clone of a remote repository, pull project test assets from the remote repository, and push changes made to your local test assets to the remote repository. Before you add a project and then add a repository to that project, you must consider some information about repositories.

Consider the following sections about version control systems that are used to manage remote repositories.

The following types of Git-based version control systems are supported:

- GitHub, which is an open-source product for storing and collaborating code.
- Azure Repos, which is part of a solution of Azure DevOps, and focuses on collaboration between DevOps professionals to develop private code.

Both these version control systems support Git repositories to store code.

You must be familiar with the various aspects of these version control systems to manage the application code that you want to test in HCL DevOps Test Hub (Test Hub).

You can learn more about installing, setting up, and using Git from the Git documentation.

You can learn more about installing, setting up, and using Azure Repos from the Azure Repos documentation.

Note: Git is the only tool that is supported to manage a repository in Azure Repos. Other tools, which can be used in Azure Repos, are not supported.

Git

You must install Git or upgrade to the supported version if you already have installed Git. See System Requirements for DevOps Test Hub 2026.09 (12.0.0) on page 94.

Repositories and user identities

After you install Git, you must set up your Git repository and set up access for members. You must ensure that the repository contains your test assets.

Optionally, you can use a command line utility or Git tool to access the repository, upload your test assets, fetch or pull from the repository, push to the repository, clone the repository, and other operations that you want to perform in Git.

Local and shared repositories

After you create a remote or shared repository in Git, you can create a local version of the repository by cloning the remote repository. You must ensure that your test assets are available in the remote repository and are also cloned to the local repository.

Alternatively, if your test assets are on your local system, you can set up a Git repository in the bare mode, add the project files to the local repository, and then commit and push from the local repository to the remote repository in Git by using your preferred method.

Note: While copying the test assets from your local system to the repository, you must ensure that you copy the entire project that contains the test assets.

User authentication for the Git repository

The administrator can set up different types of authentications for accessing the Git repository. Test Hub supports the following authentication types:

- HTTP with username and password
- HTTP without username and password
- HTTPS with username and password
- HTTPS without username and password
- SSH with SSH key and passphrase
- SSH with SSH key and without a passphrase

Based on the authentication type that is set for a repository, you must provide the same authentication values in Test Hub when you add a repository.

Test assets

You must complete the following tasks in the desktop products where you author your test before you check in and commit the test assets to the Git repository.

Test type	Task	More information
API suite in Test Integrations and APIs	Change the local stub to a remote stub.	See Test run considerations for API Suites or API tests on page 529.
	Add the library files.	

Creating a project

You can create a project and enter its details.

Before you begin

To create and manage projects for your test assets, you must log in to HCL DevOps Test Hub (Test Hub) by providing the application URL in a browser.

If you are a new user, and LDAP and Active Directory are not configured, then you must first sign up by filling out a form with your email, username, and password. You can then log in by using those details.

You must be a member of any licensed team space.

About this task

As a licensed user, you are by default assigned as both *Project Creator* and *Architect* role for the initial team space. You must select a team space to create a project.

You can view all the team spaces in the **Team Space Dashboard**. To create a project in any team space, you must be a member of that team space and assigned a *Team Space Owner* or *Project Creator* role in that team space.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **New Project** button on the **Projects** page.
   Result
   The **New Project** dialog is displayed.
3. Assign a name to your project.
4. **Optional:** Enter the description of the project.
5. Click **Create**.

Results

You have created a project.

What to do next

You can modify your project configuration. You can also add a repository to your project.

Related information

Default user administration on page 142

Changing the visibility configuration of a project on page 1219

Adding repositories to a server project on page 1214

Adding repositories to a server project

You can add a single repository individually or multiple repositories simultaneously to a server project to access assets and resources that are available in the repositories.

Before you begin

You must have completed the following tasks:

- Added a project on HCL DevOps Test Hub (Test Hub).
- Been granted permission and you must have set up your credentials to access the remote repositories in a repository domain.
- Created test assets and resources that are contained in either a single git repository or multiple git repositories.

About this task

As a project member with the role of a *Project Owner*, you can add a single repository or multiple repositories to your project.

Note: You can only add a single repository that is managed by using Git in Azure Repos. The addition of multiple repositories in Azure Repos is not supported.

You can use the information in the following table to decide the method for adding repositories to your project:

When...	**Action...**
You want to add any of the following repositories: • A single repository owned by you. • A single repository owned by other people. • The repositories that are in different domains.	You can add individual repositories one at a time. You must then provide the complete URL of the repository along with the required credentials to authenticate the connection to the repository from Test Hub. For example, the URL to a single repository can be as `https://github.com/<user>/<repository>.git`. You must provide the *username* and *password* to access this repository.
You want to add any of the following repositories:	You can add all the repositories simultaneously that are in the same domain.

When...	Action...
• Multiple repositories that are owned by you and are contained in the same domain. • Multiple repositories that are owned by others and are contained in the same domain.	You must then provide the URL of the repository domain along with the required credentials to authenticate the connection to the repository from Test Hub. For example, the URL of the repository domain can be `https://github.com`. You must select the platform and provide your *personal access token* to access the repositories.

1. Log in to Test Hub and open the team space that contains your project.
2. Open your project either from the navigation pane or from the **My Projects** page.

   The **Overview** page of the project is displayed.
3. Click **Manage > Configuration** from the navigation pane.

   The **Configuration** page of the project is displayed.
4. In the **Repositories** section, click **Add repository**.

   The **Add repository** dialog is displayed.

   You can either choose to add a single repository individually or multiple repositories simultaneously from a single domain.
5. Perform either of the following actions:
   - To add a single repository, go to Step 6 on page 1215.
   - To add multiple repositories from a domain, go to Step 7 on page 1216.
6. Perform the following steps to add individual repositories at a time:
   a. Enter the complete URL of the repository in the **Git URL** field.

      For example, the URL to a single repository can be as `https://github.com/<user>/<repository>.git`.

      The required authentication fields are displayed that depend on the protocol you specified in the URL.

If...	Then...
The repository URL contains *http* or *https*	The following fields for authentication are displayed: ▪ **Username** ▪ **Password**
The repository URL contains *git@*	The following fields for authentication are displayed:

If...	Then...
	▪ **SSH/Deploy Key** ▪ **Passphrase** You can add a Git repository by using git@ and SSH. For example: ▪ `git@github.com:account/testRepo.git` ▪ `ssh://git@10.111.111.333:2222/~/account/project.git`

Note: If you use SSH to connect to a remote repository and Test Hub displays an `Auth Fail` error, you can resolve this error by regenerating your SSH keys by using the **-m PEM** option.

b. Enter the credentials to authenticate the connection to the repository.

c. Click **Add** to add the repository to your project.

You can add other individual repositories by specifying the URL and performing the actions in Step 6 on page 1215.

7. Perform the following steps to add multiple repositories simultaneously that are contained in the same domain:

Note: You can only add a single repository that is managed by using Git in Azure Repos. The addition of multiple repositories in Azure Repos is not supported.

a. Enter the URL of the repository domain in the **Git URL** field.

For example, the URL of the repository domain can be `https://github.com`.

b. Select the platform from the list under **Platform** where your repositories reside.

c. Enter your *personal access token* to authenticate your connection to the repository domain.

Note: If you do not have your *personal access token*, you can click **Generate** to generate your *personal access token* and use the token to authenticate your connection.

d. Click **Next**.

The connection to the repository domain is established and all the repositories that you own are listed. The time when the repository was last updated is displayed in the **Last updated** column.

e. Select the repositories that you want to add to your project.

f. Click **Add**.

Note: Depending on the size of the repository that you are adding, it can take a few to several minutes to add the repository to your project on the server.

All the repositories that you selected are added to your project and are listed on the **Repositories** page.

Note: If the repository is read-only, you will be able to add it and view its assets, but you will not be able to publish changes directly to it from Test Hub.

Results

You have added individual repositories or multiple repositories to your project on the server.

What to do next

You can perform the following actions on the repository that you added:

- Update the authentication credentials if they are changed in the Git repository configuration. See Updating the authentication credentials of the repository on page 1230.
- Delete a repository if it is no longer required. See Deleting a repository on page 1231.
- Refresh a repository to fetch and synchronize changes from the remote repository. See Refreshing repositories manually on page 1231.

Secrets configuration

If you are working with an API suite and the project test asset contains environment variables that are required for the test runs, you must configure the environment variables as secrets in a security group by using the **Add Secret** option in the **Security** page. Configuring secrets enable the API suite to run correctly on Test Hub.

Related information

Protecting test assets by using secrets on page 1235

Adding users to a project

You can add users to your project in HCL DevOps Test Hub (Test Hub) so that they can access your test assets.

Before you begin

- You own a project.
- The users you want to add to your project must be the members of the team space of that project.

About this task

As a project owner, you can add other users to your public or private project. For example, if you have a large number of test assets that must be run, you might want your test team to help run them, and see the results.

1. Log in to Test Hub.
   Result
   The **Projects** page of the Initial Team Space is displayed.
2. Select the project for which you want to add the users.
3. Click the **Configure Project** icon on the Project card.

   The **Configuration** page of the project is displayed.
4. In the **Project Members** section, click **Add Members**.

   The **Add Members** dialog is displayed.
5. Enter the name or email of the user in the search bar to find the user.
6. Select **Viewer**, **Tester**, or **Owner** role for the user that you want to add.

   The added user after logging in can see the owner's project in their list of **My Projects**.
7. Click **Add Members**.

   The newly added members are listed under the **Project Members** list.

Results

You have successfully added users to your project.

What to do next

You can add or remove roles to one or more projects that you own. All users can request to be a member of another public project.

Related information

Becoming a project member on page 1220

Managing access to the server projects on page 1220

Viewing active projects

After you log in to HCL DevOps Test Hub (Test Hub), you can view both active and archived projects in the **Initial team space**.

About this task

As a project owner, you can view your projects under the **My Projects** tab, and the projects that are created by other users as public projects are displayed under the **Other Projects** tab. You can view a list of all active projects that you created under **My Projects**. You can view a list of all archived projects by selecting the **Include Archived Projects** check box.

Log in to Test Hub.

The **Projects** page is displayed.

You can view the following projects:

- The active projects that you created or added to as a member with either the *Tester* or *Viewer* role are displayed under the **My Projects** tab.
- The active projects that are created by others as public projects are displayed under the **Other Projects** tab.

Results

You have viewed the active projects that are created by you and other users in the team space.

What to do next

You can perform any of the following tasks:

- Create a project. See Adding details to a server project on page 1213.
- Request to become a member of a project that is displayed under the **Other Projects** tab. See Becoming a project member on page 1220.
- View the other pages in the team space and perform tasks on those pages. See Tasks in a team space on page 177.

Related information

Archiving or unarchiving server projects on page 1223

Changing the visibility configuration of a project

After you create a project in HCL DevOps Test Hub (Test Hub), you might want to change the visibility configuration of the project from the public, which is the default, to private.

Before you begin

You must have created a project.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Select the project from **My Projects** tab.
3. Click **Manage > Configuration**.
   Alternatively, select a project, and then click the **Configure project** icon  on the project card.
   Result
   The **Configuration** page is displayed.
4. In the **Visibility** section, select **Private** as the visibility mode of the project.
5. Click **Save**.

Results

You have successfully changed the visibility configuration of a project.

What to do next

You can add a repository to your project.

Related information

Default user administration on page 142

Becoming a project member

You can request to be a member of another project. New users without any projects can also request to be a member of an existing project.

About this task

As a project owner, you can add users to your public or private projects. All users can request to become a member of a public project.

Note: A project owner can add a licensed user as a member of the project only if that user is the member of the team space.

1. Log in to Test Hub.
   Result
   The **Projects** page of the initial team space is displayed.
2. Click the **Other Projects** tab and search for the project that you want to join.
3. Click the Project card or click **Join Project**.
   Only public projects are visible in the list of projects.
4. Confirm your request when prompted. If you agree, the project owner is notified.
5. The project owner sees a pending request **Notification** icon  on their Project card.

   After opening the notification, the owner can see the user that is requesting access and can accept or decline it.

   If the project owner accepts your request, you are added as a member of the project with a **Viewer** role. A **Viewer** role is the default role assigned and is restricted to specific actions.

   You then see the project under **My Projects** tab.

   If the project owner declines your request, follow up with the project owner.

Results

You have sent a request to become a member of a project.

Managing access to the server projects

You might want to add or remove a project member role from your project.

About this task

As a project owner, you can assign access by specifying a role when you add a user to your project. You can also assign more access or remove access for a user. Roles enable users to perform tasks on project resources such as running tests and viewing test results.

1. Choose the role in a project based on its associated actions.

Option	Description
Owner of a project	◦ All Tester's actions ◦ Update, archive, list, and view a project ◦ Assign, update, delete, and list user roles for a project ◦ Accept new member requests ◦ Create, update, and delete a dataset security group in a project ◦ Add, update, and remove an encrypted dataset from a security group ◦ Update current row for a listed dataset in a project ◦ Delete, list and view a report ◦ Update, delete, and list a repository in a project ◦ Get the test assets from a repository in a project ◦ Create, update, delete and list secrets in a project ◦ Create, update, delete, and list resource monitoring sources in a project ◦ Get the content of a resource monitoring source in a project ◦ Register, update, delete, and list resource monitoring agents ◦ Create, configure, and delete the Jira change management system ◦ Create and update defects in the Jira change management system ◦ Create and update components in the system model
Tester of a project	◦ All Viewer's actions ◦ List repositories in a project ◦ Run a test and create a report ◦ Stop a test while running ◦ Get the test assets from a repository in a project ◦ Get a list of datasets in a project ◦ Get the content of a dataset

Option	Description
	◦ Update the current row for a listed dataset in a project ◦ Add, update, and remove an encrypted dataset from a security group ◦ Delete a report if you created the report ◦ Create and update defects in the Jira change management system ◦ Create and update components in the system model
Viewer of a project	◦ List your own roles on a project ◦ List running tests ◦ List and view a report ◦ List resource monitoring sources in a project ◦ Get the content of a resource monitoring source in a project ◦ List the resource monitoring agents ◦ View components in the system model ◦ View defects in the Jira change management system
Non-member of a project	◦ Request to be a member of a project

2. Open the project that you own.
   From the **Configuration** page, you can see a list of members in your project along with their roles and then you can modify their roles and add or remove the members from your project.
3. Click **Manage > Configuration** from the navigation pane.
   From the **Configuration** page, you can see a list of members in your project along with their roles and then you can edit their roles and add or remove the members from your project.
4. To edit the roles of the existing user, perform the following steps in the **Project Members** section:
   a. Hover over the user for whom you want to modify the role and then click the **Edit roles for members** icon.
   b. Select the new role for the user and then click the **Done editing members role** icon to save the modified role for the user.
5. To delete the roles of the existing user, perform the following steps in the **Project Members** section:

a. Hover over the user for whom you want to modify the role and then click the **Remove member from the project** icon.

   The **Remove Member** dialog is displayed.

b. Click **Yes** to remove the user from the project.

Archiving or unarchiving server projects

You can archive a project when that project is no longer needed but you want to retain it for future reference. If you do want to use an archived project, you can unarchive it.

About this task

As a project owner, you can archive inactive projects in HCL DevOps Test Hub (Test Hub). Archived projects are not visible to users or project members, but owners can show or hide their archived projects. You can view the list of all archived projects from the navigation pane. Archived projects can be unarchived.

- To archive a project:

  1. Log in to Test Hub.
  2. Select your project from the **Projects** page.
  3. Click the **Configure project** icon on the project card.
     Result
     The **Configuration** page is displayed.
  4. Click the **Archive Project** icon.

     The **Archive Project** dialog is displayed.
  5. Click **Archive** to confirm that you want to archive the project.

     If you proceed, the project is archived and is hidden from **My Projects**.

- To view an archived project:

  1. Log in to Test Hub.
     Result
     The **Projects** page of the initial team space is displayed.
  2. Select **Include Archived Projects** check box.
     Result
     A list of all the projects including archived projects that are associated with you is displayed under **My Projects** and the archived projects that are associated with others are displayed under **Other Projects**.

- To unarchive a project:

  1. Log in to Test Hub.
     Result
     The **Projects** page of the initial team space is displayed.
  2. To view archived projects, check the box labeled **Include Archived Projects**.
  3. Select your archived project, and then click the **Configure project** icon on the project card.
     Result
     The **Configuration** page is displayed.
  4. Click the **Unarchive Project** icon.

     The **Unarchive Project** dialog is displayed.
  5. Click **Unarchive** to confirm that you want to unarchive the project.
     If you proceed, the project is unarchived and is shown under **My Projects** tab.

Deleting server projects

You might want to delete a project when the project is no longer needed, and you want to free up disk space.

About this task

As a project owner, you can delete a project that you no longer want. When you delete a project it is permanent. You cannot undo the delete project action.

After you delete the project, you can use the project name again.

1. Open the project that you want to delete.
2. From the **Configuration** page, click the **Delete Project** icon.

   The **Delete Project** dialog is displayed.
3. Select the **I understand project deletion is permanent.** checkbox.

   Warning:

   If you proceed, the project is deleted including all of its test assets, results, repository and members.
4. Click **Delete** to confirm that you want to delete the project.

Results

You have deleted a project.

An overview of test assets, modifications, and scheduled runs

After you configure a project with one or more Git repositories and run some test assets, you can view statistics about those test assets.

From the **Overview** page, several cards display charts, sliders, and tables of the statistics of the test assets in your server project. You can click some of the charts, sliders, or tables to see more information, such as the progress of a scheduled run or the results of a run. You can also review some statistics by date and see more details such as the proportion of test assets by type.

You can view a server project, which might contain multiple repositories and multiple branches, by selecting the name of the branch from the **Branch** drop-down list. For more information about selecting branches, see Selecting the global branch in a project on page 1233.

The **Members** area displays the total number of users who have access to the project. You can click the number to view the details of all the members. The **Owner** area displays the username of the project owner.

Test suites

This card displays a slider of the percentage of test suites by type. A test suite is a collection of tests, including schedules, Compound Tests, API Suites, and AFT Suites. You can hover over the slider and see the proportion of test suites by test suite type. The number of suites shown are relative to the selected branch.

Individual tests

This card displays a slider of the percentage of individual tests by type. You might have performances tests, API tests, or UI tests according to the desktop clients that you used to create the tests. You can hover over the slider and see the proportion of individual tests by type. The number of tests shown are relative to the selected branch.

Execution results

This card displays a chart of the proportion of test suites that were run by status and by date. If no date is selected, the card displays all the run results associated with the project since it was created. You can see the run results by clicking a test or Suite status. You can also hover over the chart and see the test or Suite status and the proportion of test assets. You can view statistics by tests or Suites that ran.

Notes:

- The results shown are generated from tests or Suites run.
- The statistics exclude canceled runs of tests or Suites.

- The number of Suites and individual tests that are associated with your project are shown on the **Home** page along with a verdict summary. You can see the run results by clicking a verdict. You can also see the last date and time when you ran one or more test assets.
- The number of suites shown are relative to the selected branch.

Last run

This card displays a table of the last three runs by date and time. You can see the run results by clicking each last run.

New or modified

This card displays a chart of new and modified test assets that were made in your Git repository for a week or for the last seven days based on a selected date range. You can hover your mouse over each bar in the chart and see the proportion of test assets by type. The data shown is relative to the selected branch.

Scheduled runs

This card displays a table of the next three scheduled runs by date and time. You can see the progress of a scheduled run by clicking each scheduled run.

Insights

This card displays a table of the insights on the supported parameters identified from the tests that were run in the project. The table lists the level of confidence with which the analysis of the specific parameter was done, the parameter analyzed, the name of the test asset, and the timestamp of the completed analysis. You can click any item in the row of a specific insight to view the details of that insight on the **Insights** page.

Related information

Managing DevOps Test Hub projects on page 1209

Managing repositories

After you create a project in HCL DevOps Test Hub (Test Hub), you can add repositories that contain test assets and resources to the project. You can also change the repository details, update the user credentials of a repository, delete a repository, create a webhook for the configured repository.

Tasks in a repository

To add a repository in your project and work on the repository, you can perform the tasks listed in the following table:

Task	More information
Adding a repository	Adding repositories to a server project on page 1214
Updating the repository credentials	Updating the authentication credentials of the repository on page 1230

Task	More information
Refreshing a repository	Refreshing repositories manually on page 1231
Creating a webhook for the repository	Creating webhooks on page 1234
Selecting the global branch	Selecting the global branch in a project on page 1233
Deleting a repository	Deleting a repository on page 1231

Adding repositories to a server project

You can add a single repository individually or multiple repositories simultaneously to a server project to access assets and resources that are available in the repositories.

Before you begin

You must have completed the following tasks:

- Added a project on HCL DevOps Test Hub (Test Hub).
- Been granted permission and you must have set up your credentials to access the remote repositories in a repository domain.
- Created test assets and resources that are contained in either a single git repository or multiple git repositories.

About this task

As a project member with the role of a *Project Owner*, you can add a single repository or multiple repositories to your project.

Note: You can only add a single repository that is managed by using Git in Azure Repos. The addition of multiple repositories in Azure Repos is not supported.

You can use the information in the following table to decide the method for adding repositories to your project:

When...	Action...
You want to add any of the following repositories: • A single repository owned by you. • A single repository owned by other people. • The repositories that are in different domains.	You can add individual repositories one at a time. You must then provide the complete URL of the repository along with the required credentials to authenticate the connection to the repository from Test Hub. For example, the URL to a single repository can be as `https://github.com/<user>/<repository>.git`. You must provide the *username* and *password* to access this repository.

When...	Action...
You want to add any of the following repositories: • Multiple repositories that are owned by you and are contained in the same domain. • Multiple repositories that are owned by others and are contained in the same domain.	You can add all the repositories simultaneously that are in the same domain. You must then provide the URL of the repository domain along with the required credentials to authenticate the connection to the repository from Test Hub. For example, the URL of the repository domain can be `https://github.com`. You must select the platform and provide your *personal access token* to access the repositories.

1. Log in to Test Hub and open the team space that contains your project.
2. Open your project either from the navigation pane or from the **My Projects** page.

   The **Overview** page of the project is displayed.
3. Click **Manage > Configuration** from the navigation pane.

   The **Configuration** page of the project is displayed.
4. In the **Repositories** section, click **Add repository**.

   The **Add repository** dialog is displayed.

   You can either choose to add a single repository individually or multiple repositories simultaneously from a single domain.
5. Perform either of the following actions:
   - To add a single repository, go to Step 6 on page 1228.
   - To add multiple repositories from a domain, go to Step 7 on page 1229.
6. Perform the following steps to add individual repositories at a time:
   a. Enter the complete URL of the repository in the **Git URL** field.

      For example, the URL to a single repository can be as `https://github.com/<user>/<repository>.git`.

      The required authentication fields are displayed that depend on the protocol you specified in the URL.

If...	Then...
The repository URL contains *http* or *https*	The following fields for authentication are displayed: ▪ **Username** ▪ **Password**

If...	Then...
The repository URL contains *git@*	The following fields for authentication are displayed: ▪ **SSH/Deploy Key** ▪ **Passphrase** You can add a Git repository by using git@ and SSH. For example: ▪ `git@github.com:account/testRepo.git` ▪ `ssh://git@10.111.111.333:2222/~/account/project.git`

Note: If you use SSH to connect to a remote repository and Test Hub displays an `Auth Fail` error, you can resolve this error by regenerating your SSH keys by using the **-m PEM** option.

b. Enter the credentials to authenticate the connection to the repository.

c. Click **Add** to add the repository to your project.

You can add other individual repositories by specifying the URL and performing the actions in Step 6 on page 1228.

7. Perform the following steps to add multiple repositories simultaneously that are contained in the same domain:

Note: You can only add a single repository that is managed by using Git in Azure Repos. The addition of multiple repositories in Azure Repos is not supported.

a. Enter the URL of the repository domain in the **Git URL** field.

For example, the URL of the repository domain can be `https://github.com`.

b. Select the platform from the list under **Platform** where your repositories reside.

c. Enter your *personal access token* to authenticate your connection to the repository domain.

Note: If you do not have your *personal access token*, you can click **Generate** to generate your *personal access token* and use the token to authenticate your connection.

d. Click **Next**.

The connection to the repository domain is established and all the repositories that you own are listed. The time when the repository was last updated is displayed in the **Last updated** column.

e. Select the repositories that you want to add to your project.

f. Click **Add**.

Note: Depending on the size of the repository that you are adding, it can take a few to several minutes to add the repository to your project on the server.

All the repositories that you selected are added to your project and are listed on the **Repositories** page.

Note: If the repository is read-only, you will be able to add it and view its assets, but you will not be able to publish changes directly to it from Test Hub.

Results

You have added individual repositories or multiple repositories to your project on the server.

What to do next

You can perform the following actions on the repository that you added:

- Update the authentication credentials if they are changed in the Git repository configuration. See Updating the authentication credentials of the repository on page 1230.
- Delete a repository if it is no longer required. See Deleting a repository on page 1231.
- Refresh a repository to fetch and synchronize changes from the remote repository. See Refreshing repositories manually on page 1231.

Updating the authentication credentials of the repository

If the authentication credentials of the Git repository which was added to your project are changed, then you must update the credentials of the repository in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Owner* role in the project with access permissions to edit repositories.

About this task

In Test Hub, the repository is refreshed at regular intervals to pull the latest changes committed to the Git repository. After you update your authentication credentials of the repository, the repository is refreshed to ensure that you have access to the Git repository and to the latest assets in the repository.

1. Log in to Test Hub and open the team space that contains your project.
2. Open your project either from the left navigation pane or from the **My Projects** page.

3. Click **Manage > Configuration** from the navigation pane.

   The **Configuration** page of the project is displayed.

4. In the **Repositories** section, click the **Change...**  icon under Actions column for the repository that you want to update.

   The **Edit Repository** dialog is displayed.

5. Enter the new credentials, and then click **Update**.

   The new credentials are updated and your access to the repository is retained.

Deleting a repository

You can delete repositories that you no longer require in your test environment.

Before you begin

You must have completed the following tasks:

- Configured the repository that contains the test assets in your project on HCL DevOps Test Hub (Test Hub).
- Been assigned a *Owner* role with access to delete repositories.

About this task

You can delete a repository that you have configured for your project.

1. Log in to Test Hub and open the team space that contains your project.
2. Open your project either from the left navigation pane or from the **My Projects** page.
3. Click **Manage > Configuration** from the navigation pane.

   The **Configuration** page of the project is displayed.

4. In the **Repositories** section, click the **Delete**  icon under **Actions** column for the repository that you want to update.

   The **Delete repository** dialog is displayed.

5. Select the checkbox, and then click the **Delete** button.

   The repository is deleted and removed from the list.

Refreshing repositories manually

You can manually trigger the refresh activity for a repository to instantly fetch and synchronize the changes from the remote repository.

Before you begin

You must have completed the following tasks:

- Configured the repository that contains the test assets in your project onHCL DevOps Test Hub (Test Hub).
- Been assigned a *Member* or a *Project Creator* role.
- Been assigned a *Owner* role.

About this task

Every two minutes, the repository is refreshed automatically. You can also manually trigger the refresh activity for a repository.

1. Log in to Test Hub and open the team space that contains your project.
2. Open your project either from the left navigation pane or from the **My Projects** page.
3. Click **Manage > Configuration** from the navigation pane.

   The **Configuration** page of the project is displayed.

4. In the **Repositories** section, click the **Refresh**  icon under **Actions** column for the repository that you want to update.

   The refresh activity is triggered.

   Note: The **Refresh** option is disabled when the refresh is in progress. After the refresh is completed, the **Refresh** option is enabled.

   The following table lists the icons and their description based on the status of the refresh activity:

Refresh icon	Description
	The last refresh activity was successful.
	The last refresh activity was successful but errors were found while processing some of the test assets. To view the error details of the test assets for a specific repository, you can click the arrow inline with the repository name. You can resolve the errors of these test assets in the Git server, and then refresh the server repository, if required. **Note:** The repository, which contains the branch that is selected in the **Branch** field, displays the unprocessed test assets for that branch. Other repositories display the unprocessed test assets for their default branch.
	The refresh activity was not successful due to any one of the following reasons:

Refresh icon	Description
	◦ Internal error ◦ Connection issue ◦ Authentication failure ◦ Disk quota exceeded ◦ Repository unreachable

Results

You have manually refreshed the repository and the **Last Refreshed** field is updated with the following details:

- The time when it was last refreshed in the remote repository.
- Latest commit ID and message of the remote repository.

Selecting the global branch in a project

After you add one or more repositories to your project on HCL DevOps Test Hub (Test Hub), you can start accessing all branches in all repositories of that project. You can select only one branch as a global branch from the list of branches in the **Execution**, **Datasets**, and **Overview** pages.

Before you begin

You must have completed the following tasks:

- Been assigned a *Member* or *Project Creator* role in a team space.
- Added at least one repository in your project.

About this task

You can access all the branches contained in all the repositories, which are added to the project, by using the **Branch** field on the **Execution**, **Datasets**, and **Overview** pages.

After you select a branch from any of the repositories as the global branch, this selection is retained for you as the global branch unless you change the branch on any of the following pages: **Overview**, **Execution**, or **Datasets** pages.

Note:

- When you add the first repository to a project, the global branch is set to the default branch of the repository.
- Branches are arranged in an increasing order of their names in the list.
- When you change the branch in one of the pages, then the same branch is displayed as selected in the other pages and the contents of that page displays the assets from the selected branch.

1. Log in to Test Hub and open the team space that contains your project.
2. Open your project either from the left navigation pane or from the **My Projects** page.

The project **Overview** page is displayed.

3. Select a branch from the **Branch** field.

   The branch that you select on any of the following pages is reflected in the rest of the other pages and the test assets are filtered based on the global branch selection on each of these pages.:
   - **Overview**
   - **Execution**
   - **Datasets**

Results

You have selected a global branch in a project.

Creating webhooks

You can configure a webhook for a repository in the Git server to notify HCL DevOps Test Hub (Test Hub) whenever there is a push event in the remote repository. Then, the refresh activity is triggered to fetch the latest changes to the server repository.

Before you begin

You must have completed the following tasks:

- Configured the repository that contains the test assets in your project.
- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Owner* role in the project.

1. Log in to Test Hub and open the team space that contains your project.
2. Open your project either from the left navigation pane or from the **My Projects** page.
3. Click **Manage > Configuration** from the navigation pane.

   The **Configuration** page of the project is displayed.

4. In the **Repositories** section, click the **Webhook**  icon under **Actions** column for the repository that you want to update.

   The **Webhook Configuration** dialog is displayed.

5. Copy the URL and the Secret of the repository that you want to configure.

   Note: The URL and the Secret details are specific to a repository.

6. Click the link of the repository for which you want to configure the webhook.

   The remote repository page is displayed.

7. Paste the details, which you copied from Test Hub, in the appropriate fields of the remote repository.

Results

You have configured the webhook for a repository in the Git server. The changes of the remote repository are synchronized in the repository of your project.

Protecting test assets by using secrets

Secrets are key-value pairs that are created for your project in HCL DevOps Test Hub (Test Hub) under a security group. You can create security group for your project that enable you or members in your project to use secrets at test runtime either in Test Hub or in HCL DevOps Test Integrations and APIs (Test Integrations and APIs).

The secrets in a project in Test Hub maintains a separate access control list managed by the members with access to the secrets added to the security group. Controlling access to secrets means controlling access to applications and systems under test. The introduction of secrets under security group for a project simplifies managing access to separate environments. If a member of a project does not have access to a secret, for example, *a server credential* then the member cannot accidentally or intentionally run tests against that server. For example, tests that must access the database server by using the server credentials to retrieve stored data can only be run by a member if the access to the secrets is granted.



Note: Secrets are applicable to server test assets created in Test Hub and test assets created in Test Integrations and APIs that facilitates running tests in defined environments. Secrets are not applicable to tests created in Test UI or Test Performance.

As a project member with the *Owner* or *Tester* role, you can add secrets to the security group in the project. You can grant or restrict access to the secrets that you added to the security group in the project.

Members with access to a secret security group can access, edit, or delete the secrets in Test Hub and can view secrets, edit secrets, or delete secrets.

Members are granted Read access to the security group at the role level, while Read and Write access can be provided to specific members.

Promoted test variables from server test assets can be mapped to secret-backed values by using security groups and secret entries. This mapping helps you reuse secure values across environments without embedding credentials directly in test steps.

Members in the project with the *Owner* or *Tester* role and with access to the secrets can use the secrets in tests at runtime.

If you are configuring a project to run an API Suite with tests that refer to secret values, you must configure the secrets under a security group by using the **Add Secret** option in the **Security** page. You must complete the following tasks:

1. Create a security group. See Creating a security group on page 404.
2. Add secrets in the security group. See Creating a secret in a security group on page 1236.
3. Grant access to project members or member roles, who can access the secrets. See Granting access to members or member roles on page 1237.

Creating a secret in a security group

You must create secrets in the security groups that is set in your project so that the members of the project with access to the secrets can use them in specific during an API Suite run.

Before you begin

You must have completed the following tasks:

- Created a project on Test Hub.
- Created a security group. See Creating a security group on page 404.
- You must be a member with access to the secrets in the security group. See Granting access to members or member roles on page 1237.

About this task

You can also configure secrets such that the secrets can be used across different test environments by members with access to the security group. Secrets correspond to the following:

- Environment variables or tags that you create in a Test Integrations and APIs project specific to an environment.
- Variables that you promote in a server test as a secret that is specific to an environment.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Click a security group for which you want to add the secrets under the **Security Group**.
5. Click **Add Secret**.
   Result
   The **Add Secret** dialog is displayed.
6. Enter a name for the secret as its *Identifier*, and then enter the `password` as the *Value* for the secret.

   For example, enter the name of the secret to access a database as `dbcred` and enter the `password` required to access the database as its value.
7. Click **Create**.

   A message is displayed that confirms the creation of the secret.

Results

You have created secrets in the selected security group for your project.

What to do next

- You can view, edit, or delete the secrets added to a security group any time you want.
- You can use the secrets in the tests that require these secrets during test runs.

Granting access to members or member roles

You can grant Read access to the secrets added to the security group in your project at the role level for members of a designated role. Additionally, you can provide Read and Write access to specific members. Without access to the security group, members cannot view, create, edit, delete, or use the secrets added to the security group.

Before you begin

You must be a project **Owner** and have completed the following tasks:

- Created a project in Test Hub. See Managing DevOps Test Hub projects on page 1209.
- Added one or more users to your project. See Adding users to a project on page 1217.
- Created at least one security group. See Creating a security group on page 404.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Click a security group in the list under **Security Group** to which the secrets are added and for which you want to grant the access.
5. Select any of the following methods to grant access to a member:
   Choose from:

   - Select the checkbox for the specific role under **Grant Read Only Access to roles** to grant read only access to the members of a specific role. For example, if you select **Tester**, then all members in the project with a tester role are granted read only access to the selected security group. You can select any role or all the roles listed.
   - Perform the following steps to grant read/write access to specific members:
     a. Click **Add Members** inline with **Members with Read/Write Access**.

        The **Additional Members** dialog is displayed.

     b. Enter the name or email ID to search the user or navigate through the list and select the member you want to grant access. You can select multiple members from the list.

        Important:

Irrespective of the role that the member (**Owner**, **Tester** or **Viewer**) was assigned in the project, the access to the security group to which the secret is added must be specifically granted to the members from the **Security** page.

c. Click **Add Members.**

You granted the read/write access to the added members.

6. Select any of the following methods to revoke access to a member:
   Choose from:
   - Clear the checkbox for the specific role under **Grant Read Only Access to roles,** to revoke read only access to the members of a specific role. For example, if you clear the **Tester** checkbox then access for all members in the project with a tester role is revoked for the selected security group. You can select any role or all the roles listed.
   - Click the **Delete** icon inline with the member name to remove the specific members from the **Members with Read/Write Access** list.

Notes:
- Any member with access to the security group can remove access to other members specifically added or of all members with a specific role.
- Members with access to the security group can remove themselves from the access list provided that there is at least one member in the list. After removing themselves, members cannot add themselves back to the access list and must be added by any of the other members in the list.

Results

You have added members from your project or members with specific role to the access list of people who can access secrets in the selected security group, or you have removed specific members or members with specific role from the access list.

What to do next

You can add secrets to the security groups for your project.

Managing secrets

You can view, edit, or delete the secrets configured under a security group any time after you created secrets or after you were granted access to the secrets security group. You can change the value of the secret by editing the secret. You can delete secrets that you no longer require in your test environment.

Before you begin

You must have completed the following tasks:

- You must have created a project on Test Hub and configured a secrets in your project.
- You must have created secrets in the selected security group or the security groups must contain secrets.
- You must be a member with access to the secrets security group.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Security**.
   The **Security** page is displayed.
4. Perform the following steps:
   a. Select the **Security Group** from the list in the **Security** page.
   b. Optionally, select the security group that you want from the list if there are multiple security groups in the project.

   The secrets configured in the selected security groups are displayed.
5. Complete the steps for the task you want to perform as listed in the following table:

Task	Action
Viewing a secret value	Click the **View** icon inline with the secret to view the entered value. The value most likely is a password for the secret. The value configured for the secret is displayed.
Editing a secret value	Perform the following steps: a. Click the **Edit** icon inline with secret that you want to edit, The **Edit secret** dialog is displayed. b. Enter a `new value` for the secret as its *Value*. The value can be a password for the secret. **Note:** You can only change the value of the secret. c. Click **Save**. The value of the selected secret is changed.
Deleting a secret	Perform the following steps: a. Click the **Delete** icon inline with the secret that you want to delete. The **Delete secret** dialog is displayed.

Task	Action
	b. Click **Delete**. After you delete the secrets, the secrets listed in the security group is removed from the list.

Results

- You viewed the password configured of the secret under a security group that you created or were granted access.
- You changed the secret value of the secret under a security group in your project.
- You deleted and removed the secret from the selected security group in your project.

What to do next

You can use secrets in the tests that require these secrets during test runs.

Related information

Becoming a project member on page 1220

Test run configurations on page 610

Notification Management

HCL DevOps Test Hub (Test Hub) provides a feature to display notifications for different events within the user interface (UI) of Test Hub. You can configure Email notifications on Test Hub to send out notifications about different events as emails to the subscribed users. You can also configure third-party applications as webhooks on Test Hub when you want Test Hub to send push notifications about the server events to the third-party applications in a preset template.

Test Hub provides you the following features to receive notifications about server events:

Channels	Description
In-app	Notifications are displayed within the UI of Test Hub.
Email	Notifications are sent in emails to subscribed users of Test Hub by an SMTP server.
Webhooks	Notifications in a preset template are sent as push notifications by Test Hub to the configured webhook of a third-party application, such as MS Teams, Slack, or others.

Managing notifications in the server UI

Viewing the in-app notifications or notifications in the server UI is a feature that is enabled automatically when you register as a user in Test Hub.

When you register as a user in Test Hub, the in-app notifications are automatically subscribed for you.

If you want to receive notifications about specific events and not all events, you can change the default subscriptions and modify your subscription for the server events. You can also view the in-app notifications that are displayed for you. See Managing in-app notifications on page 1241.

Managing Email notifications

You, as a *Server Administrator* or *Team Space Owner* can configure an existing Simple Mail Transfer Protocol (SMTP) server so that Test Hub can send out notifications about the server events in emails to the registered users who have subscribed to the email notifications. See Configuring an SMTP server to manage email notifications on page 1246.

If you, as a registered user of Test Hub want to receive the email notifications about specific events and not all events, you can change the subscriptions for the server events. See Managing email notifications on page 1244.

Managing webhook notifications

You, as a *Server Administrator*, *Team Space Owner*, or *Project Owner* can configure third-party applications as webhooks so that Test Hub can send out push notifications about the server events to the third-party applications. See Configuration of a webhook on page 1248.

Managing in-app notifications

You can manage the in-app notifications by changing the subscriptions that are enabled by default for all the events that occur on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.

About this task

You can view the default notification subscription for all server events, and then choose to unsubscribe or change the subscription for any or all the events.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **User** icon  on the menu bar.

3. Click **Account Settings**.

   Alternatively, click the **Notification** icon  on the menu bar, and then click the **Settings** icon  in the **Notifications** dialog.

4. Click the Expand  icon and identify the server events for which you want to change the subscription.
   You can notice that for each of the server events, the in-app channel is displayed as selected, which is the default selection unless you have modified the subscription.
5. Perform the actions indicated in the following table to clear the subscriptions for server events that you do not want to be notified and to select the in-app subscription for the server events that you want to be notified.

When	Action	Result
You do not want to receive the in-app notifications for a specific server event.	Perform the following steps: a. Identify the server event by expanding the channel list. b. Clear the checkbox in the **In App** row to unsubscribe. c. Click **Apply**.	You have unsubscribed the in-app notifications for a specific server event and the notification is not displayed for you when the event occurs.
You want to receive the in-app notifications for a specific server event.	Perform the following steps: a. Identify the server event by expanding the channel list. b. Select the checkbox in the **In App** row. c. Click **Apply**.	You have subscribed to the in-app notifications for the server event and the notification is displayed for you when the event occurs.
You want to subscribe to the in-app notifications for all server events.	Select the **In App** checkbox under **Channels**, and then click **Apply**.	The in-app channel is displayed as selected for all server events and notifications for all server events will be displayed.
You do not want to receive any in-app notifications for all server events.	Clear the **In App** checkbox, and then click **Apply**.	The in-app channel selections for all server events are cleared and no notifications will be displayed when the event occurs.

6. To change the notification language in which you want to receive the notifications from the default, select a language from the **Email Language** option in the **Notifications** pane and click **Apply**.

Results

You have modified the notification subscription for specific or all server events.

Related information

Managing email notifications on page 1244

Viewing in-app notifications

You can view the notifications for the events that occur within the HCL DevOps Test Hub (Test Hub).

About this task

The in-app notifications are displayed in the server UI under the following categories for the server events that occur:

Category	Notifications displayed
User	Include information about actions performed by team space members in the team space.
System	Include information about actions performed by registered users at the server level.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Notifications** icon on the menu bar.

   The **Notifications** dialog is displayed.

   You can find the information about the default notifications that you can view based on your role in the server as in-app notifications in the **USER** tab and **SYSTEM** tab.

Category	Notification details	***Server Administrator*** **or** ***Team Space Owner***	***Project Creator*** **or** ***Project Owner***	***Tester***	***Viewer***
User notifications	Information about events at the server level or across team spaces performed by other team space owners.	✔	✔	✘	✘
	Information about events in the team space performed by other team space members.	✔	✘	✘	✘

Category	Notification details	*Server Administrator* or *Team Space Owner*	*Project Creator* or *Project Owner*	*Tester*	*Viewer*
	Information about events in the project performed by other project members.	✗	✓	✓	✗
	Information about events performed by you in a project.	✗	✗	✓	✓
System notifications	Information about events at the server level or across team spaces.	✓	✓	✗	✗
	Information about events in the team space performed by other team space members.	✓	✓	✗	✗
	Information about events in the projects at the team space level performed by other project members.	✓	✓	✗	✗

The notifications in the **USER** tab are displayed as the default option.

3. View the notifications in the **USER** tab.
4. View the notifications at the server level by clicking the **SYSTEM** tab.

Results

You have viewed the in-app notifications.

Managing email notifications

You can manage the email notifications by changing the subscriptions that are automatically enabled for you for all events that occur in HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that the *Server Administrator* or *Team Space Owner* has configured a Simple Mail Transfer Protocol (SMTP) server. See Configuring an SMTP server to manage email notifications on page 1246.
- Ensured that you are assigned a role as a *Member* or *Project Creator* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you are assigned a role as a *Viewer*, *Tester*, or *Project Owner* in the project.

About this task

After the SMTP server is configured on Test Hub, you can view the server events that are subscribed in the **General Settings** page. You can either unsubscribe from the **Email** channel or change your subscriptions of the server events for which you do not want to receive email notifications.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **User** icon on the menu bar.
3. Click **Account Settings**.

   Alternatively, click the **Notification** icon on the menu bar, and then click the **Settings** icon in the **Notifications** dialog.

   The **General Settings** dialog is displayed.
4. Perform the actions indicated in the following table for the option you want for the email notifications:

When	Action	Results
You want to subscribe to email notifications for all server events.	Select the **Email** checkbox under **Channels** then click **Apply**.	The email channel for all server events is selected and you are subscribed to receive email notifications for all server events.
You do not want to receive any email notifications.	Uncheck the **Email** checkbox under **Channels**, then click **Apply**.	The email notification selection for all server events is cleared and you are unsubscribed from receiving email notifications for all the server events.

When	Action	Results
You want to receive temail notifications for a specific server event.	Perform the following steps: a. Expand the channel list in the row of the server event. b. Select the **Email** checkbox of the event you want to subscribe. c. Click **Apply**.	You have subscribed to the email channel to receive email notifications for the server event.
You do not want to receive email notifications for a specific server event, if you subscribed to it previously.	Perform the following steps: a. Expand the channel list in the row of the server event. b. Clear the checkbox for the **Email** channel option of the event you want to unsubscribe. c. Click **Apply**.	You have unsubscribed from the email channel for the server event, and you will not receive email notifications when the event occurs.

5. To change the language in which you want to receive the email notifications from the default, select a language from the **Email Language** list in the **Notifications** pane and click **Apply**.

   The default language is the language set on the computer that hosts Test Hub.

Results

You completed the following tasks:

- Subscribed to the email channel for receiving email notifications for the selected server events.
- Unsubscribed from the email channel for selected server events that you subscribed previously.
- Subscribed to the email channel for receiving email notifications for all server events.
- Unsubscribed from the email channel for receiving email notifications for all server events.

Related information

Managing in-app notifications on page 1241

Configuring an SMTP server to manage email notifications

Before you can receive email notifications about events that occur in HCL DevOps Test Hub (Test Hub), you need to configure an existing Simple Mail Transfer Protocol (SMTP) server in Test Hub. Once configured, email notifications are sent to the registered users if they are subscribed to email notifications whenever events occur in Test Hub.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that an existing SMTP server is available to be connected to Test Hub.

About this task

The configuration works for emails that are supported by the SMTP server. To use this capability, you must enable Test Hub to connect to the SMTP server to send email notifications.

As a server administrator, you can configure email notifications so that other users can receive information about actions on the server projects.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Server administration page** icon from the menu bar.

   The **Global Settings** page is displayed.
3. Click **Add Configuration**.
4. Provide the following details to configure the SMTP server:
   - A name to the configuration.
   - The hostname and port number of the SMTP server.

     Note: The default configuration uses a local SMTP server that listens to the default port 25. This SMTP server does not require any authentication.

   - The email address of the sender. The email address is displayed in email notifications. The email notifications are sent to the users based on the notification settings.
   - The encryption type for secure communications is when the SMTP server is on an external network. You can enable either **STARTTLS** or **SSL/TLS**.

     Note: The SMTP server for the secure communication listens to the default port 465.

   - Login credentials are required if the SMTP server requires authentication.
5. Click **Add**.

Note: The SMTP configuration is added after a valid connection is established with the SMTP server.

Results

You configured an SMTP server to connect with Test Hub and send out email notifications about the server events to users who have subscribed to email notifications.

What to do next

You can modify the default subscriptions for the email notifications, see Managing email notifications on page 1244.

Related information

Notification Management on page 1240

Configuration of a webhook

You can configure webhooks to send notifications to a messaging application such as Microsoft Teams, Slack, or any generic application that can receive an HTTP POST request when any of the events occur on HCL DevOps Test Hub (Test Hub). For example, when you add or remove a member from a team space or when you run a test.

A webhook is an HTTP custom callback that provides an application to send notifications to other applications. Webhooks help applications automatically communicate with other external applications.

The following table lists the tasks that you can perform and manage for the webhooks:

Task	Go to...
Configure a webhook template	Configuring a webhook template on page 1248
Configure a webhook	Configuring a project webhook on page 1260
Edit, clone, or delete a webhook template	Managing the webhook template configuration on page 1262
Edit, clone, or delete a webhook	Managing the webhook configuration on page 1264

Webhooks overview

Webhooks send a notification to the configured messaging applications when specific events occur on HCL DevOps Test Hub (Test Hub).

When the event occurs, Test Hub automatically sends an HTTP POST request with event-related information in a JSON format or in plain text to the URL that you configured. You can define a template body or use the sample template that is set on Test Hub. You must create a channel type, a template, and a webhook to send an HTTP POST request.

You can manage the webhook template configuration and webhook configuration on Test Hub. You can identify the events and view the status of the events that are posted on the messaging application whenever an event occurs.

Configuring a webhook template

You can configure a project, team space and a server webhook template on HCL DevOps Test Hub (Test Hub) to send notifications to the configured messaging application.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned the role of a *Project Owner* in the project. See Managing access to the server projects on page 1220.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Ensured that you are familiar with Apache Velocity. See Apache Velocity Documentation.

About this task

You can create a template based on the channel type that you configured. You can select a generic template to send notifications to the generic application. You can also select a JSON sample template with specific fields defined to display event-related notifications in Microsoft Teams or Slack. You can use the sample template that is available for the messaging application or you can also define a JSON template to post the message about the events that occur on Test Hub.

1. Select from the following actions you want to perform:
   - Configure a project webhook template. Go to Step 2 on page 1249.
   - Configure a team space webhook template. Go to Step 3 on page 1251.
   - Configure a server webhook template. Go to Step 4 on page 1254.
2. Perform the following steps to configure a project webhook template.

   a. Log in to Test Hub.

      The **Projects** page of the initial team space is displayed.

   b. Click **My projects > *project_name*** to open the project that contains the test assets.

      The **Overview** page of the project is displayed.

   c. Click **Manage > Webhooks**.

      The **Templates** tab of Webhooks is displayed.

   d. Click **New Template**.

      The **New template** page is displayed.

   e. Perform the following actions on the **DETAILS** tab:
      i. Enter a unique name for the template.
      ii. Click the **Channel Type** option, and then select an existing channel type.
      iii. Alternatively, you can create a new channel type by clicking **Create new channel type ...** option if there are no channel types available in the drop-down list.

      The **New channel type** dialog is displayed.

iv. Enter a unique name for the channel type, and then enter a description for the channel type.

Note: You can also create a new channel type or edit a channel type from the **Channel Type** option available within the **Templates** tab.

f. Perform the following steps if you want to select either a channel icon or a brand icon:

i. Click **Create new channel type...**, and then click **Channel type** icon  .

The **Select icon** page is displayed.

ii. View a list of icons that are displayed In the **GENERAL** tab.

iii. Search for the icons by entering any text contained in the icon name in the **Search** field, and then select the required icon.

The selected icon is displayed in the **Selected icon** box.

iv. Click the **BRANDS** tab.

View a list of icons that are displayed.

v. Search for the brands by entering any text contained in the brand name in the **Search** field, and then select the brand.

The selected brand is displayed in the **Selected icon** box.

vi. Click **Ok**.

The default icon is replaced with the selected icon.

g. Enter a description for the channel type, and then click **Apply**.

The channel type is created and added to the **Channel Type** list.

h. Enter a description for the template.

i. Perform the following steps to select an event:

i. Click **This template is suitable for events** and select All Events, or any of the subset events from the following list:

- All Events
- Team Space Events
- Project Events
- Execution Events
- License Events

ii. Search for the events by entering any text contained in the events name in the **Search** field, and then select the corresponding checkbox to select the specific event.

Note: All Events is the default event that is set for the webhook template.

j. Click the **Template** tab, and then perform any of the following actions:

- Perform the following steps to insert a generic template:
  i. Click the **cloud** icon to insert a velocity text processing template in the **Template body**.

     A message is displayed about the action that inserts a text in the template body.

  ii. Click **Ok**.

     The text is inserted in the **Template body**.
- Perform the following steps to insert a Microsoft Teams sample template:
  i. Click the **Microsoft Teams** icon to replace a template with a Microsoft Teams template.

     A message is displayed about the action that replaces the content of the template body with a Microsoft Teams template.

  ii. Click **Ok**.

     A Microsoft Teams sample template is inserted in the **Template body**.
- Perform the following steps to insert a Slack sample template:
  i. Click the **Slack** icon .
  ii. A message is displayed about the action that replaces the content of the template body with a Slack template.
  iii. Click **Ok**.

     A Slack sample template is inserted in the **Template body.**
- Perform the following steps to insert a DevOps Velocity sample template:
  i. Click the **infinity** icon .
  ii. A message is displayed about the action that replaces the content of the template body with a DevOps Velocity template.
  iii. Click **Ok**.

     A DevOps Velocity sample template is inserted in the **Template body.**

k. Click **Apply**.

The template is created and added to the **Templates** tab.

3. Perform the following steps to configure a team space webhook template:

a. Log in to Test Hub.

   The **Projects** page of the initial team space is displayed.

b. Click **Initial Team Space > Manage Team Space > > Webhooks**.

   The **Templates** tab of Webhooks is displayed.

c. Click **New Template**.

   The **New template** page is displayed.

d. Perform the following actions on the **DETAILS** tab:
   i. Enter a unique name for the template.
   ii. Click the **Channel Type** option, and then select an existing channel type.
   iii. Alternatively, you can create a new channel type by clicking **Create new channel type ...** option if there are no channel types available in the drop-down list.

      The **New channel type** dialog is displayed.

   iv. Enter a unique name for the channel type, and then enter a description for the channel type.

      Note: You can also create a new channel type or edit a channel type from the **Channel Type** option available within the **Templates** tab.

e. Perform the following steps if you want to select either a channel icon or a brand icon:
   i. Click **Create new channel type...**, and then click **Channel type** icon  .

      The **Select icon** page is displayed.

   ii. View a list of icons that are displayed In the **GENERAL** tab.
   iii. Search for the icons by entering any text contained in the icon name in the **Search** field, and then select the required icon.

      The selected icon is displayed in the **Selected icon** box.

   iv. Click the **BRANDS** tab.

      View a list of icons that are displayed.

   v. Search for the brands by entering any text contained in the brand name in the **Search** field, and then select the brand.

      The selected brand is displayed in the **Selected icon** box.

   vi. Click **Ok**.

      The default icon is replaced with the selected icon.

f. Enter a description for the channel type, and then click **Apply**.

The channel type is created and added to the **Channel Type** list.

g. Enter a description for the template.

h. Perform the following steps to select an event:

   i. Click **This template is suitable for events** and select All Events, or any of the subset events from the following list:

      - All Events
      - Team Space Events
      - Project Events
      - Execution Events
      - License Events

   ii. Search for the events by entering any text contained in the events name in the **Search** field, and then select the corresponding checkbox to select the specific event.



Note: All Events is the default event that is set for the webhook template.

i. Click the **Template** tab, and then perform any of the following actions:

- Perform the following steps to insert a generic template:

  i. Click the **cloud** icon  to insert a velocity text processing template in the **Template body**.

  A message is displayed about the action that inserts a text in the template body.

  ii. Click **Ok**.

  The text is inserted in the **Template body**.

- Perform the following steps to insert a Microsoft Teams sample template:

  i. Click the **Microsoft Teams** icon  to replace a template with a Microsoft Teams template.

  A message is displayed about the action that replaces the content of the template body with a Microsoft Teams template.

  ii. Click **Ok**.

  A Microsoft Teams sample template is inserted in the **Template body**.

- Perform the following steps to insert a Slack sample template:

  i. Click the **Slack** icon  .

  ii. A message is displayed about the action that replaces the content of the template body with a Slack template.

  iii. Click **Ok**.

  A Slack sample template is inserted in the **Template body.**

- Perform the following steps to insert a DevOps Velocity sample template:

i. Click the **infinity** icon .
ii. A message is displayed about the action that replaces the content of the template body with a DevOps Velocity template.
iii. Click **Ok**.

A DevOps Velocity sample template is inserted in the **Template body.**

j. Click **Apply**.

The template is created and added to the **Templates** tab.

4. Perform the following steps to configure a server webhook template:

a. Log in to Test Hub.

The **Projects** page of the initial team space is displayed.

b. Click the **Settings** icon on the menu bar.
c. Click **Webhooks**.

The **Templates** tab of Webhooks is displayed.

d. Click **New Template**.

The **New template** page is displayed.

e. Perform the following actions on the **DETAILS** tab:
i. Enter a unique name for the template.
ii. Click the **Channel Type** option, and then select an existing channel type.
iii. Alternatively, you can create a new channel type by clicking **Create new channel type ...** option if there are no channel types available in the drop-down list.

The **New channel type** dialog is displayed.

iv. Enter a unique name for the channel type, and then enter a description for the channel type.

Note: You can also create a new channel type or edit a channel type from the **Channel Type** option available within the **Templates** tab.

f. Perform the following steps if you want to select either a channel icon or a brand icon:
i. Click **Create new channel type...**, and then click **Channel type** icon .

The **Select icon** page is displayed.

ii. View a list of icons that are displayed In the **GENERAL** tab.
iii. Search for the icons by entering any text contained in the icon name in the **Search** field, and then select the required icon.

The selected icon is displayed in the **Selected icon** box.

iv. Click the **BRANDS** tab.

View a list of icons that are displayed.

v. Search for the brands by entering any text contained in the brand name in the **Search** field, and then select the brand.

The selected brand is displayed in the **Selected icon** box.

vi. Click **Ok**.

The default icon is replaced with the selected icon.

g. Enter a description for the channel type, and then click **Apply**.

The channel type is created and added to the **Channel Type** list.

h. Perform the following steps to select an event:

i. Click **This template is suitable for events** and select All Events, or any of the subset events from the following list:

- All Events
- Team Space Events
- Project Events
- Execution Events
- License Events

ii. Search for the events by entering any text contained in the events name in the **Search** field, and then select the corresponding checkbox to select the specific event.

Note: All Events is the default event that is set for the webhook template.

i. Click the **Template** tab, and then perform any of the following actions:

- Perform the following steps to insert a generic template:

i. Click the **cloud** icon to insert a velocity text processing template in the **Template body**.

A message is displayed about the action that inserts a text in the template body.

ii. Click **Ok**.

The text is inserted in the **Template body**.

- Perform the following steps to insert a Microsoft Teams sample template:

i. Click the **Microsoft Teams** icon to replace a template with a Microsoft Teams template.

A message is displayed about the action that replaces the content of the template body with a Microsoft Teams template.

ii. Click **Ok**.

A Microsoft Teams sample template is inserted in the **Template body**.

- Perform the following steps to insert a Slack sample template:

  i. Click the **Slack** icon .
  ii. A message is displayed about the action that replaces the content of the template body with a Slack template.
  iii. Click **Ok**.

  A Slack sample template is inserted in the **Template body.**

- Perform the following steps to insert a DevOps Velocity sample template:

  i. Click the **infinity** icon .
  ii. A message is displayed about the action that replaces the content of the template body with a DevOps Velocity template.
  iii. Click **Ok**.

  A DevOps Velocity sample template is inserted in the **Template body.**

j. Click **Apply**.

The template is created and added to the **Templates** tab.

Results

You have configured and created a project, team space and a server webhook template on Test Hub.

What to do next

You must configure a webhook to push the events from Test Hub to the messaging application. See Configuring a project webhook on page 1260.

Configuring a server webhook

You can configure a server webhook to send notifications to a messaging application whenever an event occurs on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role of a *Server Administrator*. See Managing access to the server projects on page 1220.
- Ensured that you have access to the messaging applications.
- Created a channel type.
- Configured your messaging application.
- Copied the URL of the external communication application. See Creating an incoming webhook.
- Created a webhook template. See Configuring a webhook template on page 1248.

About this task

When you configure a webhook on Test Hub, you must copy the URL of the communication application in the **Webhook URL** field and paste the outgoing URL, and then select the events. You can select either all events or a subset of events.

You can be notified of the following types of events:

- All Events
- Team space Events
- Project Events
- Execution Events
- License Events

You can configure the webhook resume policy by selecting either the Queue old messages or Discard old messages option. When a server webhook is disabled due to communication failures or network issues, the server event notifications are not delivered to the configured webhook, and you receive a notification that the webhook is disabled. By configuring the queue old messages, you can ensure that the event notifications are not lost and are delivered to the configured webhook after the server webhook is enabled again.

If you configure the Discard old messages for a server webhook, the event notifications are discarded from the moment the webhook is disabled. Only server events that occur after the webhook is re-enabled are sent to the configured webhook.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click the **Settings** icon  on the menu bar, and then click **Webhooks**.
   The **Templates** tab of Webhooks is displayed.
3. Click the **Webhooks** tab, and then click **New Webhook**.

   The **New Webhook** page is displayed.
4. Enter a name for the webhook and paste the URL that you copied in the **Webhook URL** field.
   You must copy the URL while you configure your communication application.
5. Click the **Webhook resume policy**, and then select a resume policy option from the drop-down list. You can select any of the following options:
   a. Queue old messages
   b. Discard old messages
6. Click the **Channel Type**, and then select a channel type from the list.
7. Enter a description for the webhook, and then select a template from the drop-down list.
   A corresponding event that you configured for the template is displayed in the **Events** field.
8. Click the **Templates** icon  , to add the templates that are configured for the channel type.

   Note: If you want to change the events, then click the events option and select the required events.
9. You can perform any of the following steps:

a. Click move the template to a higher priority icon  to move the template to a higher priority.

b. Click move the template to a lower priority icon  to move the template to a lower priority.

10. Click **Apply**.
    The webhook is created and added to the **Webhooks** tab.

Results

You have configured and created a server webhook on Test Hub.

What to do next

You can perform any of the following tasks:

- You can duplicate or delete the template that you created on Test Hub. See Managing the webhook template configuration on page 1262.
- You can duplicate or delete the webhook you created on Test Hub. See Managing the webhook configuration on page 1264.

Configuring a team space webhook

You can configure a team space webhook to send notifications to a messaging application whenever an event occurs on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Ensured that you have access to the messaging applications.
- Created a channel type.
- Configured your messaging application.
- Copied the URL of the external communication application. See Creating an incoming webhook.
- Created a webhook template. See Configuring a webhook template on page 1248.

About this task

When you configure a webhook on Test Hub, you must copy the URL of the communication application in the **Webhook URL** field and paste the outgoing URL, and then select the events. You can select either all events or a subset of events.

You can be notified of the following types of events:

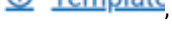
- All Events
- Team space Events
- Project Events

- Execution Events
- License Events

You can configure the webhook resume policy by selecting either the Queue old messages or Discard old messages option. When a team space webhook is disabled due to communication failures or network issues, the team space event notifications are not delivered to the configured webhook, and you receive a notification that the webhook is disabled. By configuring the queue old messages, you can ensure that the event notifications are not lost and are delivered to the configured webhook after the team space webhook is enabled again.

If you configure the Discard old messages for a team space webhook, the event notifications are discarded from the moment the webhook is disabled. Only team space events that occur after the webhook is re-enabled are sent to the configured webhook.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **Initial Team Space > Manage team space**, and then click **Webhooks**.
   The **Templates** tab of Webhooks is displayed.
3. Click the **Webhooks** tab, and then click **New Webhook**.

   The **New Webhook** page is displayed.
4. Enter a name for the webhook and paste the URL that you copied in the **Webhook URL** field.
   You must copy the URL while you configure your communication application.
5. Click the **Webhook resume policy**, and then select a resume policy option from the drop-down list. You can select any of the following options:
   a. Queue old messages
   b. Discard old messages
6. Click the **Channel Type**, and then select a channel type from the list.
7. Enter a description for the webhook, and then select a template from the drop-down list.
   A corresponding event that you configured for the template is displayed in the **Events** field.
8. Click the **Templates** icon  , to add the templates that are configured for the channel type.

   Note: If you want to change the events, then click the events option and select the required events.
9. You can perform any of the following steps:
   a. Click move the template to a higher priority icon  to move the template to a higher priority.
   b. Click move the template to a lower priority icon  to move the template to a lower priority.
10. Click **Apply**.
    The webhook is created and added to the **Webhooks** tab.

Results

You have configured and created a team space webhook on Test Hub.

What to do next

- You can duplicate or delete the template that you created on Test Hub. See Managing the webhook template configuration on page 1262.
- You can duplicate or delete the webhook you created on Test Hub. See Managing the webhook configuration on page 1264.

Configuring a project webhook

You can configure a project webhook to send notifications to a messaging application whenever an event occurs on HCL DevOps Test Hub (Test Hub).

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned the role of a *Project Owner* in the project. See Managing access to the server projects on page 1220.
- Created a project in your team space in Test Hub. See Creating a project on page 1213.
- Ensured that you have access to the messaging applications.
- Created a channel type.
- Configured your messaging application.
- Copied the URL of the external communication application. See Creating an incoming webhook.
- Created a webhook template. See Configuring a webhook template on page 1248.

About this task

When you configure a webhook on Test Hub, you must copy the URL of the communication application in the **Webhook URL** field and paste the outgoing URL, and then select the events. You can select either all events or a subset of events.

You can be notified of the following types of events:

- All Events
- Team space Events
- Project Events
- Execution Events
- License Events

  You can configure the webhook resume policy by selecting either the Queue old messages or Discard old messages option. When a project webhook is disabled due to communication failures or network issues, the project event notifications are not delivered to the configured webhook, and you receive a notification that the webhook is disabled.

By configuring the queue old messages, you can ensure that the event notifications are not lost and are delivered to the configured webhook after the project webhook is enabled again.

If you configure the Discard old messages for a project webhook, the event notifications are discarded from the moment the webhook is disabled. Only project events that occur after the webhook is re-enabled are sent to the configured webhook.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Webhooks** in the navigation pane.
   The **Templates** tab of Webhooks is displayed.
4. Click the **Webhooks** tab, and then click **New Webhook**.

   The **New Webhook** page is displayed.
5. Enter a name for the webhook and paste the URL that you copied in the **Webhook URL** field.
   You must copy the URL while you configure your communication application.
6. Click the **Webhook resume policy**, and then select a resume policy option from the drop-down list. You can select any of the following options:
   a. Queue old messages
   b. Discard old messages

      The Queue old messages is selected as the default option.
7. Click the **Channel Type**, and then select a channel type from the list.
8. Enter a description for the webhook, and then select a template from the drop-down list.
   A corresponding event that you configured for the template is displayed in the **Events** field.
9. Click the **Templates** Template icon, to add the templates that are configured for the channel type.

   Note: If you want to change the events, then click the events option and select the required events.
10. You can perform any of the following steps:
    a. Click move the template to a higher priority icon to move the template to a higher priority.
    b. Click move the template to a lower priority icon to move the template to a lower priority.
11. Click **Apply**.
    The webhook is created and added to the **Webhooks** tab.

Results

You have configured and created a project webhook on Test Hub.

What to do next

You can perform any of the following tasks:

- You can duplicate or delete the template that you created on Test Hub. See Managing the webhook template configuration on page 1262.
- You can duplicate or delete the webhook you created on Test Hub. See Managing the webhook configuration on page 1264.

Managing the webhook template configuration

You can manage the webhook template configuration that you set on HCL DevOps Test Hub (Test Hub) when you want to edit, duplicate, delete the existing webhook templates, or create new templates. Before you create a webhook, you can view the templates created at server, team space, and project levels. Additionally when you create a project, team space or server webhook you can utilize the corresponding templates associated with them.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned the role of a *Project Owner* in the project. See Managing access to the server projects on page 1220.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See Managing members and their roles in a team space on page 1207.
- Configured a webhook template. See Configuring a webhook template on page 1248.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Webhooks** in the navigation pane.

   The **Templates** tab of Webhooks is displayed.
4. Perform the tasks indicated in the following table:

Task	Action
Configuring a webhook template	Click the **New Template** option to configure a webhook template. See Configuring a webhook template on page 1248.
Viewing webhook templates	View the following details about the project, team space and server templates: a. The **Show all levels** checkbox selected. b. The following icons that display the server, team space, and project webhook templates in the **From** column.

Task	Action
	▪ Project template icon  ▪ Team space template icon  ▪ Serve template icon  As a project owner, you can view project, team space, and server templates on the project template page. Similarly, as a team space owner, you can view only team space and server templates on the team space template page. Perform the following steps to view project or team space templates: a. Clear the **Show all levels** checkbox. You can only view project webhook templates, if you are on the project template page, and you can only view team space templates, if you are on the team space template page. b. Click the **Action** icons to edit, duplicate view or delete the templates.  **Note:** As a *Project Owner*, you can edit the project webhook templates; however, you can only view and clone the details of server webhook templates or team space webhook templates. If you are a *Team Space Owner* you can edit the team space templates; however, you can only view and clone the details of server webhook templates. If you clone a webhook template, then you can edit or delete a webhook template.
Editing a webhook template	Perform the following steps to edit the webhook template: a. Click the **Edit template** icon  . The **Edit template** page is displayed. b. In the **Details** tab, you can enter a new name for the webhook template, or select any other value for the options. c. In the **Templates** tab, you can also edit or select a template based on the channel type you selected on **Details** tab. d. Click **Apply**. The edited template is saved to the **Templates** tab.
Cloning a webhook template	Perform the following steps to duplicate the webhook template: a. Click the **Duplicate template** icon  . The **Duplicate template** page is displayed. b. In the **Details** tab, you can edit or modify the values by entering a new name for the webhook template, by selecting a Channel Type, or Events.

Task	Action
	c. In the **Templates** tab, you can edit or select other webhook template based on the channel type you selected on **Details** tab. d. Click **Apply**. The cloned webhook template is added to the **Templates** tab.
Deleting a webhook template	Perform the following steps to delete the webhook template: a. Click the **Delete Template** icon  . The Delete template confirmation dialog is displayed. b. Select the **I understand template deletion is permanent** checkbox, and then click **Delete**. The webhook template is removed from the template list.

Results

You achieved the following results:

- Edited and saved a webhook template configuration.
- Cloned a webhook template configuration.
- Deleted a webhook template configuration.

Managing the webhook configuration

You can manage the webhook configuration that you set on HCL DevOps Test Hub (Test Hub) you want to edit, duplicate, delete the existing webhook configurations, or create a webhook.

Before you begin

- Ensured that you are assigned a role of a project *Owner* in the project. See Managing access to the server projects on page 1220.
- Ensured that you configured a webhook. See Configuring a project webhook on page 1260.

1. Log in to Test Hub.
   The **Projects** page of the initial team space is displayed.
2. Click **My projects > *project_name*** to open the project that contains the test assets.

   The **Overview** page of the project is displayed.
3. Click **Manage > Webhooks** in the navigation pane.
   The **Templates** tab of Webhooks is displayed.
4. Click the **Webhooks** tab.
5. Perform the tasks indicated in the following table:

Task	Action
New Webhook	Click the **New Webhook** option to configure and create a webhook. See Configuring a project webhook on page 1260.
Editing a webhook	Perform the following steps to edit the webhook: a. Click the **Edit Webhook** icon. The **Edit webhook** page is displayed. b. Enter a new name for the webhook. c. Replace the webhook URL by pasting a different URL of another messaging channel you configured. d. Select another channel type to post the notification of an event. e. Select a different template, and then select events from the **Events** list. f. Click **Close**.
Cloning a webhook	Perform the following steps to clone the webhook: a. Click the **Duplicate webhook** icon. The **Duplicate webhook** page is displayed. b. Enter a new name for the webhook, and then click **Apply**. The cloned webhook is added to the **Webhooks** tab.
Deleting a webhook	Perform the following steps to delete the webhook: a. Click the **Delete Webhook** icon. The **Delete Webhook** confirmation dialog is displayed. b. Select the **I understand webhook deletion is permanent** checkbox, and then click **Delete**. The webhook is removed from the webhook list.

Results

You achieved the following results:

- Edited and saved a webhook configuration.
- Cloned a webhook configuration.
- Deleted a webhook configuration.

Enabling a webhook

You can enable a webhook when push notifications about events occurring on HCL DevOps Test Hub (Test Hub) are not delivered to the configured webhook due to communication failures or network issues, and you receive a notification that the webhook is disabled.

Before you begin

You must have completed the following tasks:

- Ensured that you are assigned the role of a *Server Administrator*, *Team Space Owner* or a *Project Owner*. If you are a server administrator, you can have three roles-server admin, team space owner, and project owner.
- Configured a webhook. See Configuring a project webhook on page 1260.

About this task

When a webhook is disabled due to communication failure or network issues, you are notified about the disablement of the webhook. Consequently notifications for the events are not delivered to the configured webhook and instead be queued for sending after you re-enable the webhook. You have the option to either send the queued messages or discard the queued messages.

1. Select from the following actions that you want to perform:
   - Enable a project webhook. Go to Step 2 on page 1266.
   - Enable a team space webhook. Go to Step 3 on page 1266.
   - Enable a server webhook. Go to Step 4. on page 1267
2. Perform the following steps to manually enable a project webhook:
   a. Log in to Test Hub.

      The **Projects** page of the initial team space is displayed.

   b. Click **My projects > *project_name*** to open the project that contains the test assets.

      The **Overview** page of the project is displayed.

   c. Click **Manage > Webhooks**.

      The **Templates** tab of Webhooks is displayed.

   d. Click the **Webhooks** tab, and then click the **Disabled** webhook option to enable a webhook that is disabled.

   The event notifications that were queued when the webhook was disabled are delivered to the configured webhook or discarded by Test Hub.This setting is based on the resume policy configuration that you set for the specific project webhook.
3. Perform the following steps to manually enable a team space webhook:
   a. Log in to Test Hub.

      The **Projects** page of the initial team space is displayed.

b. Click the **Settings** icon on the navigation pane, and then click **Manage team space**.
c. Click **Webhooks**.

The **Templates** tab of Webhooks is displayed.

d. Click the **Webhooks** tab, and then click the **Disabled** webhook option to enable a webhook that is disabled.

The event notifications that were queued when the webhook was disabled are delivered to the configured webhook or discarded by Test Hub. This setting is based on the resume policy configuration that you set for the team space webhook.

4. Perform the following steps to manually enable a server webhook:

a. Log in to Test Hub.

The **Projects** page of the initial team space is displayed.

b. Click the **Settings** icon on the menu bar, and then click **Webhooks**.

The **Templates** tab of Webhooks is displayed.

c. Click the **Webhooks** tab, and then click the **Disabled** webhook option to enable a webhook that is disabled.

The event notifications that were queued when the webhook was disabled are delivered to the configured webhook or discarded by Test Hub. This setting is based on the resume policy configuration that you set for the server webhook.

Results

You enabled the webhook on Test Hub to push notifications of the events that failed to be delivered to the configured webhook.

Management of events

When an identified event occurs on HCL DevOps Test Hub (Test Hub), a webhook sends an HTTP POST request to the URL that you configured for the webhook. The request contains the message associated with the event, in JSON format or plain text. Each event corresponds to a set of actions that occur. For example, when a member is added to a Team space or when a Test failed with an error.

The following tables describe different events that are identified on Test Hub.

Table 36. Main

Event	Description
All Events	When any of the events occur, a webhook sends a notification.

Table 37. Team Space Events

<table>
<tr><th>Event</th><th>Description</th></tr>
<tr><td>Team Space Management</td><td>When events occur at the team space management level, a webhook sends a notification.</td></tr>
<tr><td>Team Space Created</td><td>When a team space is created, a webhook sends a notification.</td></tr>
<tr><td>Team Space Deleted</td><td>When a team space is deleted, a webhook sends a notification.</td></tr>
<tr><td>Team Space Updated</td><td>When a team space is updated, a webhook sends a notification.

Following are the subsets of the team space events:

<table><tr><th>Event</th><th>Notification sent</th></tr><tr><td>Team Space URL alias Updated</td><td>When the team space URL alias is updated.</td></tr><tr><td>Team Space Renamed</td><td>When the name of the team space is changed.</td></tr><tr><td>Team Space Becomes Permissive</td><td>When the team space is accessible.</td></tr><tr><td>Team Space Becomes Private</td><td>When an access to the team space is not shared.</td></tr><tr><td>Team Space Becomes Public</td><td>When all members are provided access to the team space.</td></tr><tr><td>Team Space Becomes Restrictive</td><td>When access to the team space is limited.</td></tr></table></td></tr>
<tr><td>Team Space Repositories</td><td>When an event occurs in the team space repositories, a webhook sends a notification.

Following are the subsets of the team space repository events:

<table><tr><th>Event</th><th>Notification sent</th></tr><tr><td>Team Space Repositories Created</td><td>When the team space repositories are created.</td></tr></table></td></tr>
</table>

Table 37. Team Space Events (continued)

<table>
<tr><th>Event</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Team Space Repositories Deleted</td><td>When the team space repositories are deleted.</td></tr>
</table>
</td></tr>
<tr><td>Team Space Member</td><td>When an event occurs in the team space, a webhook sends a notification.

Following are the subsets of the team space member events:
<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Team Space Member Added</td><td>When a member is added to the team space.</td></tr>
<tr><td>Team Space Member Removed</td><td>When a member is removed from the team space.</td></tr>
<tr><td>Team Space Member Role Added</td><td>When a specific role for a member is added.</td></tr>
<tr><td>Team Space Member Role Removed</td><td>When a specific role of a member is removed.</td></tr>
</table>
</td></tr>
<tr><td>Team Space Role Request</td><td>When a request is sent to join a team space, a webhook sends a notification.

Following are the subsets of the team space role request events:
<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Team Space Role Requested</td><td>When a member sends a request to get a role in the team space.</td></tr>
<tr><td>Team Space Role Request Accepted</td><td>When the request of the member for a role in the team space is accepted.</td></tr>
<tr><td>Team Space Role Request Rejected</td><td>When the request of the member for a role in the team space is rejected.</td></tr>
</table>
</td></tr>
</table>

Table 38. Project Events

<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Project Management</td><td>When events occur at the project level, a webhook sends a notification.

The Following are the subsets of project management events:

<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Project Archived</td><td>When a project is archived.</td></tr>
<tr><td>Project Created</td><td>When a project is created.</td></tr>
<tr><td>Project Renamed</td><td>When a project is renamed.</td></tr>
<tr><td>Project Deleted</td><td>When a project is deleted.</td></tr>
<tr><td>Project Unarchived</td><td>When a project is unarchived.</td></tr>
</table>
</td></tr>
<tr><td>Project Repositories</td><td>When an event occurs in the project repositories, a webhook sends a notification.

Following are the subsets of the project repositories events:

<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Project Repository Created</td><td>When a project repository is created.</td></tr>
<tr><td>Project Repository Deleted</td><td>When a project repository is deleted.</td></tr>
</table>
</td></tr>
<tr><td>Project Member Role</td><td>When the role of a member of a project is changed, a webhook sends a notification.

Following are the subsets of the project member role events:

<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Project Member Role Added</td><td>When a project member role is added to the project.</td></tr>
<tr><td>Project Member Role Removed</td><td>When a project member role is removed from the project.</td></tr>
</table>
</td></tr>
<tr><td>Project Member Role Requests</td><td>When a request is sent to add a member role to the project, a webhook sends a notification.</td></tr>
</table>

Table 38. Project Events (continued)

Event	Notification sent
	Following are the subsets of the project member role requests events: **Event** / **Notification sent** Project Member Role Requested — When a member sends a request to join a project. Project Member Role Request Accepted — When a request of the member to join a project is accepted. Project Member Rejected — When a request of the member to join a project is rejected.

Table 39. Execution Events

Event	Description
Test and Suite Status	Webhook sends notifications about the status of the test and test suite. Following are the subsets of the test and test suite status events: **Event** / **Notification sent** Execution Started — When you start to run a test or a test suite. Execution Stopped Manually — When the test run is manually stopped. Execution Completed with Verdict Pass — When the test run is completed with a pass verdict.
Test and Suite Status Errors	Webhook sends notifications about the error status of the test and test suite. Following are the subsets of test and test suite status error events: **Event** / **Notification sent** Execution Completed with Verdict Fail — When the test run is completed with a fail verdict. Execution Completed with Verdict Error — When the test run is completed with an error status. Execution Completed without any Verdict — When the test run is completed without any results.

Table 39. Execution Events (continued)

<table>
<tr><th>Event</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Execution Launch Failed</td><td>When the test launch fails.</td></tr>
<tr><td>Execution Stopped with Error</td><td>When the test run stops with an error.</td></tr>
<tr><td>Execution Stopped Automatically</td><td>When automatically the test run stops.</td></tr>
<tr><td>Execution Incomplete</td><td>When the test run is not completed.</td></tr>
</table>
</td></tr>
<tr><td>Virtual Service Status</td><td>Webhook sends notifications about the status of the virtual services.

Following are the subsets of the virtual service status events:

<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Virtual Service Started</td><td>When a virtual service is started.</td></tr>
<tr><td>Virtual Service Stopped Manually</td><td>When a virtual service is manually stopped.</td></tr>
</table>
</td></tr>
<tr><td>Virtual Service Errors</td><td>Webhook sends notifications about the errors that occur in the virtual services.

Following are the subsets of the virtual service error events:

<table>
<tr><th>Event</th><th>Notification sent</th></tr>
<tr><td>Virtual Service Launch Failed</td><td>When a virtual service launch fails.</td></tr>
<tr><td>Virtual Service Stopped with Error</td><td>When a virtual service is stopped with an error.</td></tr>
<tr><td>Virtual Service Stopped Automatically</td><td>When a virtual service is automatically stopped.</td></tr>
</table>
</td></tr>
</table>

Table 40. License Events

Event	Notification sent
License Failure	When a valid license is not obtained.

Chapter 8. Troubleshooting

This guide describes how to analyze and resolve some of the common problems that you might encounter while you work with HCL DevOps Test Hub (Test Hub).

Known issues are documented in the Release document and the link is in the Release Notes under the Known issues section. See Defect fixes and known issues on page 7.

You can contact HCL Support if you are unable to troubleshoot the problem. Gather all the required background information and provide the details to HCL Support for investigation. For more information, see HCL Customer Support.

Troubleshooting issues

You can find information about the issues or problems that you might encounter while working with HCL DevOps Test Hub (Test Hub). Details about issues, their causes and the resolutions that you can apply to fix the issues are described.

The troubleshooting issues are presented to you in the following tables based on where or when you might encounter these issues on Test Hub.

- Table 41: Troubleshooting issues: installation on page 1273
- Table 42: Troubleshooting issues: server administration on page 1275
- Table 43: Troubleshooting issues: configuring test runs on page 1276
- Table 44: Troubleshooting issues: test or stub runs on page 1278
- Table 46: Troubleshooting issues: Jira and Xray integrations on page 1282

Table 41. Troubleshooting issues: installation

Problem	Cause	Solution
On Ubuntu, when you are installing Test Hub and you encounter errors in the scripts that are running.	At times, scripts might not appear to be running due to any of the following reasons: • Slow connection speeds. • Insufficient CPU, memory, or disk resources. • A firewall that was configured incorrectly is already enabled.	You can complete any of the following tasks: • To identify the issue, you can perform a diagnostic check by running the following command: `journalctl -u k3s` This command displays the log that you can use to check for the problem. • Run the following command to see which pods are running and which pods are not running:

Table 41. Troubleshooting issues: installation (continued)

Problem	Cause	Solution

		`kubectl get pods -A`
		Run the following command to get details about a specific pod:
		`kubectl describe pod -n <namespace> <pod name>`
		• Follow the on-screen instructions to resolve the errors.
		• Some issues can be solved by re-running the following script:
		`sudo ./k3s-init.sh`

On Ubuntu, DNS is not working as expected.		The DNS configuration that is used by the cluster can be displayed by using the following command:
		`kubectl get cm -n kube-system coredns -ojsonpath="{.data.Corefile}"`
		The **forward** setting displays the *nameservers* that are used. For example, you might see the following in the `corefile`:
		`.:53 { : forward . 8.8.8.8 9.9.9.9 : }`
		A script (`ubuntu-set-dns.sh`) is supplied for managing these values.
		For example, to set the DNS values for the values shown in the previous example:

Table 41. Troubleshooting issues: installation (continued)

Problem	Cause	Solution

		`sudo ./ubuntu-set-dns.sh --server 8.8.8.8 --server 9.9.9.9`
		
		Note: If you do not use sudo in the command, the script runs but the configuration might be lost if the cluster is restarted.
		
		To learn more about the behavior of the script, run the following command:
		`sudo ./ubuntu-set-dns.sh --help`

When running helm install the created pods keep crashing, and the logs contain: ACCESS_REFUSED when trying to connect to RabbitMQ	In some instances, the RabbitMQ password is not automatically setup correctly.	Manually apply the necessary password: `kubectl exec -n <namespace> <release-name>-rabbitmq-0 -- rabbitmqctl change_password user \ "$(kubectl get secret -n <namespace> <release-name>-rabbitmq -o jsonpath='{.data.rabbitmq-password}' | base64 --decode)"`

Table 42. Troubleshooting issues: server administration

Problem	Cause	Solution
When a user is assigned an additional role, the change in the permissions is not observed in the browser.		You must log out of the session and log in again for the changed role to take effect.
You see the following message displayed on Test Hub: `You can't request to join a project that has no owners`	You requested to join an project that no longer has an owner. Orphaned projects occur when the project owners are deleted. This can occur, for	Ask an administrator to take ownership of the project, and then add you as a member.

Table 42. Troubleshooting issues: server administration (continued)

Problem	Cause	Solution
	example, when the person leaves the organization.	

Table 43. Troubleshooting issues: configuring test runs

Problem	Cause	Solution
When you configure a run of a schedule that matches the following conditions: • The schedule has two user groups configured to run on static agents when the schedule was created in Test Performance 10.1. • One of the user groups is disabled and the asset is committed to the remote repository. Both the static agents are displayed as available for the test run in the **Location** tab of the **Execute test asset** dialog when only one agent that is configured for the user group must be available.	The cause might be because of the following reasons: • The schedule was created in Test Performance 10.1. • The user group that is disabled is not removed or deleted from the test resources. • The agent configured on the disabled user group is already added as an agent to the server project and is available for selection.	To resolve the problem, select from either of the following methods: • By using Test Performance 10.1.1. Perform the following steps: 1. Open the schedule in Test Performance 10.1.1. 2. Save the schedule and the project. 3. Commit your test asset to the remote repository. 4. Proceed to configure a run for the schedule on Test Hub 10.1.1. • By using Test Performance 10.1. Perform the following steps: 1. Select the disabled user group. 2. Click **Remove**. 3. Save the schedule and the project. 4. Commit your test asset to the remote repository. 5. Proceed to configure a run for the schedule on Test Hub 10.1.1.

Table 43. Troubleshooting issues: configuring test runs (continued)

Problem	Cause	Solution
You have added a remote repository to your project that contains the test assets or resources of the following types: • Postman • JMeter • JUnit The test assets or resources are not displayed on the **Execution** page for you to select the asset for a run.	This problem occurs if the server extension is not enabled. Although the extension was enabled at the time of installation of Test Hub, it was disabled subsequently by the server administrator.	You must verify if the server extension is enabled and running by running the following command: kubectl get pod -n devops-system, where *devops-system* is the namespace that you created to install Test Hub. The server extensions that are running are displayed. If the server extension that you want is not running implying that the server extension is not enabled. You must enable the server extension. Contact the server administrator to enable the server extension.
When you run AFT Suites in Test Hub, the test run finishes with the `Completed with error` message in the **Status** column. On hover over this status, the `Pod failed reason OOMKilled` message is displayed.	The test run fails because of the higher memory consumption.	To resolve this issue, perform one of the following steps: • Increase the Execution Pod memory to 8Gi if you run the test in a container environment: `execution.template.resources.limits.memory: 8Gi` For details on the Helm parameter, see Set memory limit on page 119. • Decrease the number of tests in the AFT Suite if you want to run the test without an agent. Edit the `.xml` file of the AFT Suite in HCL DevOps Test UI (Test UI) to reduce the number of tests, and then commit the changes.

Table 43. Troubleshooting issues: configuring test runs (continued)

Problem	Cause	Solution
		• Use an agent to run the test if you cannot decrease the asset size of the AFT Suite. You can perform one of the following steps: ◦ When running the test, in the **LOCATION** tab, select an agent in the **Override** column. ◦ Edit the `.xml` file of the AFT Suite in Test UI to specify an agent in the locations section, and then commit the changes.

Table 44. Troubleshooting issues: test or stub runs

Problem	Cause	Solution
You encounter any of the following issues: • When many tests are run simultaneously on the default cluster location and you observe the following issues: ◦ Out of memory errors. ◦ Observe that the test runs are slow with a high CPU usage. ◦ The Kubernetes pods are getting evicted. • When you run an AFT suite that contains multiple Web UI tests and you observe the following issues:	The issue is seen when any of the following events occur: • Many tests are run in parallel. • The memory that is used by the tests during the test run exceeds the allocated default memory of 1 GB. • The default memory of the container is not adequate for the test run. • Pods are evicted due to low node memory.	To resolve the problem, you can increase the resource allocation for test runs by using the arguments listed in Table Table 45: Increasing resource allocation on page 1280. You can enter arguments in the **Additional configuration options** field in the **Advanced** settings panel of the **Execute test asset** dialog when configuring a test run. **Important:** The memory settings that you configure for a test run is persisted for the test when ever you run it. You must use this setting judiciously.

Table 44. Troubleshooting issues: test or stub runs (continued)

Problem	Cause	Solution
◦ Error stating that the browser might not be installed or the browser version is unsupported. ◦ Error stating multiple random time-outs or an internal error. • When you run VU schedules that contain multiple Web UI test or large number of virtual users or a combination of Web UI, performance, and API tests, and you observe the following issue: ◦ The test run hangs and the inadequacy of memory can be inferred as the cause in the logs.		Configuring all tests for an increased memory limit might affect subsequent test runs or cause other memory issues when tests run simultaneously. In addition, in the **JVM Arguments** field under the **Advanced** settings you can set the maximum heap size for the test run time. For example, adding the JVM argument **-Xmx3g** sets the maximum heap size to *3Gi*.
You are not able to run Web UI tests on a remote agent that runs on Linux because the browser fails to launch.	Browser launch is prevented when the remote agent runs as *root* or *sudo*.	To resolve this issue, you must perform the following steps after you have installed the remote agent: 1. Stop the agent. 2. Start the agent as the logged in user by performing the following steps: a. Open a terminal on Linux. b. Enter the following commands: cd / opt/HCL/DevOpsTest/Majordomo/ sudo ./MDStop.sh ./MDStart.sh

Note: You can refer to the Kubernetes documentation for information about the different units that can be used for resources in the **Additional configuration option** fields.

Table 45. Increasing resource allocation

Requirement	Configuration option name	Default value, if no value is set	An example value	Result of using the example value
Specifying the memory limit of the *init container*.	`init.resource.memory.limit`	*1024Mi*	*2048Mi*	Increases the memory limit of the *init container* from the default value to *2048Mi*.
Configuring a larger memory request for the *init container* to avoid pod eviction.	`init.resource.memory.request`	*64Mi*	*1024Mi*	Increases the initial memory request for the *init container* from the default value to *1024Mi*.
Specifying the cpu request for the *init container*.	`init.resource.cpu.request`	*50m*	*60m*	Increases the cpu request for the *init container* from the default value *60m*.
Specifying the memory limit of the container used for the test run. **Note:** If the memory limit you set is more than the default limit, you must run the following command to increase the allotted	`resource.memory.limit`	The larger of *3Gi* or *maximum heap size + 1Gi*	*4Gi*	Changes the memory limit of the main container from the default value to *4Gi*.

Table 45. Increasing resource allocation

(continued)

Requirement	Configuration option name	Default value, if no value is set	An example value	Result of using the example value
limit to 8 GB or any other value above the default value of 3 GB, at the time of installing the server or anytime later: `--set execution.template.resources.limits.memory=8Gi` You must be a server administrator to run this command.				
Specifying the memory request for the container used by the test run.	`resource.memory.request`	*1024Mi*	*2048Mi*	Increases the memory request for the main container from the default value to *2048Mi*.
Specifying the cpu request for the main container used by the test run.	`resource.cpu.request`	*50m*	*70m*	Increases the cpu request for the main container from the default value to *70m*.

Table 46. Troubleshooting issues: Jira and Xray integrations

Problem	Cause	Solution
You are not able to create Jira work items in Jira. You are also not able to view the created work items in Jira that you created in HCL OneTest™ Server 10.5.4 or earlier.	The cause can be due to the change in the authentication method used to connect to Jira in Test Hub 11.0.0 or later.	Reconfigure Jira as a change management system in Test Hub 11.0.0 or later. See Configuration of Atlassian Jira as a change management on page 1093.
You are not able to export results to the Xray project. You are also not able to view the results exported to the Xray project that you created in HCL OneTest™ Server 10.5.4 or earlier.	The cause can be due to the change in the authentication method used to connect to Jira in Test Hub 11.0.0 or later.	Reconfigure the Xray project after you reconfigure Jira Test Hub 11.0.0 or later. See Integration with Xray test management for Jira on page 1188.

Failed to install DevOps Test Runtime on Windows 11 25H2

When you attempt to install DevOps Test Runtime on Windows 11 version 25H2, the standard installation might fail with an error message.

The installer might display the following error message:

```
There is a problem with this Windows Installer package. A program required for this install to
 complete could not be run. Contact your support personnel or package vendor.
```

Resolution

To work around this issue, you must install the `devops-test-runtime.msi` file by using a command prompt as an administrator.

Note: You must ensure that the file name of the MSI package does not contain any spaces. For example, if you download the file multiple times, your browser might save it as `devops-test-runtime (1).msi`. Running the installation commands with spaces in the file name causes the command to fail and triggers an error dialog. To avoid this issue, you must rename the downloaded file to remove any spaces before you run the commands.

Perform the following steps:

1. Open a command prompt as an administrator.
2. Navigate to the directory where the `.msi` file is available.
3. Enter the following command in the command prompt:

   ```
   msiexec /i MSI_PATH
   ```

   The installation window with the progress bar is displayed.

4. Click **Proceed** to accept the download of the proprietary certificate if prompted.

   You have installed DevOps Test Runtime by using the command-line interface.

Related information

Recording a Web UI test on page 211

Failed to load the extension in Google Chrome while recording

When you select the Chrome browser to record a Web UI test in HCL DevOps Test Hub (Test Hub), you might get the Failed to load extension message.

Resolution

You must install the DevOps Test browser extension for Chrome that is available on the Chrome Web Store at DevOps Test Chrome extension.

You have an option to install the extension in the default profile of Chrome or use one of the other profiles available. If you use a different profile, then you must add an environment variable in your system with the following values:

- **Variable name**: DEVOPS_TEST_CHROME_PROFILE_NAME
- **Variable value**: <profile_name>

Perform the following steps:

1. Open Google Chrome.
2. Switch to the Chrome profile for which you want to add the extension by clicking the profile on the toolbar. On a Mac system, you must run the following command to create a new temporary profile by cloning the existing profile:

   - mkdir -p "$HOME/Library/Application Support/devops-test-runtime/chrome-temp-profile" && cp -R "$HOME/Library/Application Support/Google/Chrome/Default" "$HOME/Library/Application Support/devops-test-runtime/chrome-temp-profile/"

   And then open Chrome with the cloned profile.
3. Go to the Chrome Web Store.

   The extensions page is displayed.
4. Click **Add to Chrome**.

   The **Add "DevOps Test"?** dialog is displayed.
5. Click **Add extension**.

   A confirmation message is displayed.
6. Close all the Chrome browsers.

   Additionally, you can check and terminate any active browser processes in the background.

7. Open Test Hub in the Edge or Firefox browser.
8. Record a test by choosing the Chrome browser from the **Browser** drop-down list.

Chapter 9. Unable to publish test assets to a Control repository

When you attempt to publish test assets from HCL DevOps Test Hub (Test Hub) to a Control repository, the publish operation might fail with the following message: Remote rejected; pre-receive hook declined.

Cause

The branch protection rule that is configured in Control prevents Test Hub from pushing test assets directly to the **main** branch.

Resolution

You must publish the test assets to a new branch, and then merge the changes into the `main` branch through a pull request.

Perform the following steps:

1. Create a branch in Control.

   For example, `ui_test`.
2. Select the newly created Control branch from the **Branch** list in Test Hub.

   For example, `ui_test`.
3. Create an Edit branch.

   For example, `ui_test_sub`. This is a local branch that you can merge into `ui_test` that you created.
4. Create test assets or make changes to the existing assets.
5. Go to **Author > Changes**.
6. Select the Control branch from the **Branch** list.

   For example, `ui_test`. All the updated assets that are present in the Edit branch are displayed in the **Changes** page.
7. Click **Publish**.

   The **Publish** dialog is displayed.
8. Enter a description for the changes that you performed to the asset, and then click **Publish**.
9. Create a pull request from the newly created branch to the `main` branch in Control.
10. Review and merge the pull request into the `main` branch.

Related information

Publishing all changes in an Edit branch on page 493

Pull Request

Security Considerations

This document describes the actions that you can take to ensure that your installation is secure, customize your security settings, and set up user access controls in HCL DevOps Test Hub (Test Hub).

- Enabling secure communication between multiple applications on page mcclxxxvi
- Ports, protocols, and services on page mcclxxxvi
- Customizing your security settings on page mcclxxxvi
- Setting up user roles and access on page mcclxxxvii

Enabling secure communication between multiple applications

The majority of communications are sent over TLS to port 443 (see Ports, protocols, and services on page mcclxxxvi). During the installation, an X.509 certificate is generated for the user provided DNS name, which is used to connect to the server. This certificate is self-signed and hence untrusted by other applications. This self-signed certificate must be replaced by a certificate signed by a certificate authority trusted by your organization. For more information, see X.509 Certificate User Authentication in the Keycloak documentation.

For information about how the self-signed certificate was created, see the `ssl.sh` file in the `<install-directory>/prepare/` directory.

For information about importing a certificate authority trusted by your organization, see Certificate authority: Importing and extending lists on page 161.

Communications between containers are in plaintext by default, and they can be secured only if the Container Network Interface (CNI) supports encryption.

Ports, protocols, and services

TCP port 443 is used by the majority of communications with the server.

The port 7085 is the default port for communications with agents registered with Test Hub.

The ports starting from 7085, are used in pairs such as 7085 and 7086, and are allotted for the Schedule that is executed first. The next Schedule is allotted the next pair (7087,7088), and so on for the Schedules that are running simultaneously.

You must open the required ports in pairs for each of the Schedules that you want to run simultaneously.

Customizing your security settings

You can customize your security settings through user registration.

User registration

By default, users can sign up themselves with the server. In some environments, this self sign-up might be undesirable. It can be changed by switching off user registration. For more information, see User Registration in the Keycloak documentation.

By default, user email addresses are not verified. This verification must be enabled in production environments. For more information, see Email settings on page 144.

Setting up user roles and access

You can manage user roles and access through single sign on (SSO) and administration only accounts.

Single sign-on

By default, Keycloak manages users and passwords locally. In production environments, it is normally appropriate to use single sign-on. For more information, see LDAP user administration on page 144.

Administration only accounts

Users in the Administrator group can manage the team space where they can configure licenses and notifications and add a repository to store the System model. For more information, see Team space overview on page 172.

Users in the Administrator group can also discover all projects stored on the server (including private ones) and assign themselves and others roles in those projects.

For this reason, users who use the server to perform both administration and non-administration tasks must have two different accounts, one for each purpose. For more information, see Default user administration on page 142.

Notices

This document provides information about copyright, trademarks, terms and conditions for the product documentation.

© Copyright IBM Corporation 2000, 2016 / © Copyright HCL Technologies Limited 2016, 2024

This information was developed for products and services offered in the US.

HCL® may not offer the products, services, or features discussed in this document in other countries. Consult your local HCL® representative for information on the products and services currently available in your area. Any reference to an HCL® product, program, or service is not intended to state or imply that only that HCL® product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any HCL® intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-HCL® product, program, or service.

HCL® may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

For license inquiries regarding double-byte character set (DBCS) information, contact the HCL® Intellectual Property Department in your country or send inquiries, in writing, to:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

HCL TECHNOLOGIES LTD. PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. HCL® may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-HCL® websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this HCL® product and use of those websites is at your own risk.

HCL® may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by HCL® under terms of the HCL® Customer Agreement, HCL® International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

Information concerning non-HCL® products was obtained from the suppliers of those products, their published announcements or other publicly available sources. HCL® has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-HCL® products. Questions on the capabilities of non-HCL® products should be addressed to the suppliers of those products.

Statements regarding the future direction or intent of HCL® are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to HCL®, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. HCL®, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. HCL® shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:
© (your company name) (year).
Portions of this code are derived from HCL Ltd. Sample Programs.
© Copyright HCL Ltd. 2000, 2022.

Trademarks

HCL®, the HCL® logo, and hcl.com® are trademarks or registered trademarks of HCL Technologies Ltd., registered in many jurisdictions worldwide. Other product and service names might be trademarks of HCL® or other companies.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the HCL® website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of HCL®.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of HCL®.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

HCL® reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by HCL®, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

HCL® MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Index

A

accessibility
  Test Hub
    disability 112
    keyboard 112
add user action 252
Add user action for this element 239
Add user action in the test editor 252
administration default security
  Test Hub
  142
administration ldap security
  Test Hub
  144

C

Change
  Dataset 447
  Encrypt 447
  Security Group 447
Configure
  Dataset 443
Create
  Security Group 447

D

Dataset 443
datasets
  encrypting 440
  removing encryption 442
Delete
  Security Group 447

E

Edit
  Security Group 447

G

Grant
  access 447
  Security Group 447

M

Manage encrypted dataset 447

R

Remove
  Dataset 447
  Encrypt 447
  Security Group 447
Revoke
  access 447
  Security Group 447

S

security considerations
  overview mcclxxxvi
Set
  Current row 443
  Dataset 443

T

test editor overview 235
Test Hub
  accessibility 112
  default security administration 142
  disability 112
  keyboard 112
  ldap security administration 144
tests
  datasets 440, 442
  editing 235

V

View
  Dataset 443, 443
  Encrypt 443