HCLSoftware

# HCL **iAutomate**

**Integration Guide**

Version 6.4.2

HCL has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the
HCL website at www.hcltechsw.com.

# Contents

# Table of Figures

# List of Tables

# Document Revision History

This guide updates with each release of the product or when necessary.

This table provides the update history of this Integration Guide.

| Version Date | Description |
|---|---|
| October, 2019 | HCL iAutomate v4.0 Integration Guide |
| May, 2020 | HCL iAutomate v5.0 Integration Guide |
| September, 2020 | HCL iAutomate v6.0 Integration Guide |
| November, 2020 | HCL iAutomate v6.0.1 Integration Guide |
| January, 2021 | HCL iAutomate v6.0.2 Integration Guide |
| April, 2021 | HCL iAutomate v6.0.3 Integration Guide |
| October, 2021 | HCL iAutomate v6.1 Integration Guide |
| March, 2022 | HCL iAutomate v6.1.1 Integration Guide |
| August, 2022 | HCL iAutomate v6.2.1 Integration Guide |
| March, 2023 | HCL iAutomate v6.3 Integration Guide |
| October, 2023 | HCL iAutomate v6.3 Integration Guide |
| December, 2023 | HCL iAutomate v6.3.2 Integration Guide |
| June, 2024 | HCL iAutomate v6.4.0 Integration Guide |
| August, 2024 | HCL iAutomate v6.4.1 Integration Guide |
| November, 2024 | HCL iAutomate v6.4.2 Integration Guide |

# 1 Preface

This section provides information about the HCL iAutomate Integration Guide and includes the following topics-

- Intended Audience
- About This Guide
- Related Documents
- Conventions

## 1.1 Intended Audience

This information is intended for administrators authorized for configuring iAutomate and enable integrations with various ITSM tools and Runbook Automation / Orchestrator Tools.

## 1.2 About this Guide

This guide provides instructions to enable integrations with various ITSM and Runbook Automation tools, while configuring iAutomate.

## 1.3 Related Documents

The following documents can be referenced in addition to this guide for further information on the iAutomate platform.

- **iAutomate Configuration Guide**
- **iAutomate Troubleshooting Guide**
- **iAutomate Lab Manual**

## 1.4 Conventions

The following typographic conventions are used in this document:

Table 1 – Conventions

| Convention | Element |
|---|---|
| **Boldface** | Indicates graphical user interface elements associated with an action, or terms defined in text or the glossary |
| Underlined Blue face | Indicates cross-reference and links |
| *Italic* | Indicates document titles, occasional emphasis, or glossary terms |
| Courier New (Font) | Indicates commands within a paragraph, URLs, code in examples, and paths including onscreen text and text input from users |
| Numbered lists | Indicates steps in a procedure to be followed in a sequence |
| Bulleted lists | Indicates a list of items that is not necessarily meant to be followed in a sequence |

# 2 iAutomate Overview

iAutomate is an Intelligent Runbook Automation product which is equipped with Artificial Intelligence, Machine Learning and Natural Language Processing capabilities for simplifying and automating the IT Operations issues resolution lifecycle including incidents, service request tasks, change request tasks and events. It leverages its NLP capabilities for analyzing and understanding the context of a specific issue, recommends the most relevant solution and even triggers the execution, thereby enabling Zero Touch Automated Remediation. It also provides AI-driven Knowledge Recommendation by suggesting relevant knowledge articles from various repositories, both internal and external, as and when required by human agents.

When no runbook is available for automated remediation, it searches & downloads relevant executable codes and scripts for subject matter expert to validate, customize, approve and publish for future use.

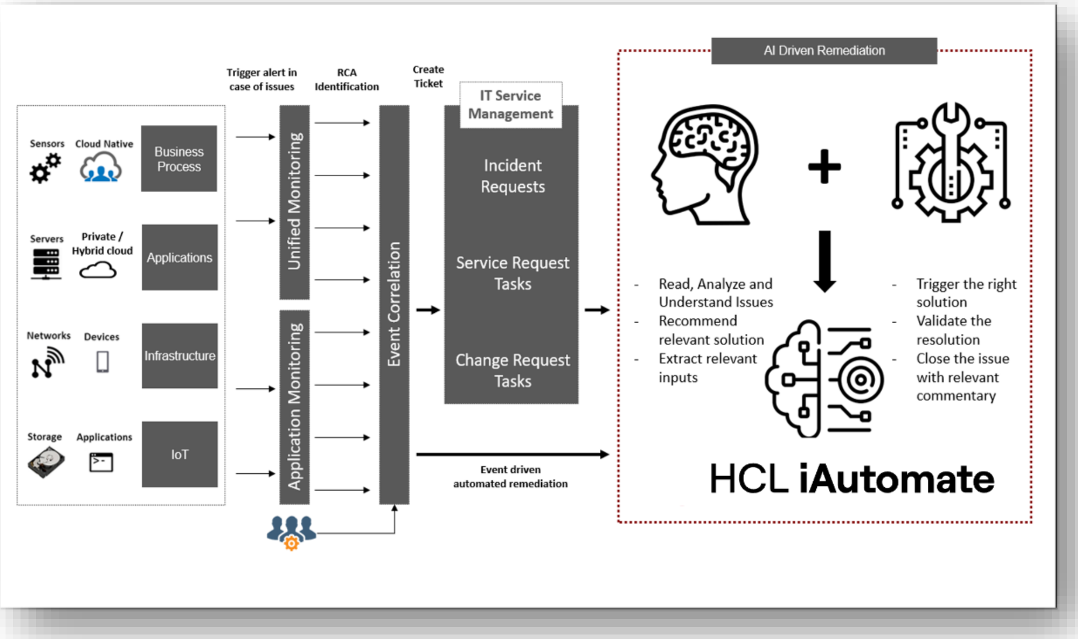


Figure 1 - iAutomate Workflow

Intelligent automation powered by HCL iAutomate can make a tremendous impact in an enterprise adjusting to the New Normal:

- **Reduce Costs**
  - Achieve up to 30% reduction in service desk related costs
  - Quick and High ROI
- **Mitigate Risks**
  - Avoid operational risks and ensure compliance by avoiding critical outages
  - Reduce escalations and improve SLA compliance by up to 20%
  - Achieve up to 85% reduction in MTTTR
- **Drive Efficiency**

- Automate redundant tasks and let employees focus on more creative activities
- Reduce manual effort by 30% to 60%
- Improve customer satisfaction by up to 50% by providing faster incident and service request resolutions.

– **Rapid Time to Value**

- Quick implementation in 6 to 8 weeks*
- Leverage 3000+ reusable and configurable runbooks out of the box
- Achieve zero-touch automation state in 4 to 5 months*

*Conditions Apply

# 3 Integration Ecosystem

This section describes the different types of tools with which iAutomate can integrate for achieving end to end issue resolution.

Primarily, iAutomate integrates with three types of tools –

- **ITSM Tools**

The purpose is to fetch the ticket data from the IT Service Management tool to read / understand the ticket and for making any changes to the ticket like updating the status, work notes, transferring to a different queue or closing the ticket.

- **ITSM Tools support**
  - ServiceNow
  - BMC Remedy
  - Cherwell ITSM
  - BMC Remedyforce
  - Jira
  - ServiceXchange(SX)
- **Event Management Tools**

The purpose is to fetch the event data from the Event Management tool to understand the issue and recommend / trigger the relevant runbook for remediation.

- Event Management Tools support
  - Moogsoft
  - Zenoss
- **RBA / Orchestrator Tools**

The purpose is to direct the RBA / orchestrator tools to trigger the runbook for resolving the incident, after iAutomate has identified the appropriate runbook. iAutomate also continuously pulls the current status of the execution from the RBA tool and reports it in its Logs section.

- **RBA Tools support**
  - Broadcom CA ITPAM
  - Microsoft System Orchestrator
  - Ansible Tower / AWX
  - Ansible CLI
  - VMware vRealize Orchestrator (vRO)
  - Microfocus Operations Orchestration
  - ServiceNow Orchestration
  - StackStorm
  - Ansbile Inside
  - BigFix

- Jenkins

Subsequent sections will cover the integrations with these tools in detail.

# 4 Integration with IT Service Management Tools

Any IT Service management tool acts as a data source for iAutomate from where it pulls the ticket data and then performs appropriate actions for resolution. Thus, to enable integration with ITSM, it requires a data source to be created as part of iAutomate configuration.

Given that the APIs for **Incident Management**, **Service Request Tasks** and **Change Request Tasks** are different, a separate data source will have to be configured for each of the previously mentioned categories.

Before proceeding with the configuration related to Data Source creation, the user has to ensure that an organization has been configured. If not done already, please refer to the Configuration Guide for the same and create the organization before proceeding ahead.

## 4.1 Common Pre-requisite

- API to Fetch tickets, Ticket in progress, Ticket Close, Ticket Release
- USER permission to query, modification on Tickets

## 4.2 Integration with ServiceNow

### 4.2.1 Incident Management

To fetch information about Incidents, usually, creation of a data source for Incident Management should suffice. However, there could be scenarios where some additional fields / values are required from CMDB for processing the tickets – recommending the relevant runbooks and parsing the tickets to extract relevant parameters, for which separate data sources for CMDB CI must be created. Here, we will cover the procedure for creating both kinds of data sources.

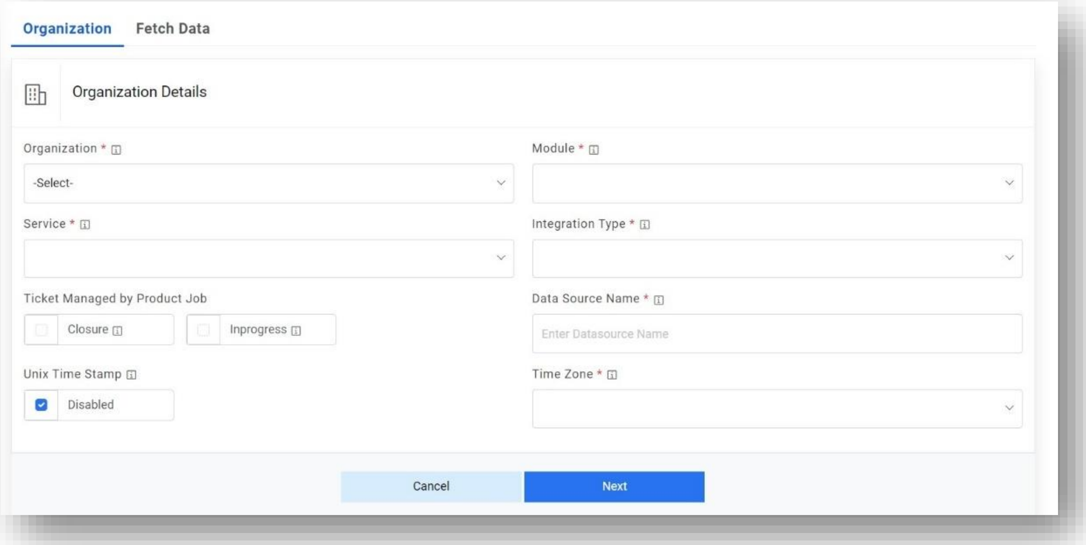
#### 4.2.1.1 Create Data Source for Incident Management

To create a data source for Incident Management, perform the following steps:

1. On the left menu bar, click **Configuration** -> **Manage Data Sources**.
2. The **Create Data Source** (Plus Icon) page appears with the following tabs:
   - Organization
   - Fetch Data

Release Rules

- Close Rules (Optional – applicable only when the ticket closure status update is managed by iAutomate directly instead of RBA tool)
- InProgress Rules (Optional – applicable only when the tickets in progress status updates is managed by iAutomate directly instead of RBA tool)



Organization Details

Organization * : -Select-

Module * :

Service * :

Integration Type * :

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name * : Enter Datasource Name

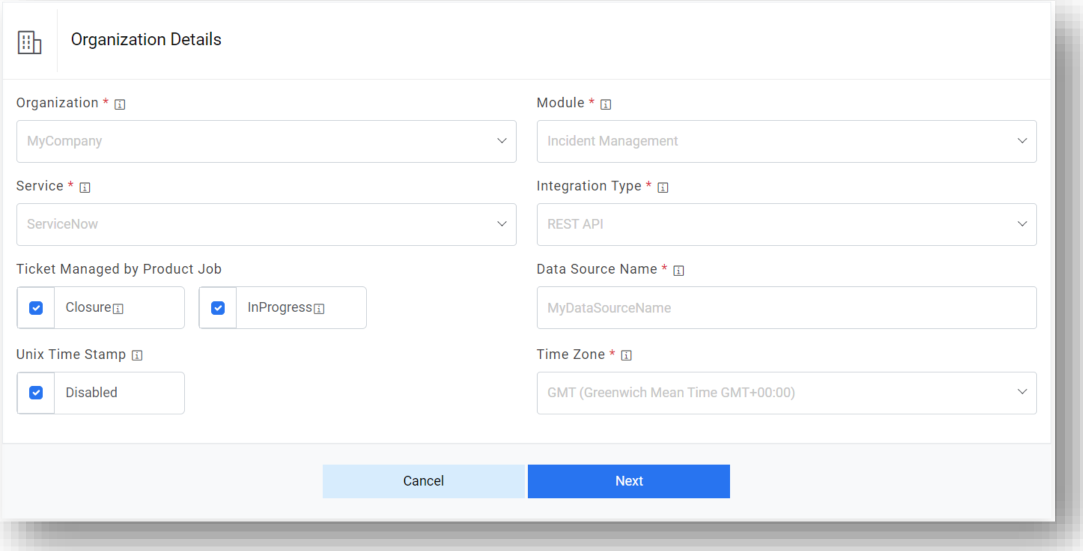
Unix Time Stamp: ☑ Disabled

Time Zone * :

Figure 2 – Create Data Source

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab,
   - Select the **Organization Name** from the dropdown.
   - Select the **Module** as **Incident Management,** since we are configuring this data source for pulling the incident tickets.
   - Select the **Service** as **Service Now Tool** as we are configuring the data source for ServiceNow
   - Select the **Integration Type** as **REST**, since we will be integrating through REST APIs.
   - Check **Is Ticket Closure Managed by iAutomate job** if you want iAutomate to manage the ticket closure updates instead of the RBA tool. In this scenario, an additional tab **Close Rules** will be activated for providing further details, steps for which are mentioned later.
   - Check "**Is ticket InProgress Managed by iAutomate job**" if you want iAutomate to manage the tickets in progress status updates instead of the RBA tool. In this scenario, an additional tab "**InProgress Rules"** will be activated for providing further details, steps for which are mentioned later.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Click **Next**.


Organization Details

Organization * MyCompany

Module * Incident Management

Service * ServiceNow

Integration Type * REST API

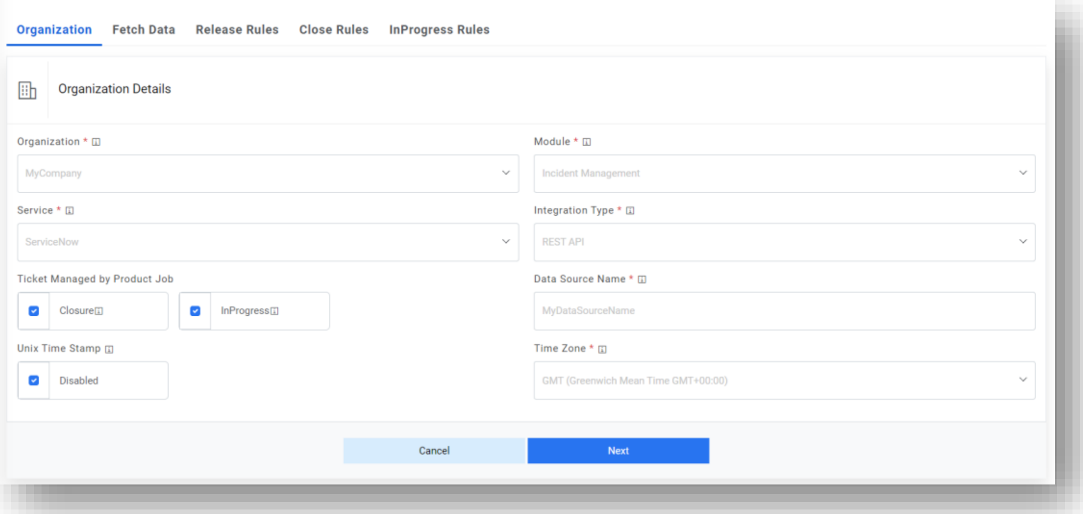
Ticket Managed by Product Job ☑ Closure ☑ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 3 – Create Data Source (Cont.)



Organization Details

Organization * MyCompany

Module * Incident Management

Service * ServiceNow

Integration Type * REST API

Ticket Managed by Product Job ☑ Closure ☑ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

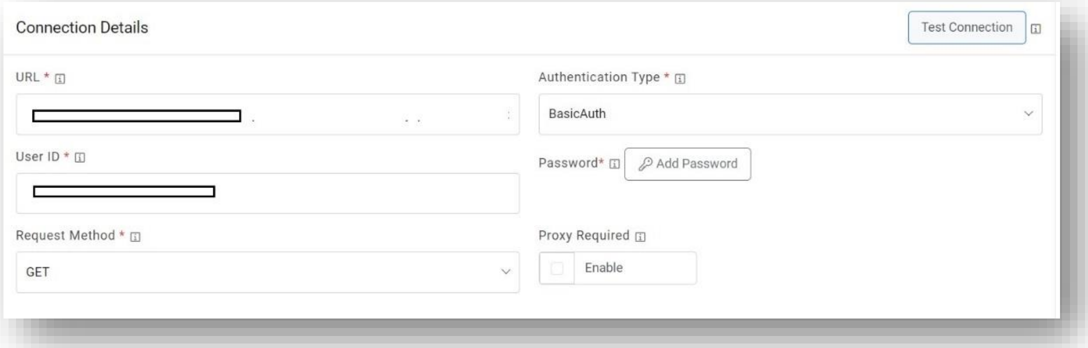
Figure 4 – Create Data Source (Cont.)

4. On the **Fetch Data** tab, type in the details as per the environment.
5. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- Sample URL – https://<url>?sysparm_fields=#Columns#&sysparm_query=sys_updated_on>=#StartDate#^sys_updated_on<=#EndDate#^ORDERBYsys_updated_on
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0.
- Selection of **Basic / Windows** requires you to enter –
  - User Id
  - Password.

- Selection of **OAuth 2** requires you to enter -
    - User Id
    - Password
    - Authentication URL
- **Request Method –** Select **GET, POST** or **PUT** as the Request method as per the URL configured.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

Test Connection

URL * [redacted]

Authentication Type * BasicAuth

User ID * [redacted]

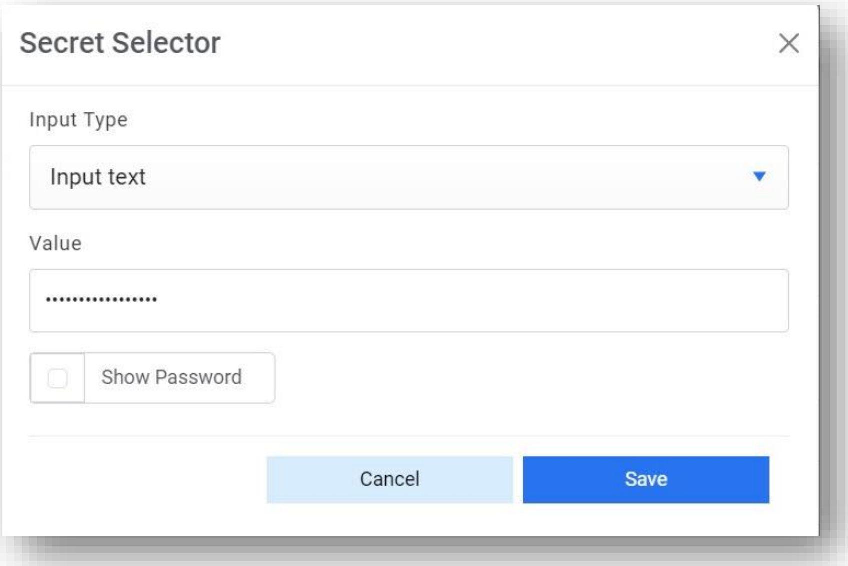
Password* Add Password

Request Method * GET

Proxy Required □ Enable

Figure 5 – Create Data Source (Connection Details)

- For **password**, click on the icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field.



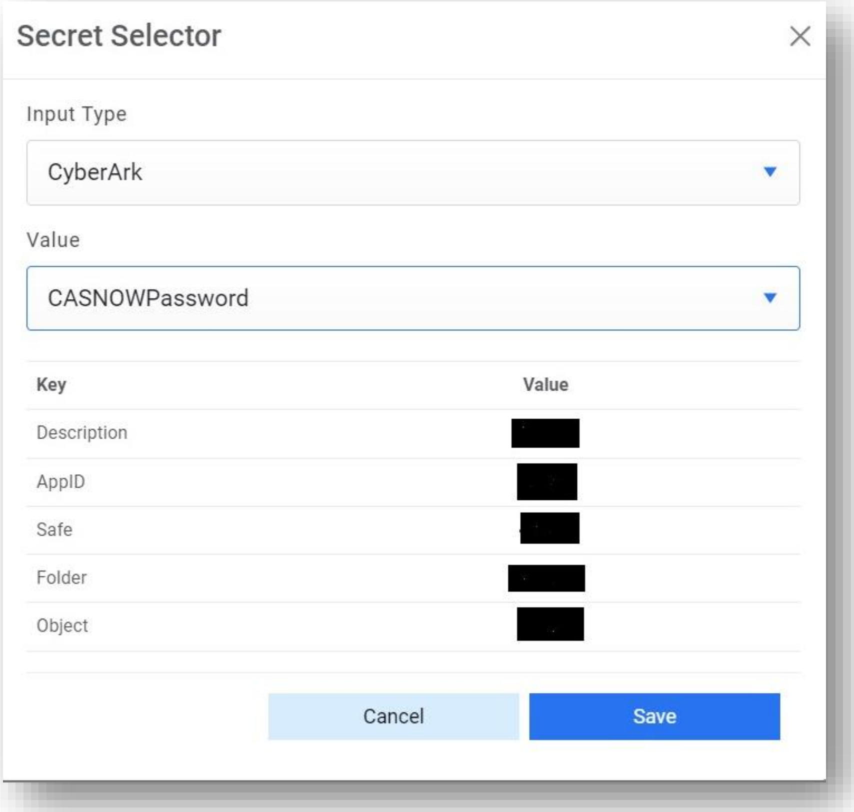
Secret Selector

Input Type: Input text

Value: ••••••••••••••••••

□ Show Password

Figure 6 - Password in Plaintext



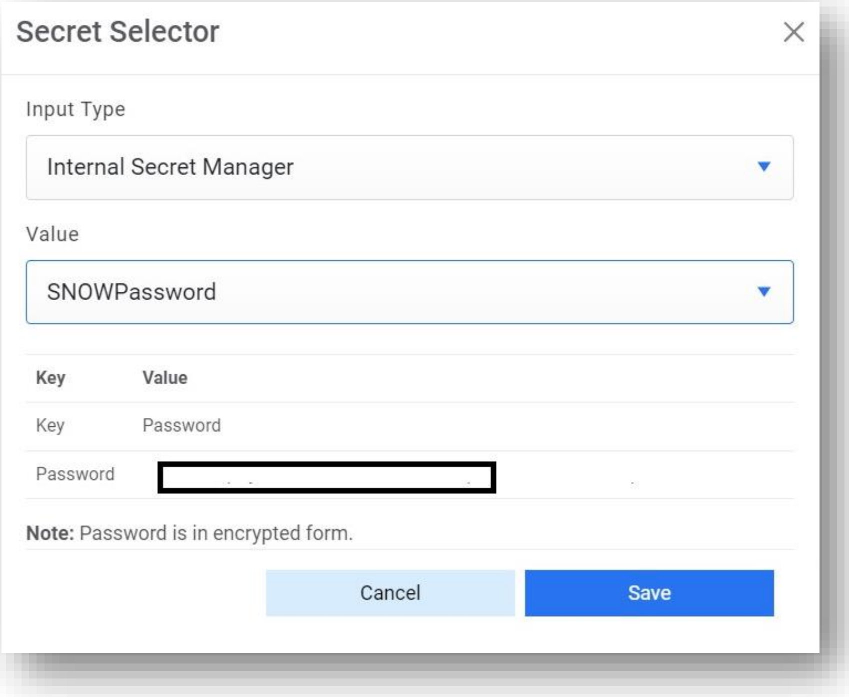
Secret Selector

Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 7 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

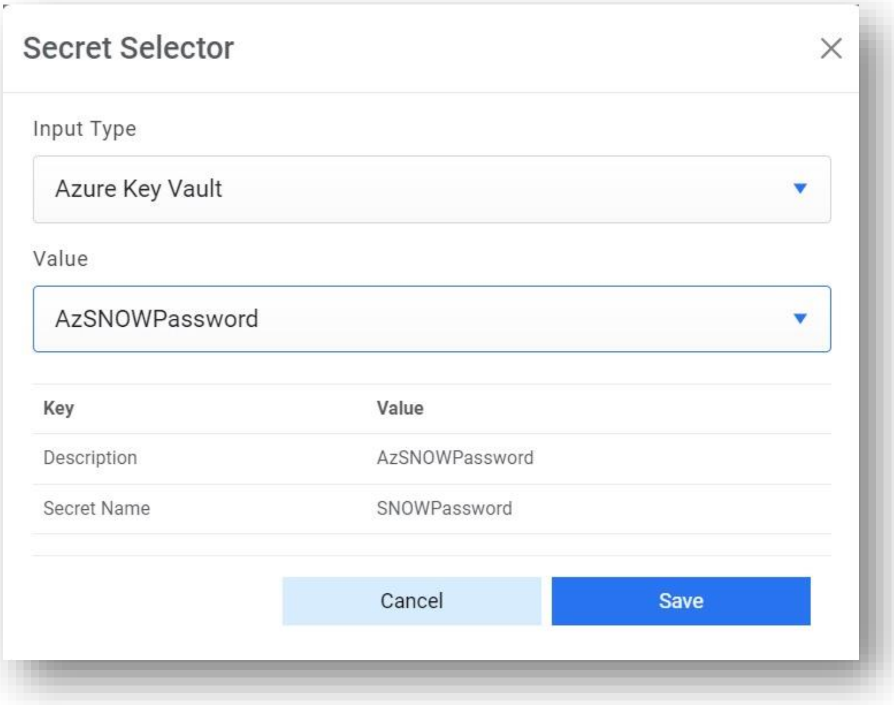
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 8 - Password from Key Vault (Secret Manager)



Secret Selector

Input Type

Azure Key Vault

Value

AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

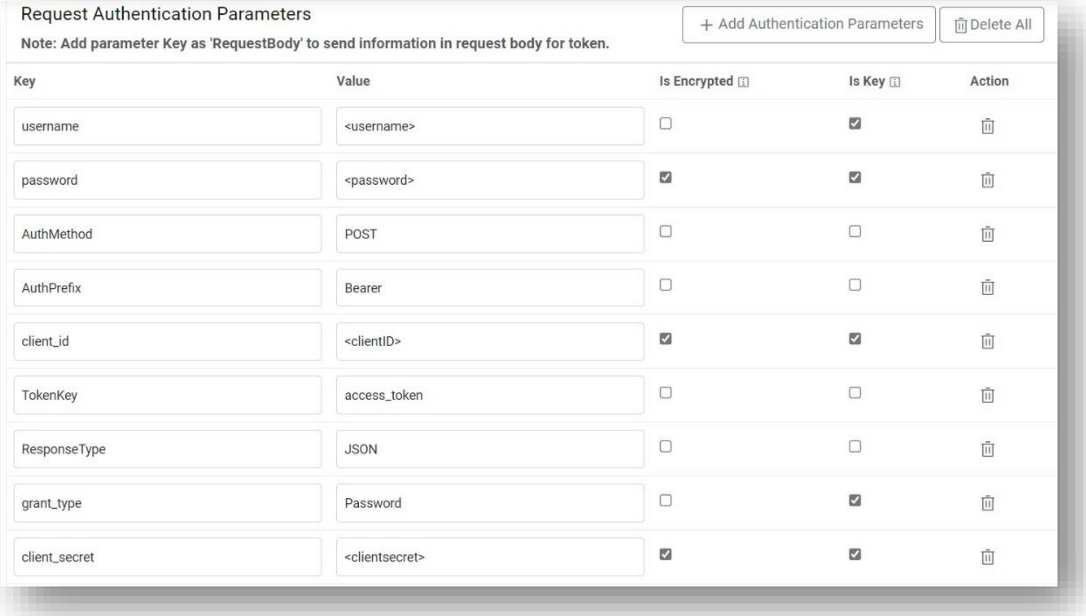
Figure 9 - Password from Azure Key Vault

- **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.

Based on the **Authentication Type**, add the parameters mentioned in the below table –

Table 2– Sample Authentication Parameters

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

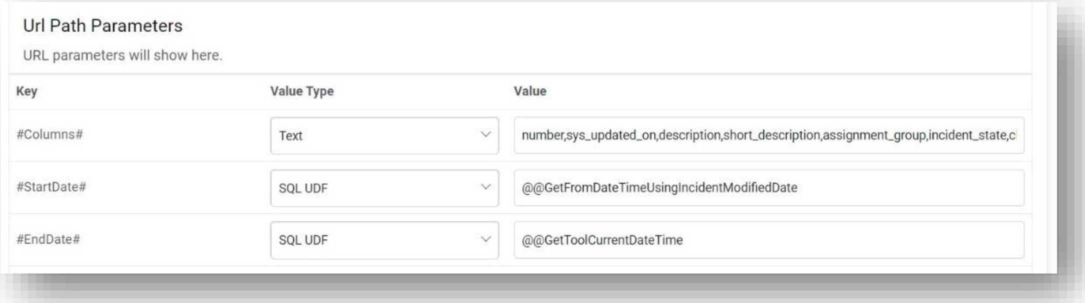
| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | □ | □ | 🗑 |
| AuthPrefix | Bearer | □ | □ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | □ | □ | 🗑 |
| ResponseType | JSON | □ | □ | 🗑 |
| grant_type | Password | □ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 10 – Create Data Source (Request Authentication Parameters for OAuth2.0)

- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Columns#

ValueType: Text

Value:

number,sys_updated_on,short_description,description,assignment_
group,incident_state,closed_at,category,dv_assigned_to,sys_id



Note – These columns are mandatory. User can add more columns if
more data is required to be fetched from ITSM tool.



Key: #StartDate#
```

```
ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingIncidentModifiedDate



Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```


Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,description,short_description,assignment_group,incident_state,c |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingIncidentModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 11 – URL Path Parameters

- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body –** Please enter the request body for the configured URL, if applicable.
- **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below -

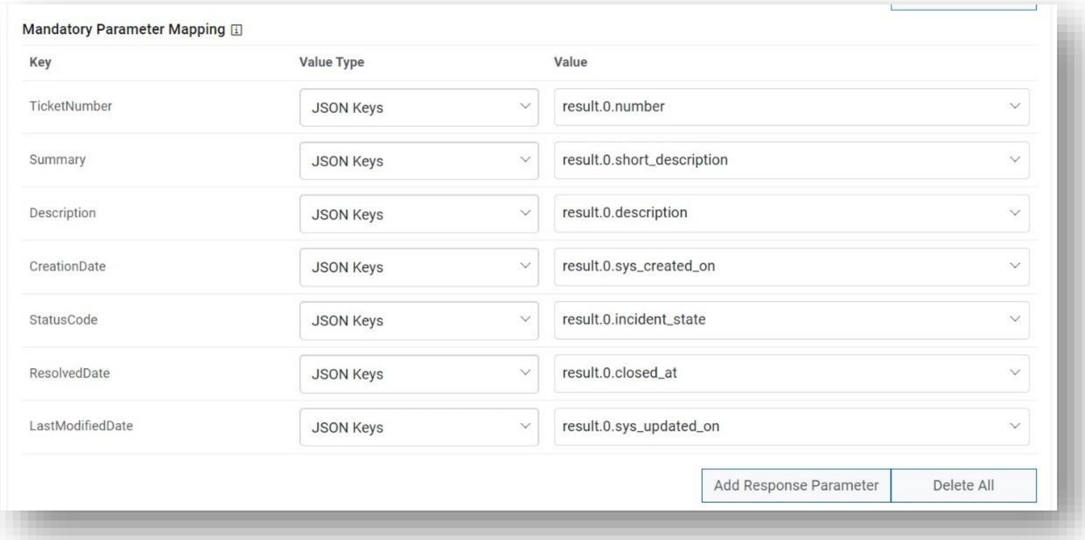
```
Response Body –

{   "result": [{    "number": "INC0079154",    "closed_at": "",
"assignment_group":  {        "link":  "<https://sample.service-
now.com/api/now/v1/table/sys_user_group/All     user     group>",
"value":  "All  user  group"      },      "incident_state":  "6",
"sys_created_on": "2017-12-22 06:59:03",  "description": "Memory
Utilization:10.0.0.11",       "short_description":        "Memory
Utilization:localhost",        "sys_updated_on":     "2018-01-02
06:39:56",    "category":  "",    "priority":  "4",    "sys_id":
"123456"  }] }
```

After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section**.**

- **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 3– Sample Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.number |
| Summary | JSON.Keys | result.0.short_description |
| Description | JSON.Keys | result.0.description |
| CreationDate | JSON.Keys | result.0.sys_created_on |
| StatusCode | JSON.Keys | result.0.incident_state |
| ResolvedDate | JSON.Keys | result.0.closed_at |
| LastModifiedDate | JSON.Keys | result.0.sys_updated_on |



Mandatory Parameter Mapping

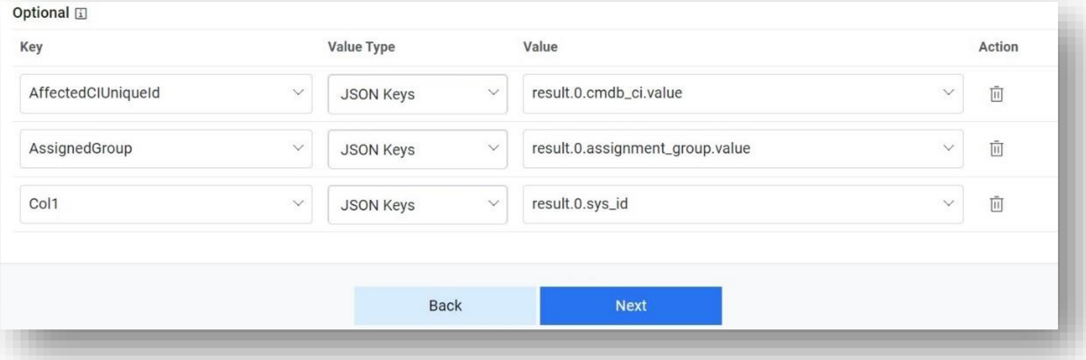
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.number |
| Summary | JSON Keys | result.0.short_description |
| Description | JSON Keys | result.0.description |
| CreationDate | JSON Keys | result.0.sys_created_on |
| StatusCode | JSON Keys | result.0.incident_state |
| ResolvedDate | JSON Keys | result.0.closed_at |
| LastModifiedDate | JSON Keys | result.0.sys_updated_on |

Figure 12 – Mandatory Parameter Mapping

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 4 – Sample Optional Parameters

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | result.0. assignment_group.value |
| Col1 | JSON.Keys | result.0.sys_id |



Optional

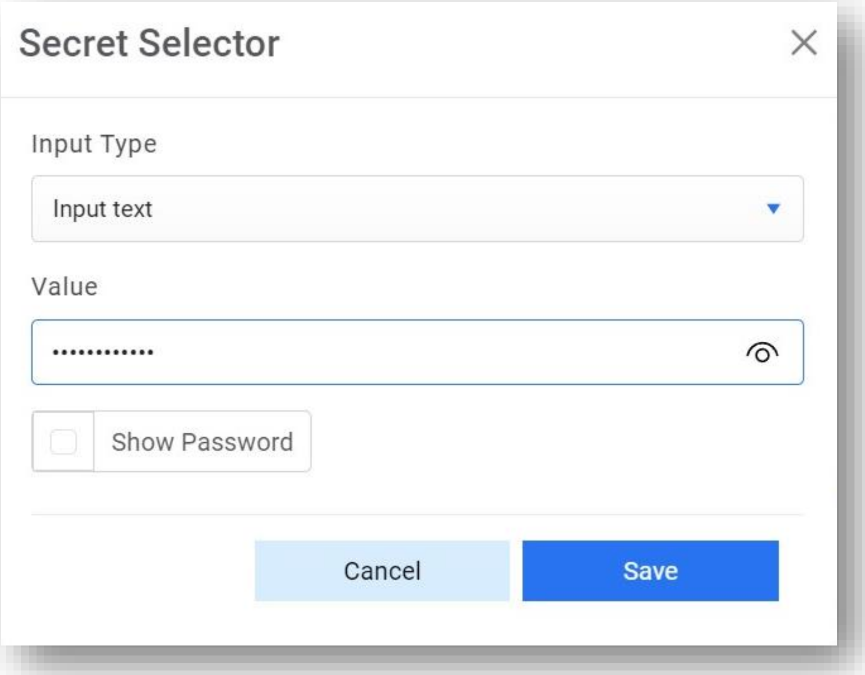
| Key | Value Type | Value | Action |
|---|---|---|---|
| AffectedCIUniqueId | JSON Keys | result.0.cmdb_ci.value | 🗑 |
| AssignedGroup | JSON Keys | result.0.assignment_group.value | 🗑 |
| Col1 | JSON Keys | result.0.sys_id | 🗑 |

Figure 13 – Optional Parameter Mapping

6. Click Next to proceed to Release Rules Configuration.

7. On **the Release Rules** tab, type in the details as per the requirement.
8. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- Sample URL - https://<URL>.service-now.com/api/now/table/incident/#incident#
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- **User Id**: Enter the user id for the configured ITSM.
- **Password**: For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.

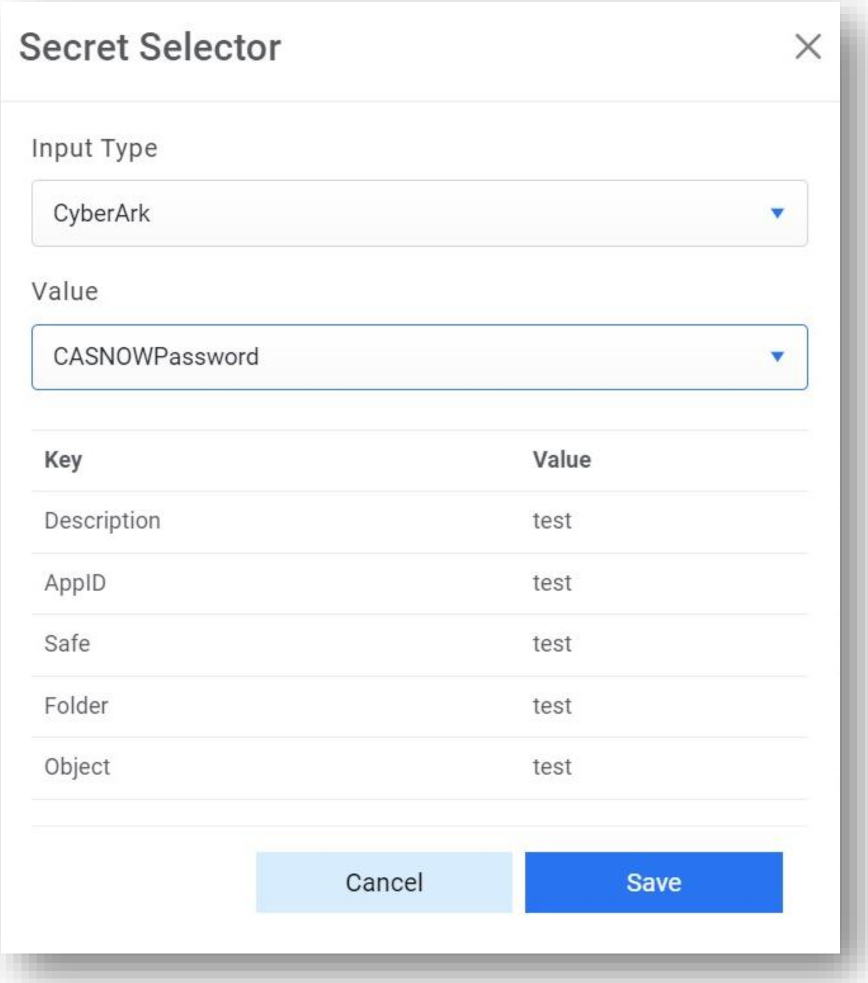


Secret Selector

Input Type: Input text

Value: ••••••••••••

Figure 14 - Password in Plaintext


Secret Selector

Input Type

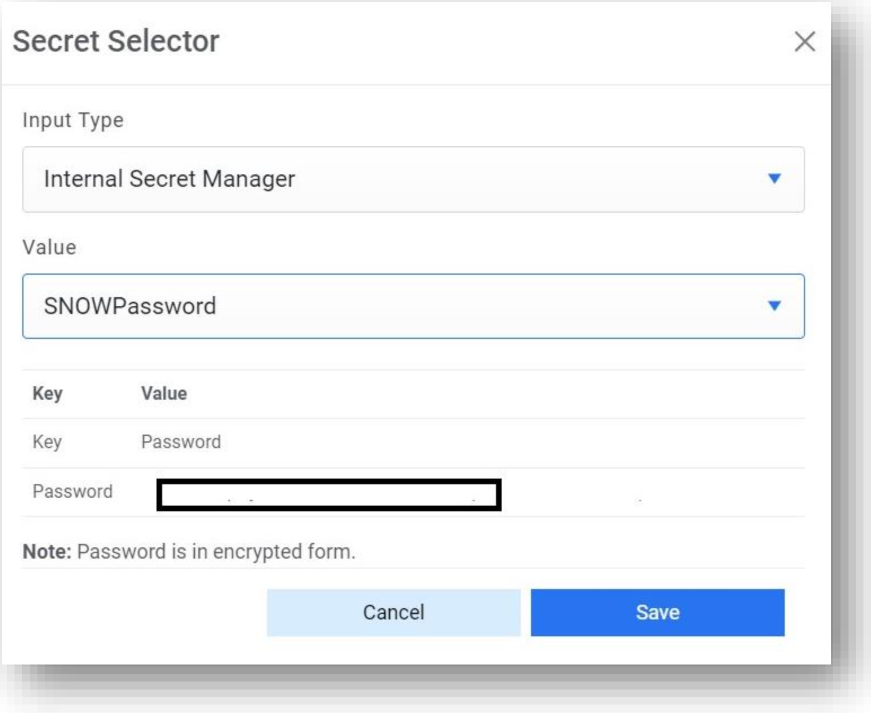
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | test |
| AppID | test |
| Safe | test |
| Folder | test |
| Object | test |

Figure 15 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

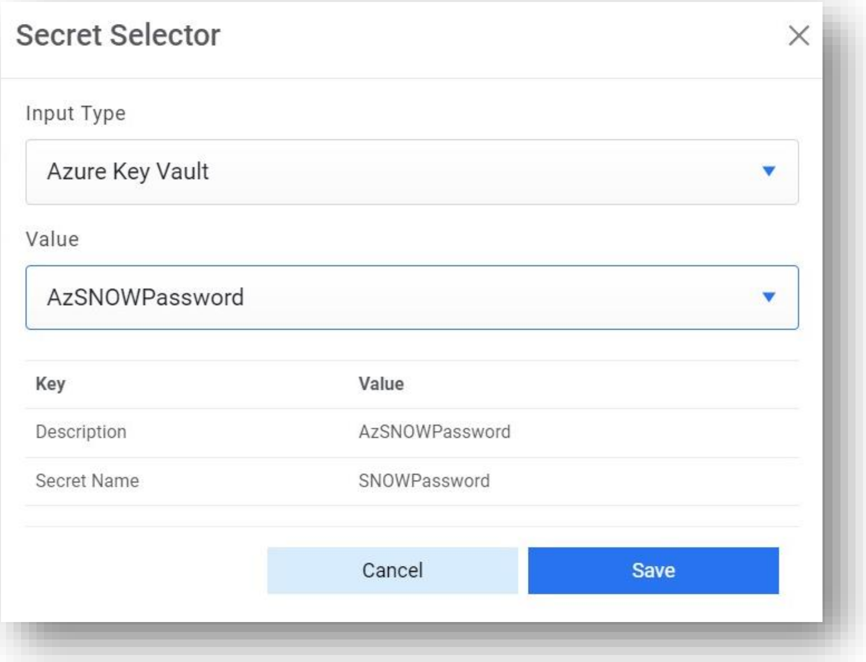
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 16 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

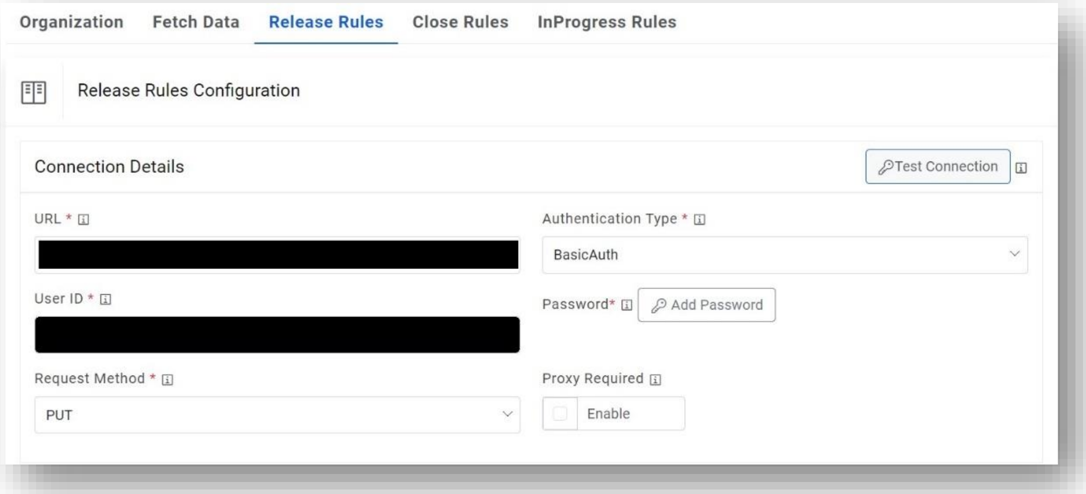
AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 17 - Password from Azure Key Vault

- **Request Method** – Select Request Method as PUT from the drop-down.
- **Proxy Required –** Check **Proxy Required,** if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Release Rules Configuration

Connection Details

URL * [redacted]

Authentication Type * BasicAuth

User ID * [redacted]

Password*

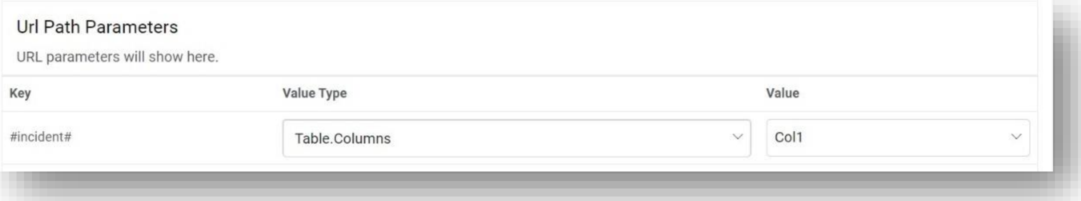
Request Method * PUT

Proxy Required □ Enable

Figure 18 – Release Rules (Connection Details)

- **URL Path Parameters** – Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #incident#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
“Col2”
```



Url Path Parameters

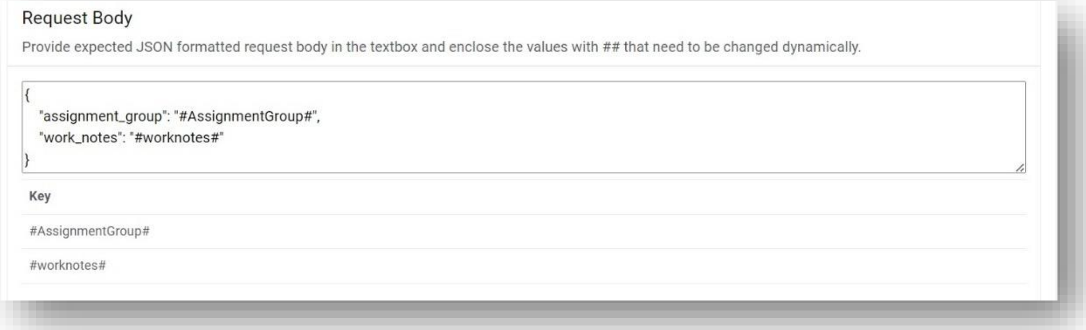
URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #incident# | Table.Columns | Col1 |

Figure 19 – Release Rules (URL Path Parameters)

- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –

{ "assignment_group" : "#AssignmentGroup#","work_notes" :
"#work_notes#" }
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

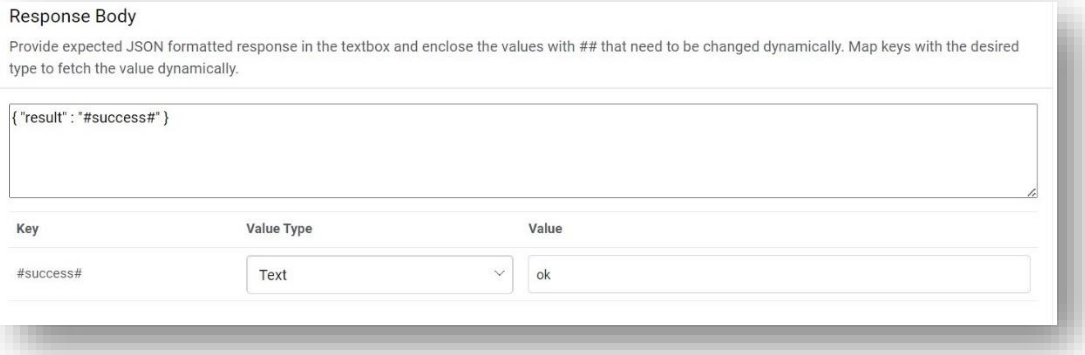
```
{
   "assignment_group": "#AssignmentGroup#",
   "work_notes": "#worknotes#"
}
```

| Key |
|---|
| #AssignmentGroup# |
| #worknotes# |

Figure 20 – Release Rules (Request Body)

- **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

Figure 21 – Release Rules (Response Body)

- **Response Key Value** mapping can be done as per the below table:

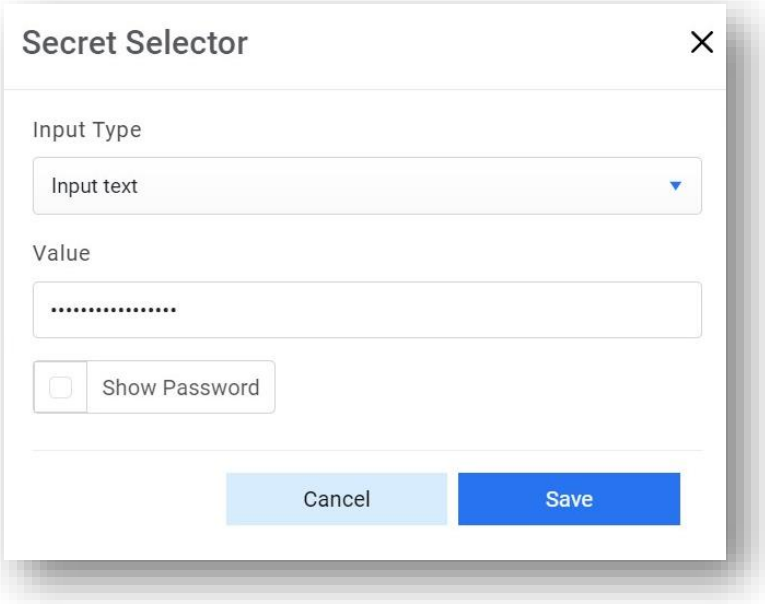
Table 5 – Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

9. On **Close Rules** tab, type in the details as per the requirement. Check **Same as Release** if similar configurations as mentioned in "Release Rules Configuration" are required, else proceed ahead.
10. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>.service-now.com/api/now/table/incident/#incident#
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- **User Id**: Enter user id for the configured ITSM tool.

- **Password**: For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



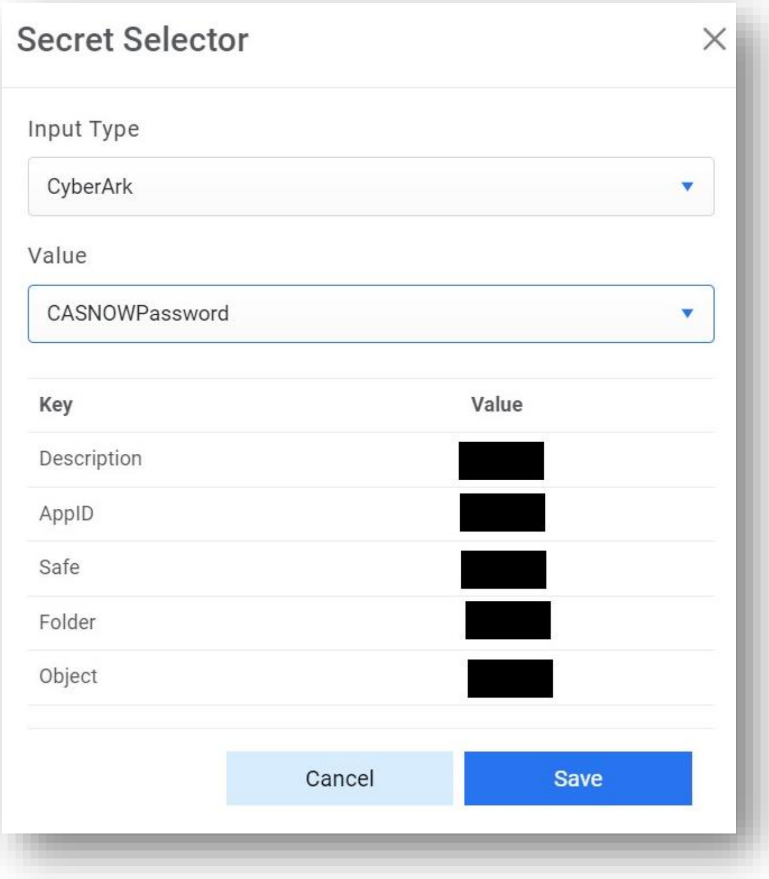
Secret Selector

Input Type: Input text

Value: ••••••••••••••••••

□ Show Password

Figure 22 - Password in Plaintext



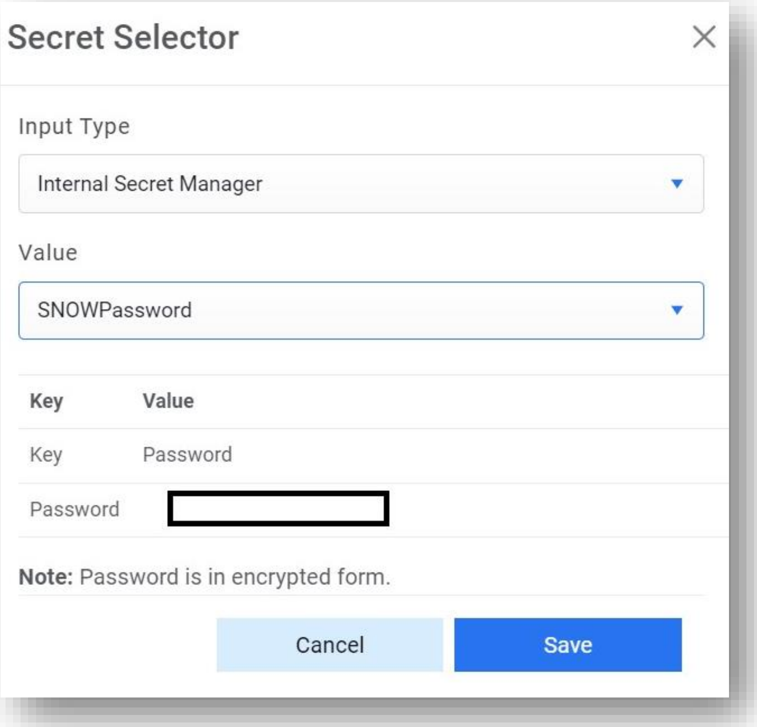
Secret Selector

Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 23 - Password from Key Vault (CyberArk)



Secret Selector

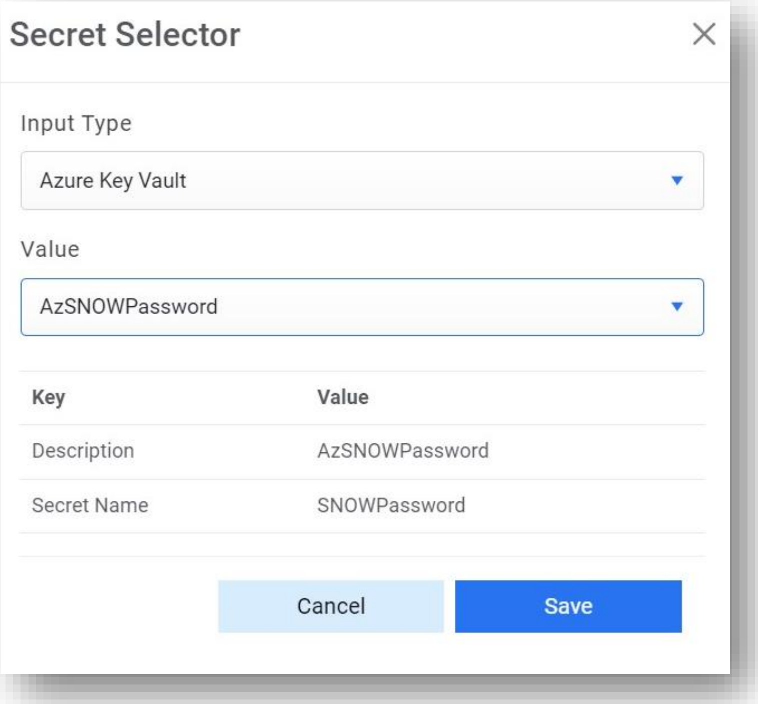
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 24 - Password from Secret Manager



Secret Selector

Input Type: Azure Key Vault

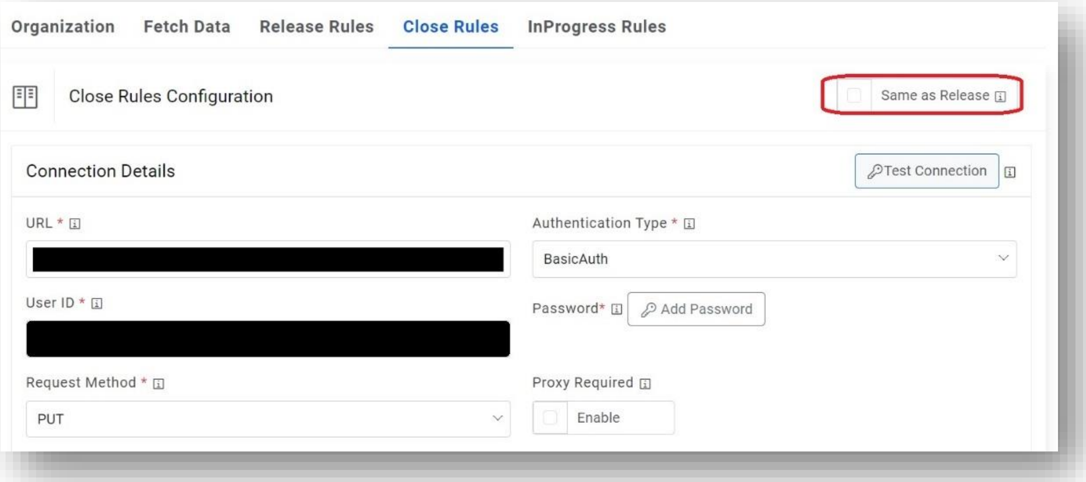
Value: AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 25 - Password from Azure Key Vault

- **Request Method** – Select Request Method as PUT from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Close Rules Configuration

□ Same as Release

Connection Details

Test Connection

URL * [redacted]

Authentication Type * BasicAuth

User ID * [redacted]

Password* Add Password

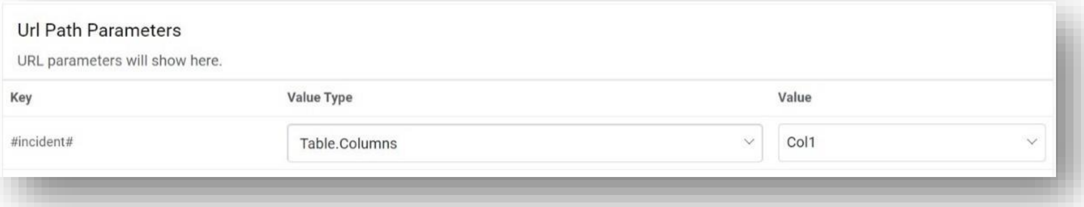
Request Method * PUT

Proxy Required □ Enable

Figure 26 – Close Rules (Connection Details)

– **URL Path Parameters** – Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #incident#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
"Col2"
```



Url Path Parameters

URL parameters will show here.

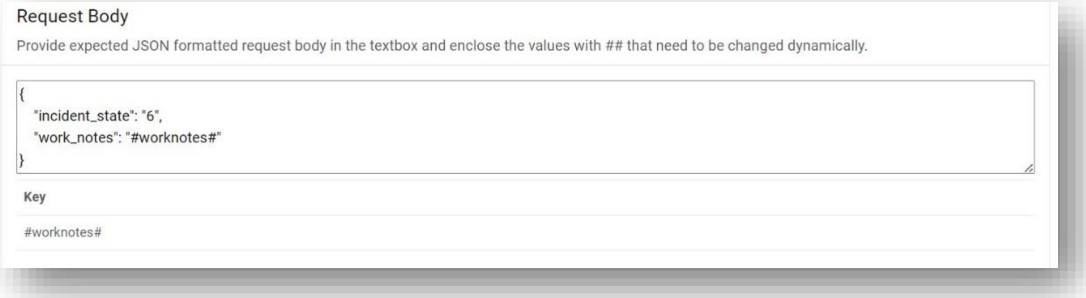
| Key | Value Type | Value |
|---|---|---|
| #incident# | Table.Columns | Col1 |

Figure 27 – Close Rules (URL Path Parameters)

– **Request Header Parameters –** Please enter the request header parameters as required.

– **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –

{ "incident_state" : "6"} If you also want to add worknotes while
Close ticket, use json {"incident_state":"6", "work_notes":
"#Notes#"}
```


Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

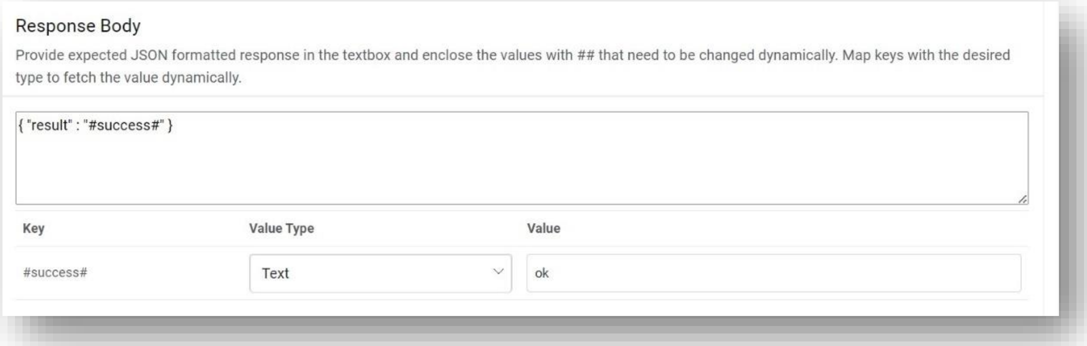
```
{
  "incident_state": "6",
  "work_notes": "#worknotes#"
}
```

| Key |
|---|
| #worknotes# |

Figure 28 – Close Rules (Request Body)

- **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```


Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

Figure 29 – Close Rules (Response Body)

- **Response Key Value** mapping can be done as per the below table:

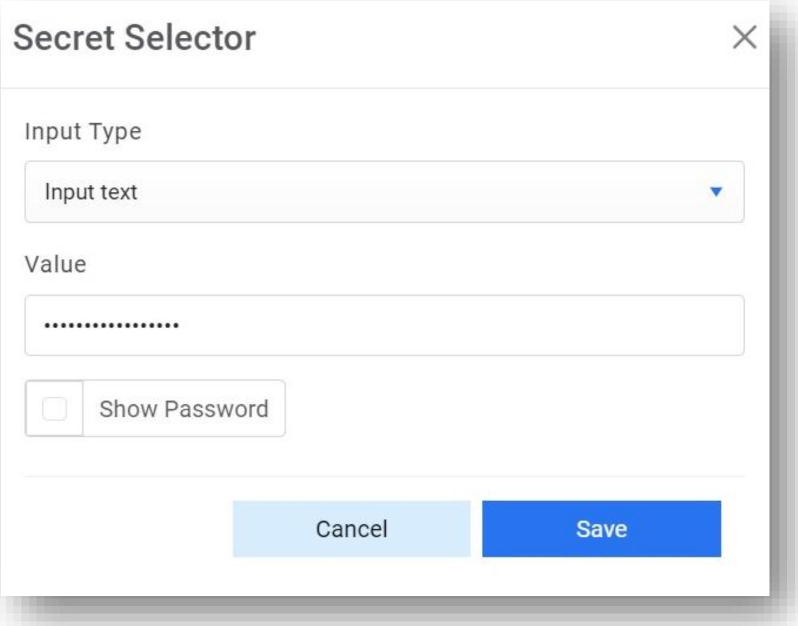
Table 6 – Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

11. On **InProgress Rules** tab, type in the details as per the requirement. Check **Same as Release** if similar configurations as mentioned in "Release Rules Configuration" are required, else proceed ahead.
12. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>.service-now.com/api/now/table/incident/#incident#
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- **User Id -** Enter the user id for the configured ITSM tool.

– **Password** – For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



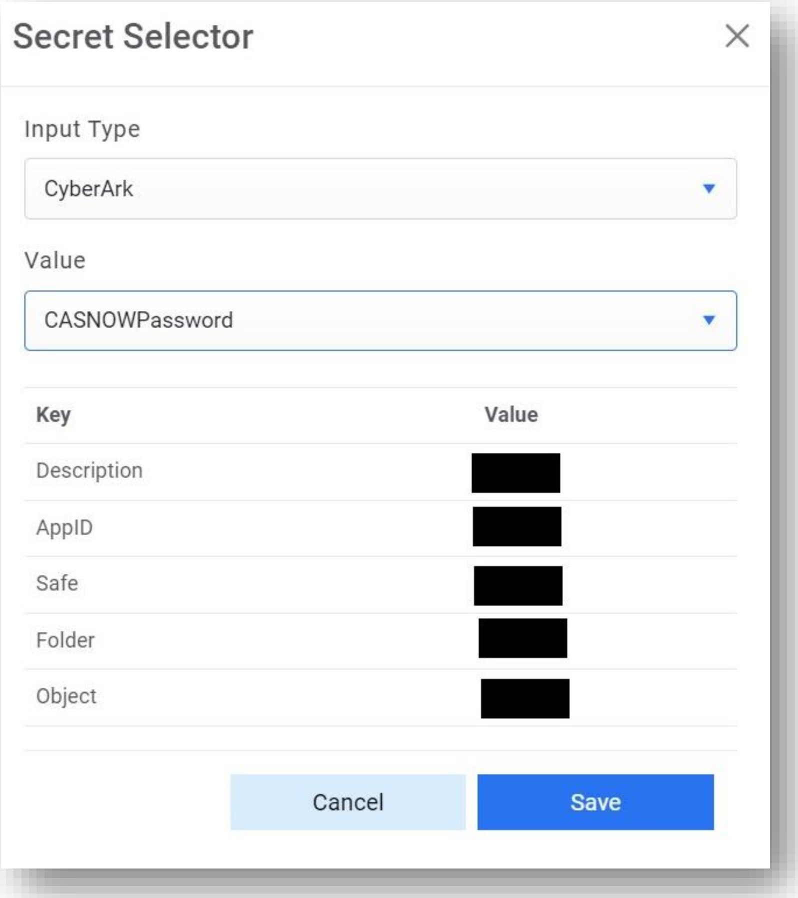
Secret Selector

Input Type: Input text

Value: •••••••••••••••••

□ Show Password

Figure 30 - Password in Plaintext



Input Type

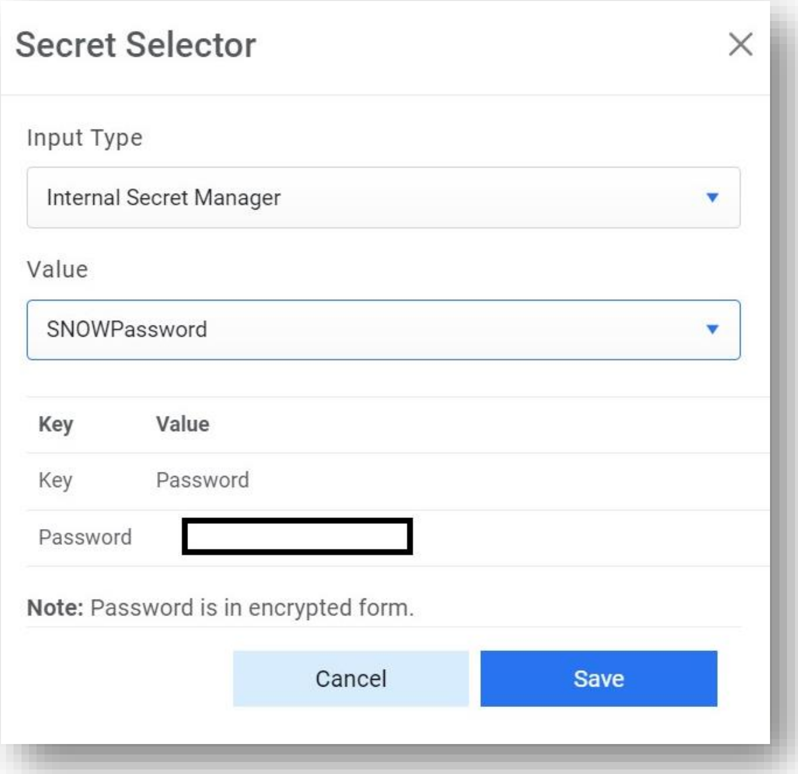
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 31 - Password from Key Vault (CyberArk)



Input Type

Internal Secret Manager

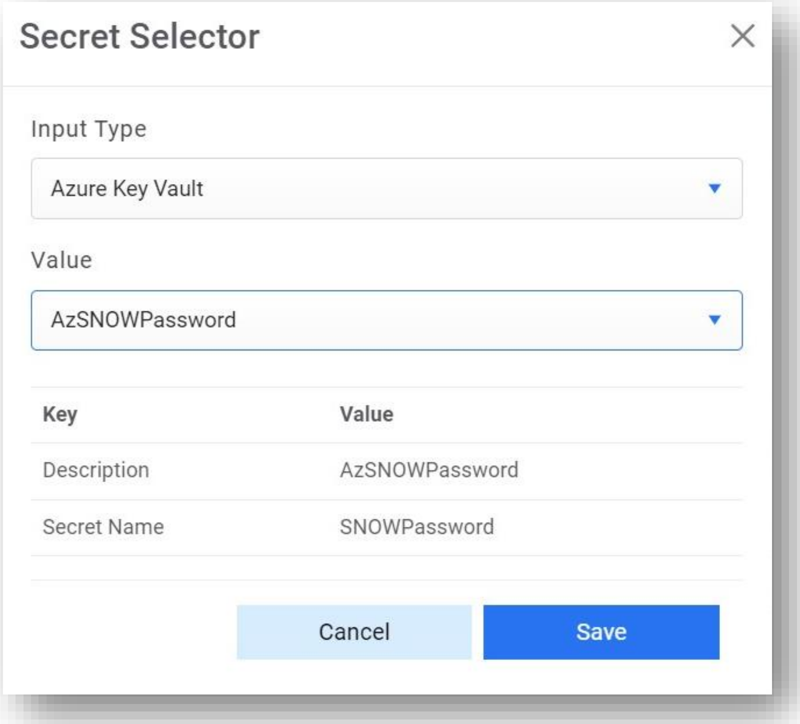
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 32 - Password from Secret Manager



Secret Selector

Input Type: Azure Key Vault

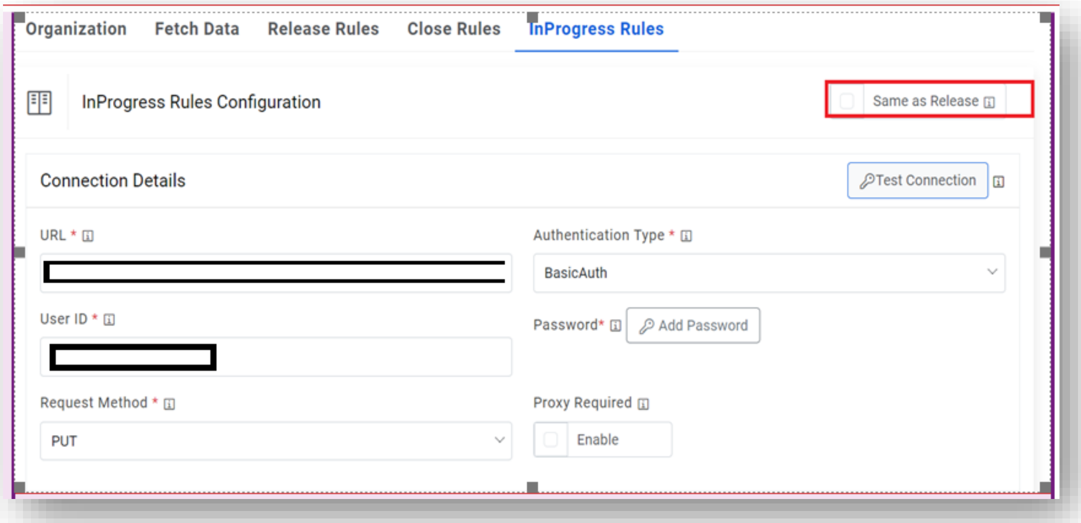
Value: AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 33 – Password from Azure Key Vault

- **Request Method** – Select Request Method as PUT from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



InProgress Rules Configuration

□ Same as Release

Connection Details

URL *: [redacted]

Authentication Type *: BasicAuth

User ID *: [redacted]

Password*: Add Password

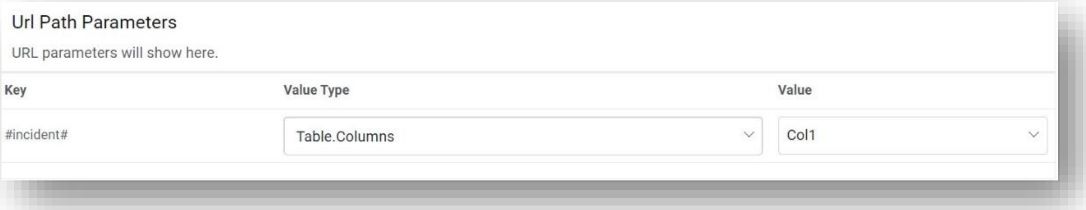
Request Method *: PUT

Proxy Required: □ Enable

Figure 34 – InProgress Rules (Connection Details)

- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #incident#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
"Col2"
```



Url Path Parameters

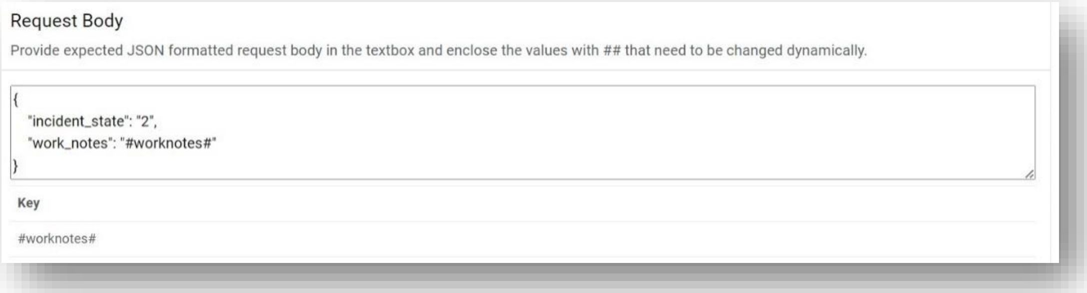
URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #incident# | Table.Columns | Col1 |

Figure 35 – InProgress Rules (URL Path Parameters)

- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –

{"incident_state" : "2"} If you also want to add worknotes while
inprogress ticket, use json {"incident_state":"2", "work_notes":
"#Notes#"}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

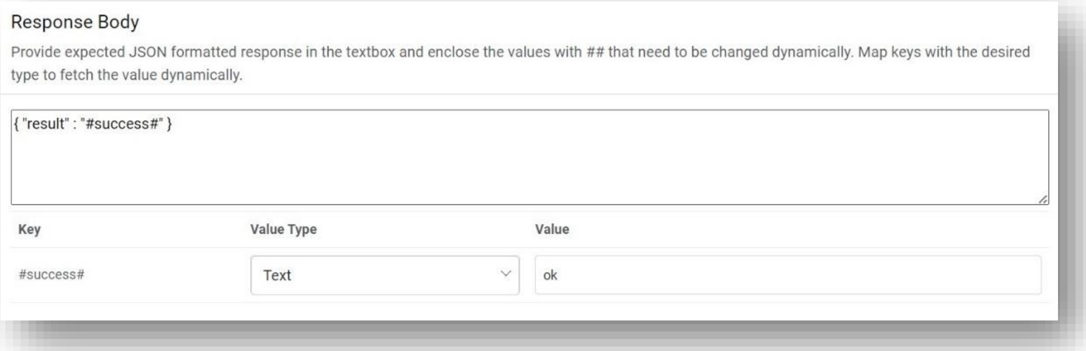
```
{
  "incident_state": "2",
  "work_notes": "#worknotes#"
}
```

| Key |
|---|
| #worknotes# |

Figure 36 – InProgress Rules (Request Body)

- **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

Figure 37 – InProgress Rules (Response Body)

– **Response Key Value** mapping can be done as per the below table:

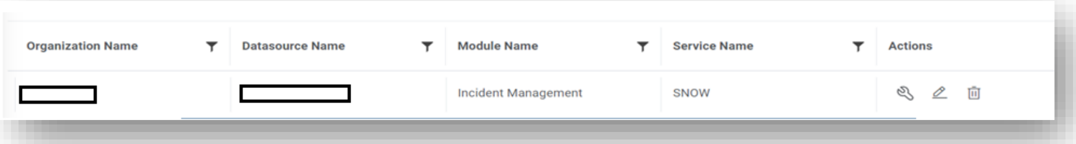
Table 7 – Sample Response Key Value Mapping

| #success# | Text | OK |
|---|---|---|

13. Click Save to add the data source.

To bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and the same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps –

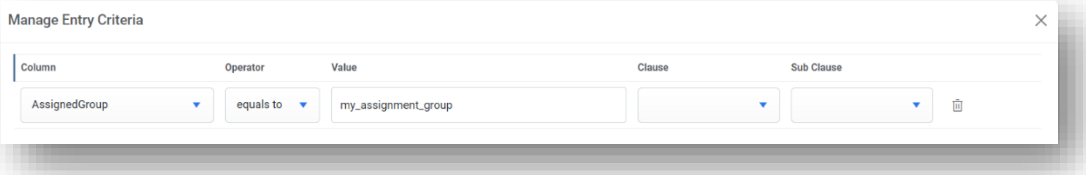
1. Go to Configuration and click Manage Data Sources.
2. On the **Data Sources** tab, click 🔧 next to the data source that user wants to manage. **Manage Entry Criteria** screen appears.



| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| [redacted] | [redacted] | Incident Management | SNOW | |

Figure 38 – Manage Entry Criteria

3. Select 'AssignedGroup' for the **Column** field and 'equals to' for the **Operator** field.
4. Enter the sys_id of the assignment group in ServiceNow in the **Value** field.
5. **Clause** and **Sub-Clause** fields can also be added based on requirement.



Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equals to | my_assignment_group | | |

Figure 39 – Manage Entry Criteria (cont.)

6. Click **Save**.

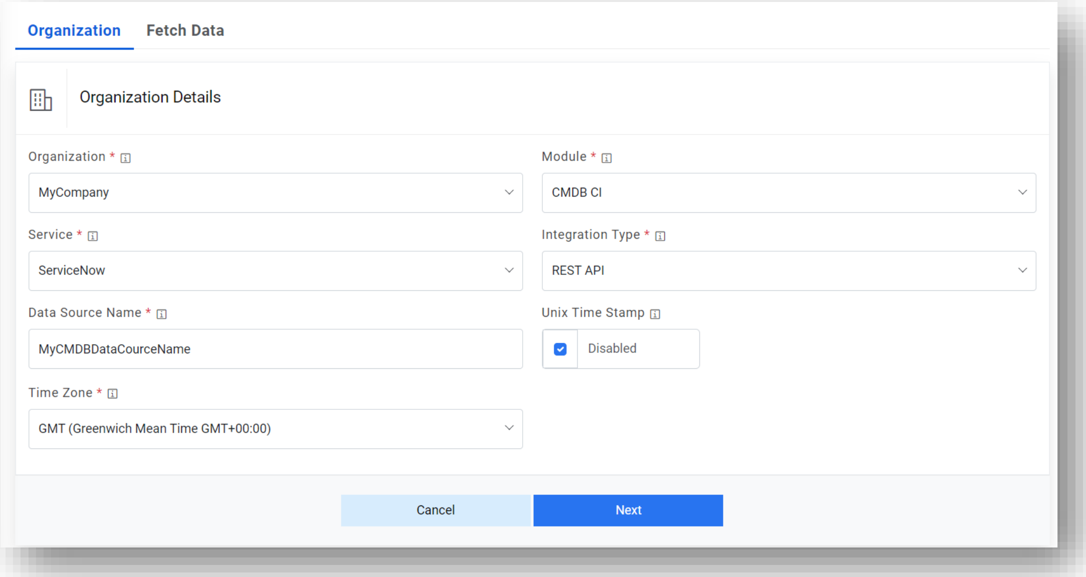
#### 4.2.1.2 Create Data Source for CMDB CI

To use the field values of CMDB CI for the purpose of Recommendation and Parsing by iAutomate services, two data sources need to be created.

To create a data source for CMDB CI, please refer to Create Data Source for Incident Management.

To create a data source for CMDB CI, perform the following steps:

1. On the left menu bar, click **Configuration** -> **Manage Data Sources.**
2. The **Create Data Source** page appears with the following tabs:
   - Organization
   - Fetch Data



Organization * : MyCompany

Module * : CMDB CI

Service * : ServiceNow

Integration Type * : REST API

Data Source Name * : MyCMDBDataCourceName

Unix Time Stamp : ☑ Disabled

Time Zone * : GMT (Greenwich Mean Time GMT+00:00)

Figure 40 – Create Data Source – CMDB CI

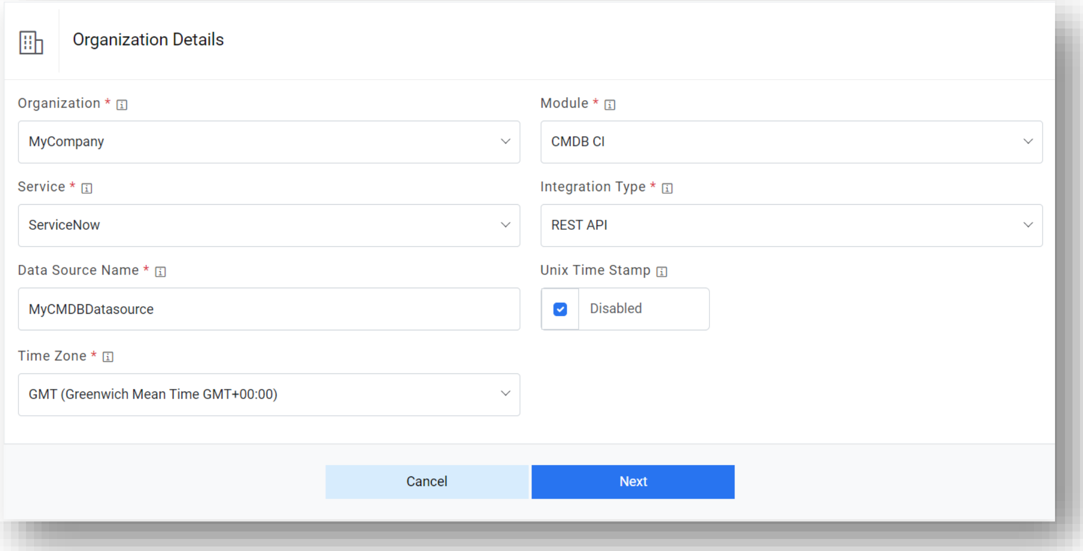
Release Rules is only applicable for the following **Module** types:

- Incident Management,

- Change Request Task and

**- Service Request Task.** (This tab will not be activated for other module types.)

3. On the **Organization** tab-
   - Select the **Organization Name** from the dropdown.
   - Select the **Module** as **CMDB CI,** since we are configuring this data source for using its field value for the incidents.
   - Select the **Service** as **Service Now Tool** as we are configuring the data source for ServiceNow
   - Select the **Integration Type** as **REST**, since we will be integrating through REST APIs.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select **Timestamp** to view the present data with date and time.
   - Click **Next**.


Organization Details

Organization * MyCompany

Module * CMDB CI

Service * ServiceNow

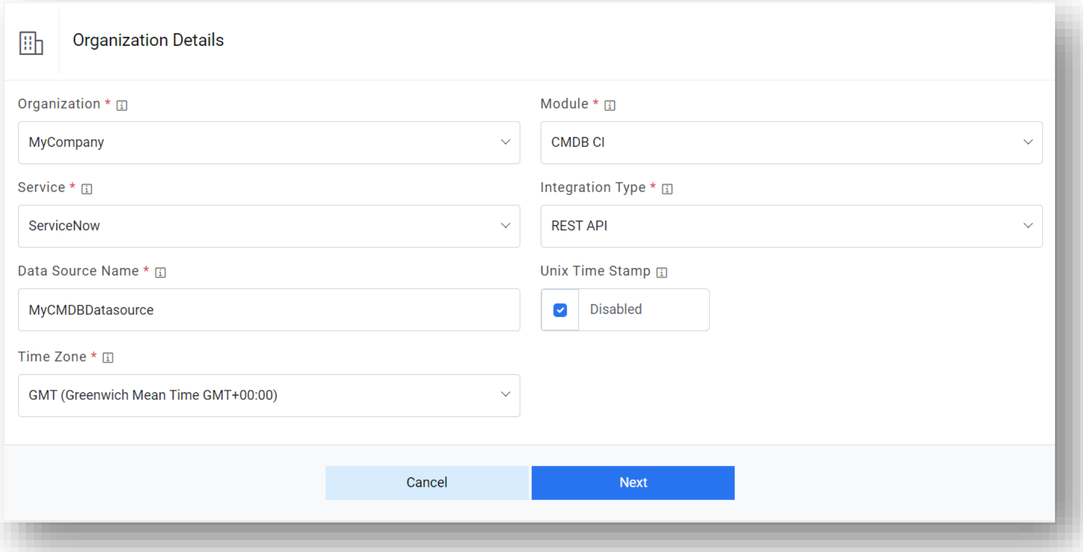
Integration Type * REST API

Data Source Name * MyCMDBDatasource

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 41 – Create Data Source – CMDB CI (Cont.)


Organization Details

Organization * MyCompany

Module * CMDB CI

Service * ServiceNow

Integration Type * REST API

Data Source Name * MyCMDBDatasource

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 42 – Create Data Source – CMDB CI (Cont.)

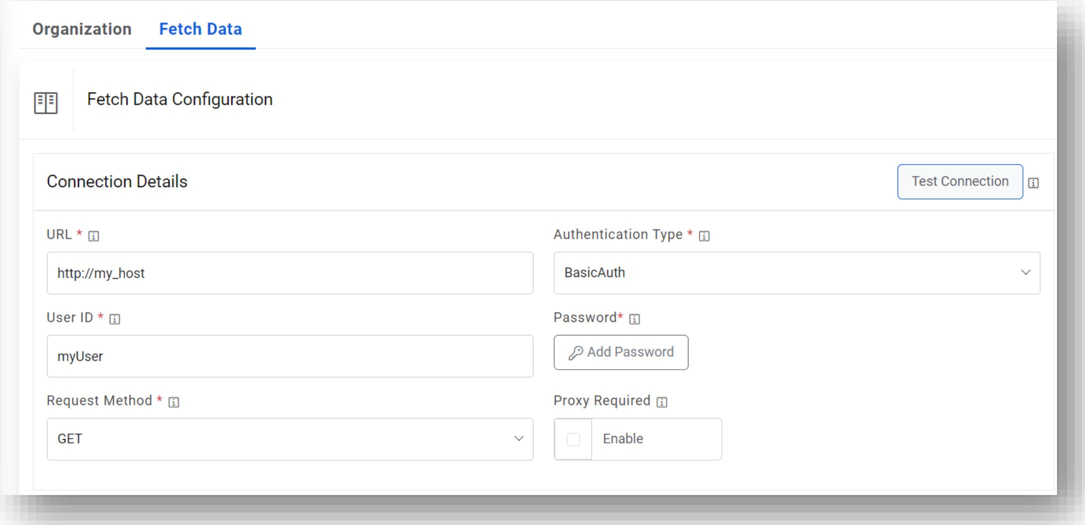
4. On the **Fetch Data** tab, type in the details as per the environment.
5. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL -** https://<url>?sysparm_fields=#Columns#&sysparm_query=sys_updated_on>=#StartDate#^sys_updated_on<=#EndDate#^ORDERBYsys_updated_on
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0
- Selection of **Basic / Windows** requires you to enter -
  - User Id

    - Password

- Selection of **OAuth 2.0** requires you to enter -
    - User Id
    - Password
    - Authentication URL
- **Request Method –** Select GET, POST or PUT as Request Method as per the configured URL.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



URL *: http://my_host

Authentication Type *: BasicAuth

User ID *: myUser

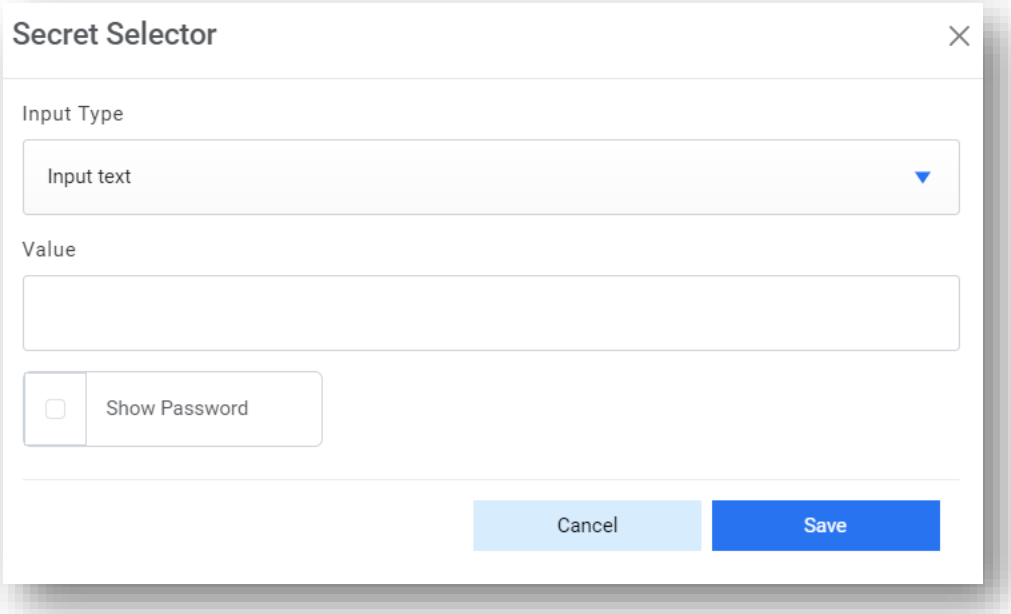
Password*: Add Password

Request Method *: GET

Proxy Required: □ Enable

Figure 43 – Create Data Source – CMDB CI (Connection Details)

- **Password** – For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



Secret Selector

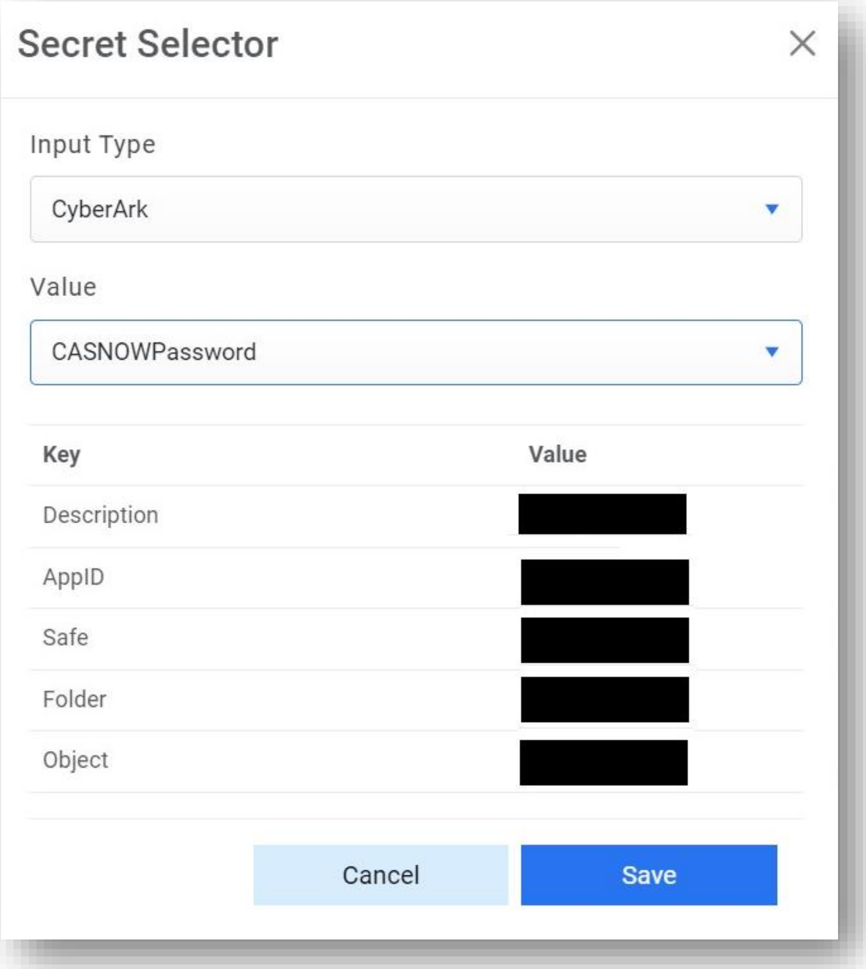
Input Type

Input text

Value

□ Show Password

Figure 44 - Password in Plaintext



Secret Selector

Input Type

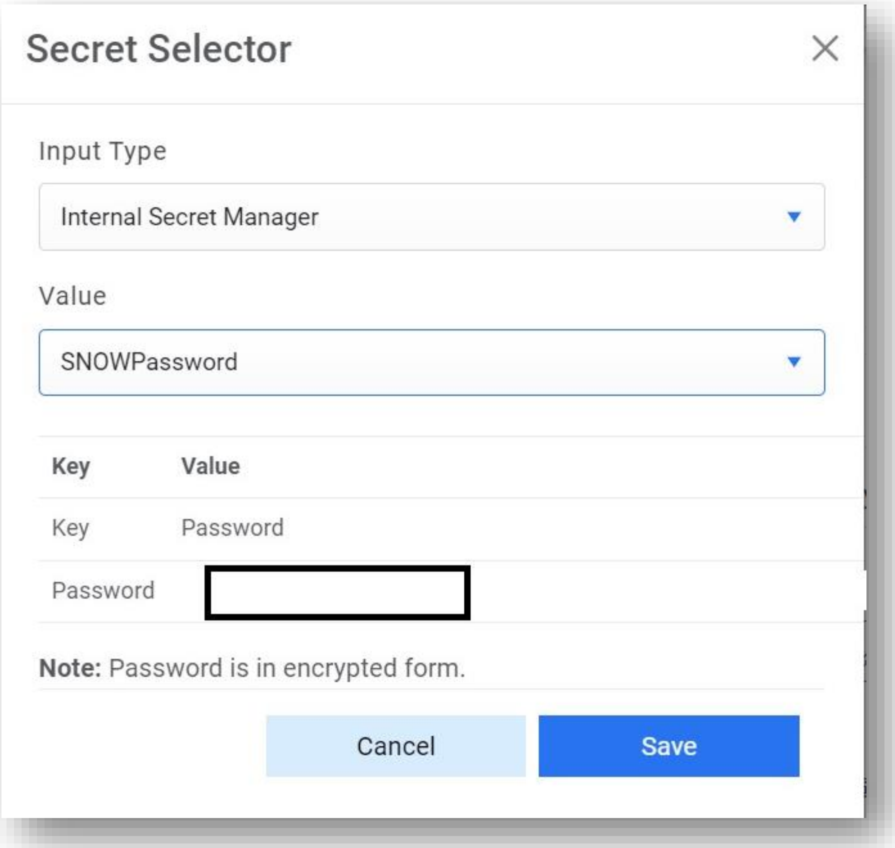
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 45 - Password from Key Vault (CyberArk)


Secret Selector

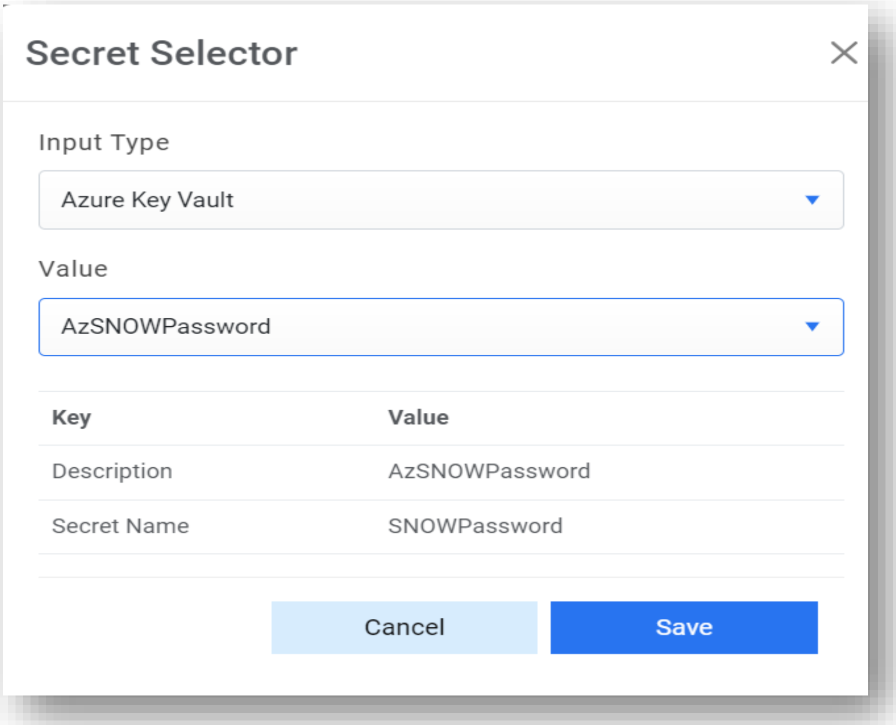
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 46 - Password from Secret Manager


Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

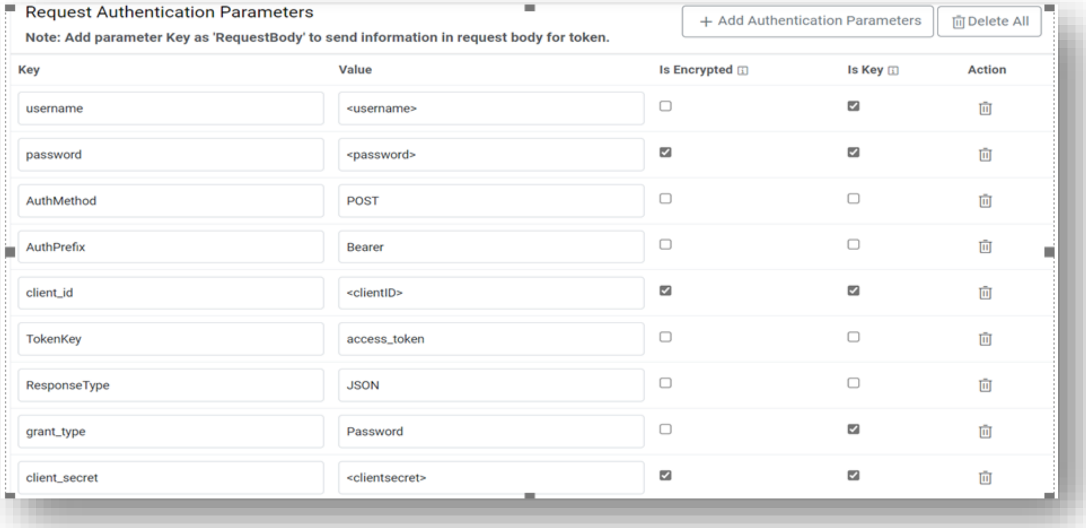
Figure 47 - Password from Azure Key Vault

- **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.

- Based on the **Authentication Type**, add the parameters mentioned in the below table:

Table 8 – Sample Authentication Parameters – CMDB CI

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

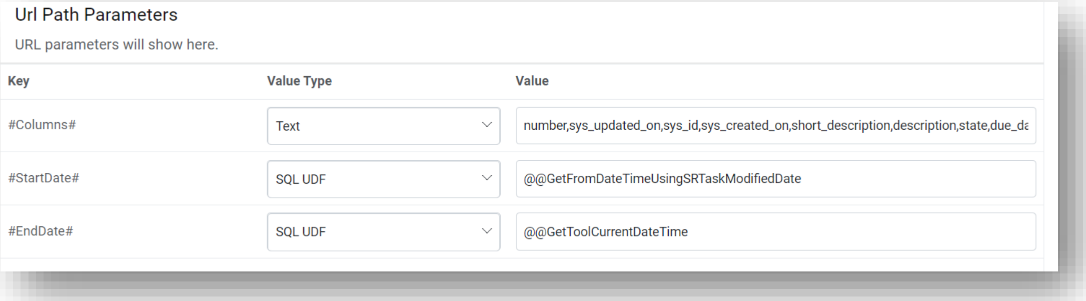
| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | □ | □ | 🗑 |
| AuthPrefix | Bearer | □ | □ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | □ | □ | 🗑 |
| ResponseType | JSON | □ | □ | 🗑 |
| grant_type | Password | □ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 48 – Create Data Source -CMDB CI (Request Authentication Parameters for OAuth2.0)

- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Columns#

ValueType: Text

Value:

sys_id,name,category,sys_updated_on,subcategory

Note - These columns are mandatory. User can add more columns if
more data is required to be fetched from ITSM tool.


Key: #StartDate#

ValueType: SQL UDF
```

```
VALUE: @@GetFromDateTimeUsingiCMDBModifiedDate


Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,sys_id,sys_created_on,short_description,description,state,due_da |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingSRTaskModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 49– URL Path Parameters – CMDB CI

- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body -** Please enter the request body as per the configured URL, if applicable.
- **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below:

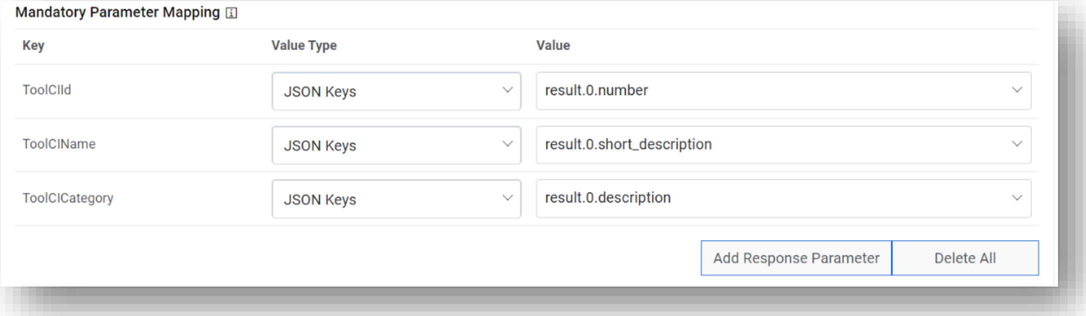
s

```
Response Body -

{

   "result": {

      "sys_id": "xxxxxxxx",

      "name": "xxxxxxxx",

      "category": "xxxxxxxx",

      "subcategory": "xxxxxxxx",

       "sys_updated_on":"2020-06-11 12:43:56"

   }

}
```

- After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
- **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 9– Sample Mandatory Parameter Mapping – CMDB CI

| Key | Value Type | Value |
|---|---|---|
| ToolCIId | JSON.Keys | result.0.sys_id |
| ToolCIName | JSON.Keys | result.0.name |
| ToolCICategory | JSON.Keys | result.0category |


Mandatory Parameter Mapping

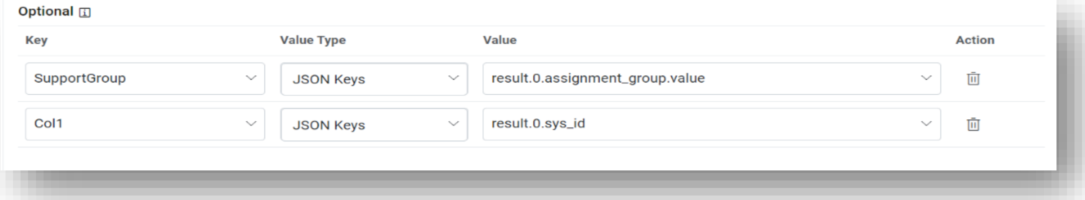
| Key | Value Type | Value |
|---|---|---|
| ToolCIId | JSON Keys | result.0.number |
| ToolCIName | JSON Keys | result.0.short_description |
| ToolCICategory | JSON Keys | result.0.description |

Figure 50 – Mandatory Parameter Mapping – CMDB CI

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 10– Sample Optional Parameters – CMDB CI

| Key | Value Type | Value |
|---|---|---|
| Col3 | JSON.Keys | result. subcategory |


Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| SupportGroup | JSON Keys | result.0.assignment_group.value | |
| Col1 | JSON Keys | result.0.sys_id | |

Figure 51 – Optional Parameter Mapping – CMDB CI

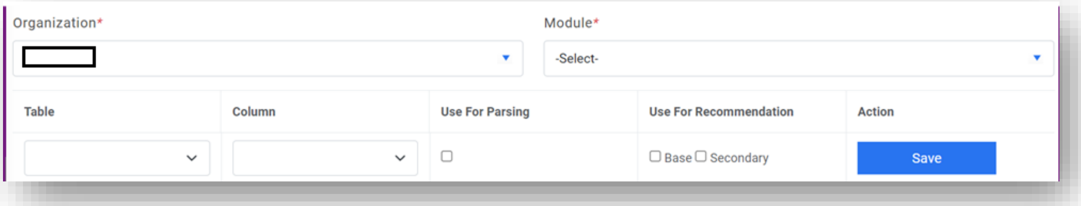
6. Click **Save** to add the data source.

### 4.2.1.3 Configuration of Additional Parameters for Recommendation and Parsing

To use the field values of CMDB CI for the purpose of Recommendation and Parsing by iAutomate services, they need to be mapped to Incident Management.

To do so, perform the following steps –

1. On the left menu bar, click Advance Configuration -> Parameter -> Manage Column.



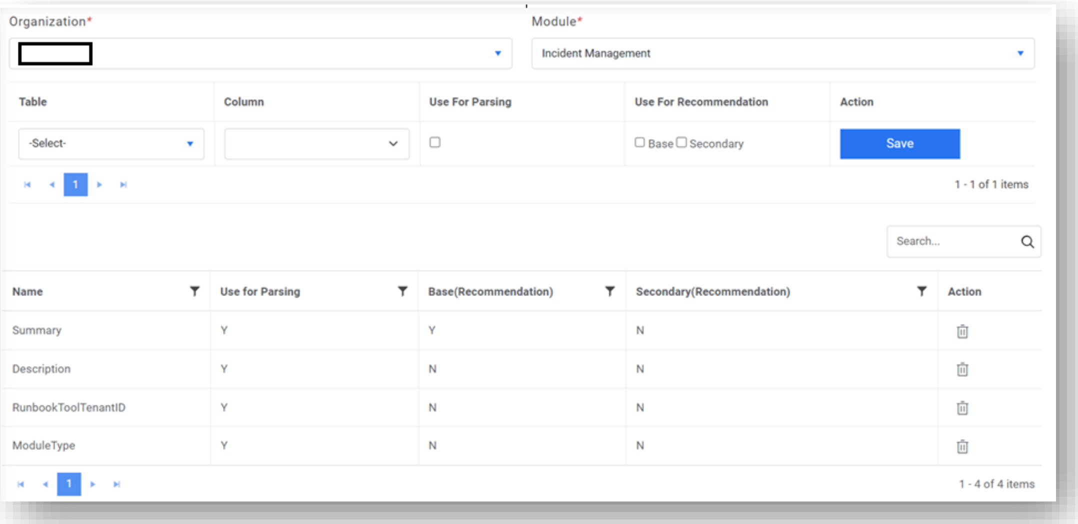
Organization*

Module* -Select-

| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| | | □ | □ Base □ Secondary | Save |

Figure 52 – Map CMDB CI to Incident Management

2. Select **Organization Name** from dropdown. Select I**ncident Management** as the **Module**.



Organization*

Module* Incident Management

| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| -Select- | | □ | □ Base □ Secondary | Save |

1 - 1 of 1 items

Search...

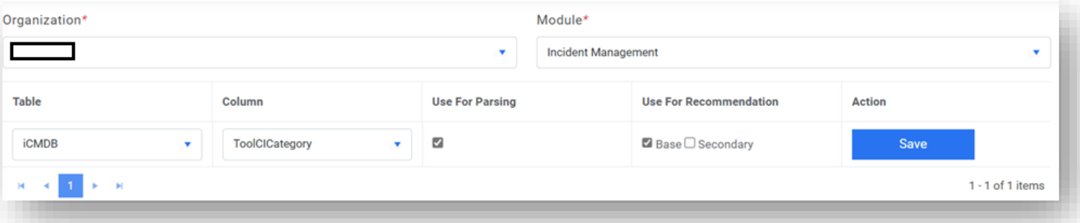
| Name | Use for Parsing | Base(Recommendation) | Secondary(Recommendation) | Action |
|---|---|---|---|---|
| Summary | Y | Y | N | 🗑 |
| Description | Y | N | N | 🗑 |
| RunbookToolTenantID | Y | N | N | 🗑 |
| ModuleType | Y | N | N | 🗑 |

1 - 4 of 4 items

Figure 53 – Map CMDB CI to Incident Management (Cont.)

Summary, Description, RunbookToolTenantID, ModuleType are the default entries.

3. Select **iCMDB** in Table dropdown.
4. Select the column of CMDB which has to be mapped to incident in the **Column** dropdown. In this case, we are selecting **ToolCICategory**.
5. Check the fields Use For Parsing and 'Base' in Use For Recommendation.



Organization*

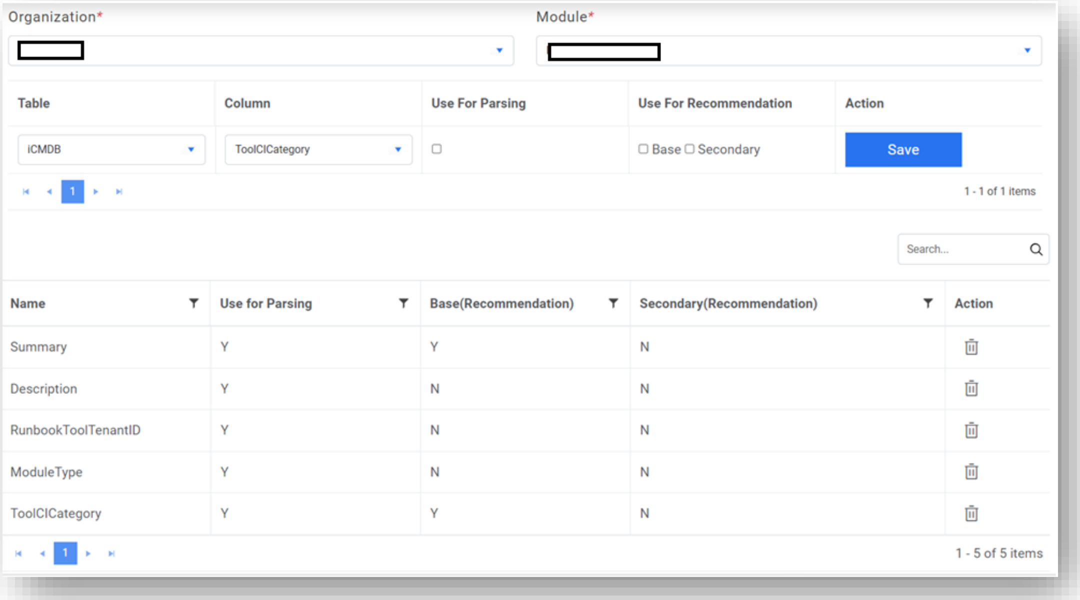
Module* Incident Management

| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| iCMDB | ToolCICategory | ☑ | ☑ Base □ Secondary | Save |

1 - 1 of 1 items

Figure 54 – Map CMDB CI to Incident Management (cont.)

6. Click **Save.** The page lists one additional entry i.e. ' ToolCICategory, as depicted below:



Organization*

Module*

| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| iCMDB | ToolCICategory | □ | □ Base □ Secondary | Save |

1 - 1 of 1 items

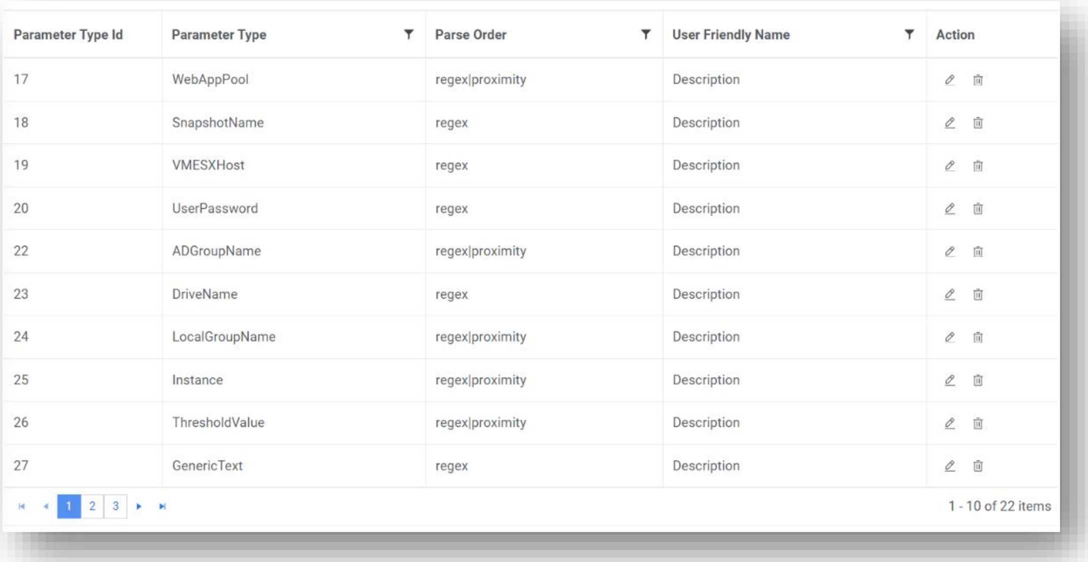
Search...

| Name | Use for Parsing | Base(Recommendation) | Secondary(Recommendation) | Action |
|---|---|---|---|---|
| Summary | Y | Y | N | 🗑 |
| Description | Y | N | N | 🗑 |
| RunbookToolTenantID | Y | N | N | 🗑 |
| ModuleType | Y | N | N | 🗑 |
| ToolCICategory | Y | Y | N | 🗑 |

1 - 5 of 5 items

Figure 55 – Map CMDB CI to Incident Management (cont.)

7. For Recommendation, the above steps are sufficient. But for Parsing, additional steps are required to be performed.
8. On the left menu bar, click Advance Configuration → Parameter → Configure Parameter Type.
9. Click **Configure Parameter Type**. By default, there are several entries already defined.

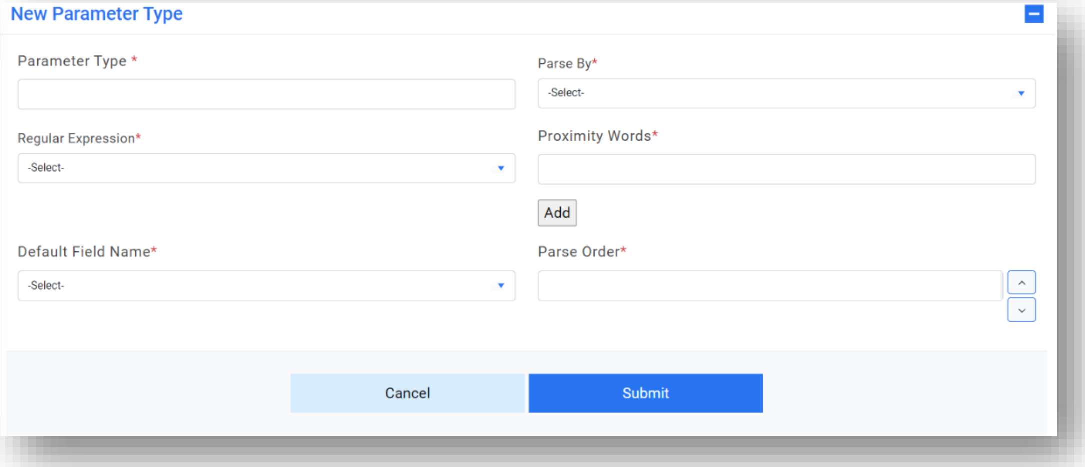


| Parameter Type Id | Parameter Type | Parse Order | User Friendly Name | Action |
|---|---|---|---|---|
| 17 | WebAppPool | regex\|proximity | Description | ✎ 🗑 |
| 18 | SnapshotName | regex | Description | ✎ 🗑 |
| 19 | VMESXHost | regex | Description | ✎ 🗑 |
| 20 | UserPassword | regex | Description | ✎ 🗑 |
| 22 | ADGroupName | regex\|proximity | Description | ✎ 🗑 |
| 23 | DriveName | regex | Description | ✎ 🗑 |
| 24 | LocalGroupName | regex\|proximity | Description | ✎ 🗑 |
| 25 | Instance | regex\|proximity | Description | ✎ 🗑 |
| 26 | ThresholdValue | regex\|proximity | Description | ✎ 🗑 |
| 27 | GenericText | regex | Description | ✎ 🗑 |

1 - 10 of 22 items

Figure 56 – Map CMDB CI to Incident Management (cont.)

10. Click **on** + icon to add new.



New Parameter Type

Parameter Type *

Parse By* -Select-

Regular Expression* -Select-

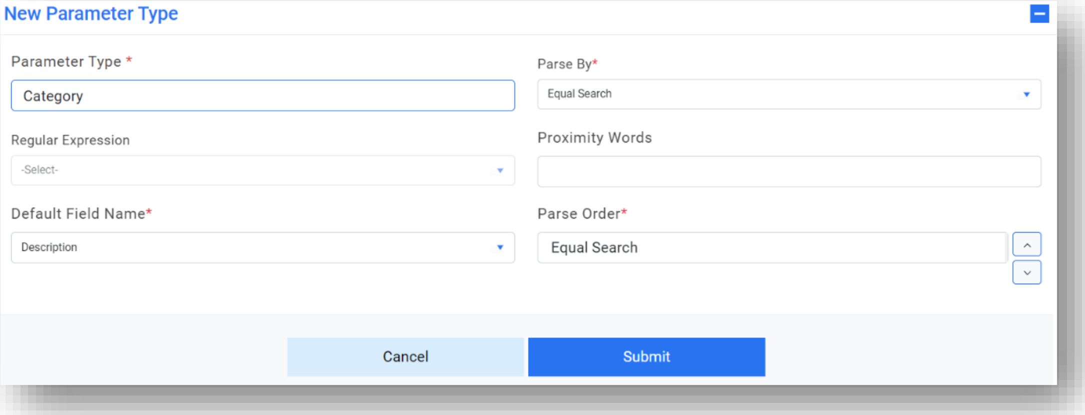
Proximity Words*

Default Field Name* -Select-

Parse Order*

Figure 57 – Map CMDB CI to Incident Management (Cont.)

11. Mention **Parameter Type**, for e.g. Category
12. Select 'Equal Search' in the **Parse By** field.
13. Select 'Description' in the **Default Field Name** field.
14. Click **Submit**.



New Parameter Type

Parameter Type * Category

Parse By* Equal Search

Regular Expression -Select-

Proximity Words

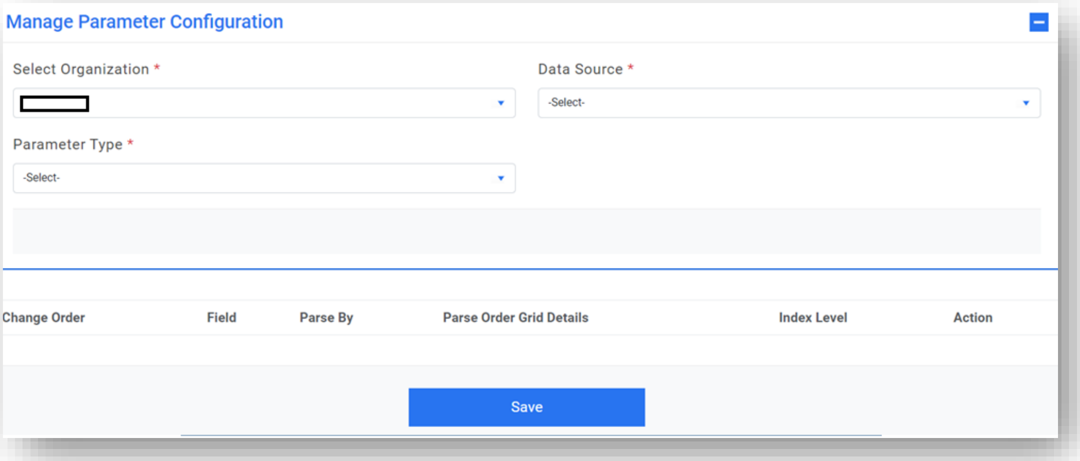
Default Field Name* Description

Parse Order* Equal Search

Figure 58 – Map CMDB CI to Incident Management (Cont.)

15. Next step is to map this **Parameter Type** i.e. '**Category'**, to the one that was created via **Manage Columns** in earlier step by the name **subcategory**. To do that, perform the following steps:

1. On the left menu bar, click Advance Configuration -> Parameter.
2. Click Manage Parameter Configuration.


Manage Parameter Configuration

Select Organization * [redacted]

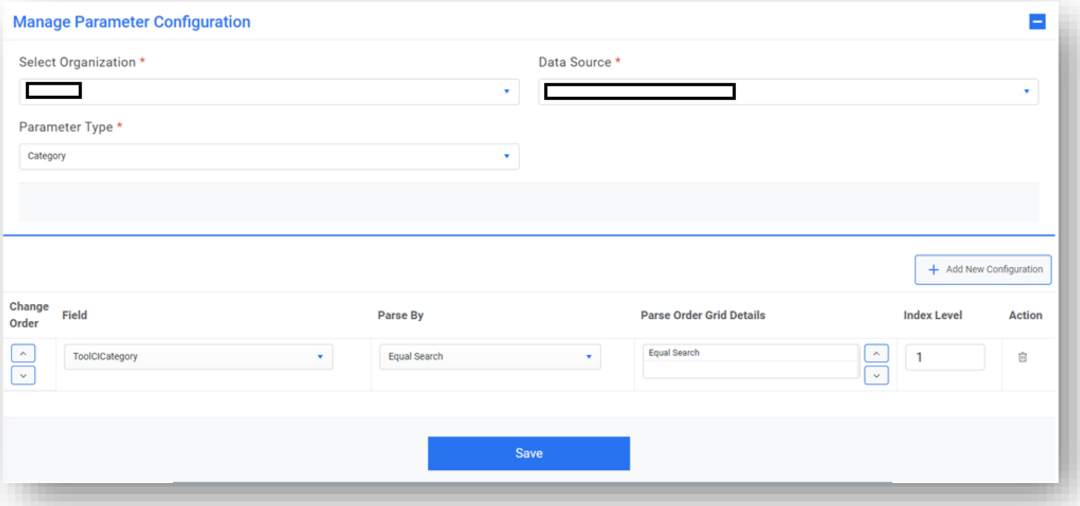
Data Source * -Select-

Parameter Type * -Select-

| Change Order | Field | Parse By | Parse Order Grid Details | Index Level | Action |
|---|---|---|---|---|---|

Figure 59 – Map CMDB CI to Incident Management (Cont.)

3. Select Organization.
4. Select 'Incident Management' as the **Data Source**.
5. Select the newly created parameter 'Category' from **Parameter Type** dropdown**.**
6. From the **Field** dropdown, select 'subcategory'**,** the parameter that has been mapped via **Manage Columns.**


Manage Parameter Configuration

Select Organization * [redacted]

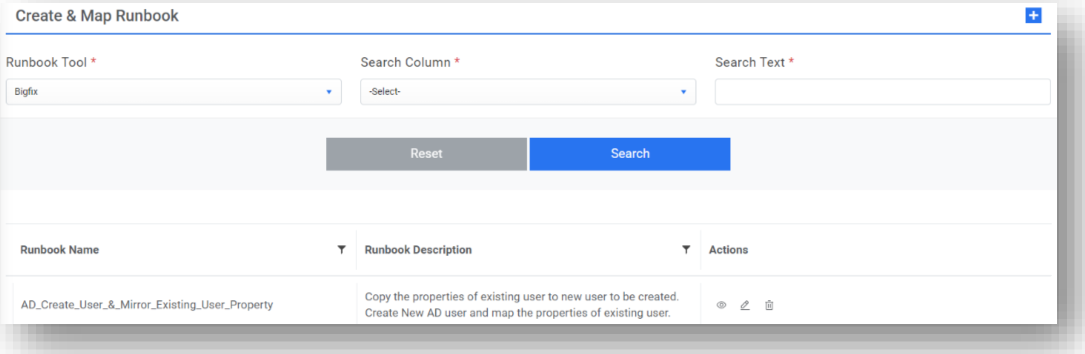
Data Source * [redacted]

Parameter Type * Category

| Change Order | Field | Parse By | Parse Order Grid Details | Index Level | Action |
|---|---|---|---|---|---|
| ˄ ˅ | ToolCICategory | Equal Search | Equal Search | 1 | 🗑 |

Figure 60 – Map CMDB CI to Incident Management (Cont.)

7. Click **Save**.
8. To verify whether this parameter is successfully parsed or not, perform the following steps -
   - On the left menu bar, click **Runbooks.**
   - Click Manage Runbooks.
   - Select the **Runbook Tool** mapped with the organization.



Create & Map Runbook

Runbook Tool * Bigfix

Search Column * -Select-

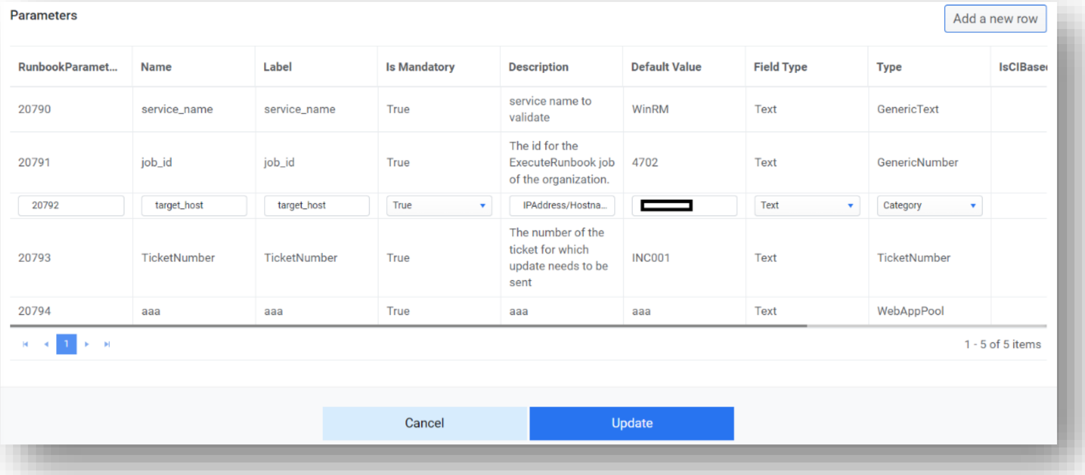
Search Text *

| Runbook Name | Runbook Description | Actions |
|---|---|---|
| AD_Create_User_&_Mirror_Existing_User_Property | Copy the properties of existing user to new user to be created. Create New AD user and map the properties of existing user. | |

Figure 61 – Map CMDB CI to Incident Management (Cont.)

The parameter, **Category**, which was created in earlier steps, has to be added as one of the parameters to the existing runbook. You can also create a new runbook with **Category** as one of the parameters.

- Click the **Edit** icon to edit the runbook.
- In the Parameters section, add a new parameter with any relevant **Parameter Name, Parameter Label, Parameter Description, Default Parameter Value**. Ensure that Parameter Type is selected as **Category**.



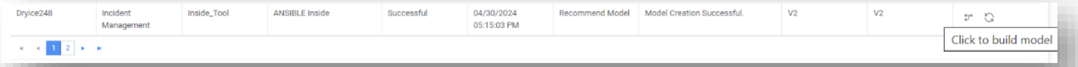
Parameters

| RunbookParamet... | Name | Label | Is Mandatory | Description | Default Value | Field Type | Type | IsCIBase |
|---|---|---|---|---|---|---|---|---|
| 20790 | service_name | service_name | True | service name to validate | WinRM | Text | GenericText | |
| 20791 | job_id | job_id | True | The id for the ExecuteRunbook job of the organization. | 4702 | Text | GenericNumber | |
| 20792 | target_host | target_host | True | IPAddress/Hostna... | [redacted] | Text | Category | |
| 20793 | TicketNumber | TicketNumber | True | The number of the ticket for which update needs to be sent | INC001 | Text | TicketNumber | |
| 20794 | aaa | aaa | True | aaa | aaa | Text | WebAppPool | |

1 - 5 of 5 items

Figure 62 – Map CMDB CI to Incident Management (Cont.)

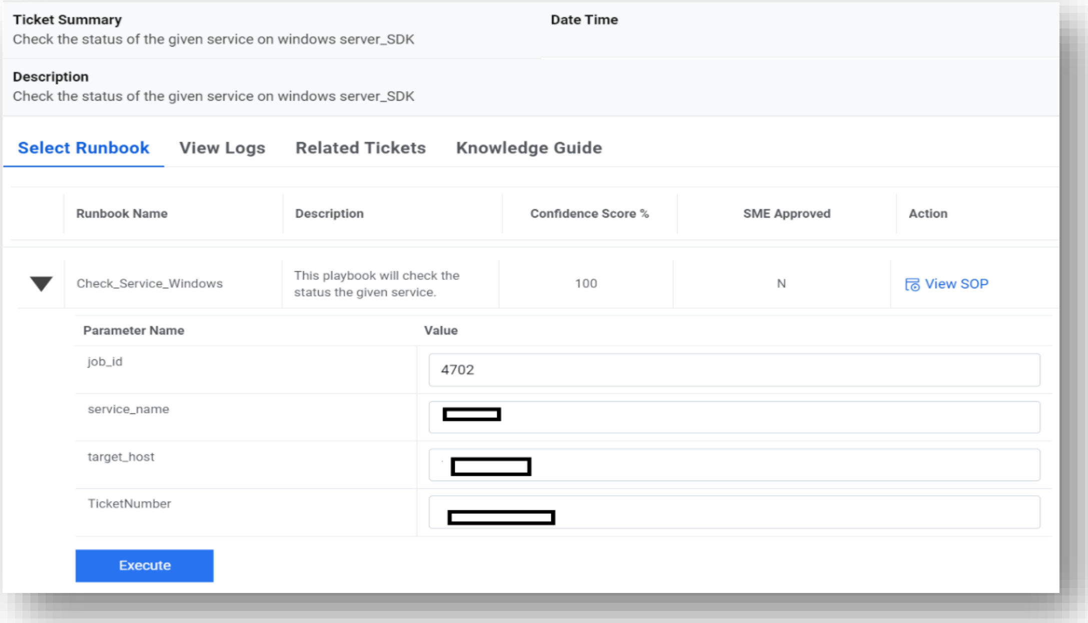
9. Add the parameter and click **Update**.
10. Ensure that the runbook in which the parameter is added is mapped with the organization.
11. Next step is to build the Recommendation model and to do that perform the following steps:
    - On the left menu bar, click **Configuration→ Build Model.**
    - ReBuild / Re-build the model for the Organization under **Incident Management** module for the mapped runbook tool.



| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dryice248 | Incident Management | Inside_Tool | ANSIBLE Inside | Successful | 04/30/2024 05:15:03 PM | Recommend Model | Model Creation Successful. | V2 | V2 |

Figure 63 – Map CMDB CI to Incident Management (Cont.)

- Run the entire flow and see if the runbook recommended for the ticket in which the parameter was added has the parameter **Category** with its expected value.



**Ticket Summary**
Check the status of the given service on windows server_SDK

**Date Time**

**Description**
Check the status of the given service on windows server_SDK

| Runbook Name | Description | Confidence Score % | SME Approved | Action |
|---|---|---|---|---|
| Check_Service_Windows | This playbook will check the status the given service. | 100 | N | View SOP |

| Parameter Name | Value |
|---|---|
| job_id | 4702 |
| service_name | [redacted] |
| target_host | [redacted] |
| TicketNumber | [redacted] |

Figure 64 – Map CMDB CI to Incident Management (Cont.)

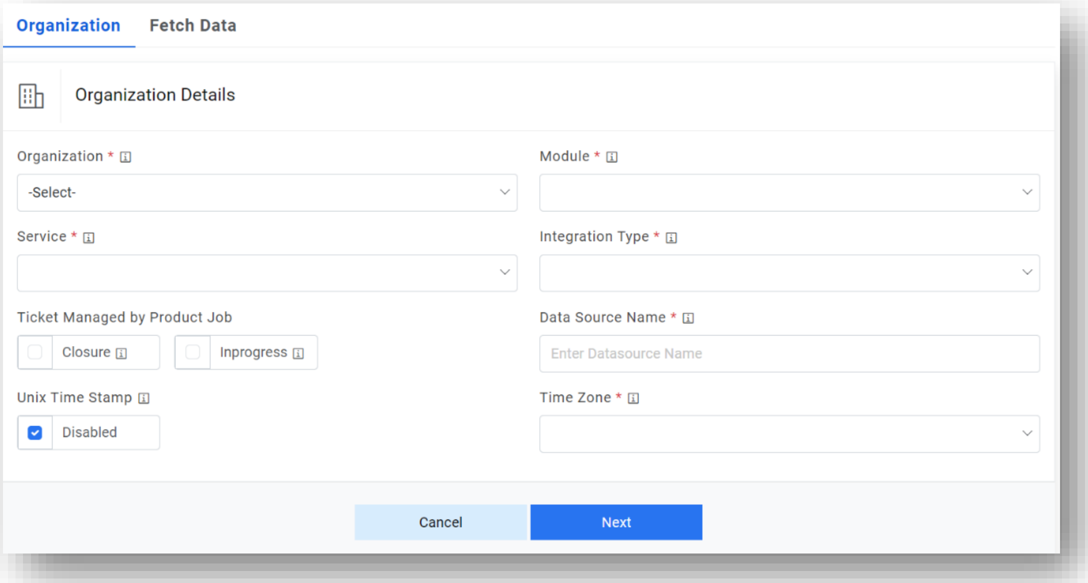
## 4.2.2 Service Request Management

To fetch information about Service Requests, usually, creation of a data source for Service Request Tasks should suffice. However, there could be scenarios where some additional fields / values are required for processing the tickets – recommending the relevant runbooks and parsing the tickets to extract relevant parameters, for which separate data sources for Service Request and Service Request Item must be created. Here, we will cover the procedure for creating all 3 kinds of data sources.

### 4.2.2.1 Create Data Source for Service Request

To create a data source for Service Requests, perform the following steps:

1. On the left menu bar, click **Configuration → Manage Data** Sources.
2. The **Create Data Source** page appears with the following tabs:
   - Organization
   - Fetch Data



Organization | Fetch Data

Organization Details

Organization * -Select-

Module *

Service *

Integration Type *

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name * Enter Datasource Name

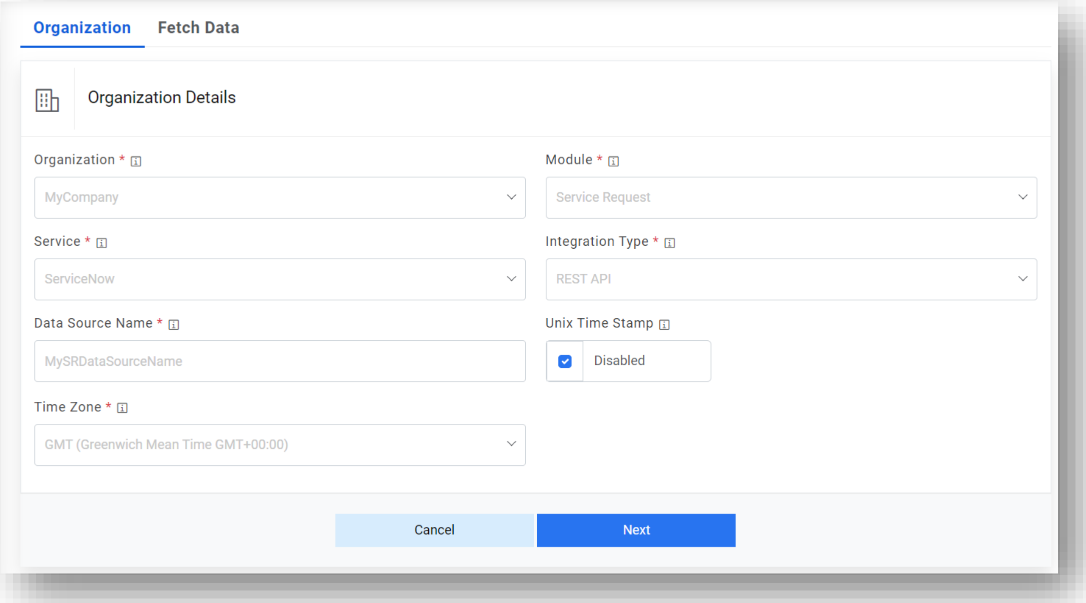
Unix Time Stamp: ☑ Disabled

Time Zone *

Figure 65 - Create Data Source – Service Request

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab:
   - Select the **Organization Name** from the dropdown.
   - In the **Module** field, select 'Service Request', since we are using this data source for using its field value for the **Service Request Tasks**.
   - In the **Service** field, select **Service Now** as we are configuring the data source for ServiceNow.
   - In the **Integration Type** field, select **REST API**, since we will be integrating through REST APIs.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select **Unix Time** Stamp to view the present data with date and time.
   - Click **Next**.



Organization Details

Organization * MyCompany

Module * Service Request

Service * ServiceNow

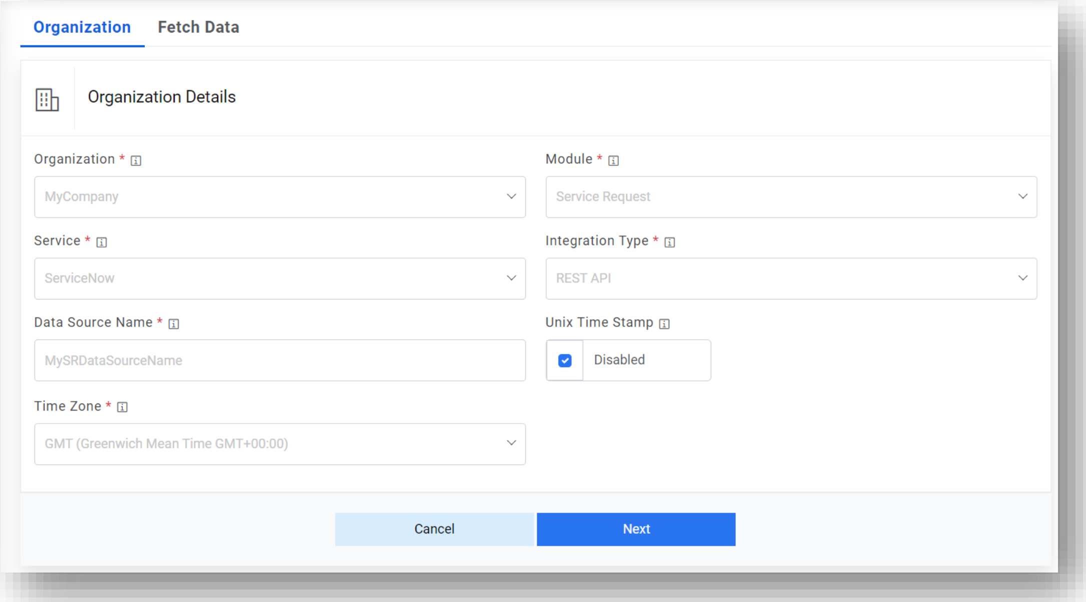
Integration Type * REST API

Data Source Name * MySRDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 66 - Create Data Source – Service Request (Cont.)



Organization Details

Organization * MyCompany

Module * Service Request

Service * ServiceNow

Integration Type * REST API

Data Source Name * MySRDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 67 - Create Data Source – Service Request (Cont.)

4. On the **Fetch Data** tab, populate the details as per the environment.
5. In the **Connection Details** section, enter the following details:

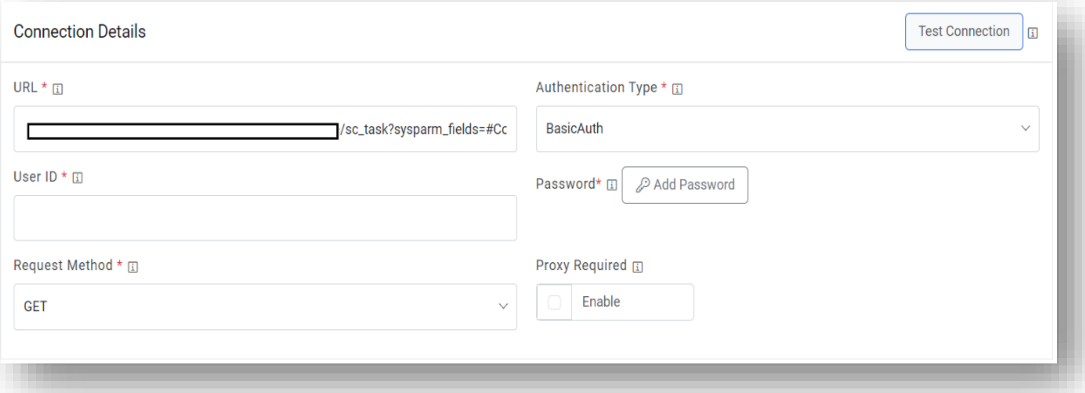
– **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.

– **Sample URL** – https://URL.service-now.com/api/now/v1/table/sc_request?sysparm_fields=#Columns#&sysparm_query=sys_updated_on>=#StartDate#^sys_updated_on<=#EndDate#^ORDERBYsys_updated_on

– **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0

- Selection of **Basic / Windows** requires you to enter -
  - User Id
  - Password
- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL
- **Request Method –** Select the **GET**, **POST** or **PUT** as Request Method as per the configured URL.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

URL *: [redacted]/sc_task?sysparm_fields=#Cc

Authentication Type *: BasicAuth

User ID *:

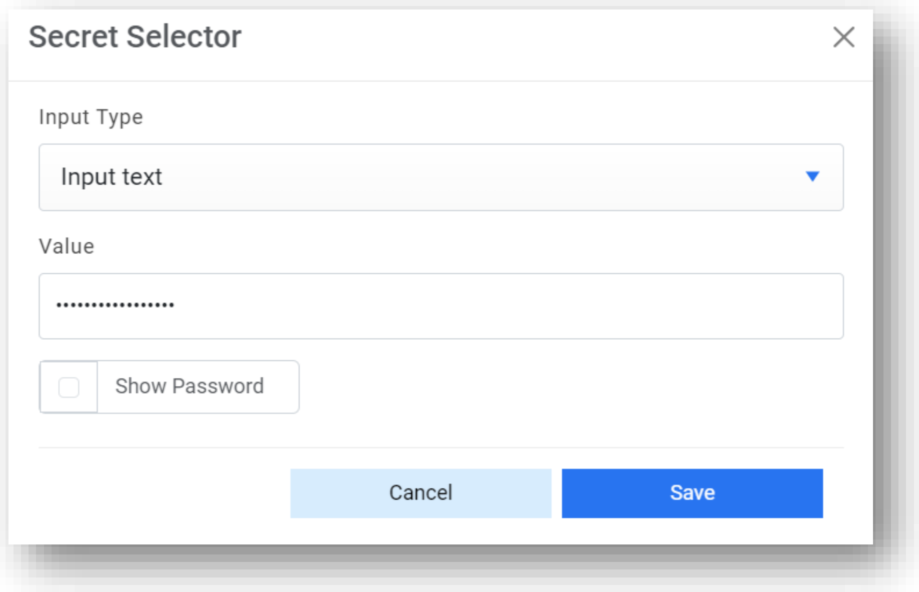
Password*

Request Method *: GET

Proxy Required: □ Enable

Figure 68 – Create Data Source – Service Request (Connection Details)

- **Password** – For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



Secret Selector

Input Type

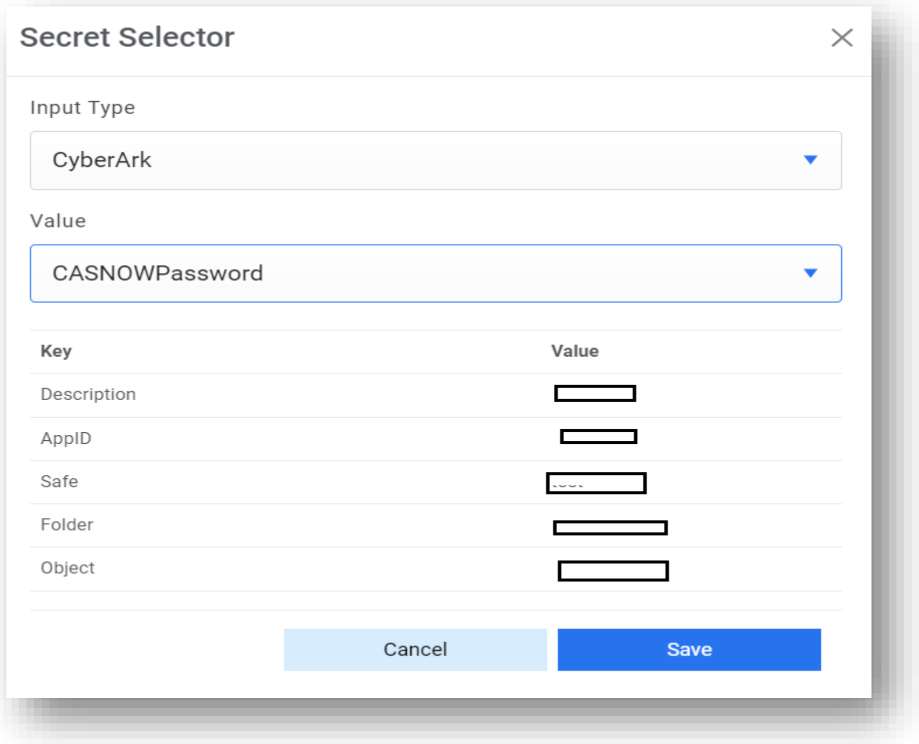
Input text

Value

•••••••••••••••••

□ Show Password

Figure 69 - Password in plaintext



Secret Selector

Input Type

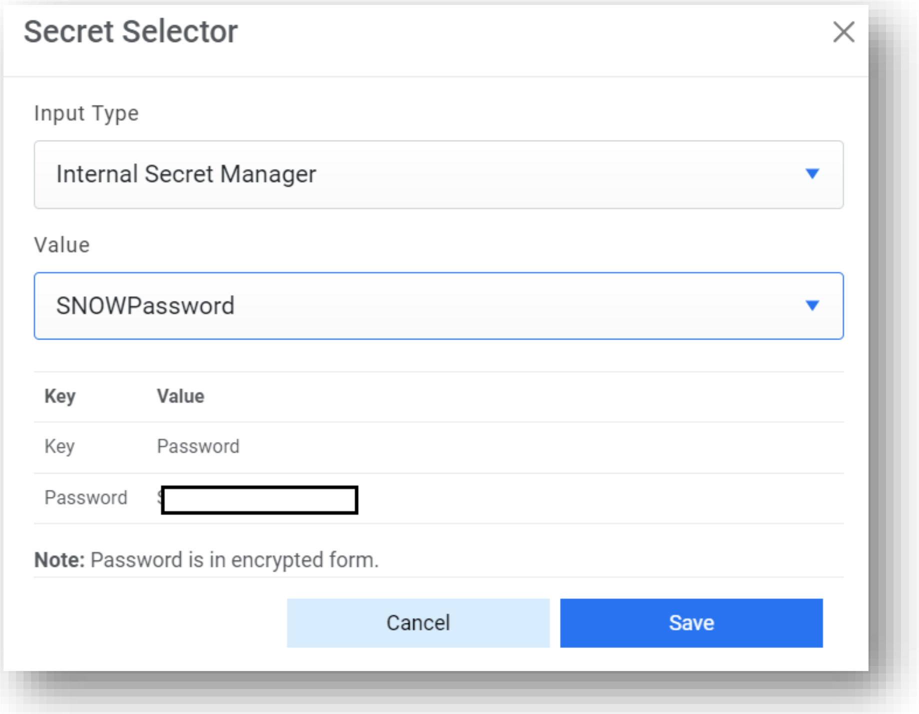
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 70 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

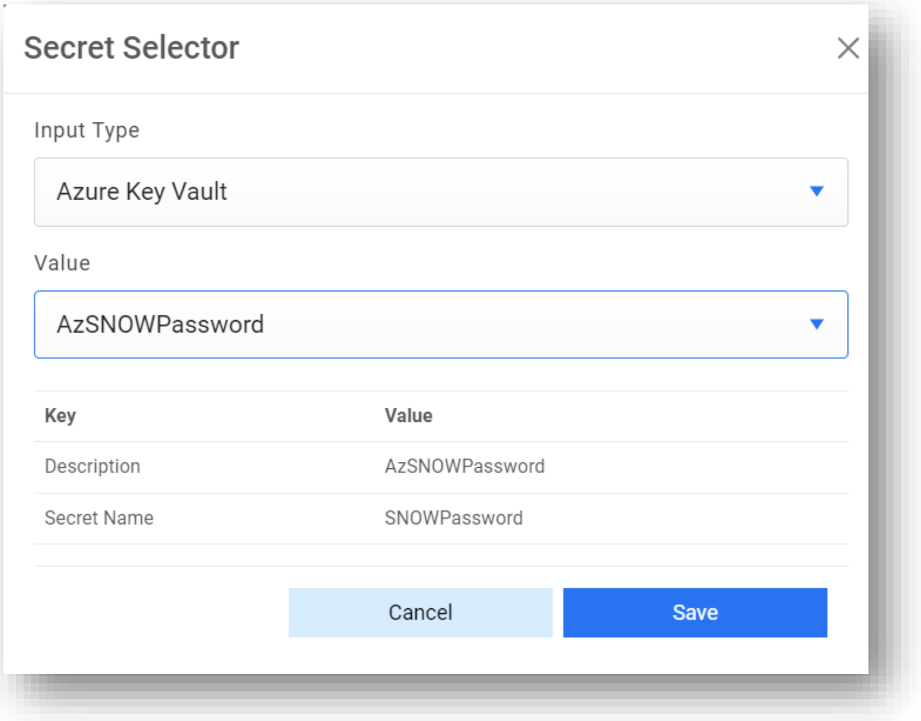
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 71 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

AzSNOWPassword

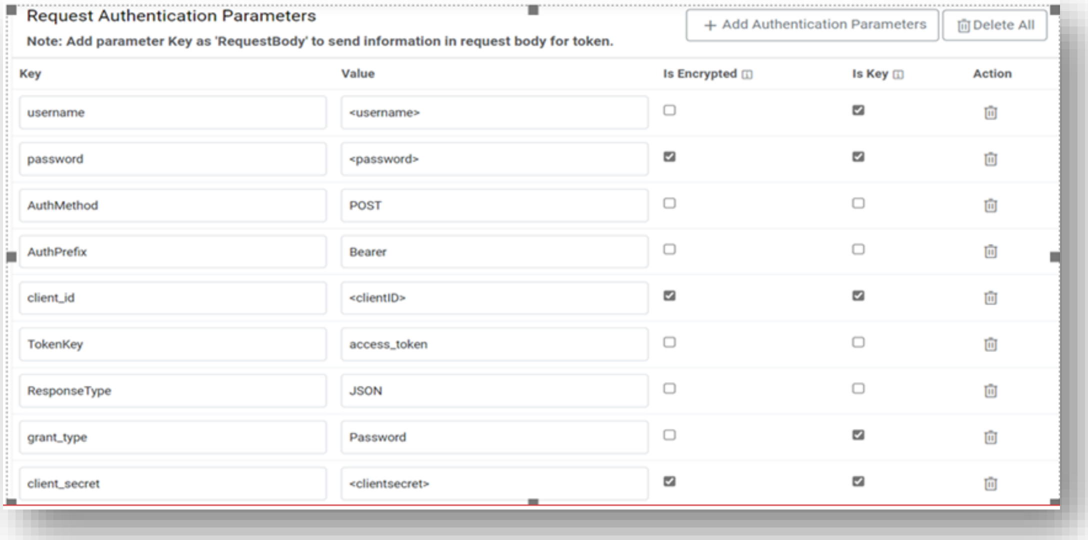
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 72- Password from Azure Key Vault

- **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
- Based on the **Authentication Type**, add the parameters mentioned in the below table:

Table 11 – Sample Authentication Parameters – Service Request

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

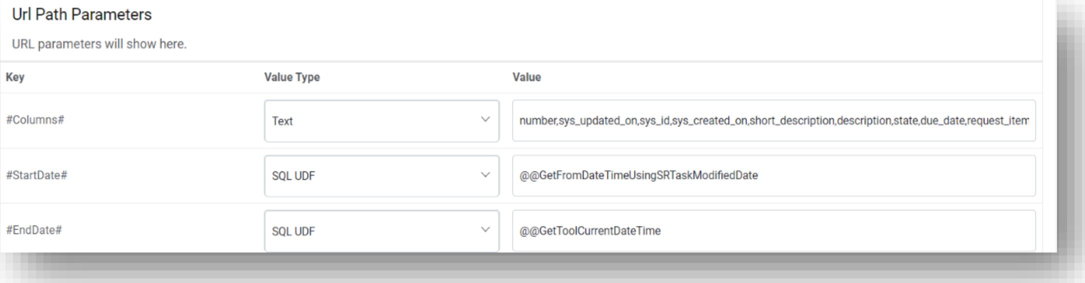
| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | ☐ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | ☐ | ☐ | 🗑 |
| AuthPrefix | Bearer | ☐ | ☐ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | ☐ | ☐ | 🗑 |
| ResponseType | JSON | ☐ | ☐ | 🗑 |
| grant_type | Password | ☐ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 73 – Create Data Source – Service Request (Request Authentication Parameters for OAuth2.0)

- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Columns#

ValueType: Text

Value:

number,sys_updated_on,sys_id,sys_created_on,short_description,d
escription,state,request_state



Note – These columns are mandatory. User can add more columns if
more data is required to be fetched from ITSM tool.



Key: #StartDate#

ValueType: SQL UDF
```

```
VALUE: @@GetFromDateTimeUsingServiceRequestModifiedDate


Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,sys_id,sys_created_on,short_description,description,state,due_date,request_item |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingSRTaskModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 74 – URL Path Parameters – Service Request (Service Request Task Management)

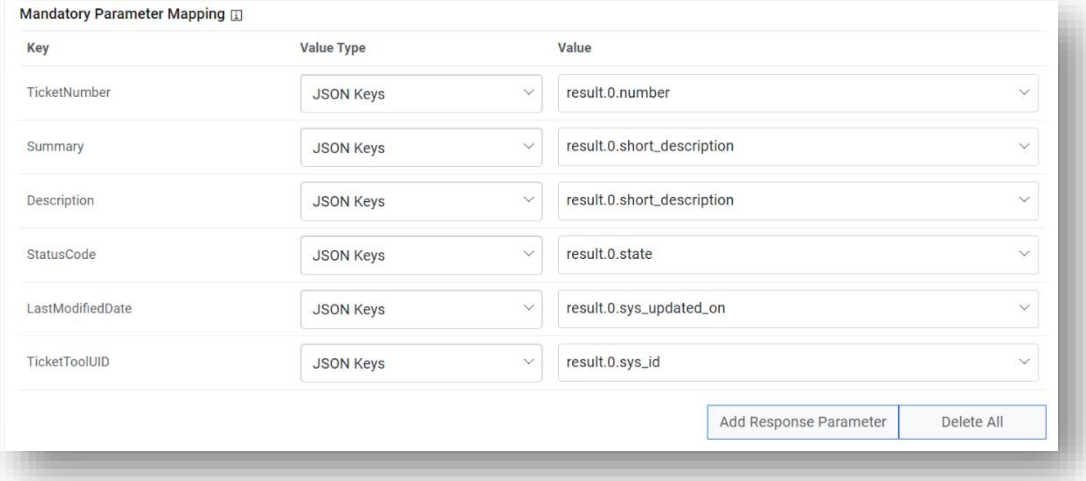
- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body -** Enter the request body in JSON format as per the configured URL, if applicable.
- **Response Body –** In this section, please enter the output of URL query for one of the service requests tasks in JSON format. A sample response is mentioned below:

```
Response Body -

{

   "result": {

      "number": "REQ0011787",

      "sys_id": "xxxxxxxx",

      "short_description": "Test",

      "request_state": "in_process",

      "sys_created_on": "2020-06-08 10:34:54",

      "description": "test",

      "sys_updated_on": "2020-06-08 10:34:56",

      "state": "2"

   }

}
```

- After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
- **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 12– Sample Mandatory Mapping Parameters – Service Request

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.number |
| Summary | JSON.Keys | result.0.short_description |
| Description | JSON.Keys | result.0.description |
| StatusCode | JSON.Keys | result.0.state |
| LastModifiedDate | JSON.Keys | result.0.sys_updated_on |
| TicketToolUID | JSON.Keys | result.0.sys_id |



Mandatory Parameter Mapping

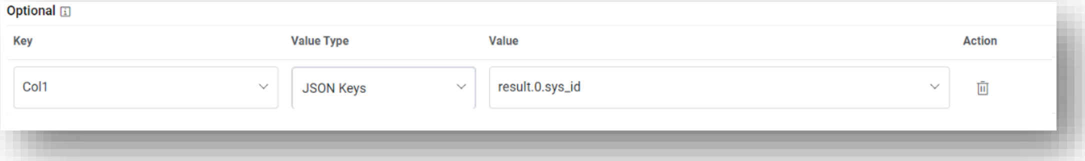
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.number |
| Summary | JSON Keys | result.0.short_description |
| Description | JSON Keys | result.0.short_description |
| StatusCode | JSON Keys | result.0.state |
| LastModifiedDate | JSON Keys | result.0.sys_updated_on |
| TicketToolUID | JSON Keys | result.0.sys_id |

Figure 75 – Mandatory Parameter Mapping (Service Request Management)

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 13 – Sample Optional Mapping Parameters – Service Request

| Key | Value Type | Value |
|---|---|---|
| Col1 | JSON.Keys | Result.0.sys.id |



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| Col1 | JSON Keys | result.0.sys_id | |

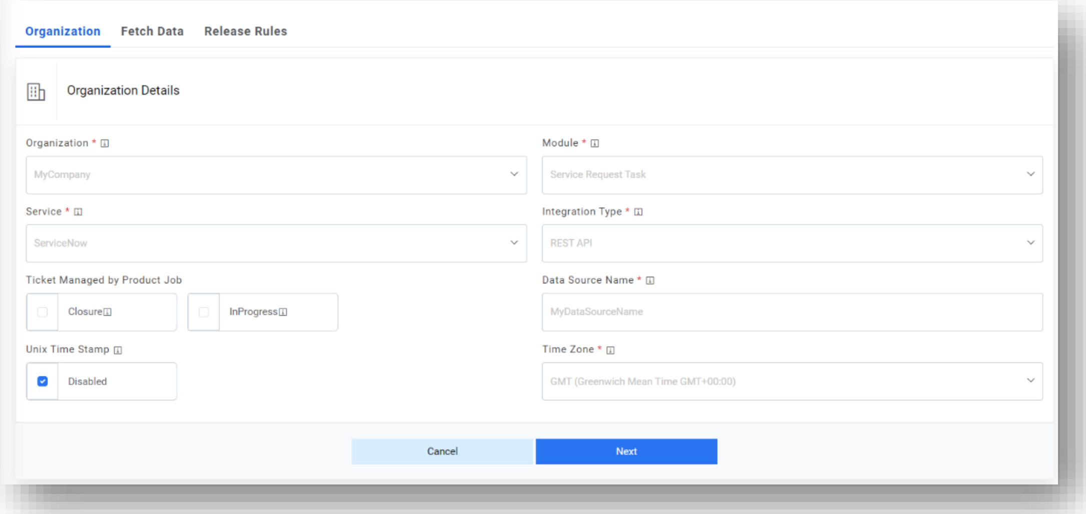
Figure 76 – Optional Parameter Mapping (Service Request Management)

6. Click Save to add the data source.

### 4.2.2.2 Create Data Source for Service Request Tasks

To create a data source for Service Requests Tasks Management, perform the following steps:

1. On the left menu bar, click Configuration Tab -> **Manage Data Sources**.
2. The **Create Data Source** page appears with the following tabs:
   - Organization
   - Fetch Data
   - Release Rules



Organization Details

Organization * : MyCompany

Module * : Service Request Task

Service * : ServiceNow

Integration Type * : REST API

Ticket Managed by Product Job: □ Closure □ InProgress

Data Source Name * : MyDataSourceName

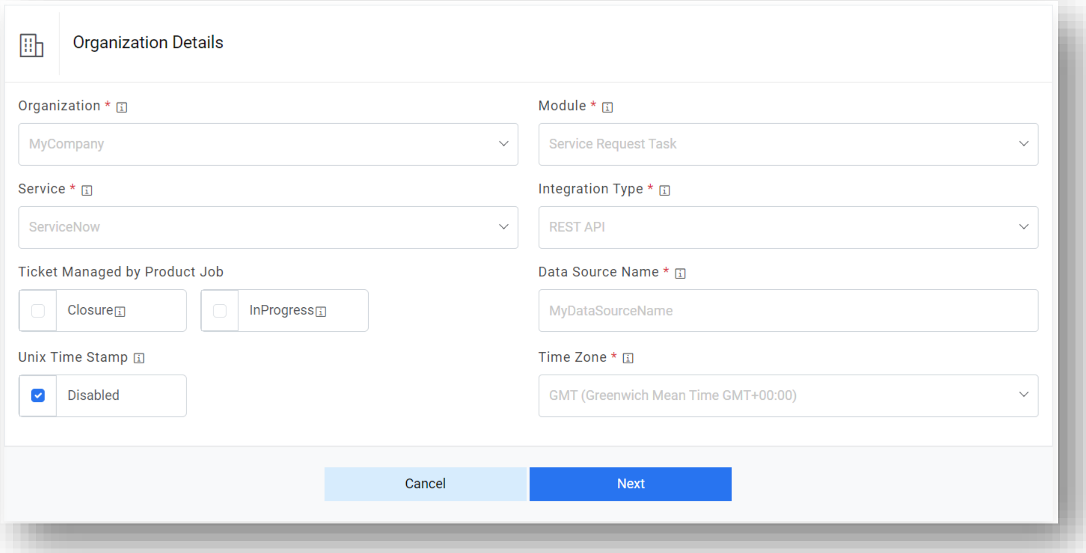
Unix Time Stamp: ☑ Disabled

Time Zone * : GMT (Greenwich Mean Time GMT+00:00)

Figure 77 - Create Data Source – Service Request Tasks

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab,
   - Select the **Organization Name** from the dropdown.
   - In the **Module** field, select 'Service Request Task'**,** since we are configuring this data source for pulling the service requests tasks.
   - In the **Service** field, select **Service Now Tool** as we are configuring the data source for ServiceNow
   - In the **Integration Type** field, select **REST**, since we will be integrating through REST APIs.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select Unix **Timestamp** to view the present data with date and time.
   - Click **Next**.



Organization Details

Organization * MyCompany

Module * Service Request Task

Service * ServiceNow

Integration Type * REST API

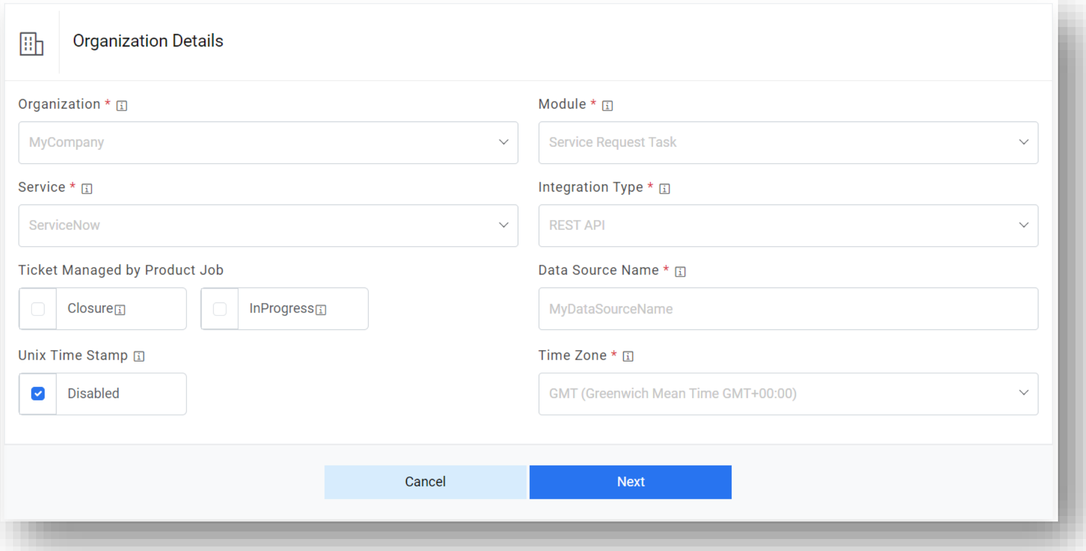
Ticket Managed by Product Job: □ Closure □ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp: ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 78 – Create Data Source – Service Request Tasks (Cont.)



Organization Details

Organization * MyCompany

Module * Service Request Task

Service * ServiceNow

Integration Type * REST API

Ticket Managed by Product Job: □ Closure □ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp: ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 79 – Create Data Source – Service Request Tasks (Cont.)

4. On the **Fetch Data** tab, type in the details as per the environment.
5. In the **Connection Details** section, enter the following details:

– **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.

– **Sample URL** – https://<url>?sysparm_fields=#Columns#&sysparm_query=active=true^sys_updated_on>=#StartDate#^sys_updated_on<=#EndDate#^ORDERBYsys_updated_on

– **Authentication Type** – Select one of the Authentication Types from Basic / Windows,  OAuth 2.0

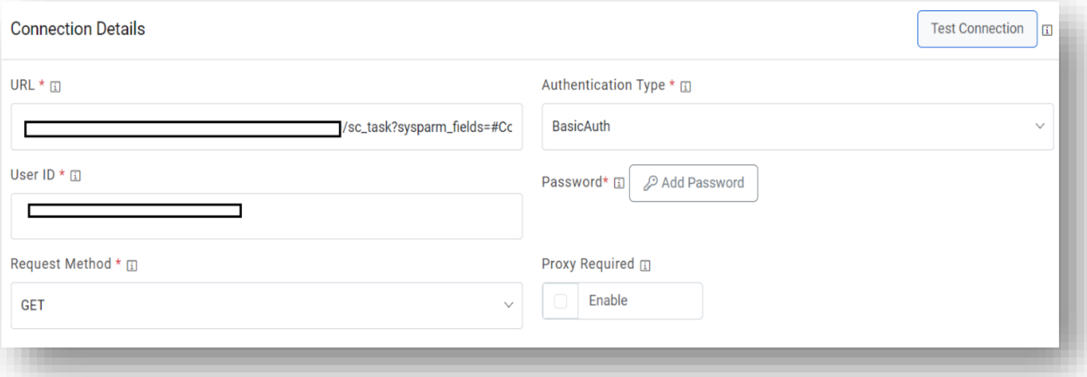
– Selection of **Basic / Windows** requires you to enter -

  - User Id

  - Password

- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL

- **Request Method –** Enter the request method as **GET**, **POST** or **PUT** as per the configured URL.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.
- Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

Test Connection

URL *: [redacted]/sc_task?sysparm_fields=#Cc

Authentication Type *: BasicAuth

User ID *: [redacted]

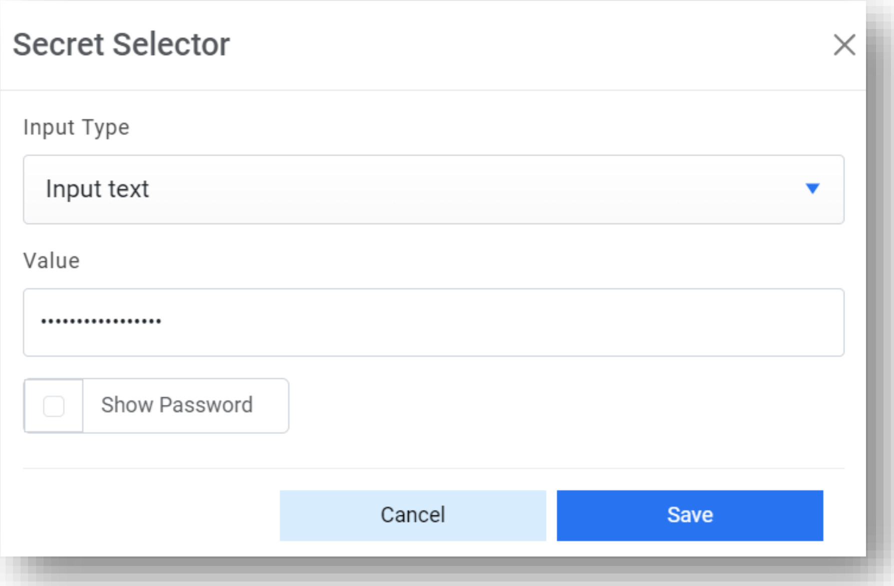
Password*: Add Password

Request Method *: GET

Proxy Required: □ Enable

Figure 80 – Create Data Source – Service Request Tasks (Connection Details)

- **Password** – For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



Secret Selector

Input Type

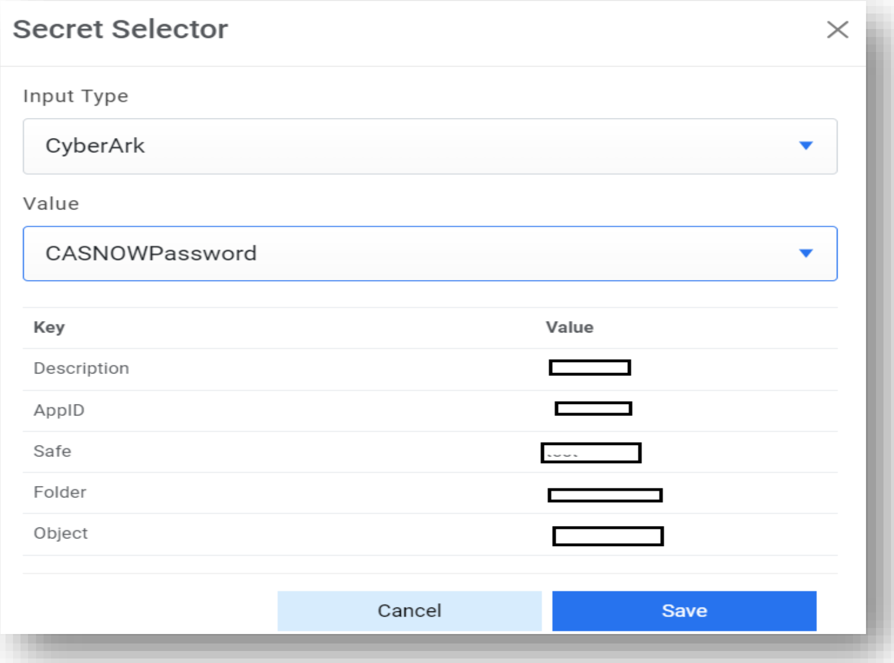
Input text

Value

•••••••••••••••••

□ Show Password

Figure 81 - Password in Plaintext



Secret Selector

Input Type

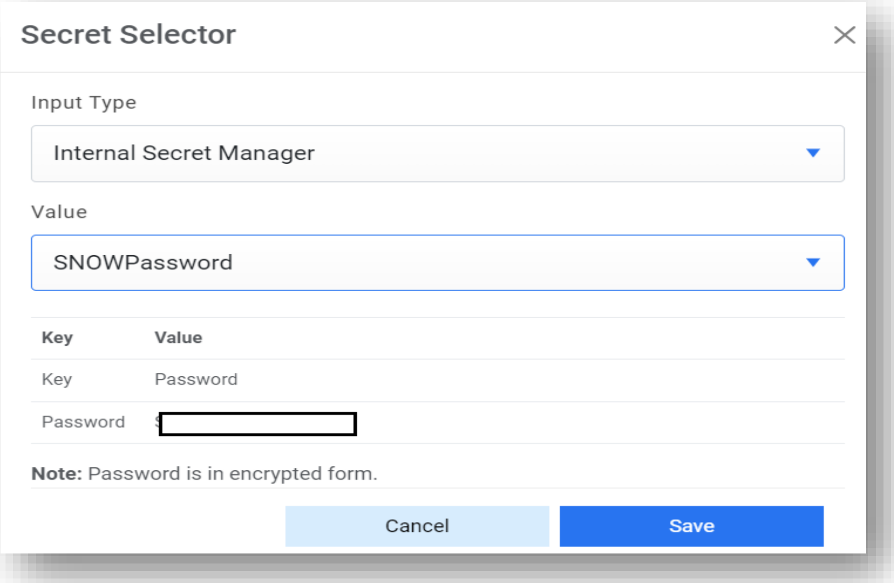
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 82 - Password from Key Vault (CyberArk)



Secret Selector

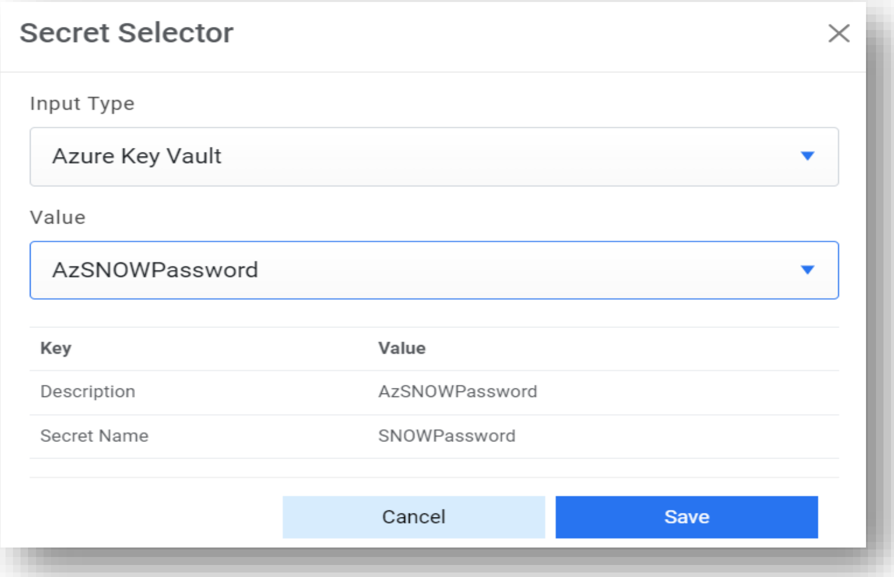
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 83 - Password from Secret Manager



Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

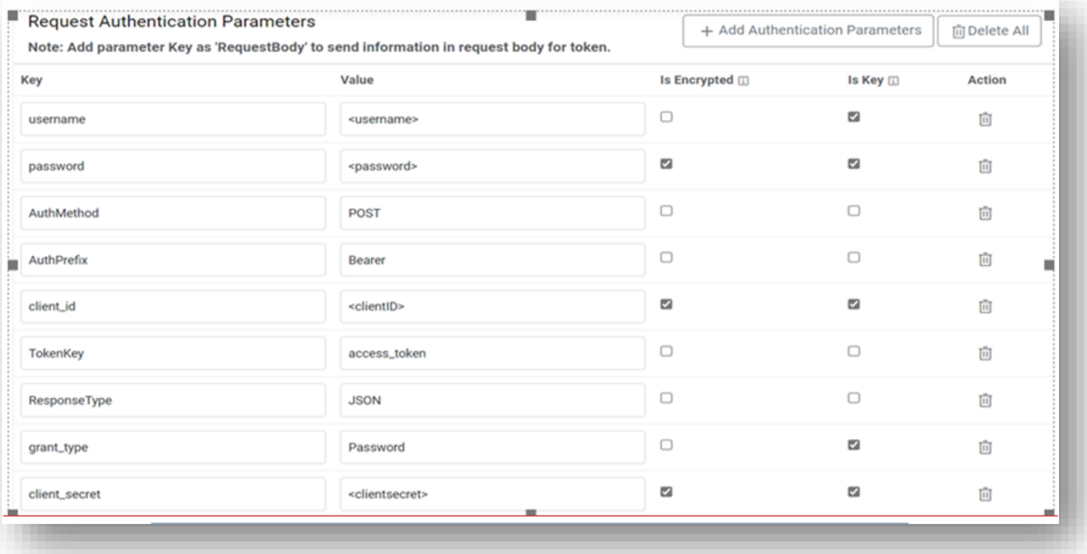
Figure 84 - Password from Azure Key Vault

- **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
- Based on the **Authentication Type**, add the parameters mentioned in the below table:

Table 14 – Sample Authentication Parameters – Service Request Tasks

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |

| | | | | |
|---|---|---|---|---|
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

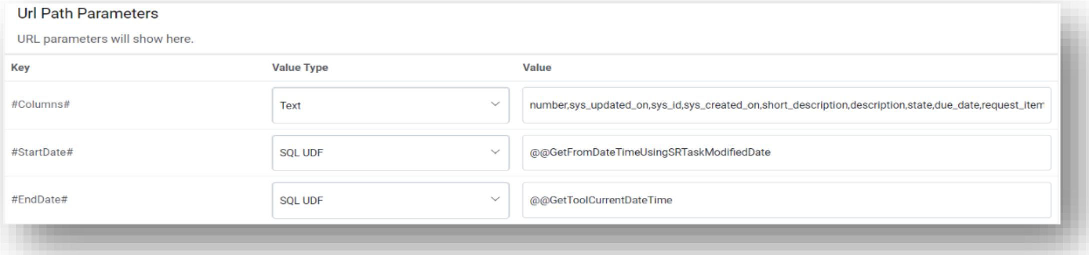
| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | □ | □ | 🗑 |
| AuthPrefix | Bearer | □ | □ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | □ | □ | 🗑 |
| ResponseType | JSON | □ | □ | 🗑 |
| grant_type | Password | □ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 85 – Create Data Source – Service Request Tasks (Request Authentication Parameters for OAuth2.0)

- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Columns#

ValueType: Text

Value:

number,    sys_updated_on,    short_description,    description,
assignment_group,
closed_at,category,dv_assigned_to,sys_id,sys_created_on,state,r
equest,request_item,sys_id


Note – These columns are mandatory. User can add more columns if
more data is required to be fetched from ITSM tool.


Key: #StartDate#

ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingSRTaskModifiedDate


Key: #EndDate#
```

```
ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,sys_id,sys_created_on,short_description,description,state,due_date,request_item |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingSRTaskModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 86 – URL Path Parameters (Service Request Task)

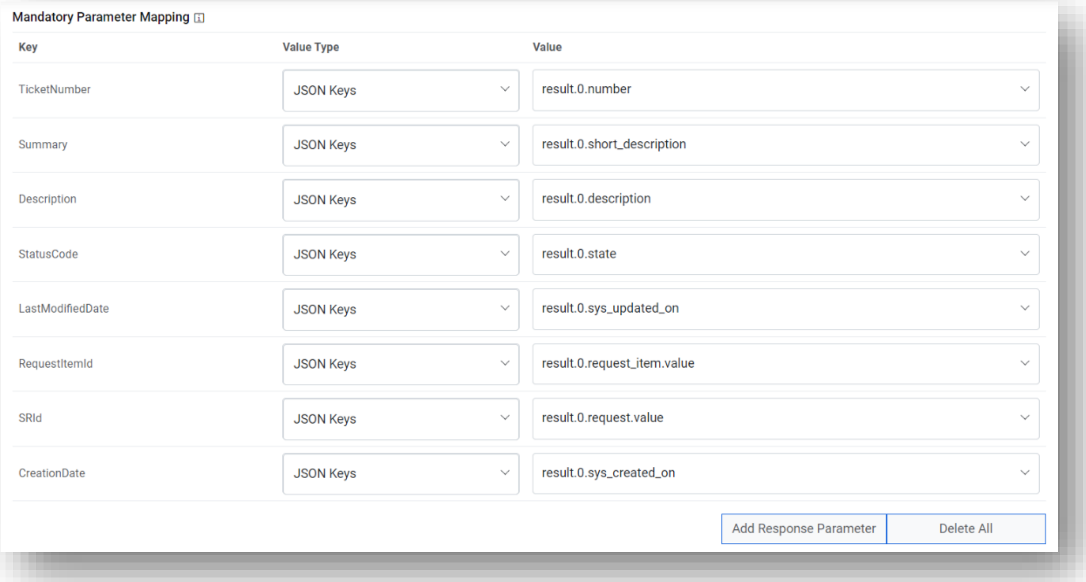
- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body -** Enter the request body in JSON format as per the configured URL, if applicable.
- **Response Body –** In this section, please enter the output of URL query for one of the service requests tasks in JSON format. A sample response is mentioned below:

```
Response Body –

{

   "result": [{"number": "TASK2190188","short_description": "For
fullfillment","description": "Test",       "state": "1","active":
"true","sys_created_on":"2019-12-31          05:45:39","sys_id":
"xxxxxxxx","approval":  "not  requested","sys_updated_on":"2020-
01-31 05:45:39”,"request": {

"link":"https://my_host",  "value": "xxxxxxxx"},

 "request_item": {"link": "https://my_host ", "value": "xxxxxxxx"

        }

   }]

}
```

- After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
- **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 15– Sample Mandatory Mapping Parameters – Service Request Tasks

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.number |
| Summary | JSON.Keys | result.0.short_description |
| Description | JSON.Keys | result.0.description |
| StatusCode | JSON.Keys | result.0.state |
| LastModifiedDate | JSON.Keys | result.0.sys_updated_on |
| RequestItemId | JSON.Keys | result.0.request_item.value |
| SRId | JSON.Keys | result.0.request.value |
| CreationDate | JSON.Keys | result.0.sys_created_on |



Mandatory Parameter Mapping

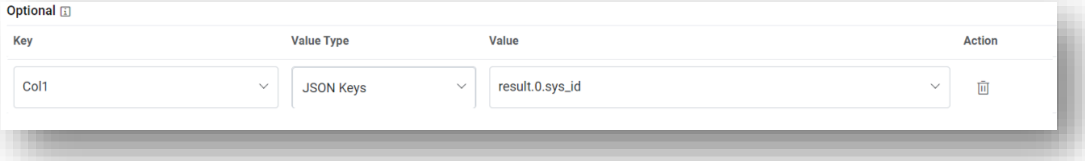
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.number |
| Summary | JSON Keys | result.0.short_description |
| Description | JSON Keys | result.0.description |
| StatusCode | JSON Keys | result.0.state |
| LastModifiedDate | JSON Keys | result.0.sys_updated_on |
| RequestItemId | JSON Keys | result.0.request_item.value |
| SRId | JSON Keys | result.0.request.value |
| CreationDate | JSON Keys | result.0.sys_created_on |

Figure 87 – Mandatory Parameter Mapping (Service Request Task)

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 16– Sample Optional Mapping Parameters – Service Request Tasks

| Key | Value Type | Value |
|---|---|---|
| Col1 | JSON.Keys | result.sys_id |



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| Col1 | JSON Keys | result.0.sys_id | 🗑 |

Figure 88 – Optional Parameter Mapping (Service Request Task)

6. Click Next to proceed to Release Rules.
7. On **the Release Rules** tab, type in the details as per the requirement.
8. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>.service-now.com/api/now/table/sc_task/#incident#
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- **Request Method** – Select Request Method as PUT from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.

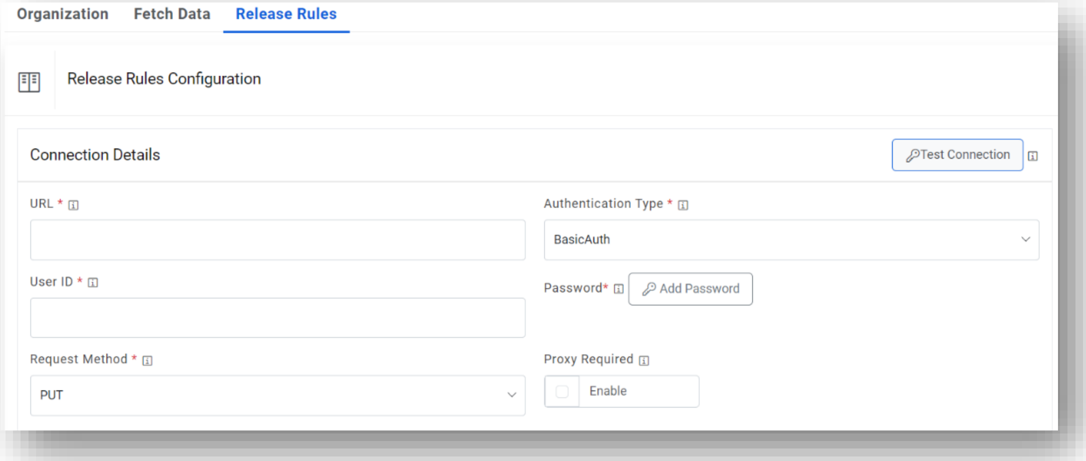
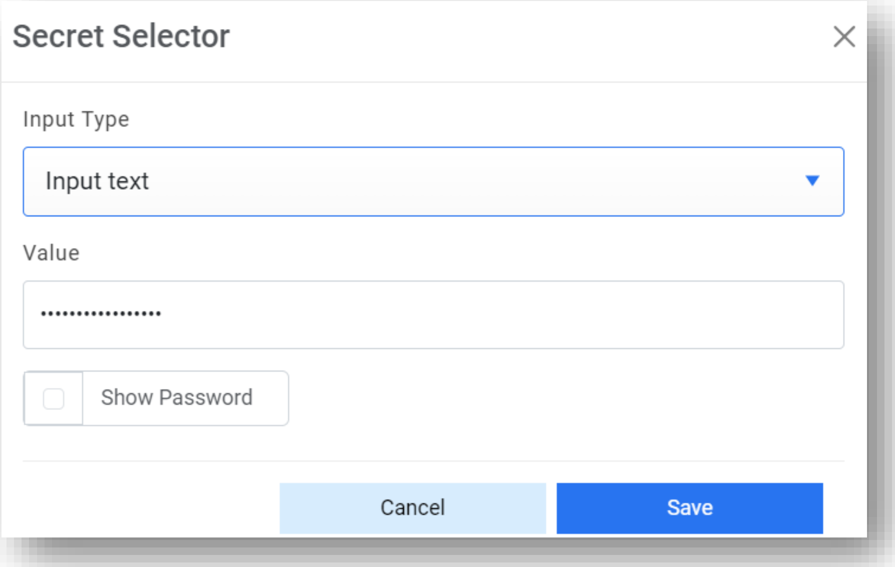


Figure 89 – Release Rules – Service Request Tasks (Connection Details)

- **Password** - For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.

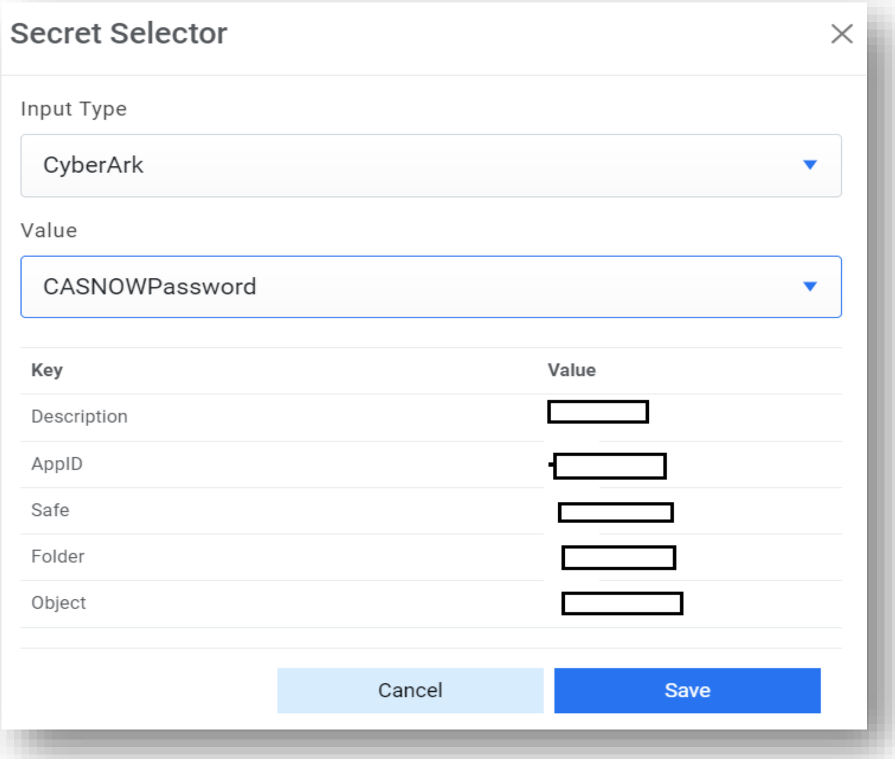


Input Type: Input text

Value: ••••••••••••••••

□ Show Password

Figure 90 - Password in plaintext

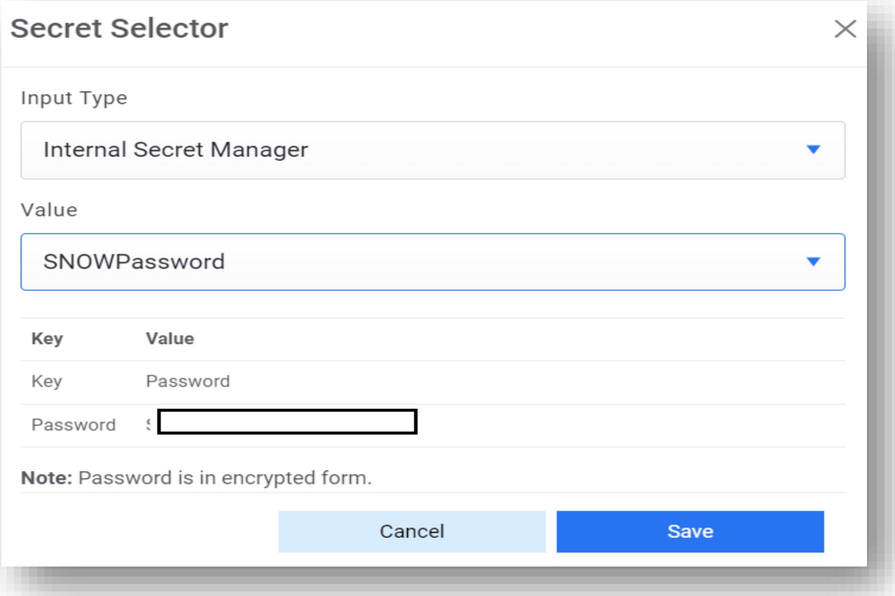


Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 91 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

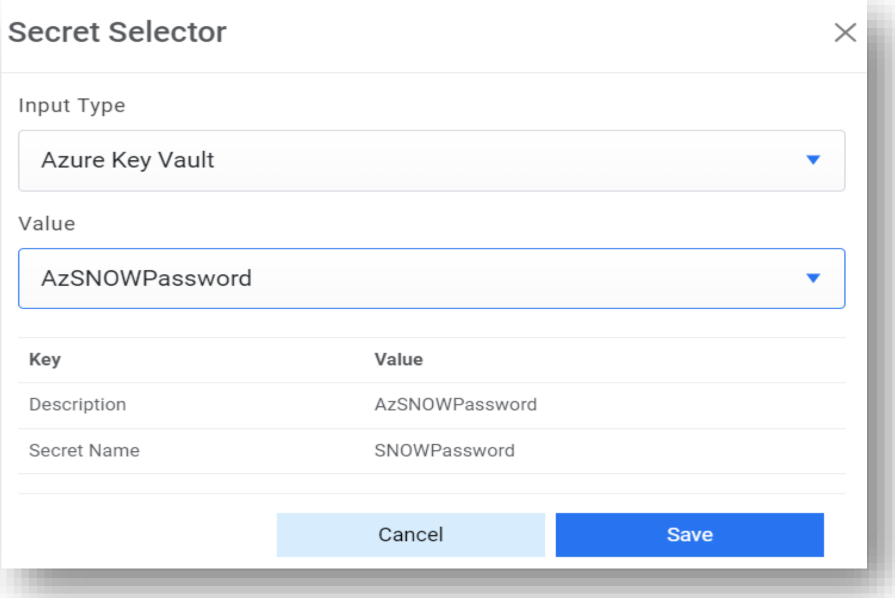
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 92 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

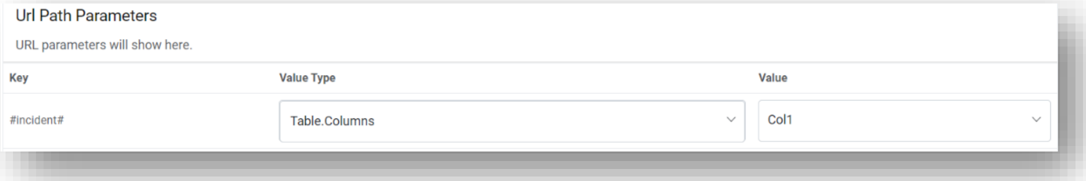
AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 93 - Password from Azure Key Vault

- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #incident#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
"Col2"
```



Url Path Parameters

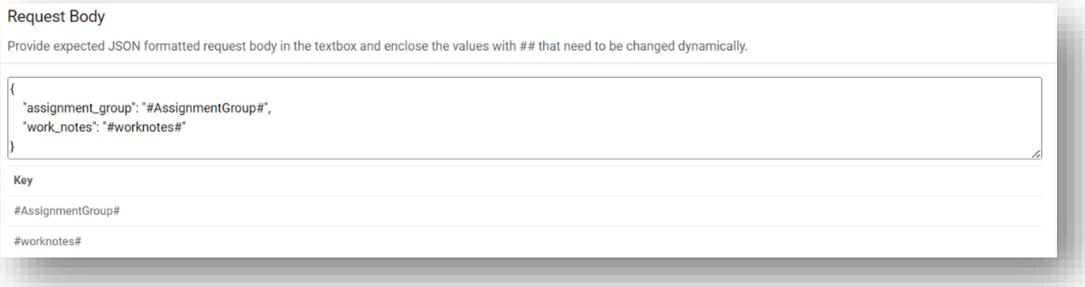
URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #incident# | Table.Columns | Col1 |

Figure 94 – Release Rules – Service Request Tasks (URL Path Parameters)

- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –
{  "assignment_group"  :  "#AssignmentGroup#","work_notes"  :
"#work_notes#" }
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

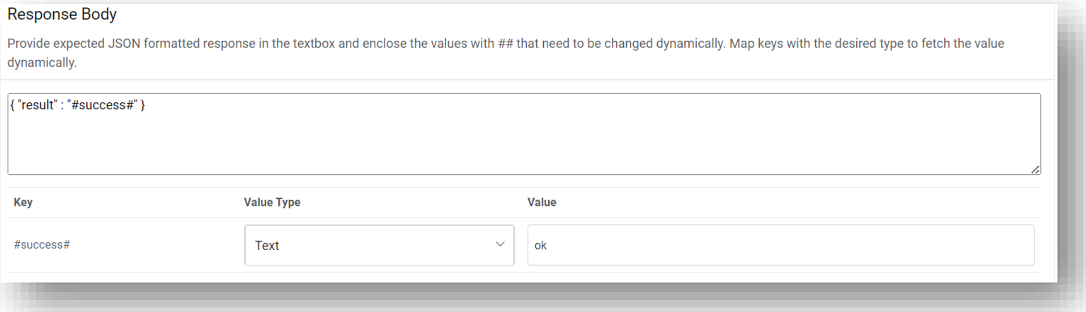
```
{
  "assignment_group": "#AssignmentGroup#",
  "work_notes": "#worknotes#"
}
```

| Key |
|---|
| #AssignmentGroup# |
| #worknotes# |

Figure 95 – Release Rules – Service Request Tasks (Request Body)

- **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below:

```
Response Body –
{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

Figure 96 – Release Rules – Service Request Tasks (Response Body)

- **Response Key Value** mapping can be done as per the below table:

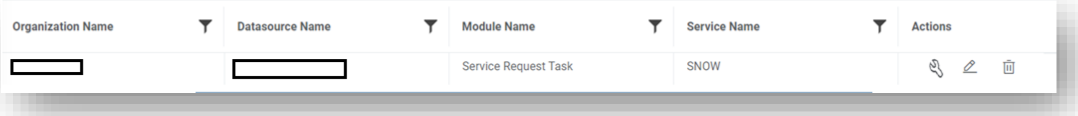
Table 17– Sample Response Key Value Mapping – Service Request Tasks

| | | |
|---|---|---|
| #success# | Text | OK |

9. Click **Save** to add the data source.

To bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps –

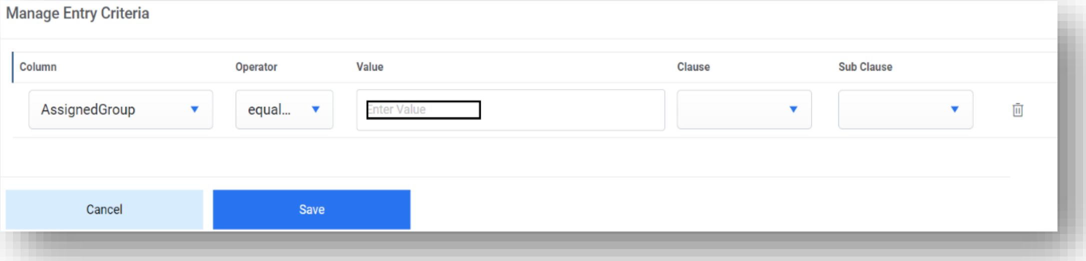
1. Go to Configuration tab and click **Manage Data Sources**.
2. On the **Data Sources** tab, click 🔧 next to the data source user wants to manage. **Manage Entry Criteria** screen appears.



| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| [redacted] | [redacted] | Service Request Task | SNOW | 🔧 ✎ 🗑 |

Figure 97 – Manage Entry Criteria (Service Request Task)

3. Select 'AssignedGroup' for the **Column** field and 'equals to' for the **Operator** field.
4. Enter the sys_id of the assignment group in ServiceNow in the **Value** field.
5. **Clause** and **Sub-Clause** fields can also be added based on requirement.



Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | Enter Value | | |

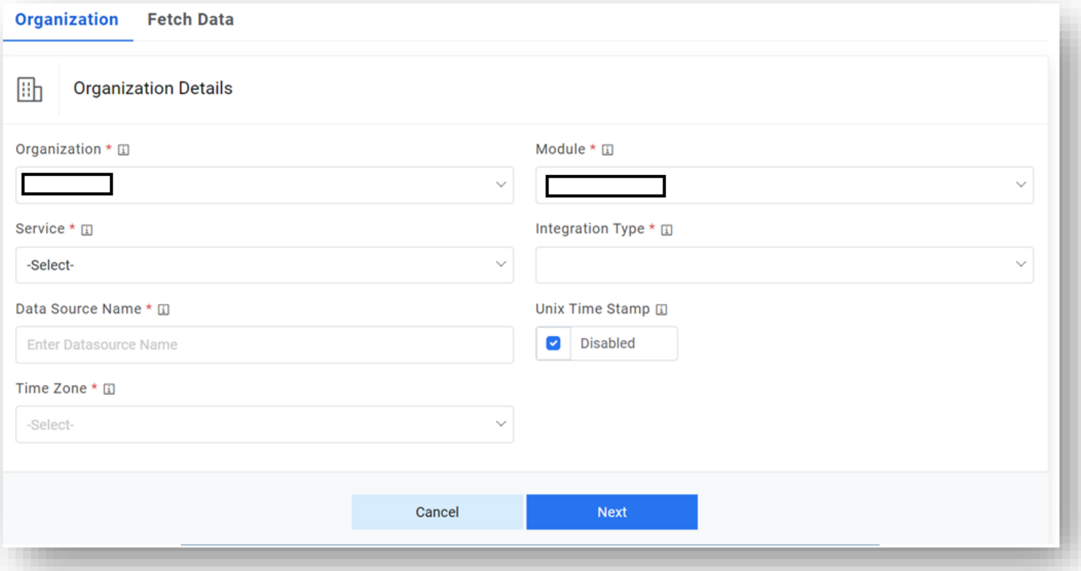
Figure 98 – Manage Entry Criteria (Service Request Task) cont.

6. Click **Save**.

### 4.2.2.3 Create Data Source for Service Request Item

To create a data source for Service Requests Items, perform the following steps:

1. On the left menu bar, click Configuration -> **Manage Data Sources**.
2. The **Create Data Source** page appears with the following tabs:
   - Organization
   - Fetch Data



Organization Details

Organization * [redacted]

Module * [redacted]

Service * -Select-

Integration Type *

Data Source Name * Enter Datasource Name

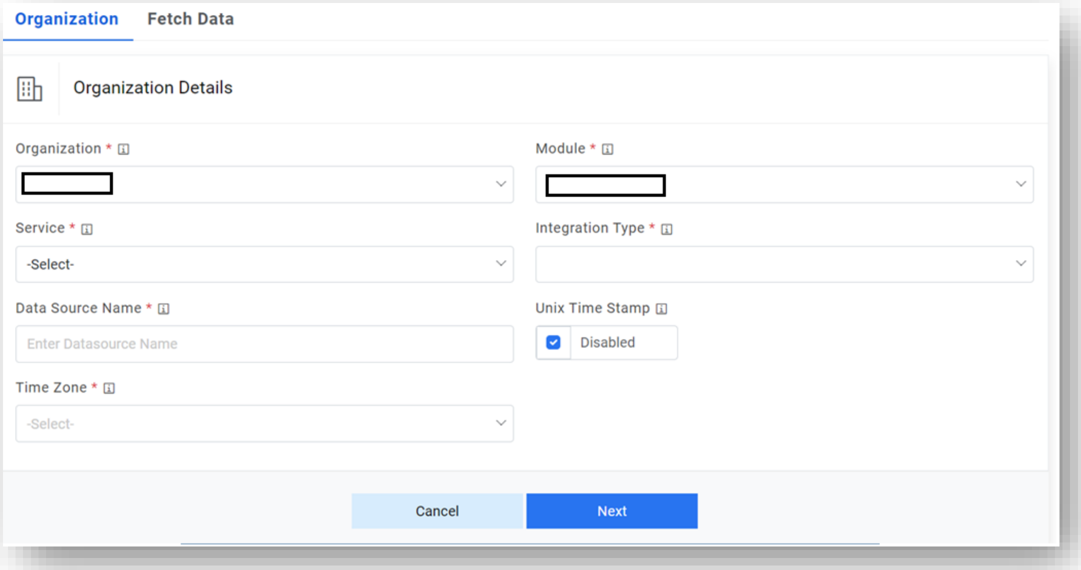
Unix Time Stamp ☑ Disabled

Time Zone * -Select-

Figure 99 - Create Data Source – Service Request Item

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab:
   - Select the **Organization Name** from the dropdown.
   - In the **Module** field, select 'Service Request Item', since we are using this data source for using its field value for the Service Request Tasks.
   - In the **Service** field, select 'Service Now Tool' as we are configuring the data source for ServiceNow.
   - In the **Integration Type** field, select 'REST API', since we will be integrating through REST APIs.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select Unix **Timestamp** to view the present data with date and time
   - Click **Next**.



Organization | Fetch Data

Organization Details

Organization * [redacted]

Module * [redacted]

Service * -Select-

Integration Type *

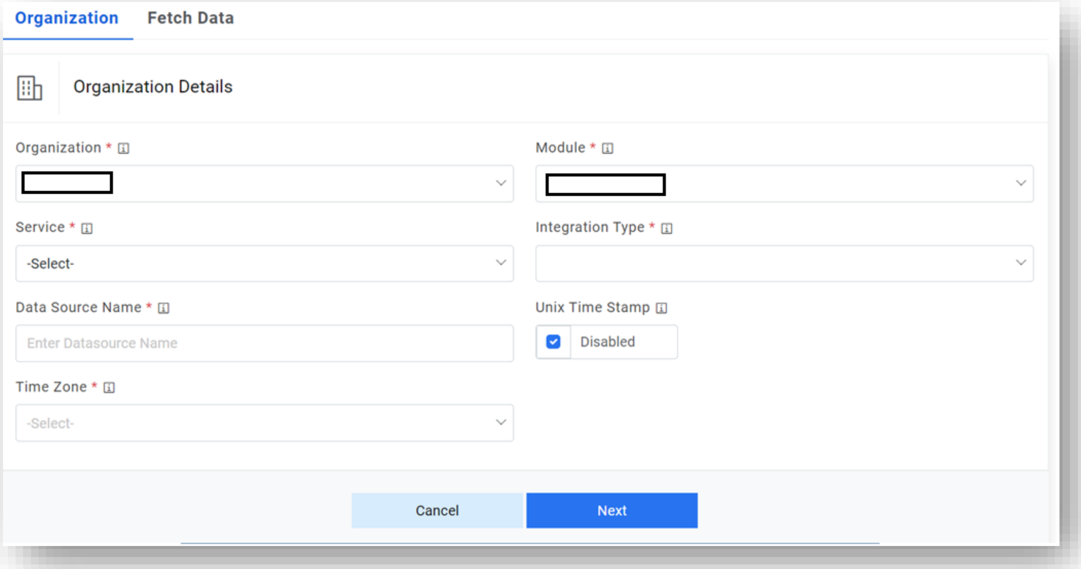
Data Source Name * Enter Datasource Name

Unix Time Stamp ☑ Disabled

Time Zone * -Select-

Cancel | Next

Figure 100 – Create Data Source – Service Request Item (cont.)



Organization | Fetch Data

Organization Details

Organization * [redacted]

Module * [redacted]

Service * -Select-

Integration Type *

Data Source Name * Enter Datasource Name

Unix Time Stamp ☑ Disabled

Time Zone * -Select-

Cancel | Next

Figure 101 – Create Data Source – Service Request Item (cont.)

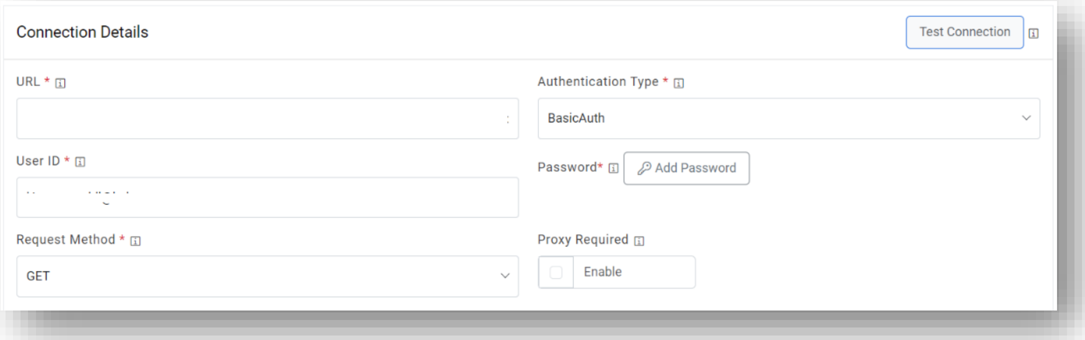
4. On the **Fetch Data** tab, type in the details as per the environment.
5. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL** – https://<url>?sysparm_fields=#Columns#&sysparm_query=sys_updated_on>=#StartDate#^sys_updated_on<=#EndDate#^ORDERBYsys_updated_on
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0
- Selection of **Basic / Windows** requires you to enter –
    - User Id

  - Password.

- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL
- **Request Method –** Select the request method as **GET**, **POST** or **PUT** as per the configured URL.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

Test Connection

URL *

Authentication Type * BasicAuth

User ID *

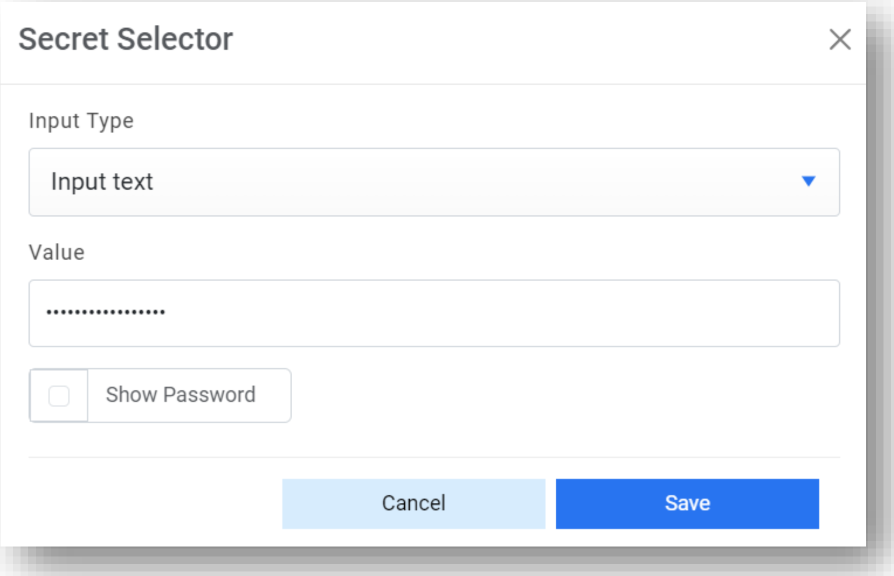
Password* Add Password

Request Method * GET

Proxy Required □ Enable

Figure 102 – Create Data Source – Service Request Item (Connection Details)

- **Password** - For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



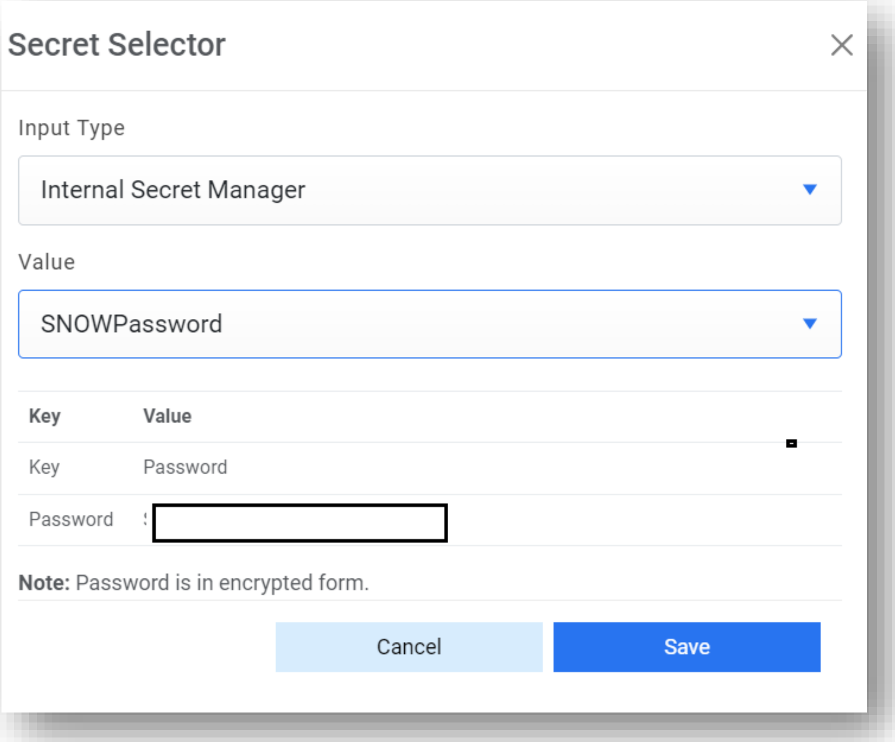
Secret Selector

Input Type: Input text

Value: ••••••••••••••••

□ Show Password

Figure 103 - Password in plaintext



Secret Selector

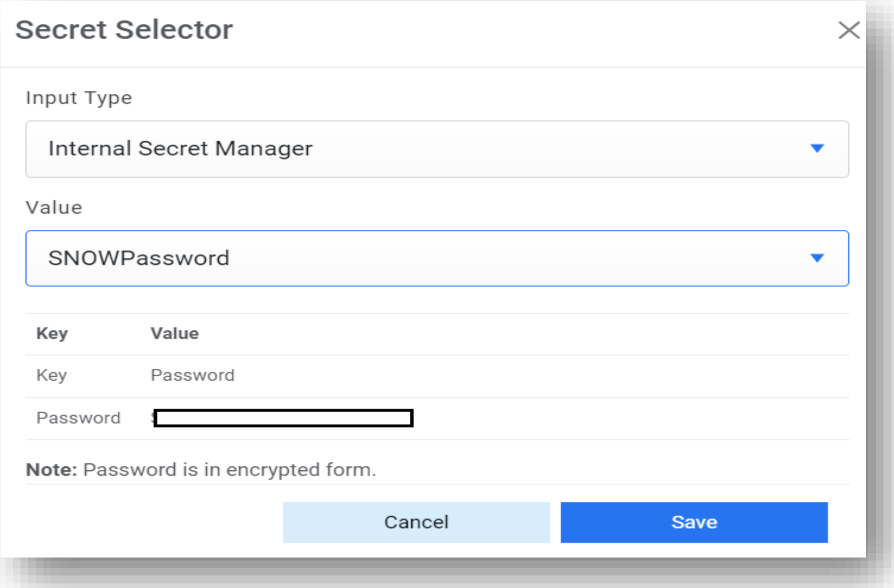
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 104 - Password from Key Vault (CyberArk)



Secret Selector

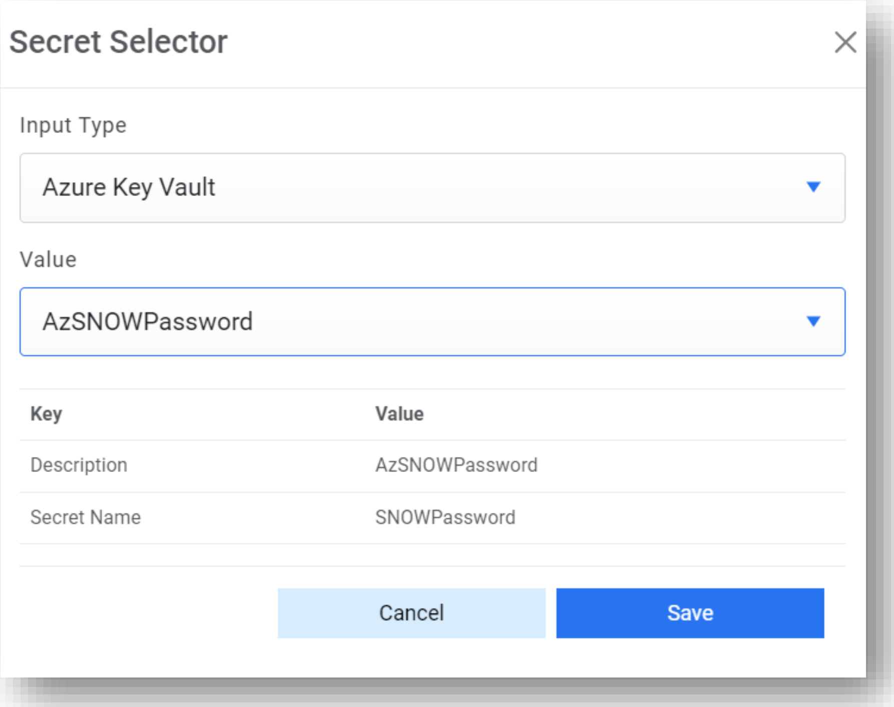
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 105 - Password from Secret Manager



Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

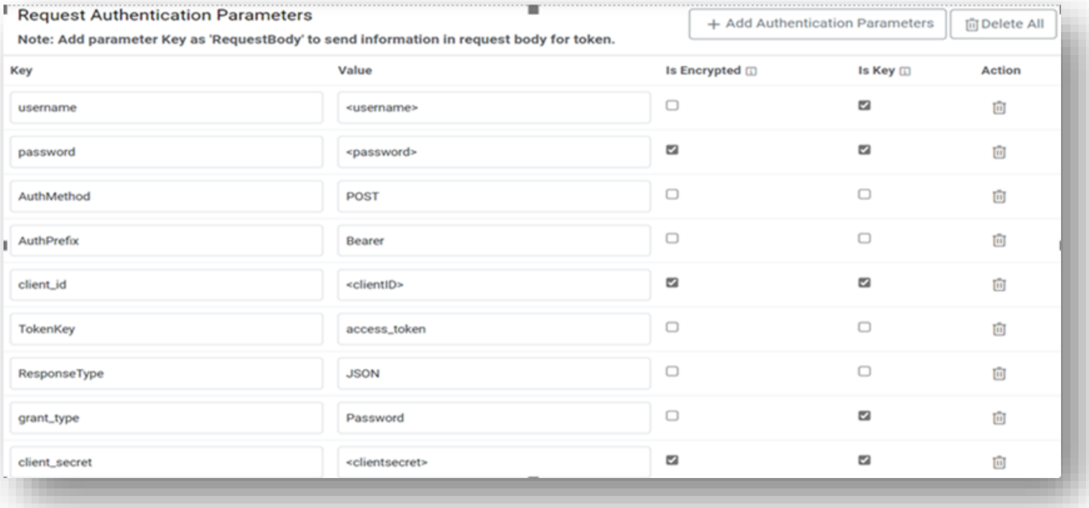
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 106 - Password from Azure Key Vault

- **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
- Based on the **Authentication Type**, add the parameters mentioned in the below table:

Table 18 – Sample Authentication Parameters – Service Request Item

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

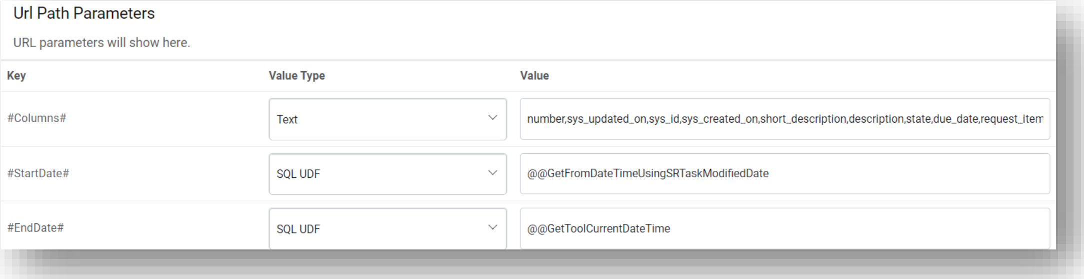
| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | ☐ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | ☐ | ☐ | 🗑 |
| AuthPrefix | Bearer | ☐ | ☐ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | ☐ | ☐ | 🗑 |
| ResponseType | JSON | ☐ | ☐ | 🗑 |
| grant_type | Password | ☐ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 107 – Create Data Source – Service Request Item (Request Authentication Parameters for OAuth2.0)

- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Columns#

ValueType: Text

Value:

number,sys_updated_on,sys_id,sys_created_on,short_description,d
escription,state,request,approval


Note – These columns are mandatory. User can add more columns if
more data is required to be fetched from ITSM tool.


Key: #StartDate#

ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingiRequestItemModifiedDate
```

```
Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,sys_id,sys_created_on,short_description,description,state,due_date,request_item |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingSRTaskModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 108 – URL Path Parameters – Service Request Item (Service Request Task Management)

- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body –** Enter the request body in JSON format as per the configured URL, if applicable.
- **Response Body –** In this section, please enter the output of URL query for one of the service requests tasks in JSON format. A sample response is mentioned below:

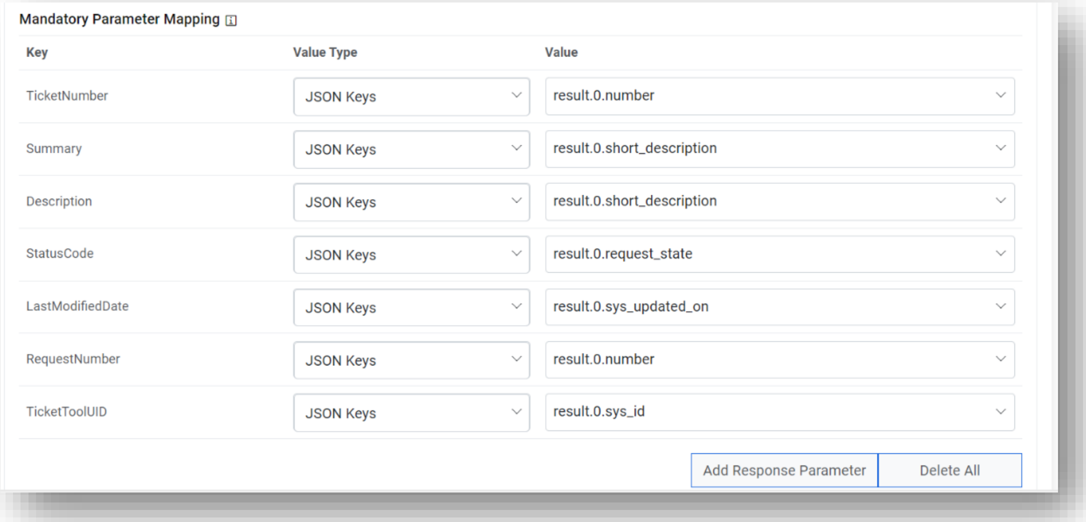
```
Response Body –

{

   "result": {

      "number": "RITM0011964",

      "sys_id": "xxxxxxxx",

      "short_description": "Can't find the right request?TEST",

      "request": {

         "link":"https://my_host ",

         "value": "2ae764d5db199c14e3bbde06f496195a"

      },

      "sys_created_on": "2020-06-08 10:34:54",

      "approval": "approved",

      "description": "Test",

      "sys_updated_on": "2020-06-08 10:35:17",

      "state": "2"
```

```
        }

}
```

- After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
- **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 19– Sample Mandatory Mapping Parameters – Service Request Item

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | Result.0.number |
| Summary | JSON.Keys | Result.0.short_description |
| Description | JSON.Keys | Result.0.description |
| StatusCode | JSON.Keys | Result.0.state |
| LastModifiedDate | JSON.Keys | Result.0.sys_updated_on |
| RequestNumber | JSON.Keys | Result.0.number |
| TicketToolUID | JSON.Keys | Result.0.sys_id |



Mandatory Parameter Mapping

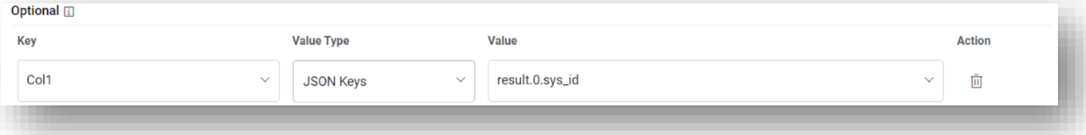
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.number |
| Summary | JSON Keys | result.0.short_description |
| Description | JSON Keys | result.0.short_description |
| StatusCode | JSON Keys | result.0.request_state |
| LastModifiedDate | JSON Keys | result.0.sys_updated_on |
| RequestNumber | JSON Keys | result.0.number |
| TicketToolUID | JSON Keys | result.0.sys_id |

Figure 109 – Mandatory Parameter Mapping (Service Request Item)

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 20 – Sample Optional Mapping Parameters – Service Request Item

| Key | Value Type | Value |
|---|---|---|
| Col1 | JSON.Keys | Result.0.sys_id |



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| Col1 | JSON Keys | result.0.sys_id | |

Figure 110 – Optional Parameter Mapping (Service Request Item)

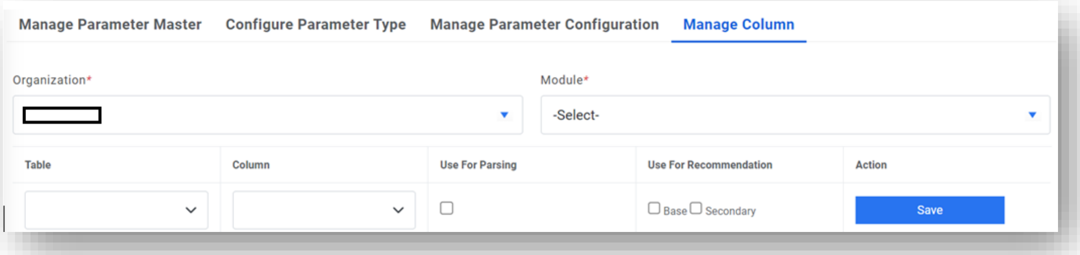
6. Click **Save** to add the data source.

### 4.2.2.4 Configuration of Additional Parameters for Recommendation and Parsing

To use the field values of Service Request and Service Request Item for the purpose of Recommendation and Parsing by iAutomate services, they need to be mapped to Service Request Task.

To do so, perform the following steps -

1. On the left menu bar, click Advance Configuration -> Parameter -> Manage Column.



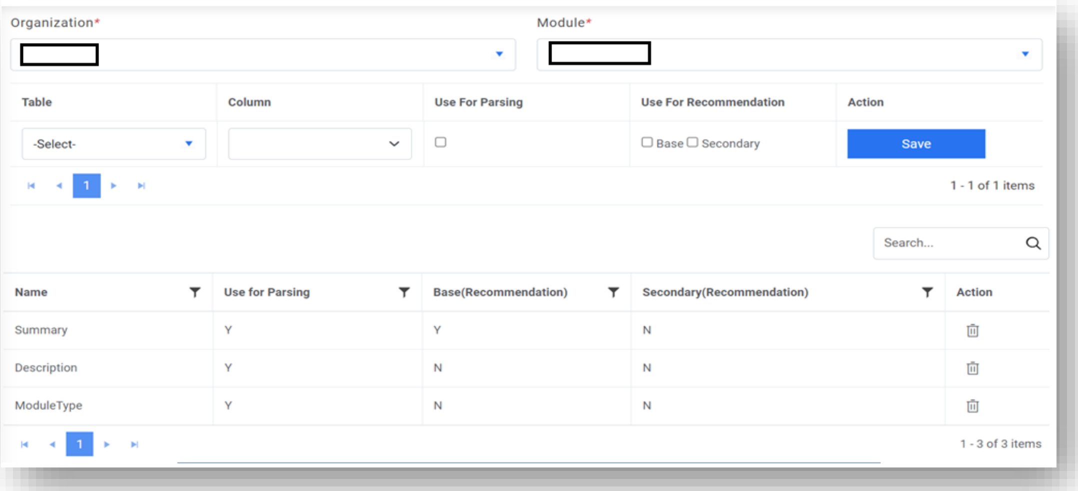
Organization* [redacted]

Module* -Select-

| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| | | ☐ | ☐ Base ☐ Secondary | Save |

Figure 111 – Map fields of Service Request and Service Request Item to Service Request Task

2. Select **Organization Name** from dropdown. Select Service Request Task as the **Module**.



Organization* [redacted]

Module* [redacted]

| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| -Select- | | ☐ | ☐ Base ☐ Secondary | Save |

1 - 1 of 1 items

Search...

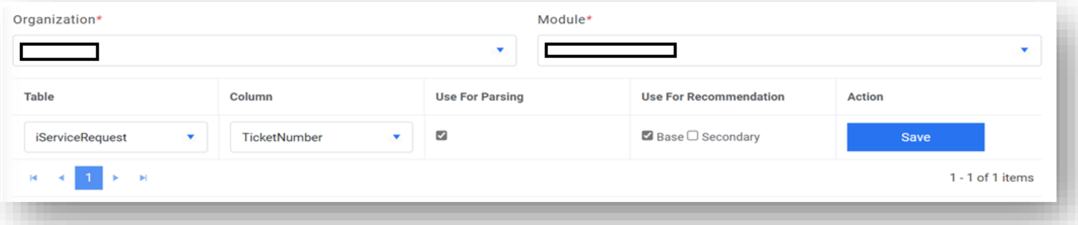
| Name | Use for Parsing | Base(Recommendation) | Secondary(Recommendation) | Action |
|---|---|---|---|---|
| Summary | Y | Y | N | |
| Description | Y | N | N | |
| ModuleType | Y | N | N | |

1 - 3 of 3 items

Figure 112 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

Summary, Description, RunbookToolTenantID, ModuleType are the default entries.

3. To map the column of Service Request, select **iServiceRequest** in Table dropdown.
4. Select the column of Service Request which has to be mapped to Service Request Task in the Column dropdown. In this case, we are selecting **TicketNumber**.
5. Check the fields **Use For Parsing** and select 'Base' in **Use For Recommendation**.



Organization*

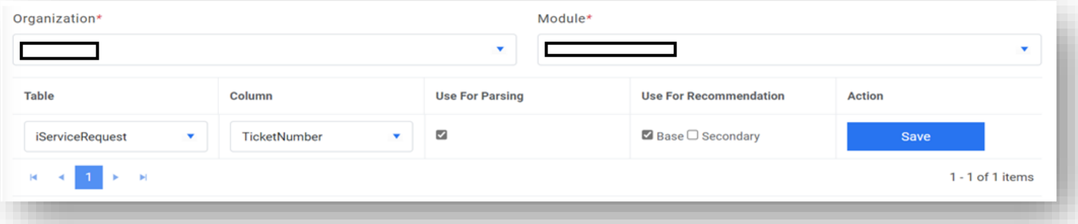
Module*

| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| iServiceRequest | TicketNumber | ☑ | ☑ Base ☐ Secondary | Save |

1 - 1 of 1 items

Figure 113 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

6. Click **Save.**
7. To map the column of Service Request Item, select **iRequestItem** in Table dropdown.
8. Select the column of Service Request Item which has to be mapped to Service Request Task in the Column dropdown. In this case, we are selecting **Summary**.
9. Check the fields **Use For Parsing** and select 'Base' in **Use For Recommendation**.



Organization*

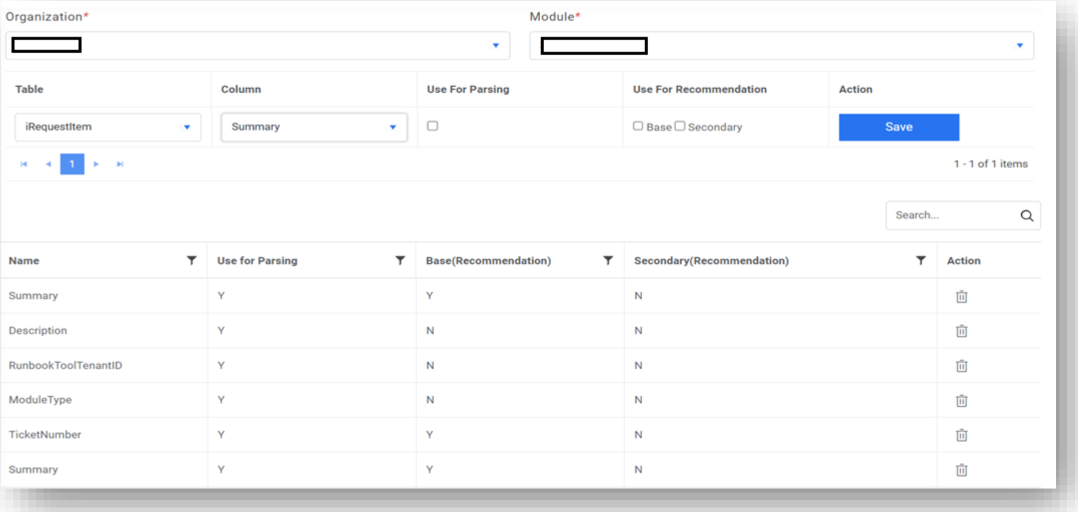
Module*

| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| iServiceRequest | TicketNumber | ☑ | ☑ Base ☐ Secondary | Save |

1 - 1 of 1 items

Figure 114 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

10. Click **Save**. The page lists two additional entries, TicketNumber and **Summary**, as depicted below.



Organization* [redacted]

Module* [redacted]

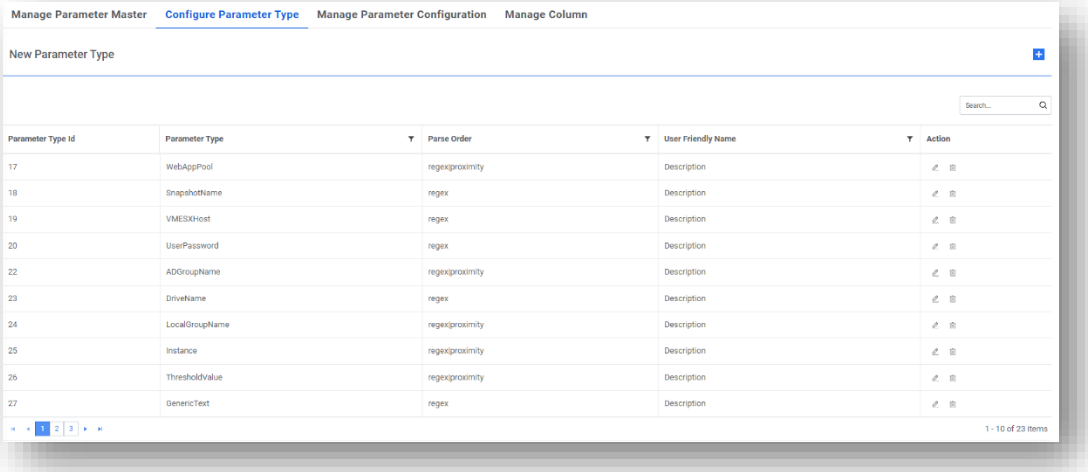
| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| iRequestItem | Summary | □ | □ Base □ Secondary | Save |

1 - 1 of 1 items

| Name | Use for Parsing | Base(Recommendation) | Secondary(Recommendation) | Action |
|---|---|---|---|---|
| Summary | Y | Y | N | 🗑 |
| Description | Y | N | N | 🗑 |
| RunbookToolTenantID | Y | N | N | 🗑 |
| ModuleType | Y | N | N | 🗑 |
| TicketNumber | Y | Y | N | 🗑 |
| Summary | Y | Y | N | 🗑 |

Figure 115 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

11. For Recommendation, the above steps are sufficient. But for Parsing, additional steps are required to be performed.
12. On the left menu bar, click Advance Configuration.
13. Click **Configure Parameter Type**. By default, there are several entries already defined.

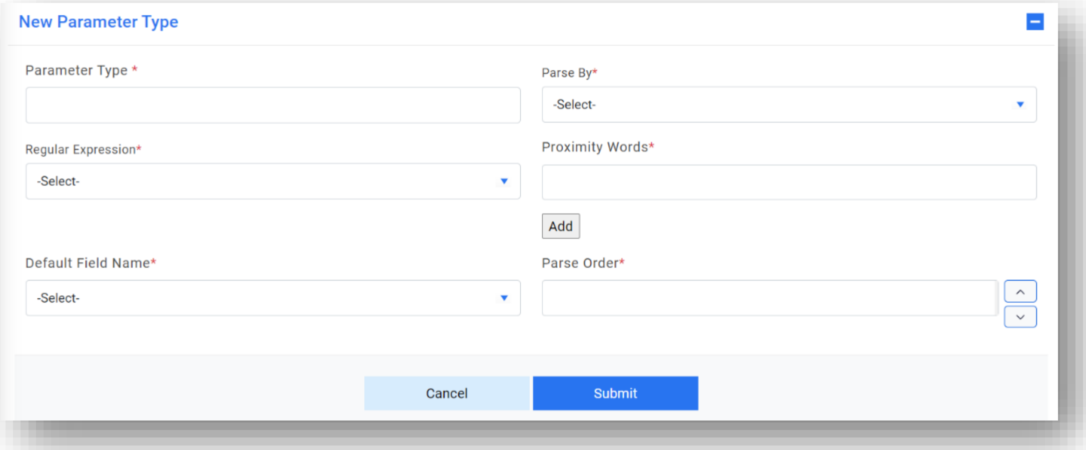


| Parameter Type Id | Parameter Type | Parse Order | User Friendly Name | Action |
|---|---|---|---|---|
| 17 | WebAppPool | regex\|proximity | Description | ✎ 🗑 |
| 18 | SnapshotName | regex | Description | ✎ 🗑 |
| 19 | VMESXHost | regex | Description | ✎ 🗑 |
| 20 | UserPassword | regex | Description | ✎ 🗑 |
| 22 | ADGroupName | regex\|proximity | Description | ✎ 🗑 |
| 23 | DriveName | regex | Description | ✎ 🗑 |
| 24 | LocalGroupName | regex\|proximity | Description | ✎ 🗑 |
| 25 | Instance | regex\|proximity | Description | ✎ 🗑 |
| 26 | ThresholdValue | regex\|proximity | Description | ✎ 🗑 |
| 27 | GenericText | regex | Description | ✎ 🗑 |

1 - 10 of 23 items

Figure 116 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

14. Click ➕ Icon to **Add New**.

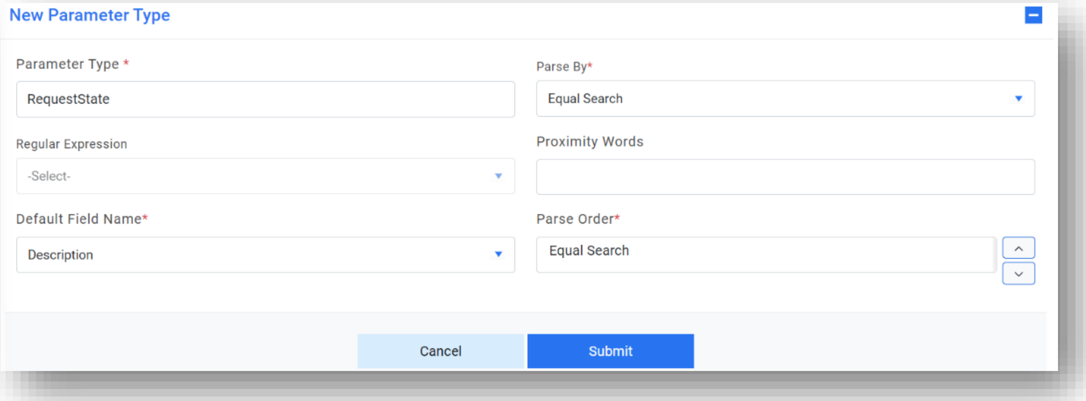


New Parameter Type

| | |
|---|---|
| Parameter Type * | Parse By* |
| | -Select- |
| Regular Expression* | Proximity Words* |
| -Select- | |
| | Add |
| Default Field Name* | Parse Order* |
| -Select- | |

Figure 117 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

15. Mention **Parameter Type** for Service Request column, for e.g. **RequestState**
16. Select 'Equal Search' in the **Parse By** field.
17. Select 'Description' in the **Default Field Name** field.
18. Click **Submit**.

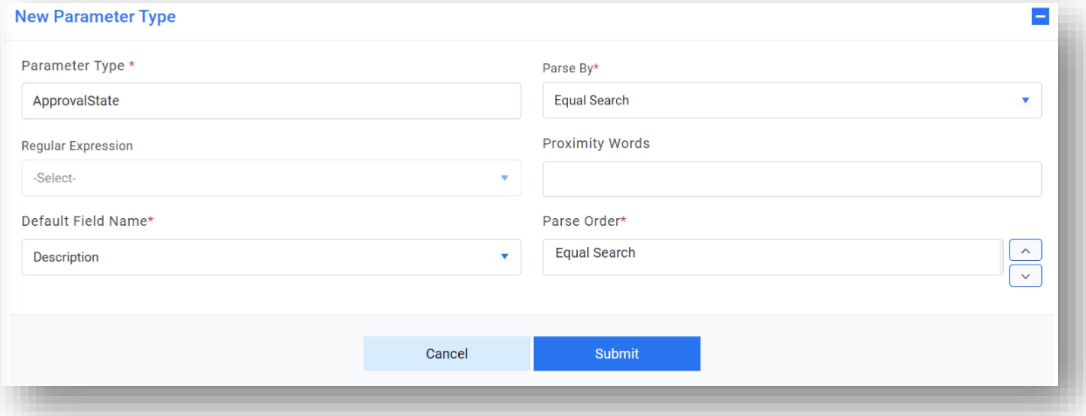


New Parameter Type

| | |
|---|---|
| Parameter Type * | Parse By* |
| RequestState | Equal Search |
| Regular Expression | Proximity Words |
| -Select- | |
| Default Field Name* | Parse Order* |
| Description | Equal Search |

Figure 118 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

19. Click **Add New**.
20. Mention **Parameter Type** for Service Request Item column, for e.g. **ApprovalState.**
21. In the **Parse By** field, select 'Equal Search'.
22. In the **Default Field Name** field, select 'Description'.
23. Click **Submit**.



New Parameter Type

Parameter Type *: ApprovalState

Parse By*: Equal Search

Regular Expression: -Select-

Proximity Words:

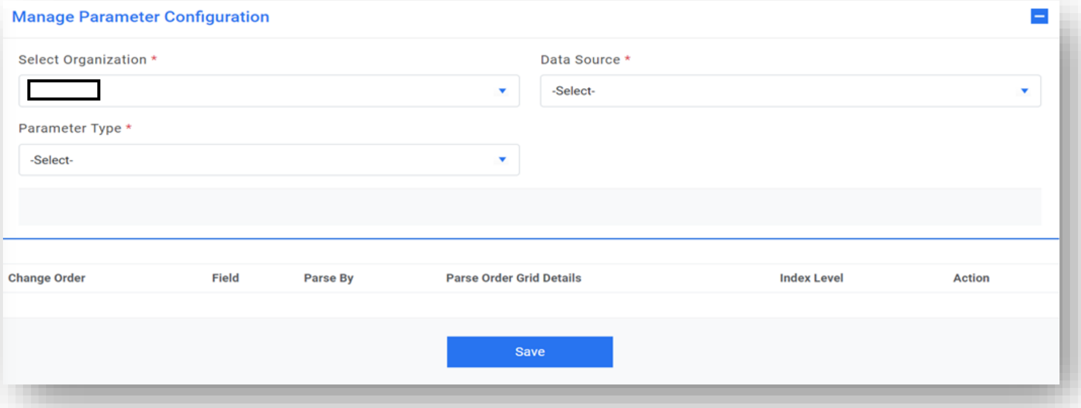
Default Field Name*: Description

Parse Order*: Equal Search

Figure 119 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

24. Next step is to map this **Parameter Type** i.e. 'RequestState' and 'ApprovalState', to the one that was created via **Manage Columns** in earlier step by the name 'TicketNumber' and 'Summary', respectively. To do that, perform the following steps:

1. On the main menu bar, click Advance Configurations Parameter.
2. Click Manage Parameter Configuration.



Manage Parameter Configuration

Select Organization *:

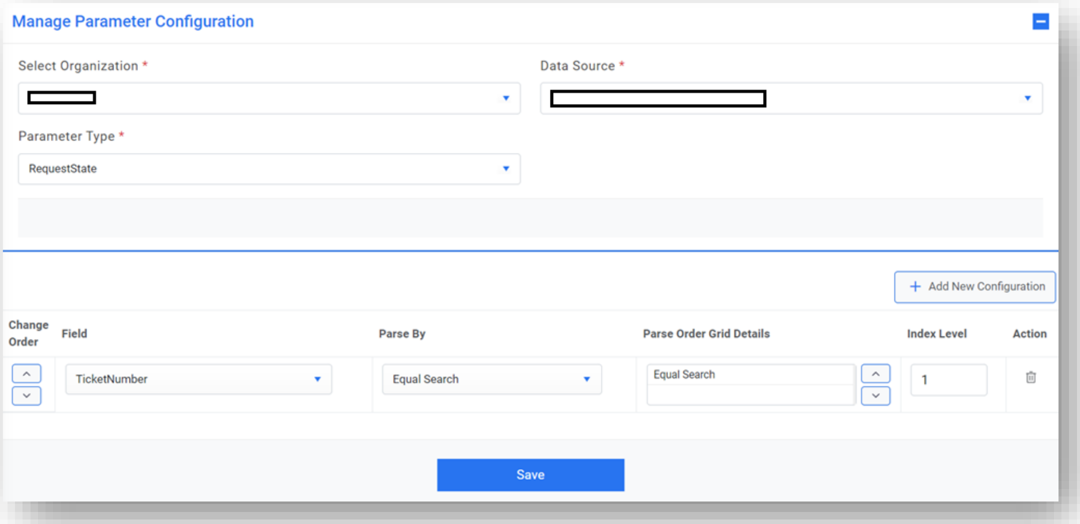
Data Source *: -Select-

Parameter Type *: -Select-

| Change Order | Field | Parse By | Parse Order Grid Details | Index Level | Action |
|---|---|---|---|---|---|

Figure 120 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

3. Selection **Organization**.
4. Select relevant 'Service Request Task' as the **Data Source.**
5. Select the newly created parameter **RequestState** from **Parameter Type** dropdown**.**
6. From the **Field** dropdown, select 'TicketNumber', the parameter that has been mapped via **Manage Columns.**



Select Organization * [redacted]

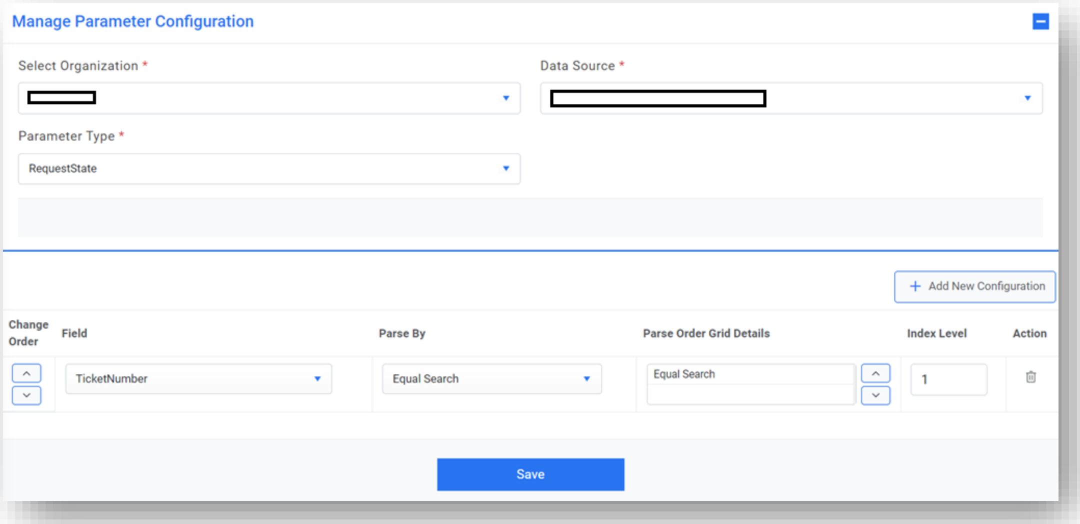
Data Source * [redacted]

Parameter Type * RequestState

| Change Order | Field | Parse By | Parse Order Grid Details | Index Level | Action |
|---|---|---|---|---|---|
| | TicketNumber | Equal Search | Equal Search | 1 | |

Figure 121 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

7. Click **Save**.
8. Selection Organization.
9. Select relevant 'Service Request Task' as the **Data Source.**
10. Select the newly created parameter i.e. 'ApprovalState' from **Parameter Type** dropdown.
11. From the **Field** dropdown, select 'Summary', the parameter that has been mapped via **Manage Columns.**
12. Click **Save**.



Select Organization * [redacted]

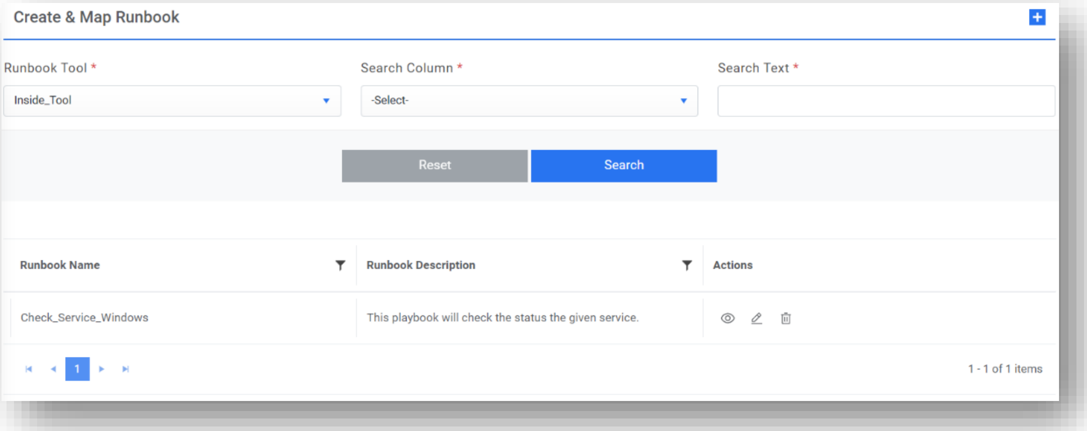
Data Source * [redacted]

Parameter Type * RequestState

| Change Order | Field | Parse By | Parse Order Grid Details | Index Level | Action |
|---|---|---|---|---|---|
| | TicketNumber | Equal Search | Equal Search | 1 | |

Figure 122 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

13. To verify whether this parameter is successfully parsed or not, perform the following steps –
    - On the left menu bar, click **Runbooks**.
    - Click Manage Runbooks.
    - Select the **Runbook Tool** mapped with the organization.



Create & Map Runbook

Runbook Tool *: Inside_Tool

Search Column *: -Select-

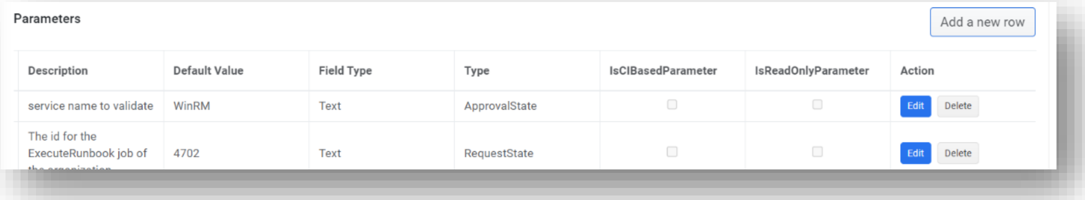
Search Text *:

| Runbook Name | Runbook Description | Actions |
|---|---|---|
| Check_Service_Windows | This playbook will check the status the given service. | |

Figure 123 – Map fields of Service Request and Service Request Item to Service Request Task (cont.)

The parameters, **RequestState** and **ApprovalState**, which were created in earlier steps, have to be added as the parameters to the existing runbook. You can also create a new runbook with **RequestState** and **ApprovalState** as the parameters.

14. Click the **Edit** icon to edit the runbook.
15. In the Parameters section, add two new parameters with relevant **Parameter Name, Parameter Label, Parameter Description, Default Parameter Value**. Ensure that Parameter Type is selected as **RequestState** and **ApprovalState** respectively.

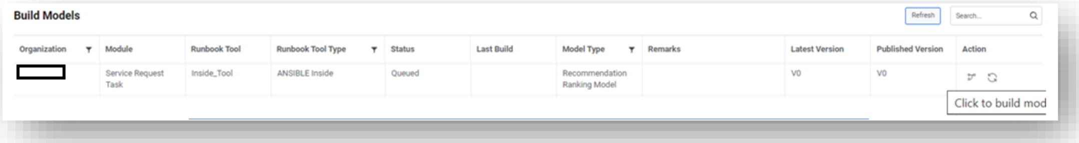


Parameters

| Description | Default Value | Field Type | Type | IsCIBasedParameter | IsReadOnlyParameter | Action |
|---|---|---|---|---|---|---|
| service name to validate | WinRM | Text | ApprovalState | □ | □ | Edit Delete |
| The id for the ExecuteRunbook job of | 4702 | Text | RequestState | □ | □ | Edit Delete |

Figure 124 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

16. Add the parameters and click **Save**.
17. Ensure that the runbook in which the parameters are added is mapped with the organization.
18. Next step is to build the Recommendation model and to do that perform the following steps:
    - On the left menu bar, click **Configuration**.
    - Click Build Models.
    - ReBuild / Re-build the model for the **Organization** under **Service Request Task** module for the mapped runbook tool.

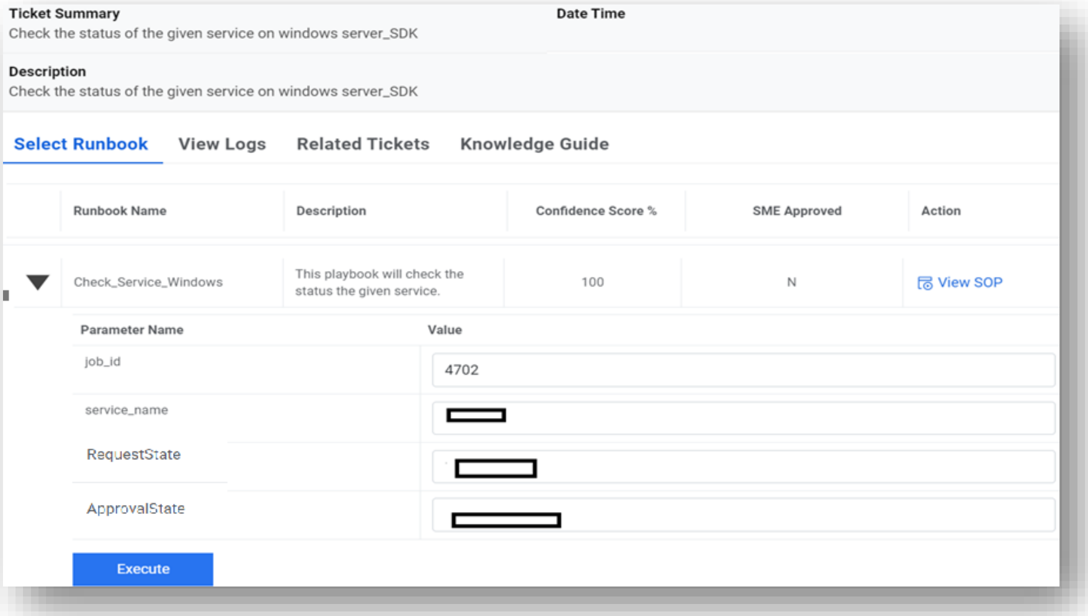


Build Models

| Organization | Module | Runbook Tool | Runbook Tool Type | Status | Last Build | Model Type | Remarks | Latest Version | Published Version | Action |
|---|---|---|---|---|---|---|---|---|---|---|
| [redacted] | Service Request Task | Inside_Tool | ANSIBLE Inside | Queued | | Recommendation Ranking Model | | V0 | V0 | |

Figure 125 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

19. Run the entire flow and see if the runbook recommended for the ticket in which the parameters were added have the parameter **RequestState** and **ApprovalState** with their expected values.



**Ticket Summary**
Check the status of the given service on windows server_SDK

**Date Time**

**Description**
Check the status of the given service on windows server_SDK

| Runbook Name | Description | Confidence Score % | SME Approved | Action |
|---|---|---|---|---|
| Check_Service_Windows | This playbook will check the status the given service. | 100 | N | View SOP |

| Parameter Name | Value |
|---|---|
| job_id | 4702 |
| service_name | [redacted] |
| RequestState | [redacted] |
| ApprovalState | [redacted] |

Figure 126 – Map fields of Service Request and Service Request Item to Service Request Task (Cont.)

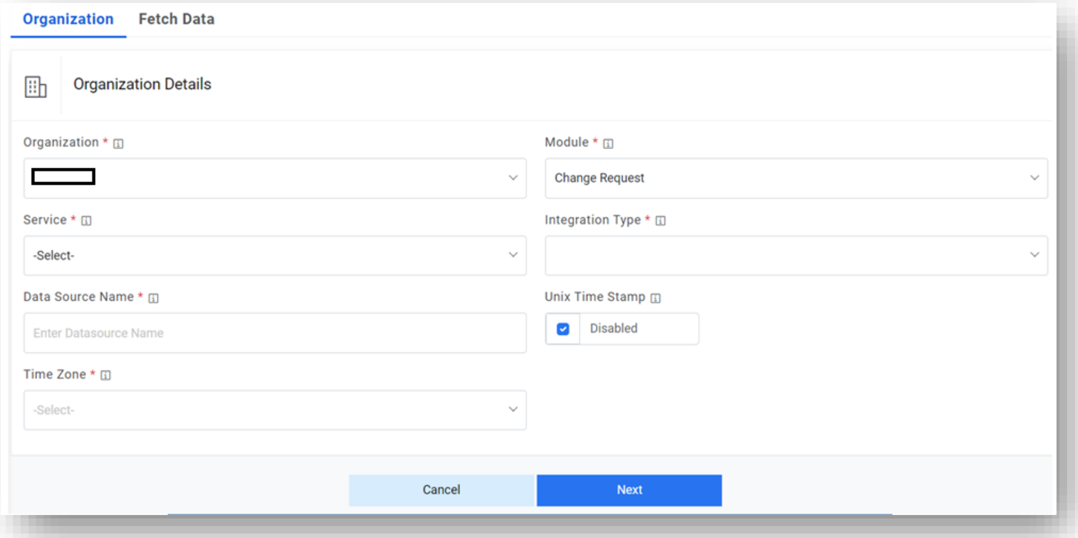
### 4.2.3 Change Request Management

To fetch information about Change Requests, usually, creation of a data source for Change Request Task should suffice. However, there could be scenarios where some additional fields / values are required from Change Request for processing the tickets – recommending the relevant runbooks and parsing the tickets to extract relevant parameters, for which separate data source for Change Request has to be created. Here, we will cover the procedure for creating both kinds of data sources.

#### 4.2.3.1 Create Data Source for Change Request

To create a data source for Change Request, perform the following steps:

1. On the left menu bar, click Configuration -> Manage Data Sources.
2. The **Create Data Source** page appears with the following tabs:
    - Organization
    - Fetch Data



Organization Details

Organization *: [redacted]

Module *: Change Request

Service *: -Select-

Integration Type *:

Data Source Name *: Enter Datasource Name

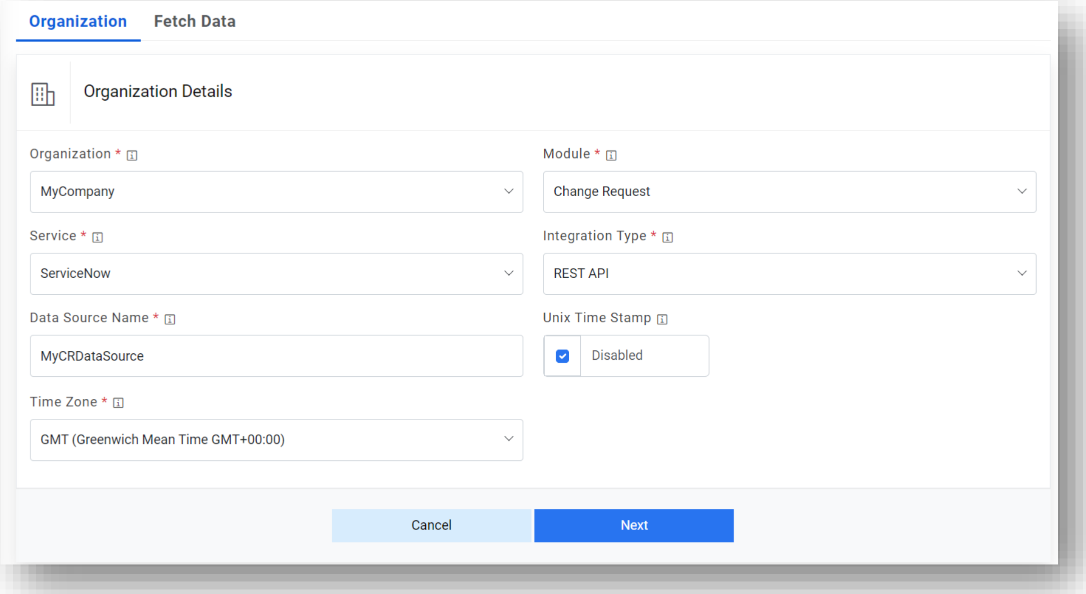
Unix Time Stamp: ☑ Disabled

Time Zone *: -Select-

Figure 127 - Create Data Source – Change Request

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab,
    - Select the **Organization Name** from the dropdown.
    - Select the **Module** as **Change Request** since we are using this data source for using its field value for the change requests.
    - Select the **Service** as **Service Now Tool** as we are configuring the data source for ServiceNow
    - Select the **Integration Type** as **REST**, since we will be integrating through REST APIs.
    - Select the **Timezone** to specify the time zone of the selected data source.
    - Select **Timestamp** to view the present data with date and time.
    - Click **Next**.



Organization * MyCompany

Module * Change Request

Service * ServiceNow

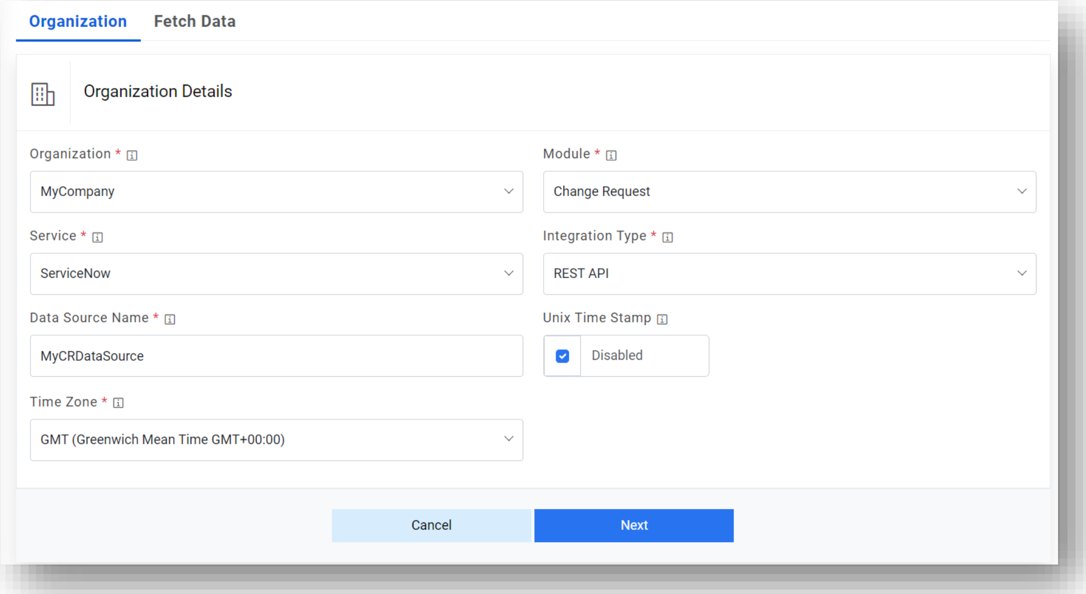
Integration Type * REST API

Data Source Name * MyCRDataSource

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 128 - Create Data Source – Change Request (Cont.)



Organization * MyCompany

Module * Change Request

Service * ServiceNow

Integration Type * REST API

Data Source Name * MyCRDataSource

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 129 - Create Data Source – Change Request (Cont.)

4. On the **Fetch Data** tab, type in the details as per the environment.
5. In the **Connection Details** section, enter the following details:

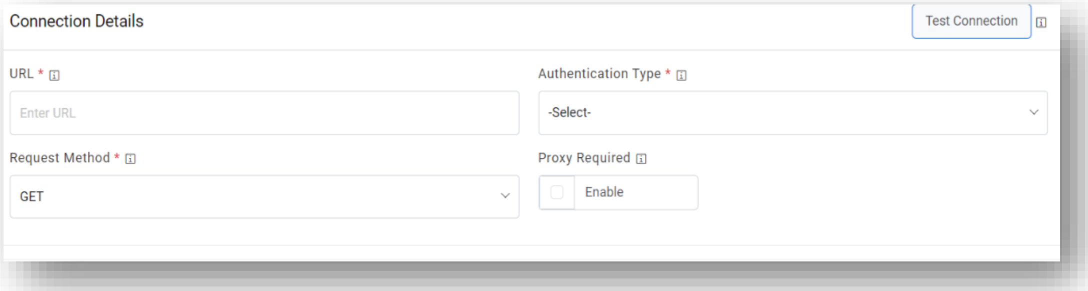
- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>?sysparm_fields=#Columns#&sysparm_query=active=true^ sys_updated_on >=#StartDate#^ sys_updated_on <=#EndDate#^ORDERBYsys_updated_on
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0
- Selection of **Basic / Windows** requires you to enter -
  - User Id

- Password

- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL

- **Request Body –** Select the request method as GET, POST or PUT as per the configured URL.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

URL * : Enter URL

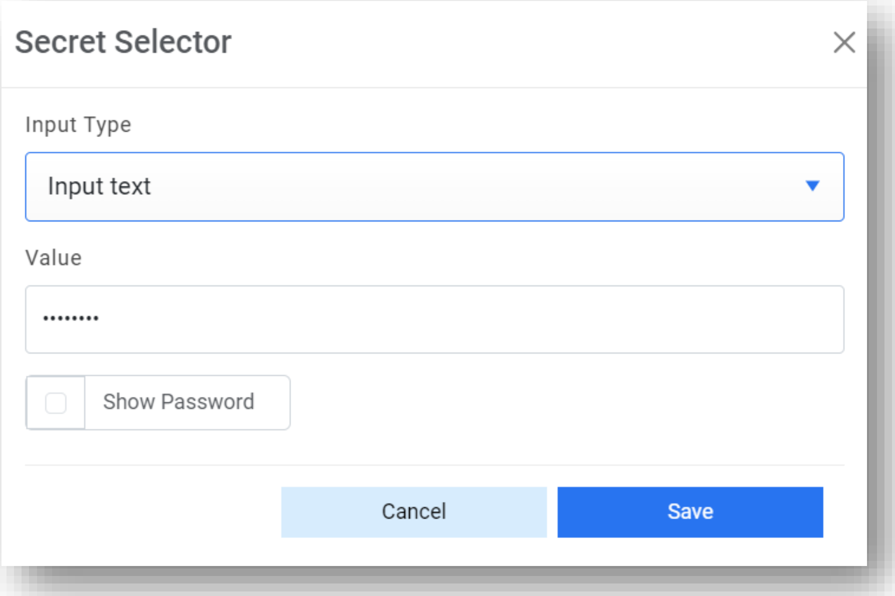
Authentication Type * : -Select-

Request Method * : GET

Proxy Required : □ Enable

Figure 130 – Create Data Source – Change Request (Connection Details)

- **Password** - For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.

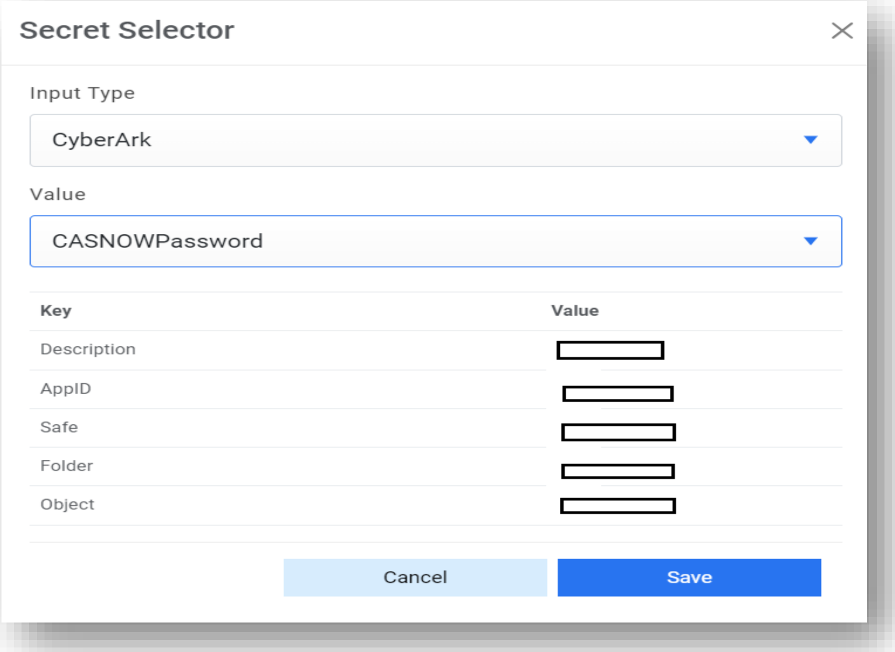


Input Type: Input text

Value: ••••••••

□ Show Password

Figure 131 - Password in Plaintext

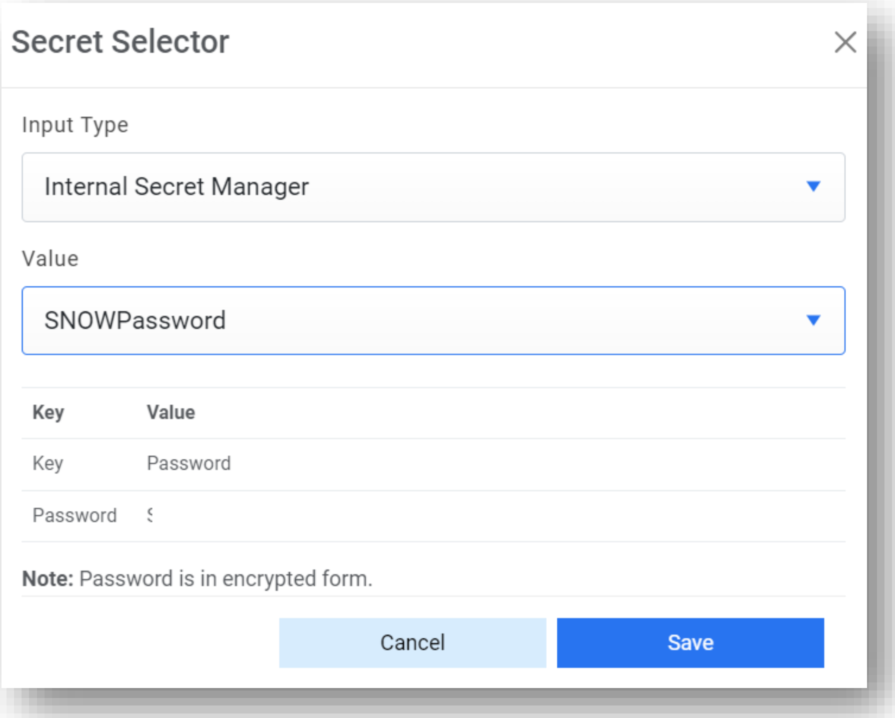


Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | |
| AppID | |
| Safe | |
| Folder | |
| Object | |

Figure 132 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

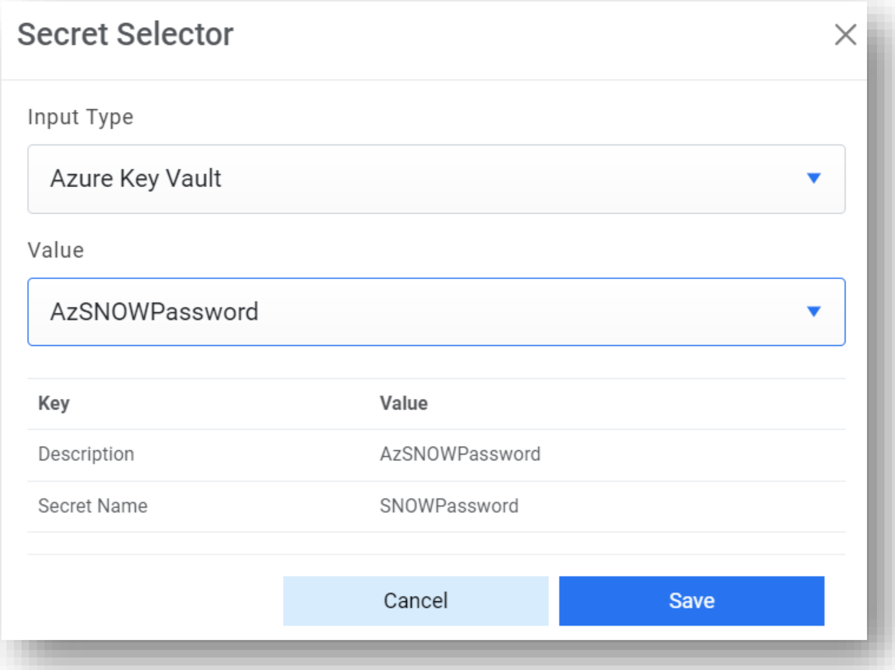
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | ʢ |

**Note:** Password is in encrypted form.

Figure 133 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

AzSNOWPassword

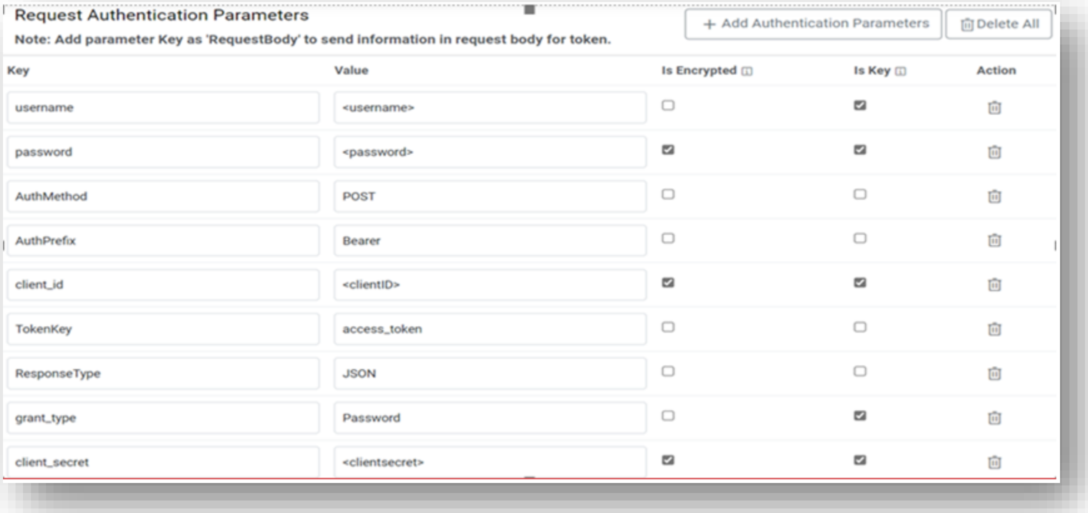
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 134 - Password from Azure Key Vault

- **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
- Based on the **Authentication Type**, add the parameters mentioned in the below table:

Table 21– Sample Authentication Parameters– Change Request

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsrcret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

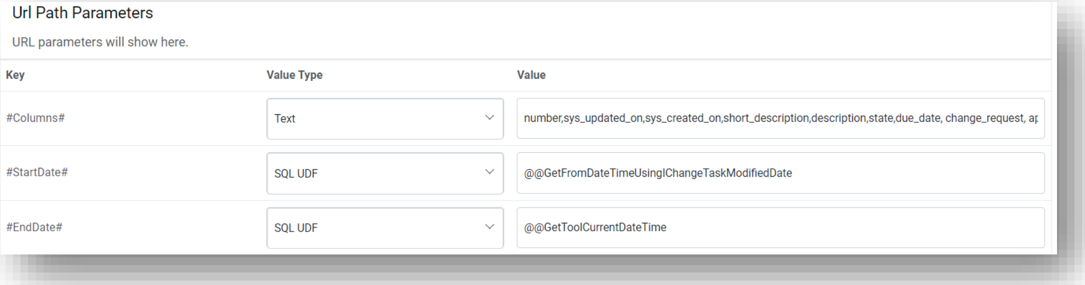
| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | ☐ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | ☐ | ☐ | 🗑 |
| AuthPrefix | Bearer | ☐ | ☐ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | ☐ | ☐ | 🗑 |
| ResponseType | JSON | ☐ | ☐ | 🗑 |
| grant_type | Password | ☐ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 135 – Create Data Source– Change Request (Request Authentication Parameters for OAuth2.0)

- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Columns#

ValueType: Text

Value:

number,
approval,sys_updated_on,sys_created_on,short_description,
description,state,due_date,
change_request,sys_id,assignment_group,priority


Note – These columns are mandatory. User can add more columns if
more data is required to be fetched from ITSM tool.

Key: #StartDate#

ValueType: SQL UDF
```

```
VALUE: @@GetFromDateTimeUsingIChangeRequestModifiedDate


Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,sys_created_on,short_description,description,state,due_date, change_request, ap |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingIChangeTaskModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 136 – URL Parameters (Change Request)

- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body -** Enter the request body in JSON format as per the configured URL, if applicable.
- **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below-

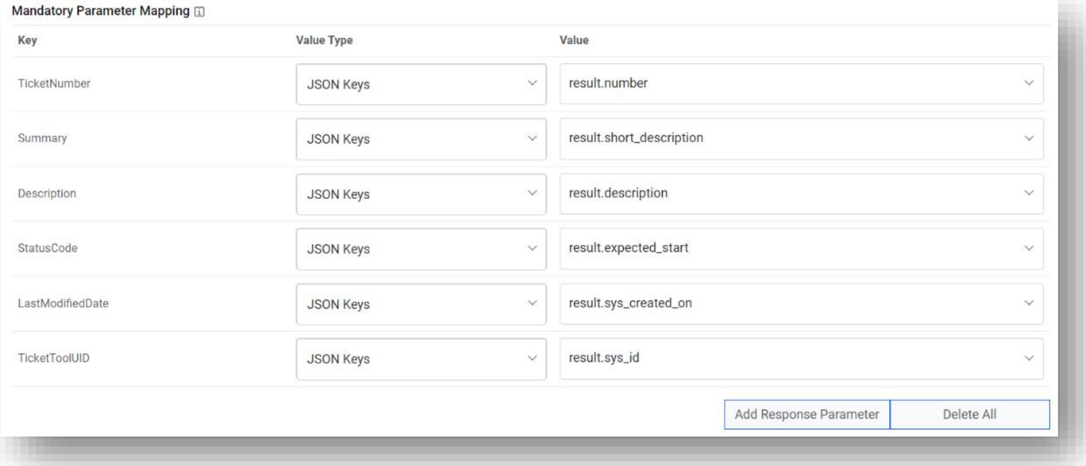
```
Response Body –
{“result": {"sys_updated_on": "2018-03-18 13:59:04","number":
"CHG556563","approval":"approved","priority":"4","sys_created_o
n": "2018-03-18 13:59:02","state": "1", "short_description":
"Implementation Task", "description": "Please initiate the
Implementation process.","sys_id": "xxxxxxxx","expected_start":
"2018-03-19 13:58:31",
"change_request": {"link": "https://my_host ","value":
"xxxxxxxx”},
"assignment_group":{
   "link": "https://my_host_link",
   "value": "73be6572db1bdf00ce29b6bffe96193d"
}
 }}
```

6. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.

7. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 22– Sample Mandatory Mapping Parameters– Change Request

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.number |
| Summary | JSON.Keys | result.short_description |
| Description | JSON.Keys | result.description |
| StatusCode | JSON.Keys | result.state |
| LastModifiedDate | JSON.Keys | result.sys_updated_on |
| TicketToolUID | JSON.Keys | result.sys_id |



Mandatory Parameter Mapping

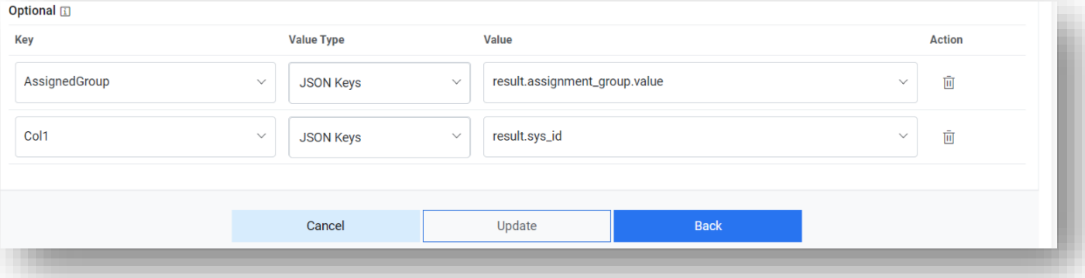
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.number |
| Summary | JSON Keys | result.short_description |
| Description | JSON Keys | result.description |
| StatusCode | JSON Keys | result.expected_start |
| LastModifiedDate | JSON Keys | result.sys_created_on |
| TicketToolUID | JSON Keys | result.sys_id |

Figure 137 – Mandatory Parameter Mapping (Change Request)

8. If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 23 – Sample Optional Mapping Parameters– Change Request

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | result.assignment_group.value |
| Col1 | JSON.Keys | result.sys_id |



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | result.assignment_group.value | 🗑 |
| Col1 | JSON Keys | result.sys_id | 🗑 |

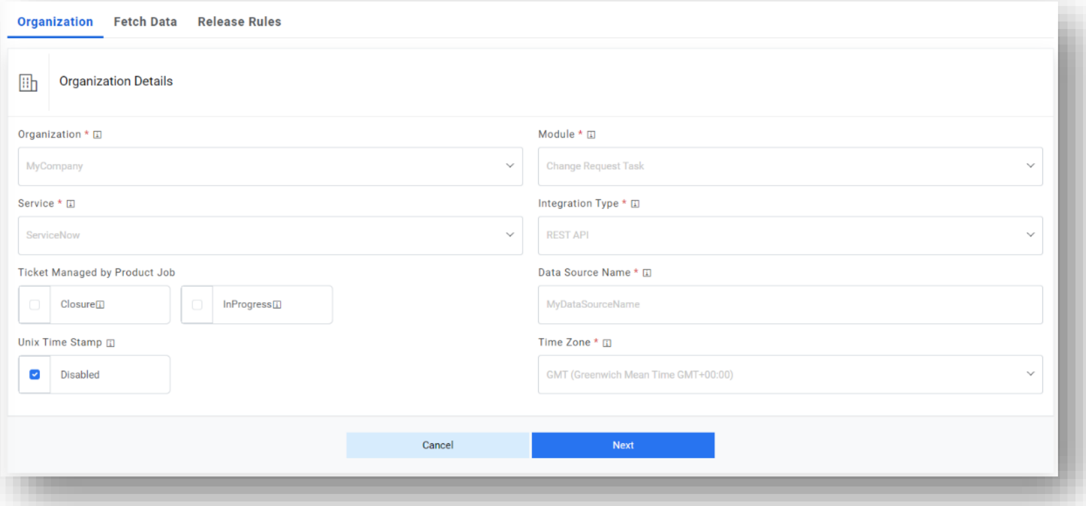
Figure 138 – Optional Parameter Mapping (Change Request)

9. Click **Save** to add the data source.

### 4.2.3.2 Create Data Source for Change Request Task

To create a data source for Change Request Task Management, perform the following steps:

1. On the left menu bar, click Configuration -> Manage Data Sources.
2. The **Create Data Source** page appears with the following tabs:
    - Organization
    - Fetch Data
    - Release Rules



Organization Details

Organization * : MyCompany

Module * : Change Request Task

Service * : ServiceNow

Integration Type * : REST API

Ticket Managed by Product Job: □ Closure □ InProgress

Data Source Name * : MyDataSourceName

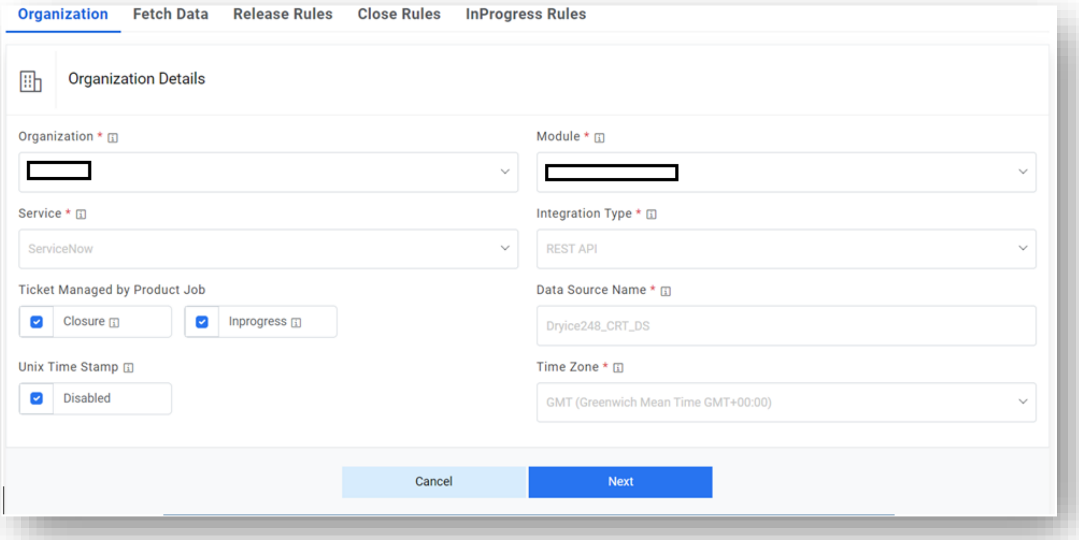
Unix Time Stamp: ☑ Disabled

Time Zone * : GMT (Greenwich Mean Time GMT+00:00)

Figure 139 - Create Data Source – Change Request Task

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab,
    - Select the **Organization Name** from the dropdown.
    - In the **Module** field, select 'Change Request Task', since we are configuring this data source for pulling the change requests.
    - In the **Service** field, select 'Service Now Tool' as we are configuring the data source for ServiceNow.
    - In the **Integration Type** field, select **REST**, since we will be integrating through REST APIs.
    - Select the **Timezone** to specify the time zone of the selected data source.
    - Select **Timestamp** to view the present data with date and time.
    - Click **Next**.



Organization Details

Organization *: [redacted]

Module *: [redacted]

Service *: ServiceNow

Integration Type *: REST API

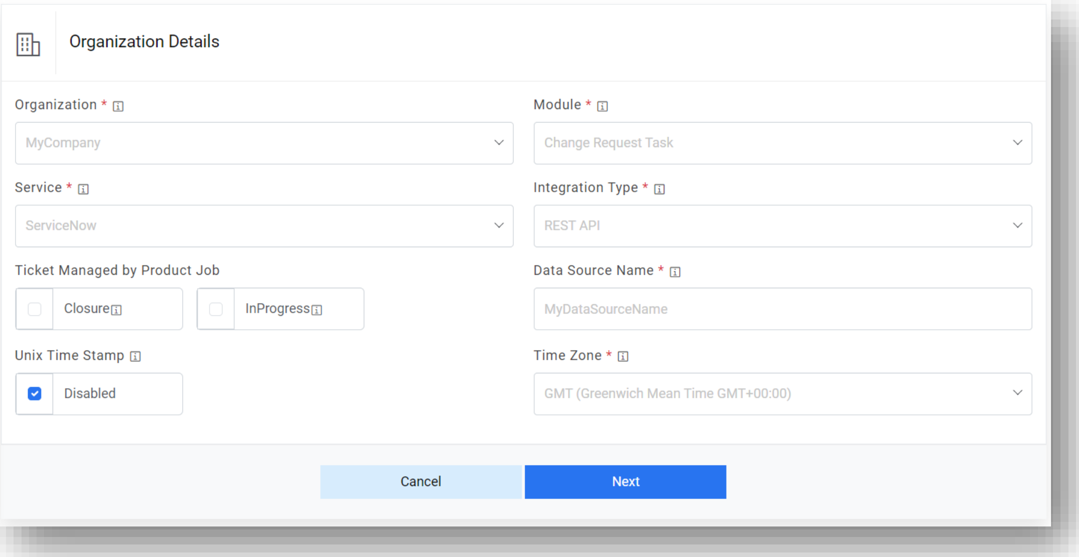
Ticket Managed by Product Job: ☑ Closure ☑ Inprogress

Data Source Name *: Dryice248_CRT_DS

Unix Time Stamp: ☑ Disabled

Time Zone *: GMT (Greenwich Mean Time GMT+00:00)

Figure 140 - Create Data Source – Change Request Task (Cont.)



Organization Details

Organization *: MyCompany

Module *: Change Request Task

Service *: ServiceNow

Integration Type *: REST API

Ticket Managed by Product Job: □ Closure □ InProgress

Data Source Name *: MyDataSourceName

Unix Time Stamp: ☑ Disabled

Time Zone *: GMT (Greenwich Mean Time GMT+00:00)

Figure 141 - Create Data Source – Change Request Task (Cont.)

4. On the **Fetch Data** tab, type in the details as per the environment.

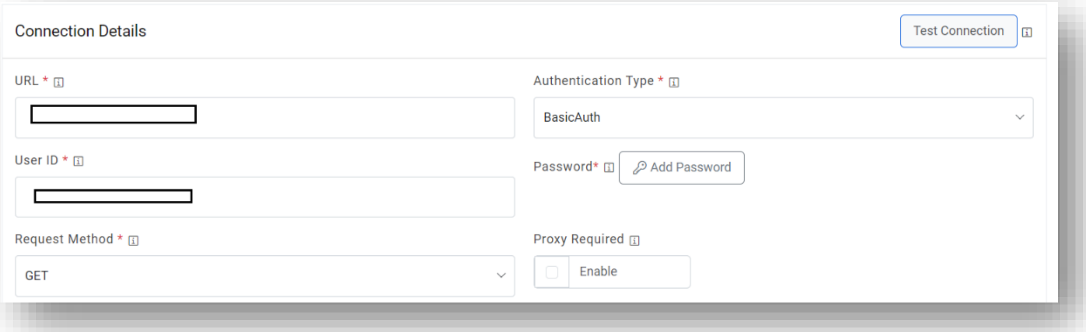
5. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL** – https://<URL>?sysparm_fields=#Columns#&sysparm_query=active=true^sys_updated_on>=#StartDate#^sys_updated_on<=#EndDate#^ORDERBYsys_updated_on
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0
- Selection of **Basic / Windows** requires you to enter -
  - User Id

- Password

- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

URL * : [redacted]

Authentication Type * : BasicAuth

User ID * : [redacted]

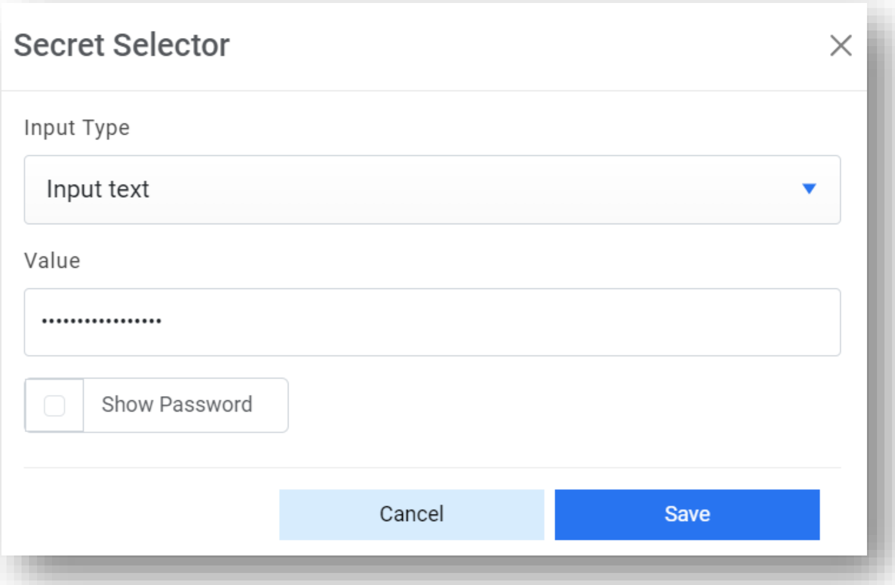
Password* : Add Password

Request Method * : GET

Proxy Required : □ Enable

Figure 142 – Create Data Source – Change Request Task (Connection Details)

- **Password** - For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



Secret Selector

Input Type

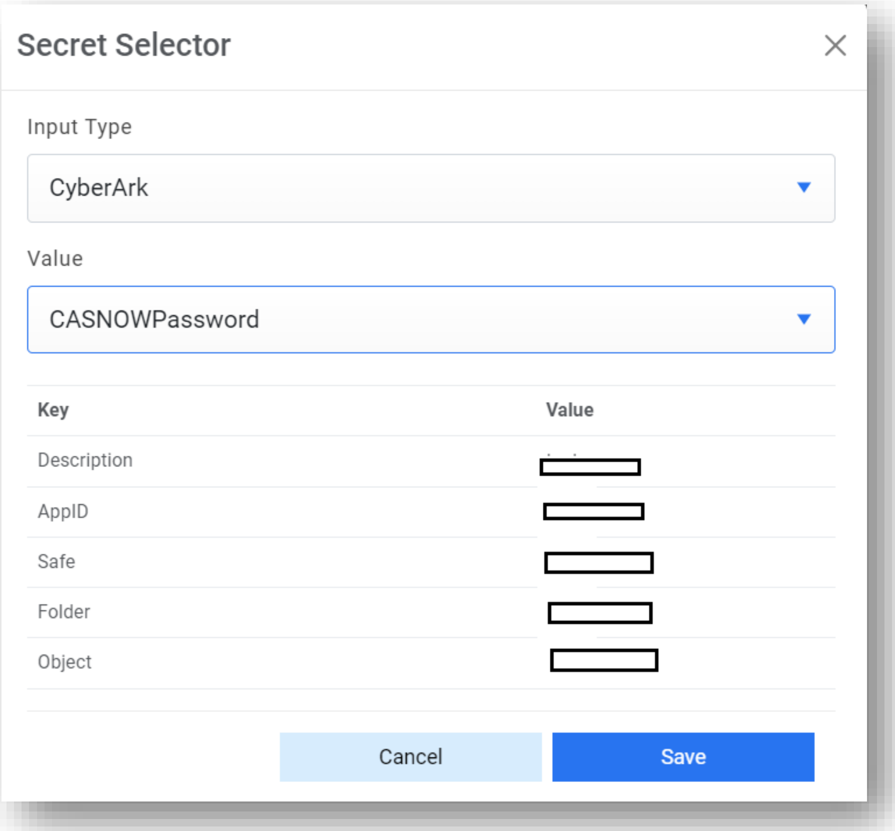
Input text

Value

••••••••••••••••

□ Show Password

Figure 143 - Password in Plaintext



Secret Selector

Input Type

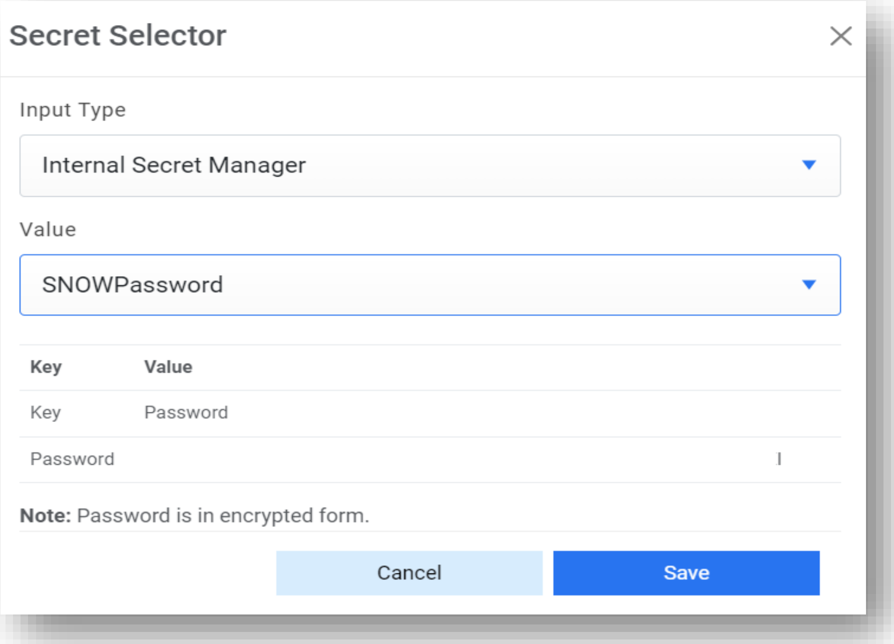
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 144 - Password from Key Vault (CyberArk)



Secret Selector

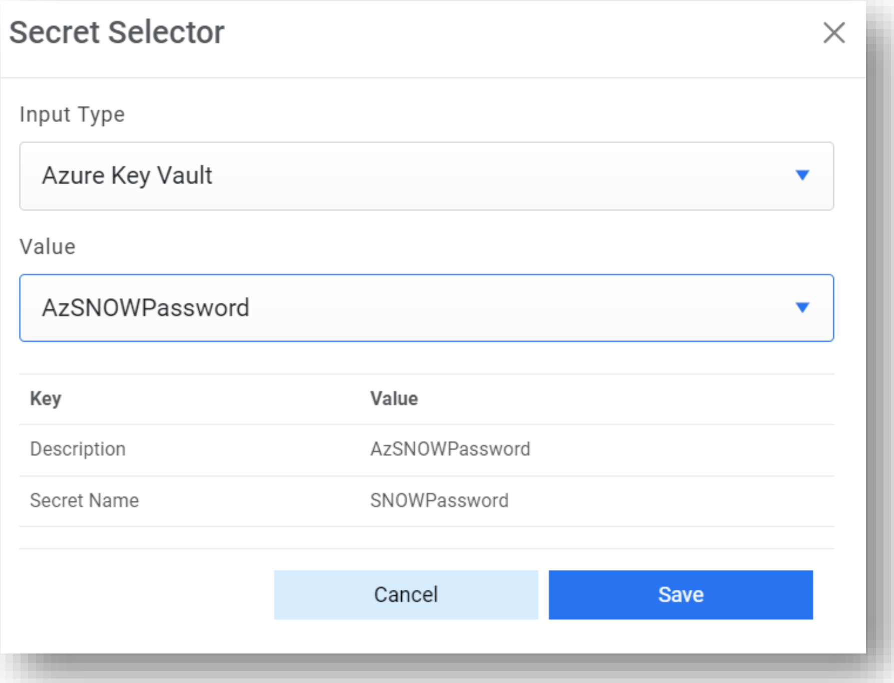
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | I |

**Note:** Password is in encrypted form.

Figure 145 - Password from Secret Manager



Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

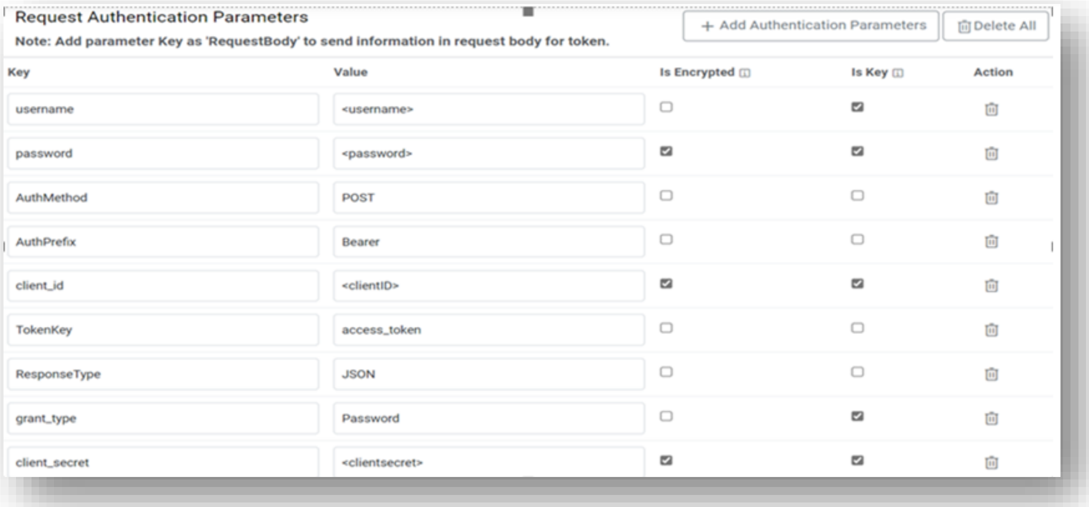
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 146 - Password from Azure Key Vault

- **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
- Based on the **Authentication Type**, add the parameters mentioned in the below table:

Table 24 – Sample Authentication Parameters– Change Request Task

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsrcret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | □ | □ | 🗑 |
| AuthPrefix | Bearer | □ | □ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | □ | □ | 🗑 |
| ResponseType | JSON | □ | □ | 🗑 |
| grant_type | Password | □ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 147 – Create Data Source – Change Request Task (Request Authentication Parameters for OAuth2.0)

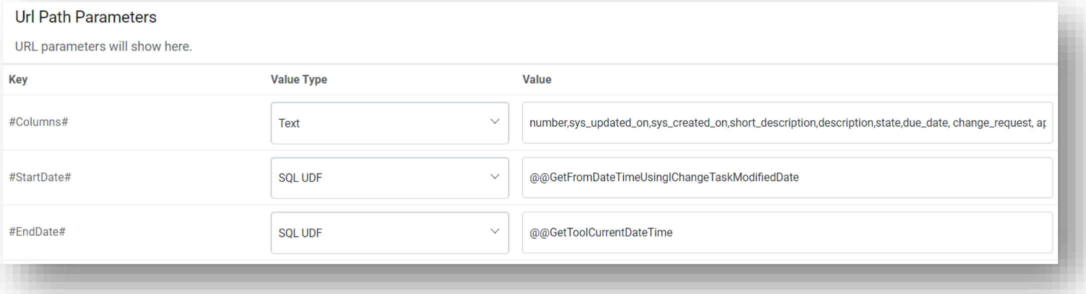
- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Columns#

ValueType: Text

Value:

number, short_description, description, state, change_request,
sys_updated_on, sys_created_on
```

These columns are mandatory. Users can add more columns if more data is required to be fetched from ITSM tool.

```
Key: #StartDate#

ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingIChangeTaskModifiedDate



Key: #EndDate#
```

```
ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,sys_created_on,short_description,description,state,due_date, change_request, ap |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingIChangeTaskModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 148 – URL Parameters (Change Request Task)

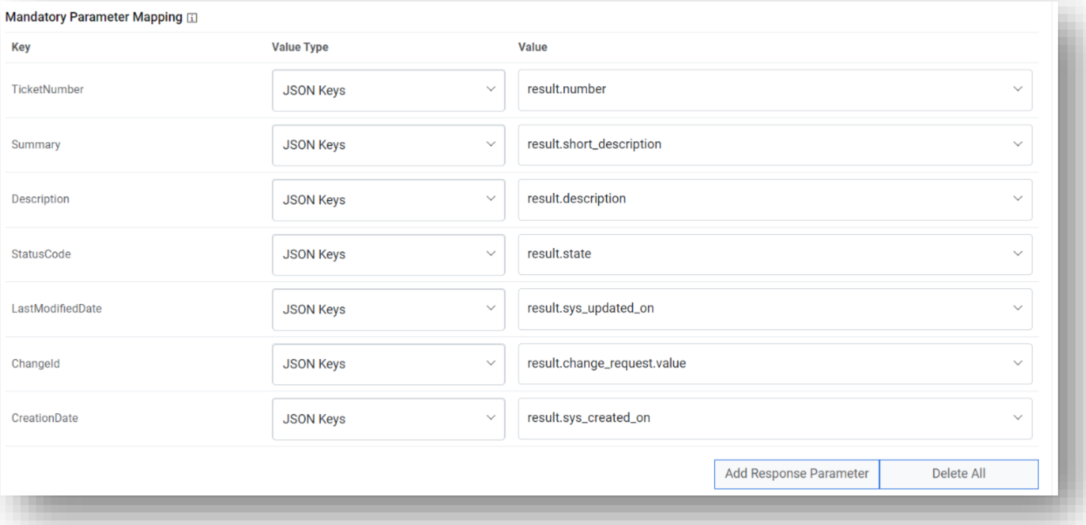
- **Request Header Parameters –** Please enter the request header parameters as required.
- **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below:

```
Response Body –

{

   "result": {"sys_updated_on": "2018-03-18 13:59:04","number":
"CTASK0039760","sys_created_on":

2018-03-18    13:59:02","state":    "1",    "short_description":
"Implementation  Task",  "description":  "Please  initiate  the
Implementation process.","sys_id": "xxxxxxxx”,

"change_request":                                      {"link":
"https://<ipaddress>:<port>/api/now/v1/table/change_request/xxx
xxxxx ","value": "xxxxxxxx"} }

}
```

6. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
7. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 25 – Sample Mandatory Mapping Parameters– Change Request Task

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.number |
| Summary | JSON.Keys | result.short_description |
| Description | JSON.Keys | result.description |
| StatusCode | JSON.Keys | result.state |
| LastModifiedDate | JSON.Keys | result.sys_updated_on |
| ChangeId | JSON.Keys | result.change_request.value |
| CreationDate | JSON.Keys | result.sys_created_on |



Mandatory Parameter Mapping

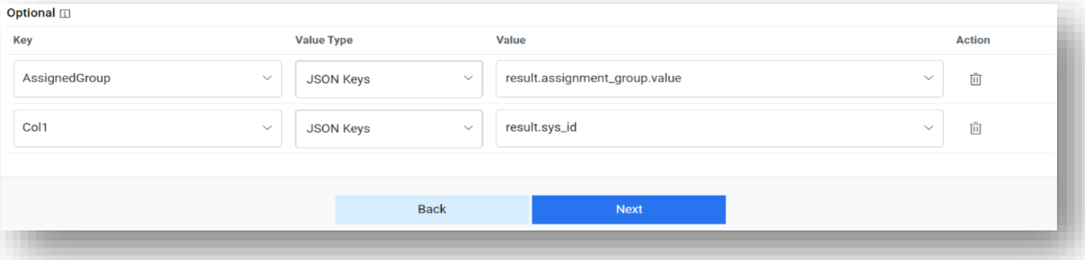
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.number |
| Summary | JSON Keys | result.short_description |
| Description | JSON Keys | result.description |
| StatusCode | JSON Keys | result.state |
| LastModifiedDate | JSON Keys | result.sys_updated_on |
| ChangeId | JSON Keys | result.change_request.value |
| CreationDate | JSON Keys | result.sys_created_on |

Figure 149 – Mandatory Parameter Mapping (Change Request Task)

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 26 – Sample Optional Mapping Parameters– Change Request Task

| Key | Value Type | Value |
|---|---|---|
| Col1 | JSON.Keys | result.sys_id |
| AssignedGroup | JSON.Keys | result.assignment_group.value |



Optional

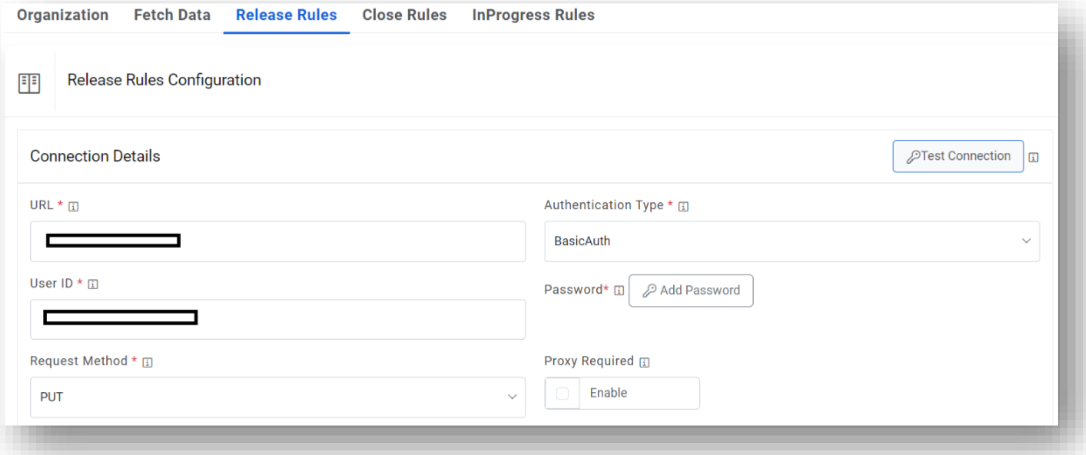
| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | result.assignment_group.value | |
| Col1 | JSON Keys | result.sys_id | |

Figure 150 – Optional Parameter Mapping (Change Request Task)

8. Click Next to proceed to Release Rules.
9. On the **Release Rules** tab, type in the details as per the requirement.
10. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>.service-now.com/api/now/table/change_task/#incident#
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- **Request Method** – Select Request Method as PUT from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Release Rules Configuration

Connection Details

URL * [redacted]

Authentication Type * BasicAuth

User ID * [redacted]

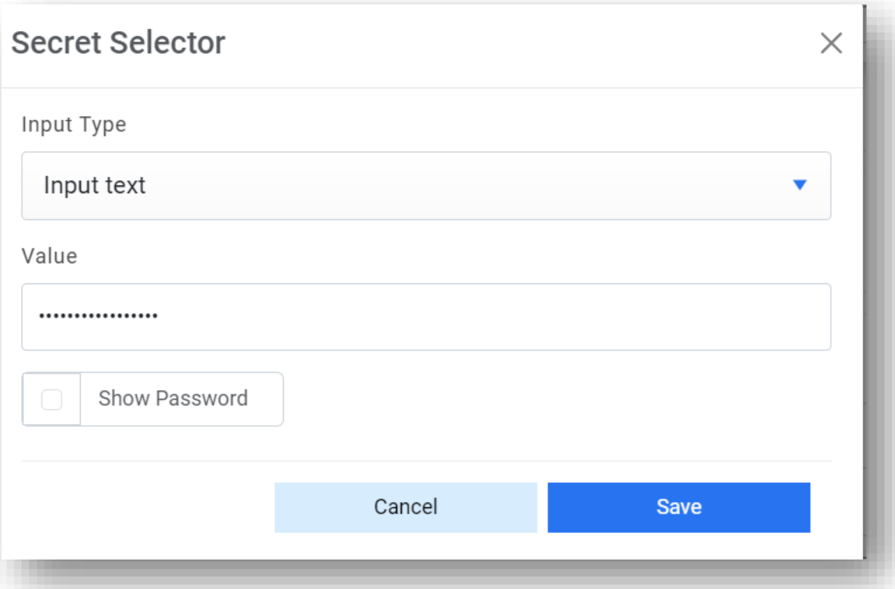
Password*

Request Method * PUT

Proxy Required □ Enable

Figure 151 – Release Rules – Change Request Task (Connection Details)

- **Password** - For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



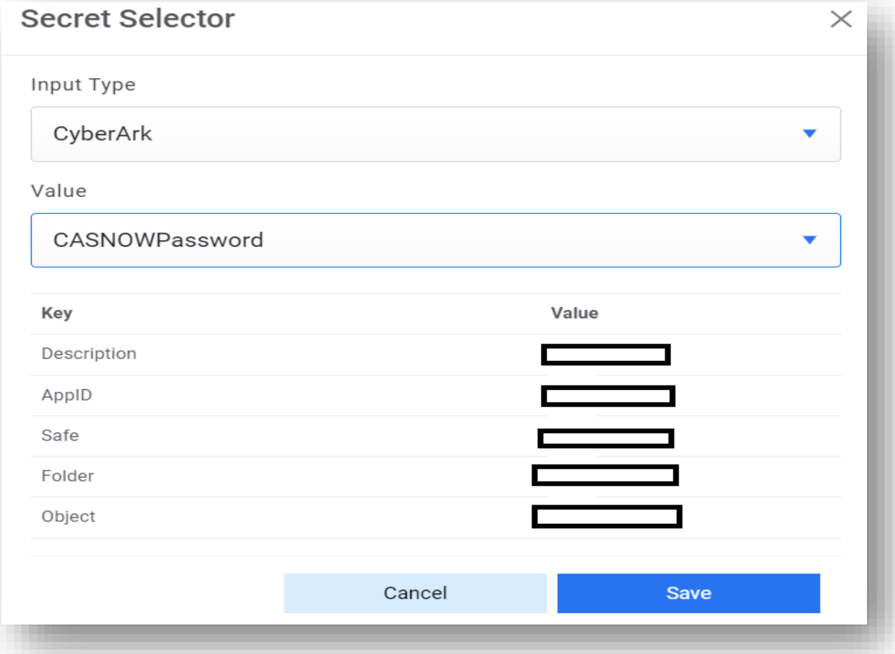
Secret Selector

Input Type: Input text

Value: ••••••••••••••••

□ Show Password

Figure 152 - Password in Plaintext



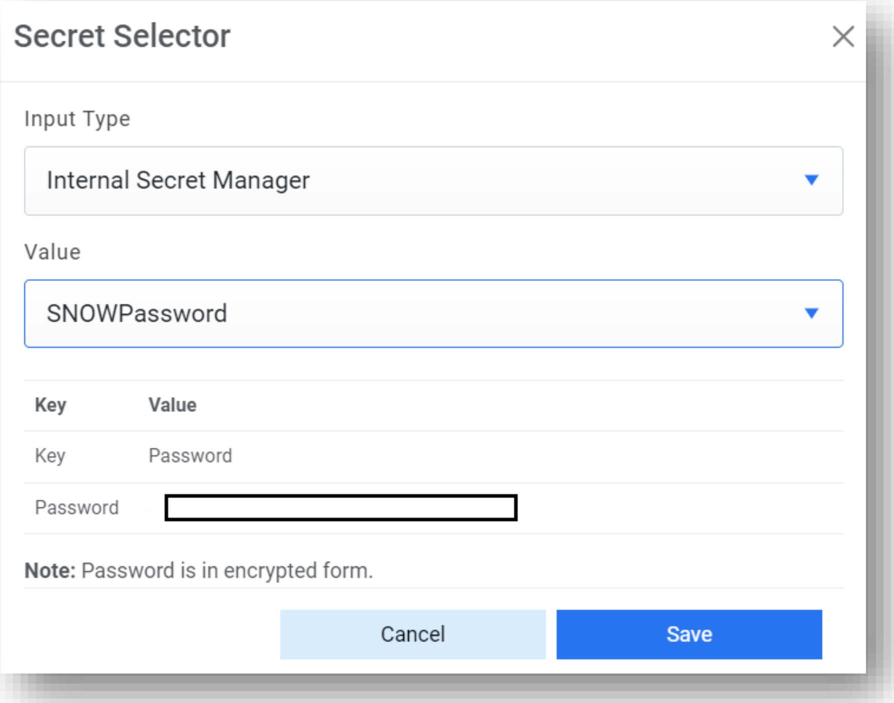
Secret Selector

Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 153 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

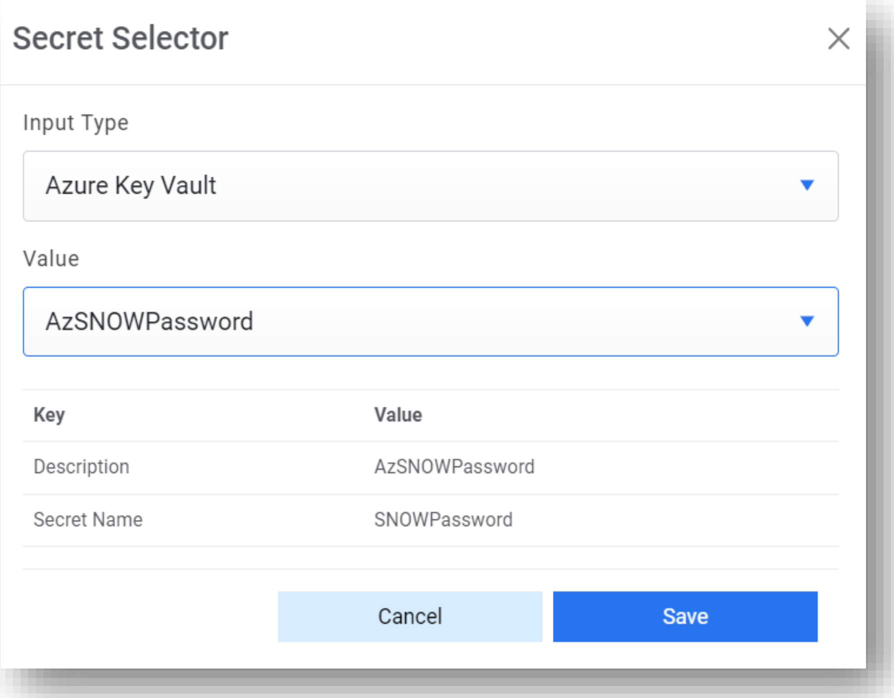
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 154 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

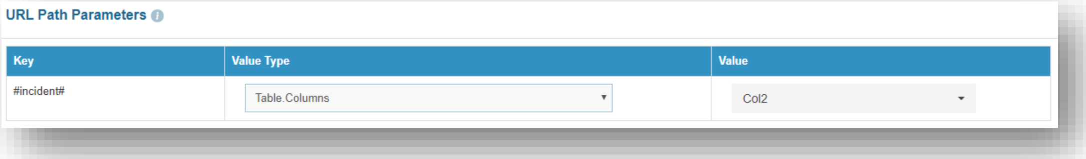
AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 155 - Password from Azure Key Vault

– **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #incident#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
"Col2"
```



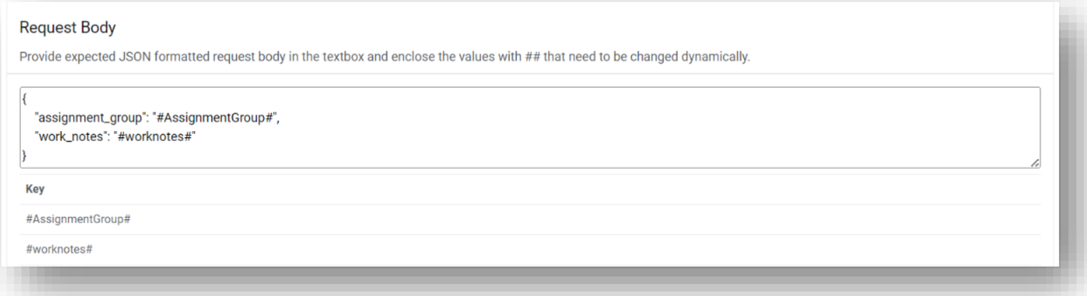
URL Path Parameters

| Key | Value Type | Value |
|---|---|---|
| #incident# | Table.Columns | Col2 |

Figure 156 – Release Rules – Change Request Task (URL Path Parameters)

- **Request Header Parameters –** Please enter the request header parameters as required.
- **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –

{   "assignment_group"   :   "#AssignmentGroup#","work_notes"   :
"#work_notes#" }
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

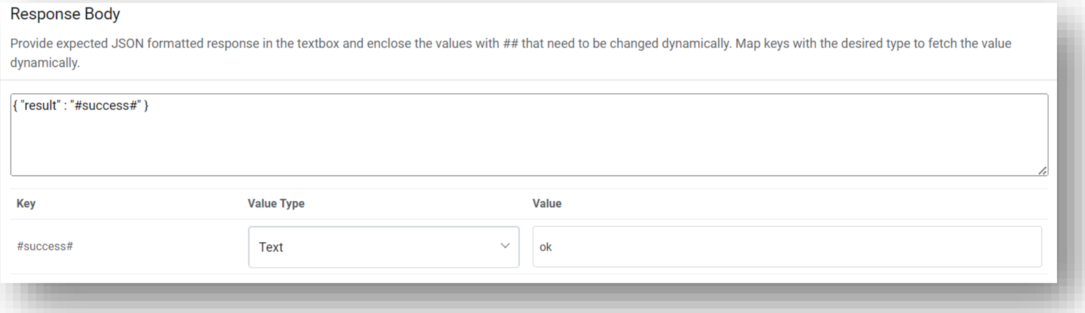
```
{
  "assignment_group": "#AssignmentGroup#",
  "work_notes": "#worknotes#"
}
```

| Key |
|---|
| #AssignmentGroup# |
| #worknotes# |

Figure 157 – Release Rules – Change Request Task (Request Body)

- **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

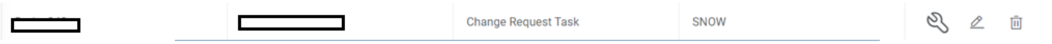
Figure 158 – Release Rules – Change Request Task (Response Body)

– **Response Key Value** mapping can be done as per the below table:

Table 27– Sample Response Key Value Mapping Parameters– Change Request Task

| | | |
|---|---|---|
| #success# | Text | OK |

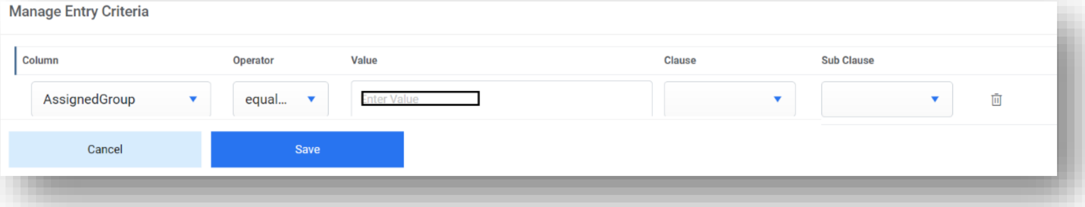
11. Click **Save** to add the data source.
12. To bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and same has to be configured in iAutomate. This is achieved through **Manage Entry Criteria**. Please perform the below steps:
    - Go to Configuration and click Manage Data Sources.
    - On the **Data Sources** tab, click 🔧 next to the data source user wants to manage. **Manage Entry Criteria** screen appears.



| | | | | |
|---|---|---|---|---|
| [redacted] | [redacted] | Change Request Task | SNOW | |

Figure 159 – Manage Entry Criteria (Change Request Task)

- Click on 'Add Response Parameter'
- Select 'AssignedGroup' for the **Column** field and 'equals to' for the **Operator** field.
- Enter the sys_id of the assignment group in ServiceNow in the **Value** field.
- **Clause** and **Sub-Clause** fields can also be added based on requirement.



Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | [redacted] | | |

Figure 160 – Manage Entry Criteria – Change Request Task (Cont.)

13. Click **Save**.

### 4.2.3.3 Configuration of additional parameters for Recommendation and Parsing

To use the field values of Change Request for the purpose of Recommendation and Parsing by iAutomate services, they need to be mapped to Change Request Task.

To do so, perform the following steps –

1. On the left menu bar, click Advance Configuration -> Parameter -> Manage Column.

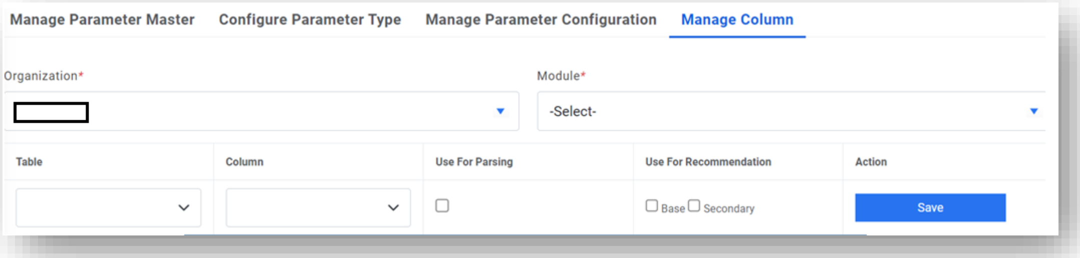
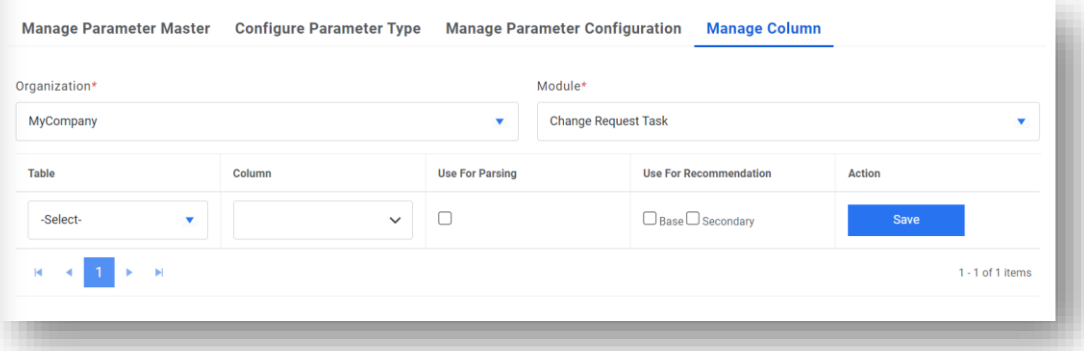


Figure 161 – Map fields of Change Request to Change Request Task

2. Select **Organization Name** from dropdown. Select 'Change Request Task' as the **Module**.



Organization*: MyCompany

Module*: Change Request Task

Table: -Select-

Figure 162 – Map fields of Change Request to Change Request Task (Cont.)

Summary, Description, RunbookToolTenantID, ModuleType are the default entries.

3. Select 'iChangeRequest' in **Table** dropdown.
4. Select the column of Change Request which has to be mapped to Change Request in the **Column** dropdown. In this case, we are selecting 'StatusCode.
5. Check the fields **Use for Parsing** and select 'Base' for **Use For Recommendation** field.

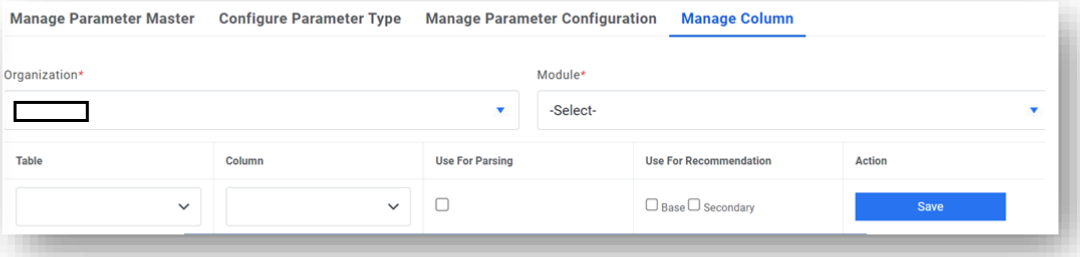
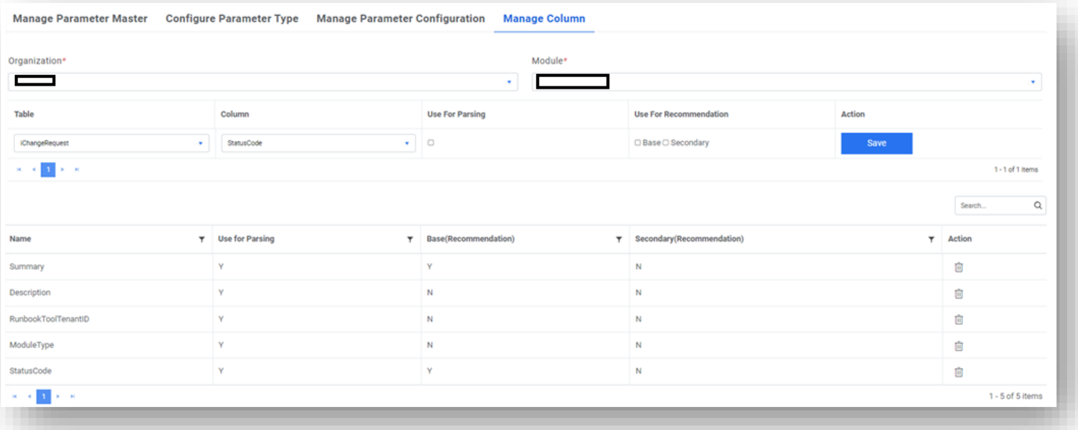


Figure 163 – Map fields of Change Request to Change Request Task (Cont.)

6. Click **Save.** The page lists one additional entry i.e. 'StatusCode', as depicted below.



Organization* [redacted]

Module* [redacted]

| Table | Column | Use For Parsing | Use For Recommendation | Action |
|---|---|---|---|---|
| iChangeRequest | StatusCode | □ | □ Base □ Secondary | Save |

1 - 1 of 1 items

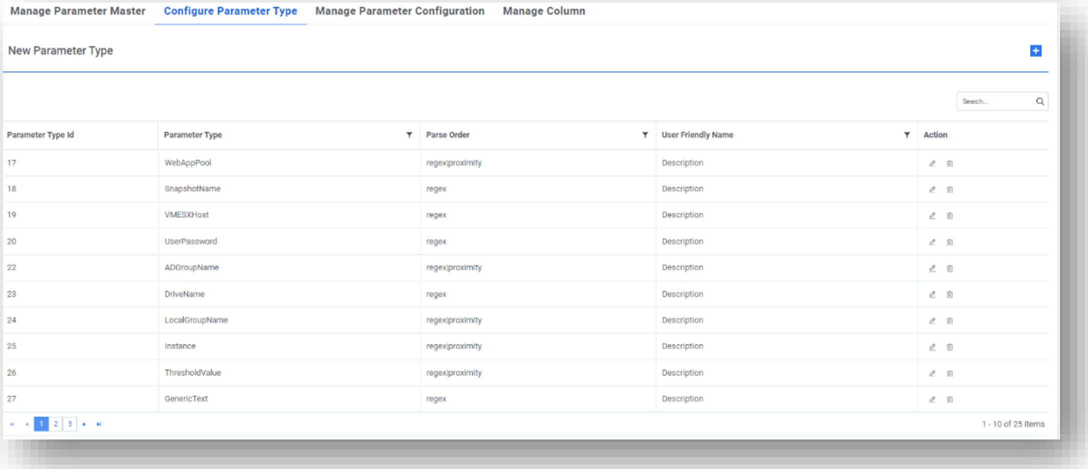
Search...

| Name | Use for Parsing | Base(Recommendation) | Secondary(Recommendation) | Action |
|---|---|---|---|---|
| Summary | Y | Y | N | 🗑 |
| Description | Y | N | N | 🗑 |
| RunbookToolTenantID | Y | N | N | 🗑 |
| ModuleType | Y | N | N | 🗑 |
| StatusCode | Y | Y | N | 🗑 |

1 - 5 of 5 items

Figure 164 – Map fields of Change Request to Change Request Task (Cont.)

7. For Recommendation, the above steps are sufficient. But for Parsing, additional steps are required to be performed.
8. On the main menu bar, click **Advance Configuration**.
9. Click **Configure Parameter Type**. By default, there are several entries already defined.



New Parameter Type

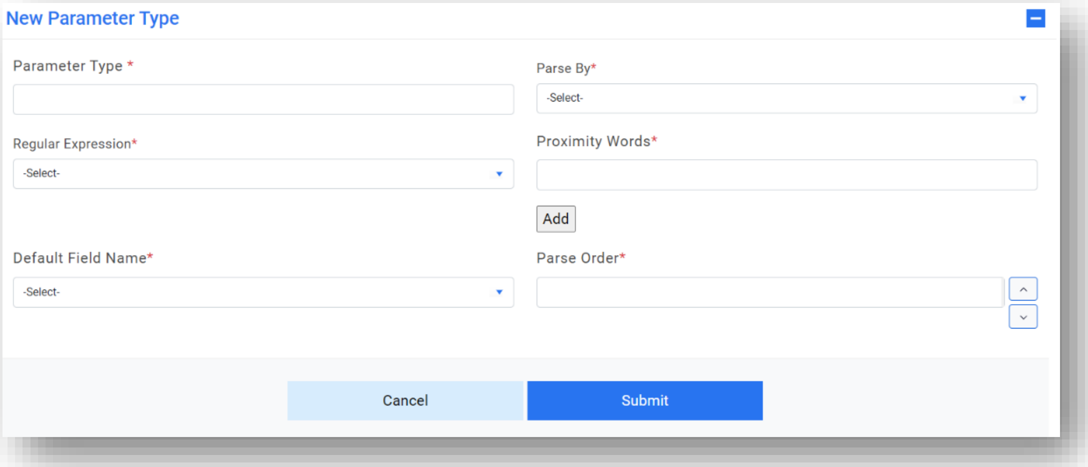
Search...

| Parameter Type Id | Parameter Type | Parse Order | User Friendly Name | Action |
|---|---|---|---|---|
| 17 | WebAppPool | regex\|proximity | Description | ✎ 🗑 |
| 18 | SnapshotName | regex | Description | ✎ 🗑 |
| 19 | VMESXHost | regex | Description | ✎ 🗑 |
| 20 | UserPassword | regex | Description | ✎ 🗑 |
| 22 | ADGroupName | regex\|proximity | Description | ✎ 🗑 |
| 23 | DriveName | regex | Description | ✎ 🗑 |
| 24 | LocalGroupName | regex\|proximity | Description | ✎ 🗑 |
| 25 | Instance | regex\|proximity | Description | ✎ 🗑 |
| 26 | ThresholdValue | regex\|proximity | Description | ✎ 🗑 |
| 27 | GenericText | regex | Description | ✎ 🗑 |

1 - 10 of 25 items

Figure 165 – Map fields of Change Request to Change Request Task (Cont.)

10. Click ➕ Icon to **Add New**.



New Parameter Type

Parameter Type *

Parse By* -Select-

Regular Expression* -Select-

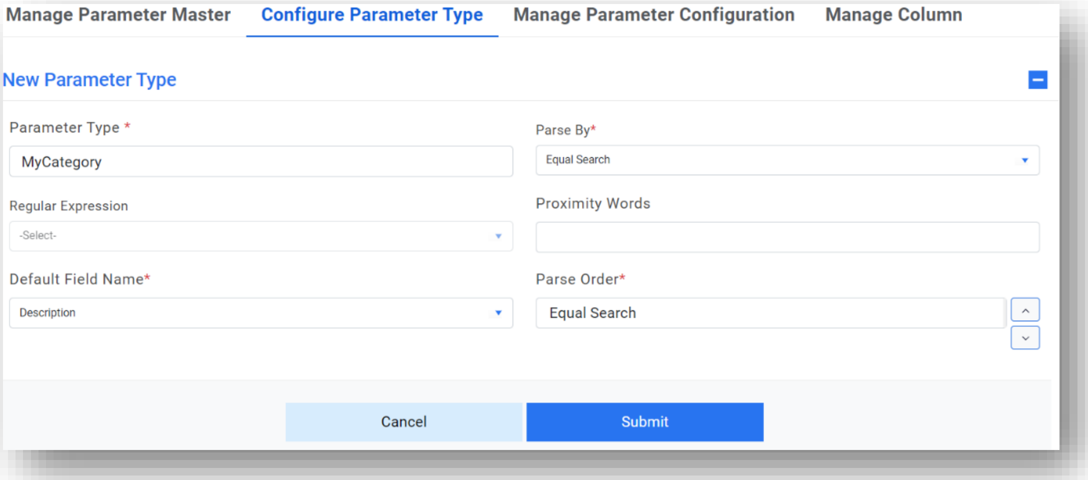
Proximity Words*

Default Field Name* -Select-

Parse Order*

Figure 166 – Map fields of Change Request to Change Request Task (Cont.)

11. Type **Parameter Type,** for e.g. MyCategory
12. Select 'Equal Search' as **Parse By**.
13. Select 'Description' as **Default Field Name**.
14. Click **Submit**.



New Parameter Type

Parameter Type * MyCategory

Parse By* Equal Search

Regular Expression -Select-

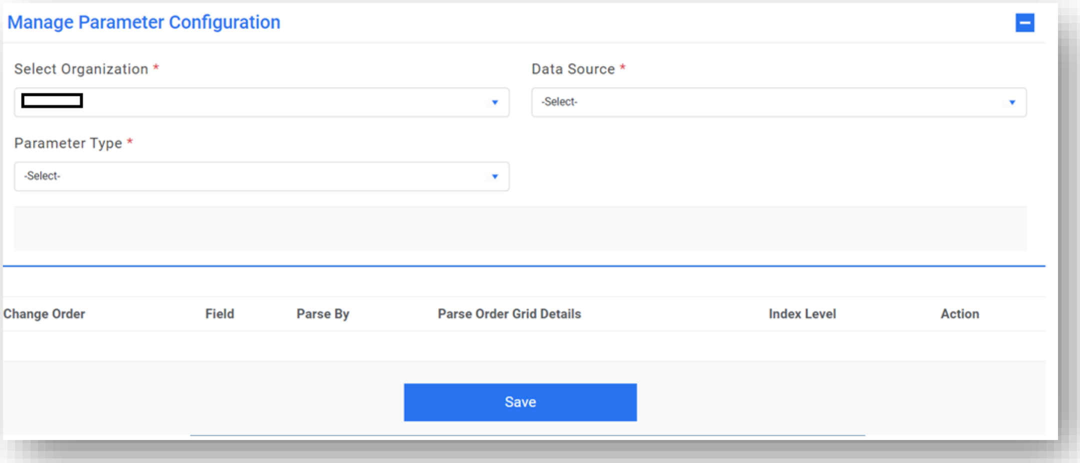
Proximity Words

Default Field Name* Description

Parse Order* Equal Search

Figure 167 – Map fields of Change Request to Change Request Task (Cont.)

15. Next step is to map this **Parameter Type** 'MyCategory', to the one that was created via **Manage Columns** in earlier step by the name **StatusCode**. To do that, perform the following steps:
16. On the main menu bar, click **Advance Configuration.**
17. Click Manage Parameter Configuration.



Manage Parameter Configuration

Select Organization * [redacted]

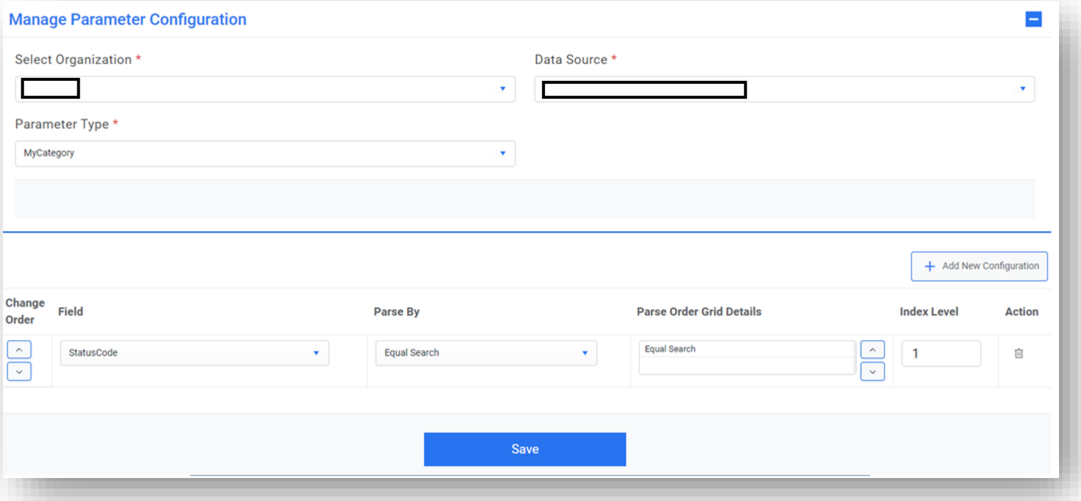
Data Source * -Select-

Parameter Type * -Select-

| Change Order | Field | Parse By | Parse Order Grid Details | Index Level | Action |
|---|---|---|---|---|---|

Figure 168 – Map fields of Change Request to Change Request Task (Cont.)

18. Selection **Organization.** Select 'Change Request Task' as the **Data Source.**
19. Select the newly created parameter 'MyCategory from **Parameter Type** dropdown.
20. From the **Field** dropdown, select 'StatusCode**,** the parameter that has been mapped via **Manage Columns.**



Manage Parameter Configuration

Select Organization * [redacted]

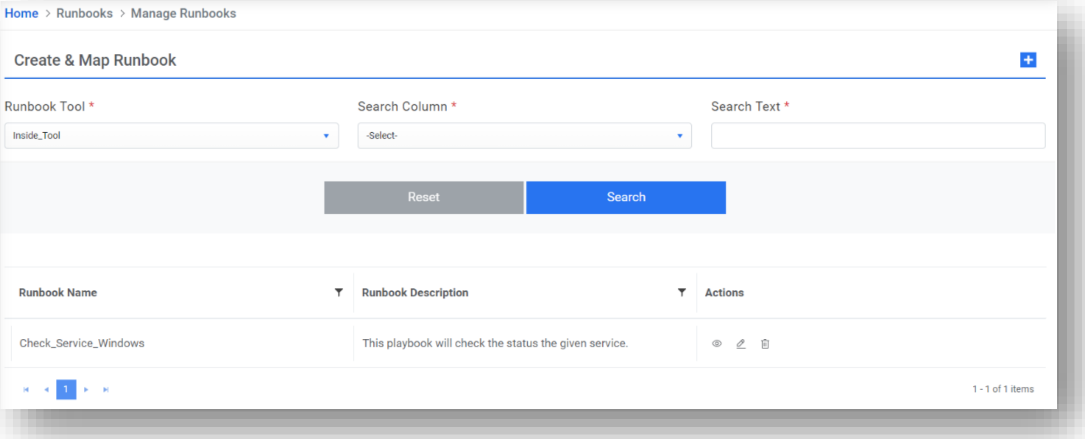
Data Source * [redacted]

Parameter Type * MyCategory

| Change Order | Field | Parse By | Parse Order Grid Details | Index Level | Action |
|---|---|---|---|---|---|
| | StatusCode | Equal Search | Equal Search | 1 | |

Figure 169 – Map fields of Change Request to Change Request Task (Cont.)

21. Click **Save**.
22. To verify whether this parameter is successfully parsed or not, perform the following steps:
    - On the main menu bar, click **Runbooks**.
    - Click Manage Runbooks.
    - Select the **Runbook Tool** mapped with the organization.



Home > Runbooks > Manage Runbooks

Create & Map Runbook

Runbook Tool * Inside_Tool

Search Column * -Select-

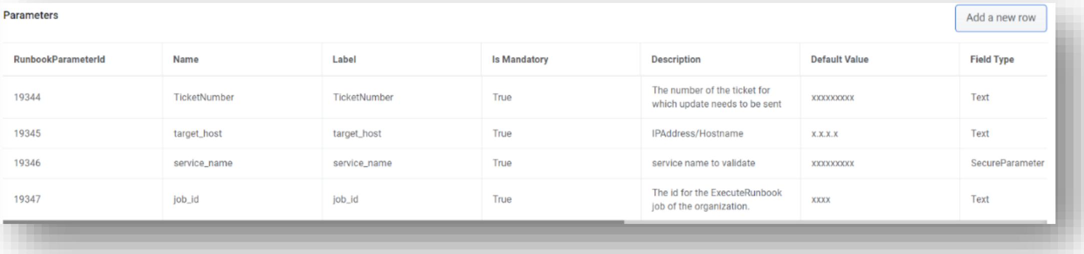
Search Text *

| Runbook Name | Runbook Description | Actions |
|---|---|---|
| Check_Service_Windows | This playbook will check the status the given service. | |

1 - 1 of 1 items

Figure 170 – Map fields of Change Request to Change Request Task (Cont.)

- The parameter, **StatusCode**, which was created in earlier steps, has to be added as one of the parameters to the existing runbook. You can also create a new runbook with **StatusCode** as one of the parameters.
- Click the **Edit** icon to edit the runbook.
- In the Parameters section, add a new parameter with any relevant **Parameter Name**, **Parameter Label**, **Parameter Description**, **Default Parameter Value**. Ensure that Parameter Type is selected as **Priority.**



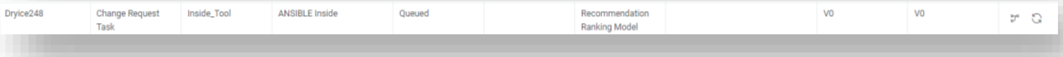
Parameters

| RunbookParameterId | Name | Label | Is Mandatory | Description | Default Value | Field Type |
|---|---|---|---|---|---|---|
| 19344 | TicketNumber | TicketNumber | True | The number of the ticket for which update needs to be sent | xxxxxxxxx | Text |
| 19345 | target_host | target_host | True | IPAddress/Hostname | x.x.x.x | Text |
| 19346 | service_name | service_name | True | service name to validate | xxxxxxxxx | SecureParameter |
| 19347 | job_id | job_id | True | The id for the ExecuteRunbook job of the organization. | xxxx | Text |

Figure 171 – Map fields of Change Request to Change Request Task (Cont.)

- Add the parameter and click **Update**.
- Ensure that the runbook in which the parameter is added is mapped with the organization.

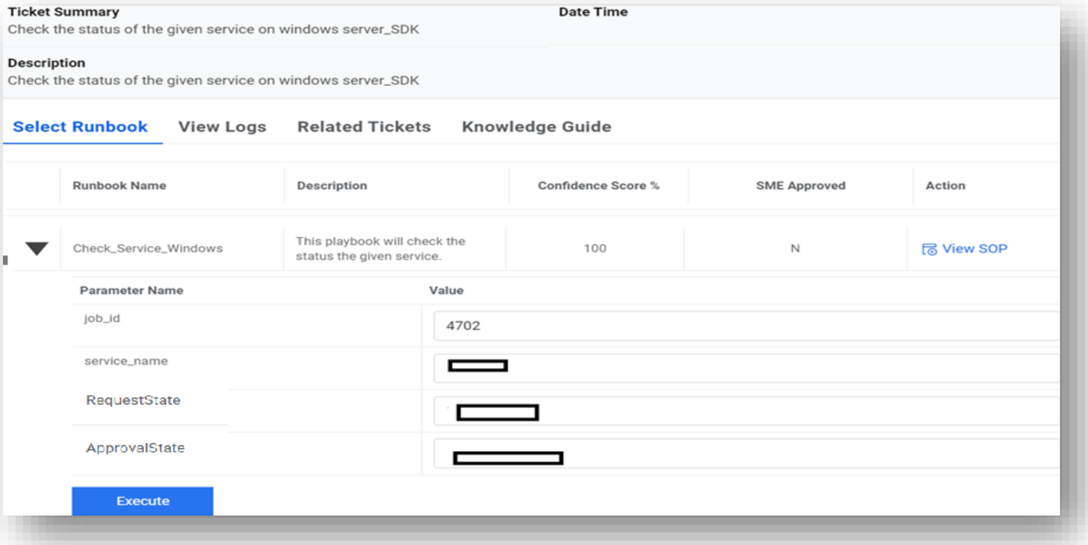
23. Next step is to build the Recommendation model and to do that perform the following steps:
    - On the left menu bar, click **Configuration.**
    - Click Build Model.
    - ReBuild / Re-build the model for the Organization under Change Request Task module for the mapped runbook tool.



| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Dryice248 | Change Request Task | Inside_Tool | ANSIBLE Inside | Queued | | Recommendation Ranking Model | | V0 | V0 | |

Figure 172 – Map fields of Change Request to Change Request Task (Cont.)

- Run the entire flow and see if the runbook recommended for the ticket in which the parameter was added has the parameter **MyCategory** with its expected value.



**Ticket Summary**
Check the status of the given service on windows server_SDK

**Date Time**

**Description**
Check the status of the given service on windows server_SDK

| Runbook Name | Description | Confidence Score % | SME Approved | Action |
|---|---|---|---|---|
| Check_Service_Windows | This playbook will check the status the given service. | 100 | N | View SOP |

| Parameter Name | Value |
|---|---|
| job_id | 4702 |
| service_name | [redacted] |
| RequestState | [redacted] |
| ApprovalState | [redacted] |

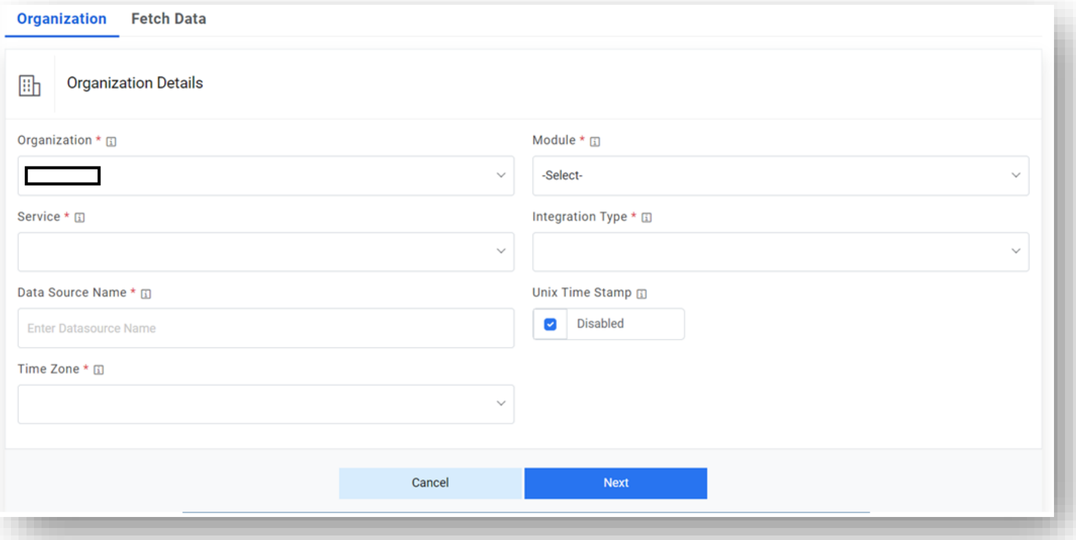
Figure 173 – Map fields of Change Request to Change Request Task (Cont.)

## 4.3 Integration with BMC Remedy

### 4.3.1 Incident Management

To create a data source for Incident Management, perform the following steps:

1. On the left menu bar, click Configuration -> Manage Data Sources.
2. The **Create Data Source** page appears with the following tabs:
   - Organization
   - Fetch Data
   - Release Rules



Organization Details

Organization * [redacted]

Module * -Select-

Service *

Integration Type *

Data Source Name * Enter Datasource Name

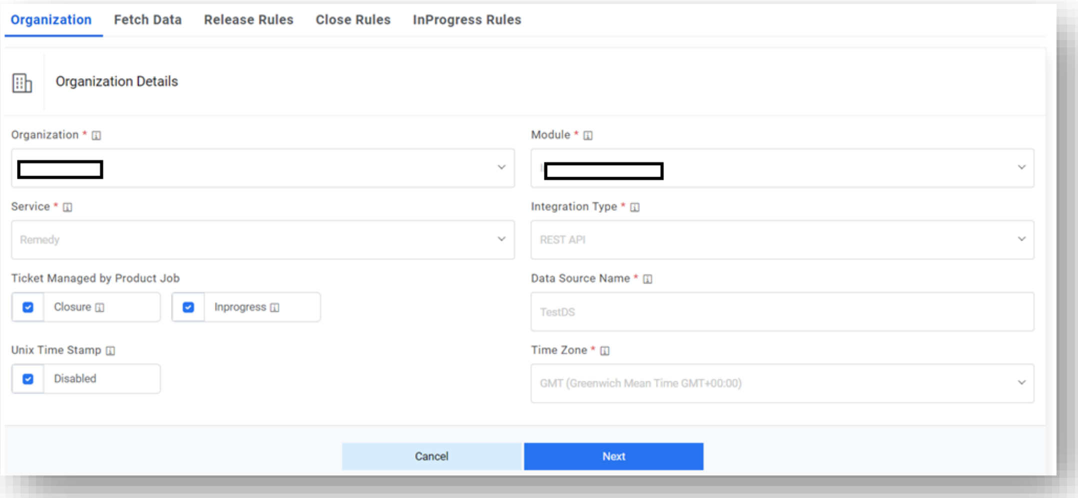
Unix Time Stamp ☑ Disabled

Time Zone *

Figure 174 - Create Data Source

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab:
   - Select the **Organization Name** from the dropdown.
   - Select the **Module** as **Incident Management** since we are configuring this data source for pulling the incident tickets.
   - Select the **Service** as **Remedy** as we are configuring the data source for BMC Remedy
   - Select the **Integration Type** as **REST**, since we will be integrating through REST APIs.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select **Timestamp** to view the present data with date and time.
   - Click **Next**.



Organization * [redacted]

Module * [redacted]

Service * Remedy

Integration Type * REST API

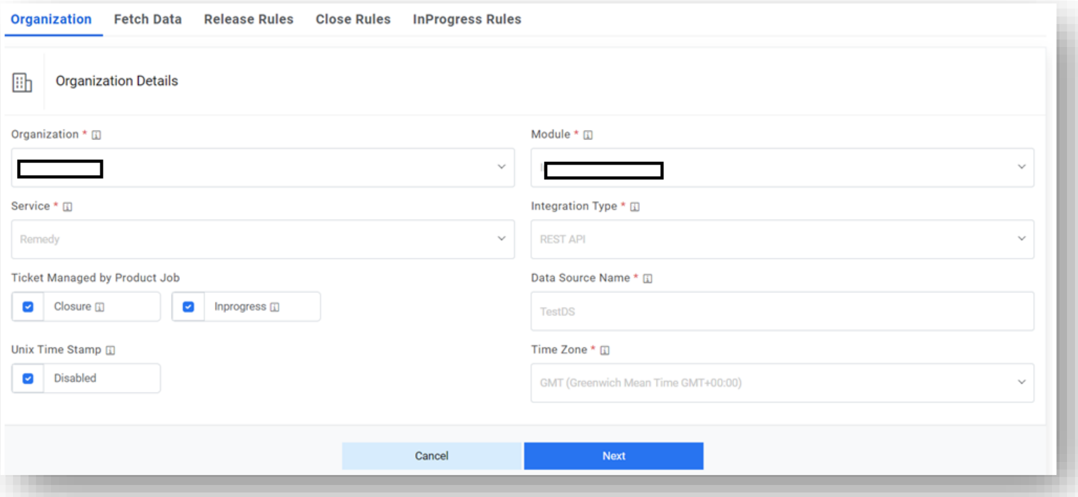
Ticket Managed by Product Job ☑ Closure ☑ Inprogress

Data Source Name * TestDS

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 175 - Create Data Source (Cont.)



Organization * [redacted]

Module * [redacted]

Service * Remedy

Integration Type * REST API

Ticket Managed by Product Job ☑ Closure ☑ Inprogress

Data Source Name * TestDS

Unix Time Stamp ☑ Disabled

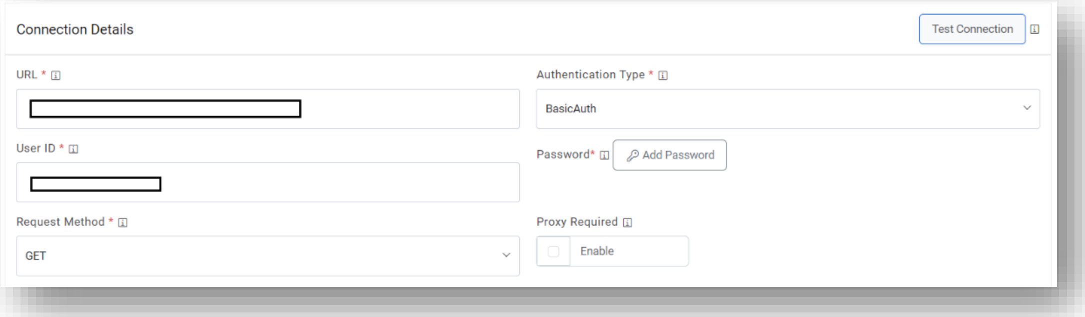
Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 176 - Create Data Source (Cont.)

4. On the **Fetch Data** tab, type in the details as per the environment.
5. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL** - http://URL/api/arsys/v1/entry/HPD:Help%20Desk/?q='Assigned Group'="#Group#" AND 'Last Modified Date'>"#StartDate#" AND 'Last Modified Date'<"#EndDAte#"&fields=values(#Columns#)
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0
- Selection of **Basic / Windows** requires you to enter -
  - User Id
  - Password
- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

Test Connection

URL *

[redacted]

Authentication Type *

BasicAuth

User ID *

[redacted]

Password* Add Password

Request Method *

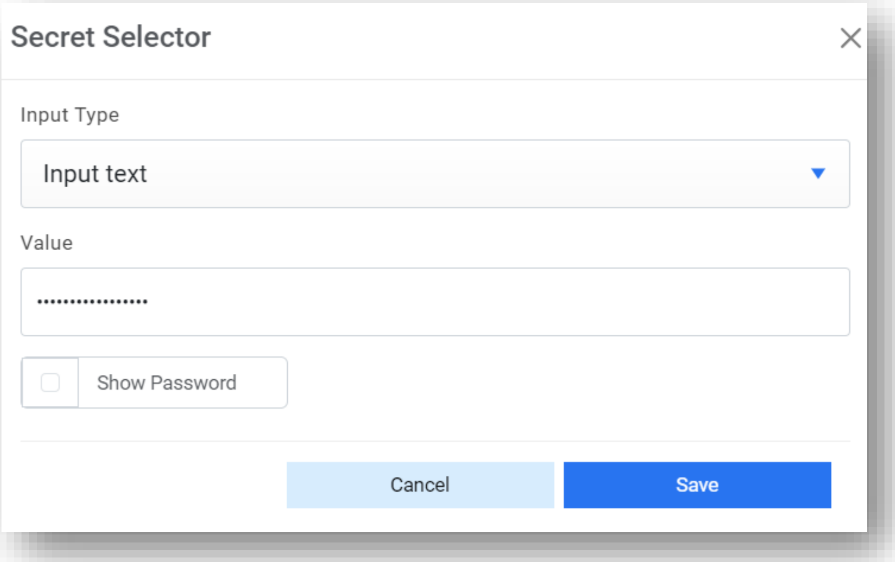
GET

Proxy Required

□ Enable

Figure 177 – Create Data Source (Connection Details)

- **Password** - For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



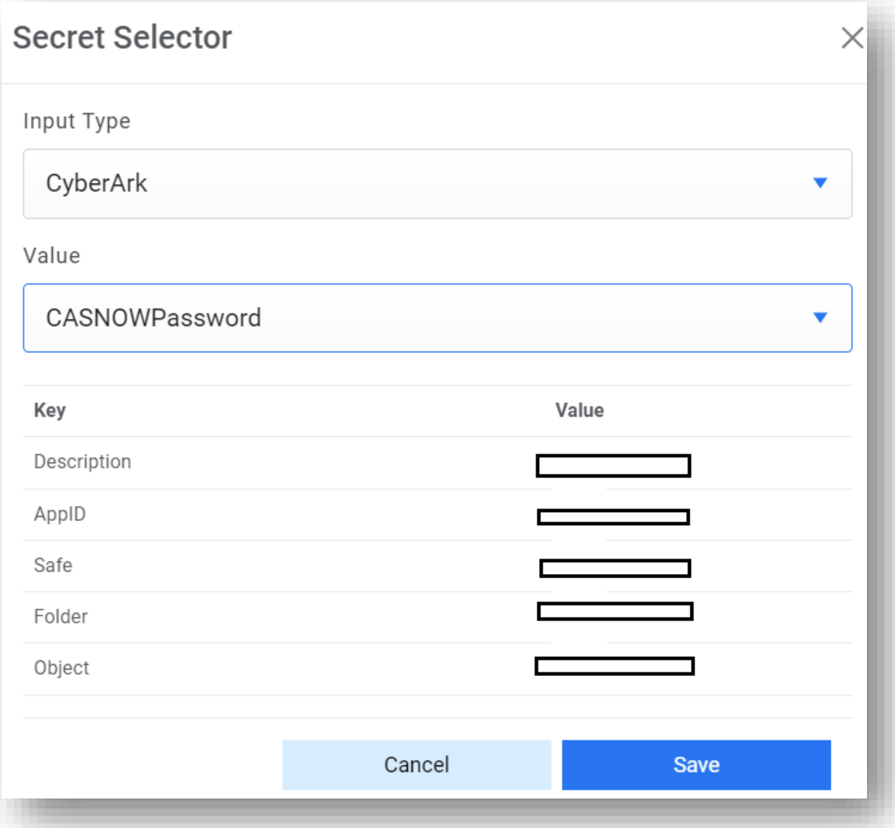
Secret Selector

Input Type: Input text

Value: ·················

☐ Show Password

Figure 178 - Password in Plaintext



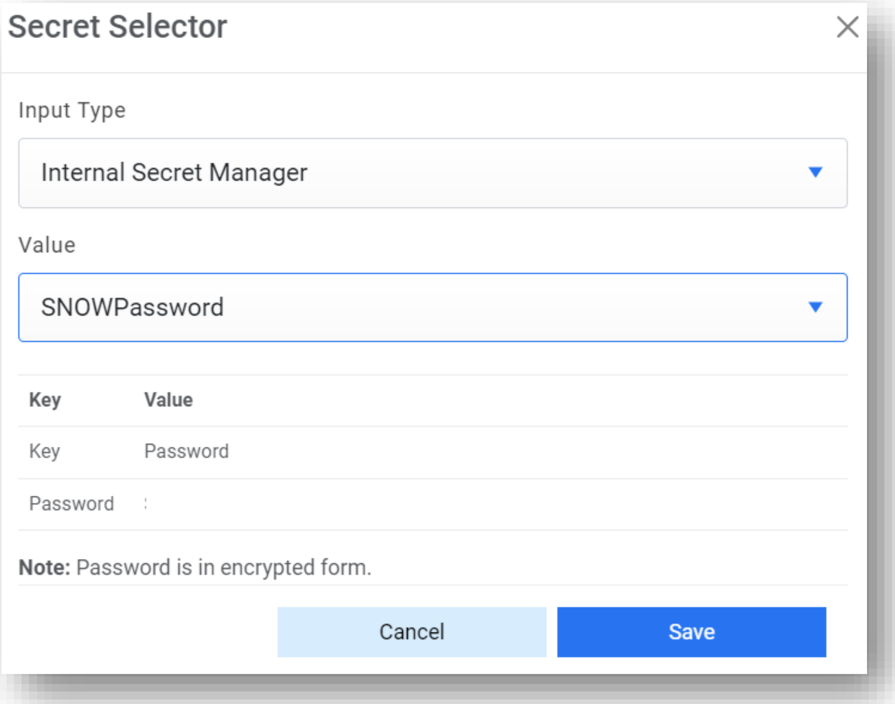
Secret Selector

Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 179 - Password from Key Vault (CyberArk)



Secret Selector

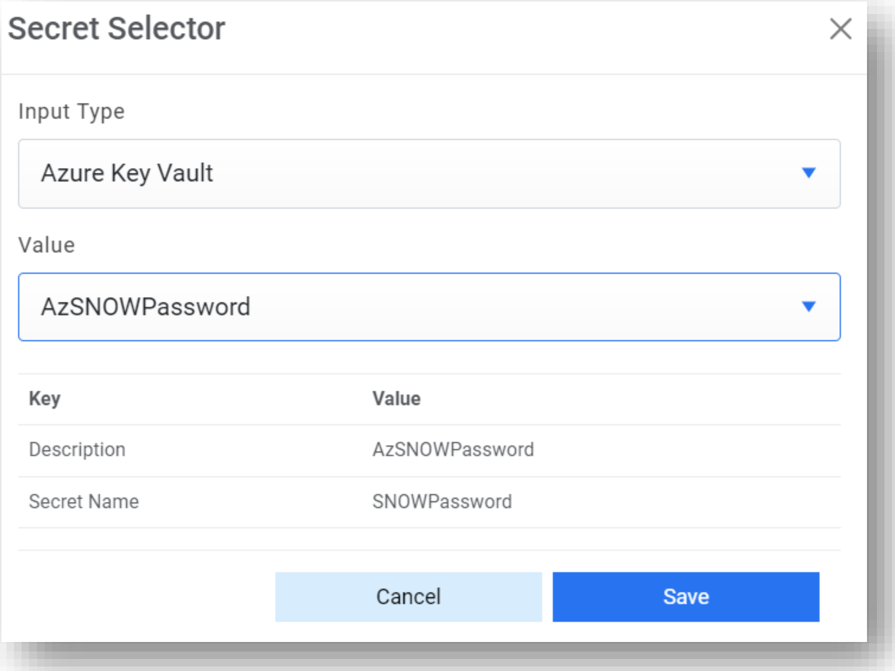
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | : |

**Note:** Password is in encrypted form.

Figure 180 - Password from Secret Manager



Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

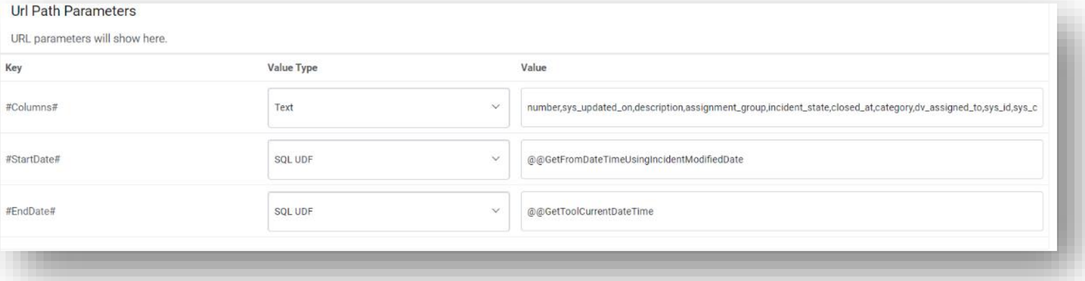
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 181 - Password from Azure Key Vault

- **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
- Based on the **Authentication Type**, add the parameters.

6. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Columns#

ValueType: Text

Value:

Incident Number,Description,Entry ID,Detailed Decription,Submit
Date,Status,Last Resolved Date,Assigned Group, Last Modified
Date,Parent_SAP_ID,Fraud Alert No.


Note – These columns are mandatory. User can add more columns if
more data is required to be fetched from ITSM tool.


Key: #StartDate#

ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingIncidentModifiedDate_Remedy


Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime_Remedy
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,description,assignment_group,incident_state,closed_at,category,dv_assigned_to,sys_id,sys_c |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingIncidentModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 182 – URL Path Parameters (BMC Remedy – Incident Management)

7. **Request Header Parameters –** Please enter the request header parameters as required.
8. **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below:

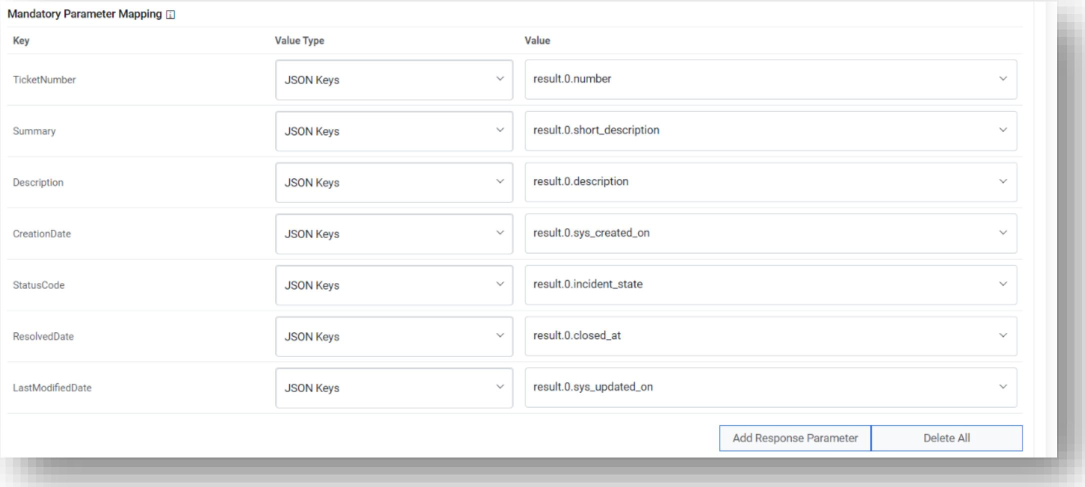
```
Response Body –

{

    "entries": [
        {
            "values": {
                "Incident Number": "xxxxxxxx",
                "Description": "Test ticket please ignore",
                "Entry ID": "xxxxxxxx",
                "Detailed Decription": "Test ticket please
ignore",
                "Submit Date": "2018-12-06T16:43:52.000+0000",
                "Status": "Assigned",
                "Last Resolved Date": "dummy",
                "Assigned Group": "xxxxxxxx",
                "Last Modified Date": "2018-12-
06T16:43:52.000+0000"
, "Fraud Alert No.": "xxxxxxxx"
, "Parent_SAP_ID": "xxxxxxxx"
            },
            "_links": {
                "self": [
                    {
                        "href":
"http://<ipaddress>:<port>/api/arsys/v1/entry/HPD:Help%20Desk/I
NC000000454748"
                    }
                ]
            }
        }
    ],
    "_links": {
        "next": [
            {
                "href":
"http://<ipaddress>:<port>/api/arsys/v1/entry/HPD:Help%20Desk/?
q=%27Assigned%20Group%27=%22NOC%22%20AND%20%27Last%20Modified%2
0Date%27%3E%222018-11-
01T15:48:00%22%20AND%20%27Last%20Modified%20Date%27%3C%222018-
12-
07T15:48:00%22&offset=1&limit=1&fields=values(Incident%20Number
,Description,Entry%20ID,Detailed%20Decription,Submit%20Date,Sta
tus,Last%20Resolved%20Date,Assigned%20Group,%20Last%20Modified%
20Date)"

            }

        ],

        "self": [

            {

                "href":
"http://<ipaddress>:<port>/api/arsys/v1/entry/HPD:Help%20Desk/?
q=%27Assigned%20Group%27=%22NOC%22%20AND%20%27Last%20Modified%2
0Date%27%3E%222018-11-
01T15:48:00%22%20AND%20%27Last%20Modified%20Date%27%3C%222018-
12-
07T15:48:00%22&fields=values(Incident%20Number,Description,Entr
y%20ID,Detailed%20Decription,Submit%20Date,Status,Last%20Resolv
ed%20Date,Assigned%20Group,%20Last%20Modified%20Date)&limit=1"

            }

        ]

    }

}
```

9. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
10. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below.

Table 28– Sample Mandatory Mapping Parameters

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | entries.0. Number |
| Summary | JSON.Keys | entries.0.short_Description |
| Description | JSON.Keys | entries.0 Description |

| | | |
|---|---|---|
| CreationDate | JSON.Keys | entries.0.sys_create_on |
| StatusCode | JSON.Keys | entries.0.incident_Status |
| ResolvedDate | JSON.Keys | entries.0.closed_at |
| LastModifiedDate | JSON.Keys | entries.0.updated_on |



Mandatory Parameter Mapping

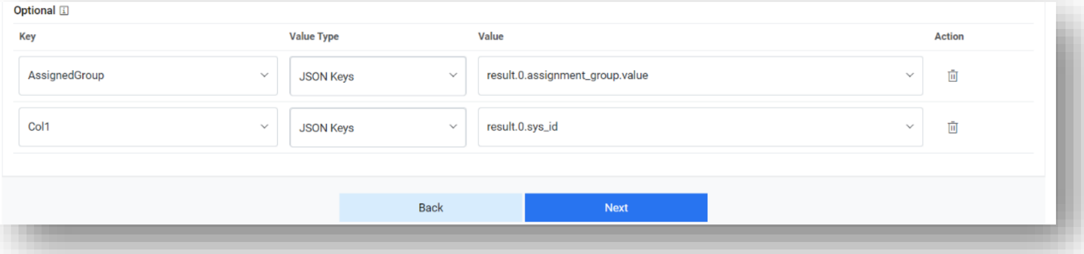
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.number |
| Summary | JSON Keys | result.0.short_description |
| Description | JSON Keys | result.0.description |
| CreationDate | JSON Keys | result.0.sys_created_on |
| StatusCode | JSON Keys | result.0.incident_state |
| ResolvedDate | JSON Keys | result.0.closed_at |
| LastModifiedDate | JSON Keys | result.0.sys_updated_on |

Figure 183 – Mandatory Parameter Mapping

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 29 – Sample Optional Mapping Parameters

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | entries.0..assignment_group.value |
| Col1 | JSON.Keys | entries.0.sys_id |



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | result.0.assignment_group.value | |
| Col1 | JSON Keys | result.0.sys_id | |

Figure 184 – Optional Parameter Mapping

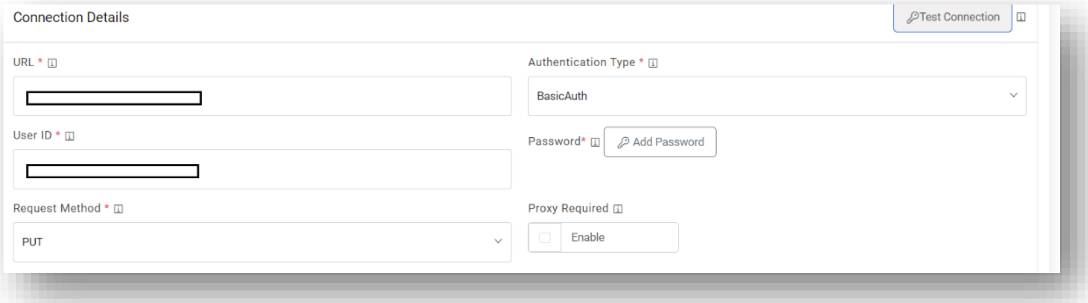
11. Click Next to proceed to Release Rules Configuration.
12. On **Release Rules** tab, type in the details as per the requirement.
13. In the **Connection Details** section, enter the following details:

– **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.

– **Sample URL** - http://URL/api/arsys/v1/entry/HPD:IncidentInterface/#TicketID#|#TicketID1#

- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- Request Method – Select Request Method as PUT from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

Test Connection

URL * [redacted]

Authentication Type * BasicAuth

User ID * [redacted]

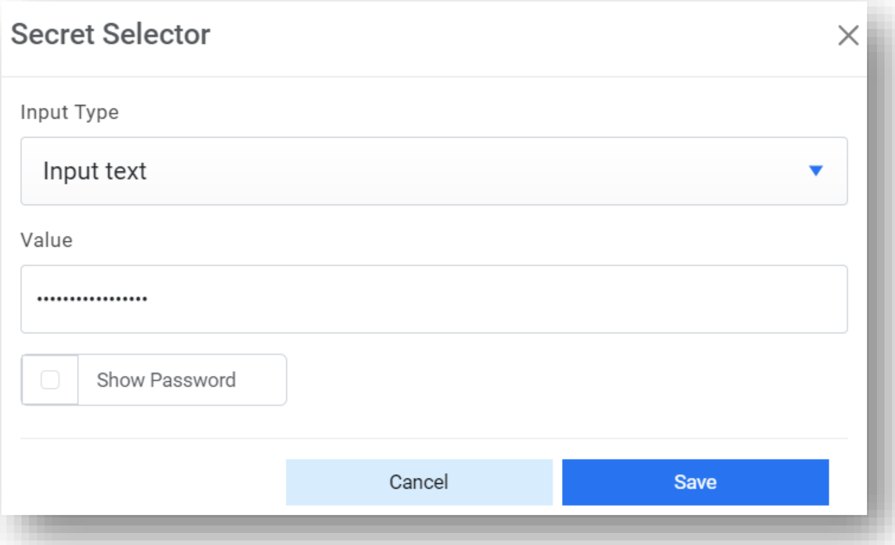
Password* Add Password

Request Method * PUT

Proxy Required □ Enable

Figure 185 – Create Data Source (Connection Details)

- **For password**, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



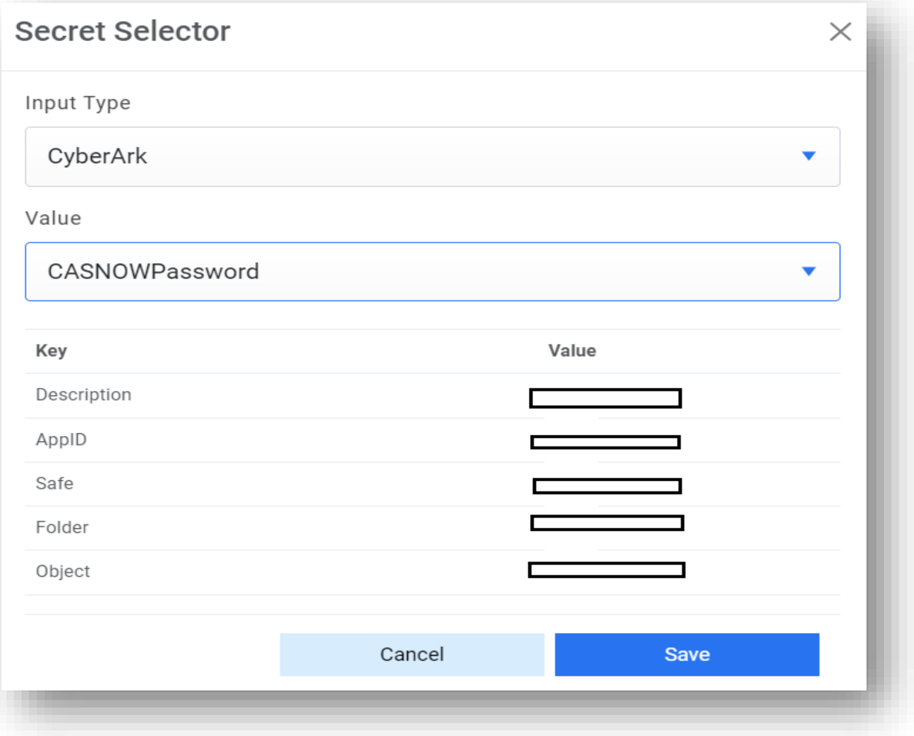
Secret Selector

Input Type: Input text

Value: ••••••••••••••••

□ Show Password

Figure 186 - Password in Plaintext



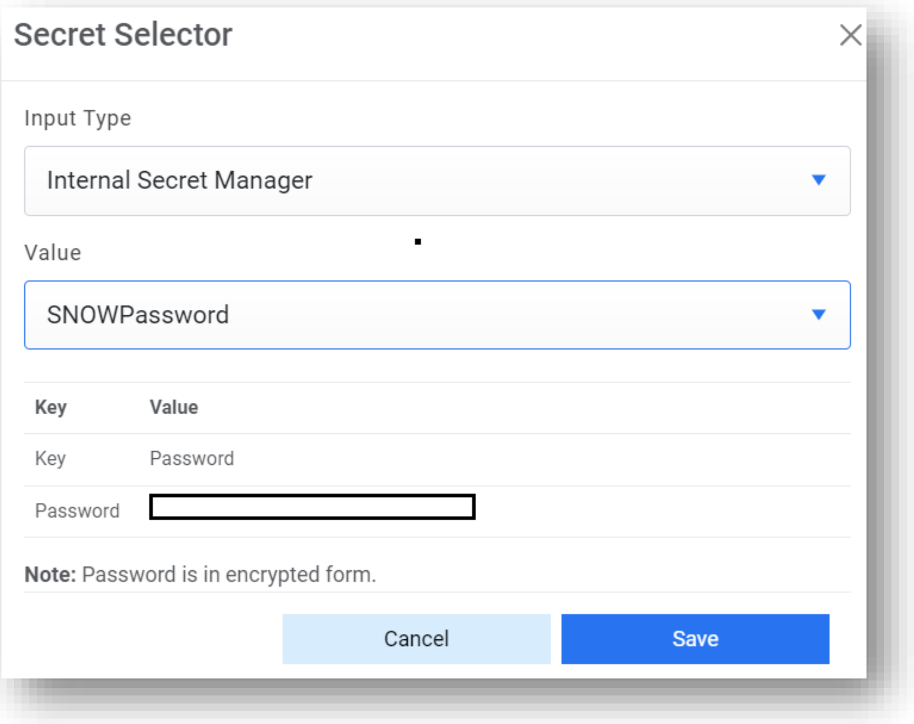
Secret Selector

Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 187 - Password from Key Vault (CyberArk)



Secret Selector

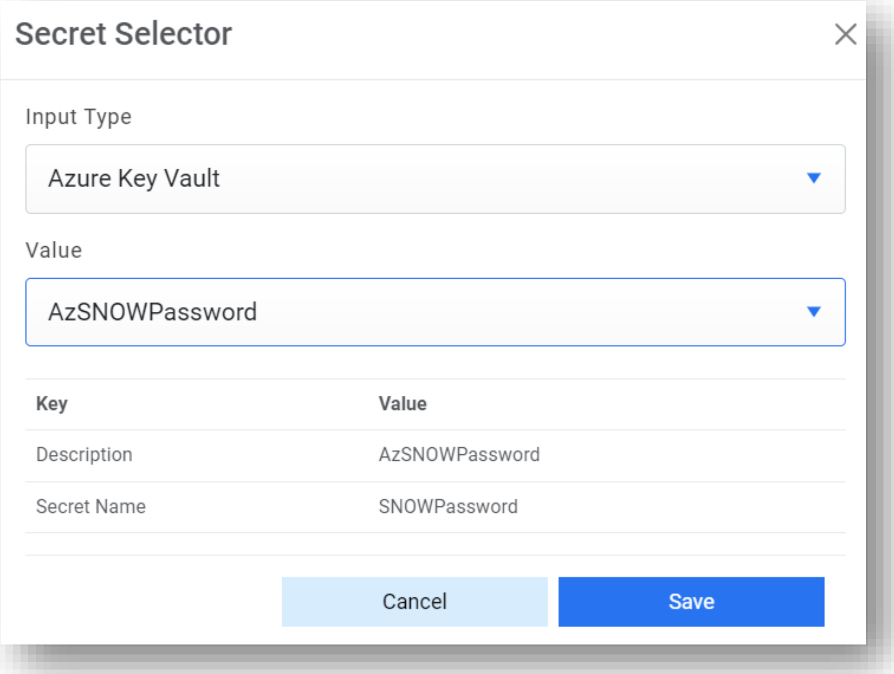
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 188 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

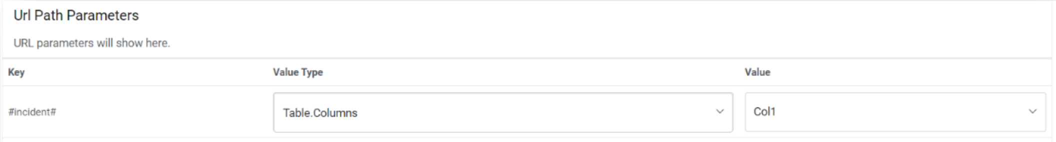
AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 189 - Password from Azure Key Vault

– **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs.

```
Key: #TicketID#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
“Col1”


Key: #TicketID1#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
“Col1”
```



Url Path Parameters

URL parameters will show here.

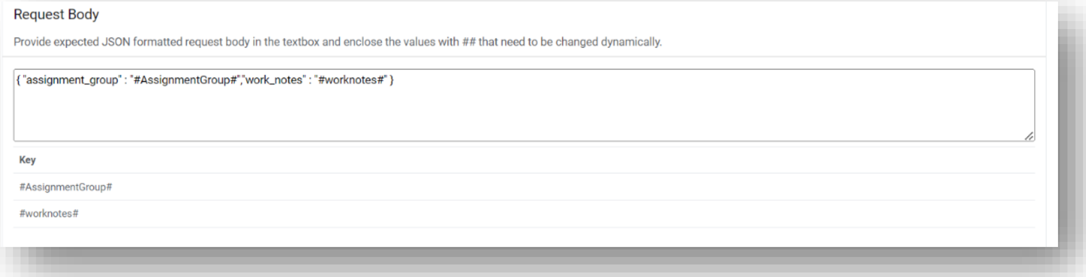
| Key | Value Type | Value |
|---|---|---|
| #incident# | Table.Columns | Col1 |

Figure 190 – Release Rules (URL Path Parameters)

– **Request Header Parameters –** Please enter the request header parameters as required.

14. **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below -

```
{ "assignment_group" : "#AssignmentGroup#","work_notes"       :
"#worknotes#" }
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

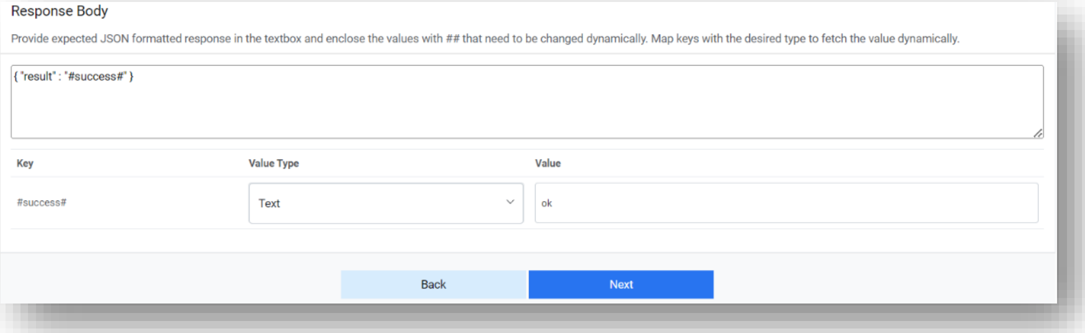
```
{ "assignment_group" : "#AssignmentGroup#","work_notes" : "#worknotes#" }
```

| Key |
|---|
| #AssignmentGroup# |
| #worknotes# |

Figure 191 – Release Rules (Request Body)

15. **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below:

```
{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

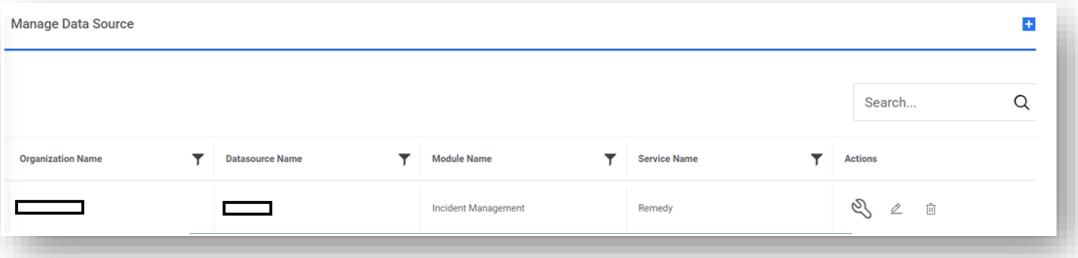
Figure 192 – Release Rules (Response Body)

16. **Response Key Value** mapping can be done as per the below table.

Table 30– Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | Success |

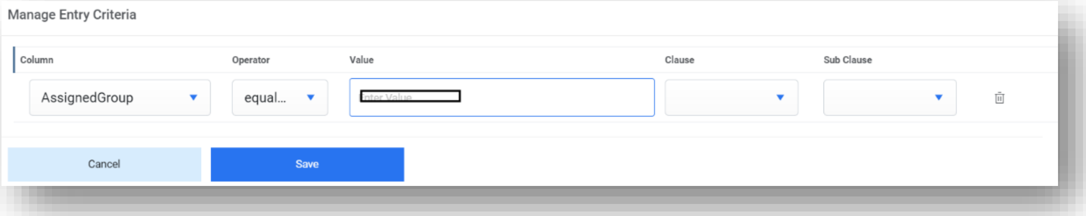
17. Click **Save** to add the data source.
18. In order to bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps –
    - Go to Configuration and click manage Data Sources.
    - On the **Data Sources** tab, click 🔧 next to the data source user wants to manage. **Manage Entry Criteria** screen appears.


Manage Data Source

| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| [redacted] | [redacted] | Incident Management | Remedy | |

Figure 193 – Manage Entry Criteria

- Select ‘AssignedGroup’ for the **Column** field and ‘equals to’ for the **Operator** field.
- Enter the sys_id of the assignment group in ServiceNow in the **Value** field.
- **Clause** and **Sub-Clause** fields can also be added based on requirement.


Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | [redacted] | | |

Figure 194 – Manage Entry Criteria (Cont.)

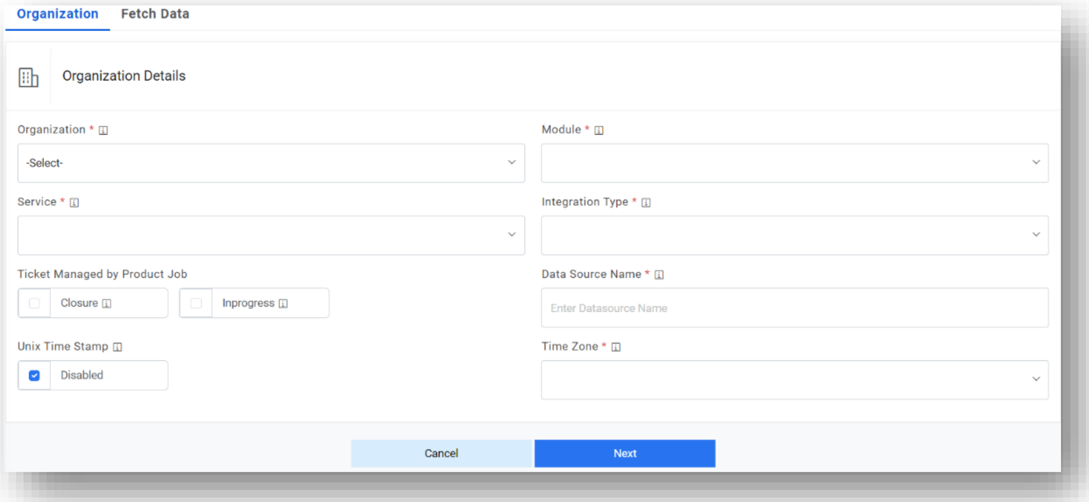
19. Click **Save**.

## 4.4 Integration with Cherwell ITSM

### 4.4.1 Incident Management

To create a data source for Incident Management, perform the following steps:

1. On the left menu bar, click Configuration -> Manage Data Sources.
2. The **Create Data Source** page appears with the following tabs:
   - Organization
   - Fetch Data
   - Release Rules



Organization | Fetch Data

Organization Details

Organization * -Select-

Module *

Service *

Integration Type *

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name * Enter Datasource Name

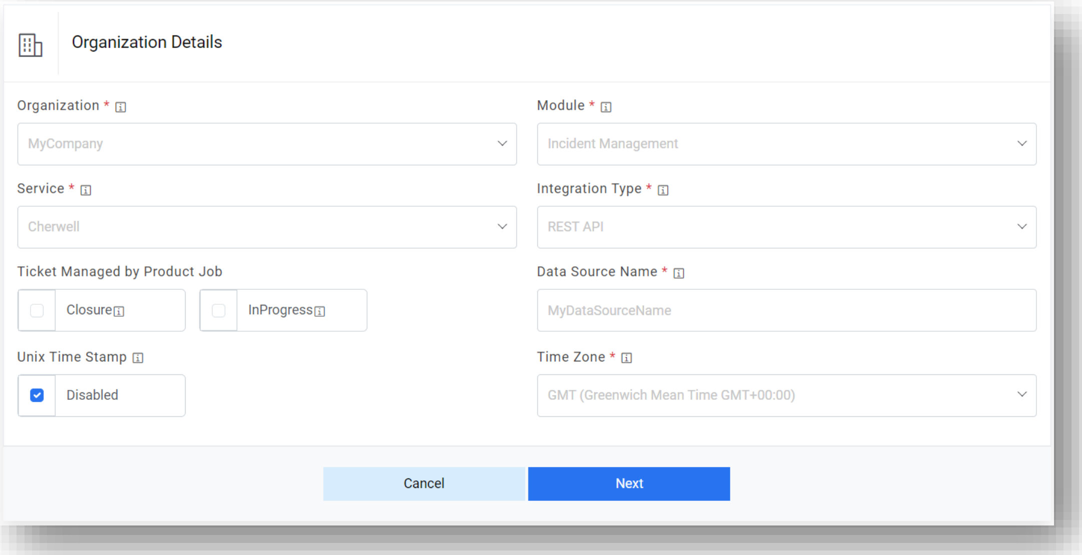
Unix Time Stamp: ☑ Disabled

Time Zone *

Figure 195 – Create Data Source

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab,
   - Select the **Organization Name** from the dropdown.
   - Select the **Module** as **Incident Management,** since we are configuring this data source for pulling the incident tickets.
   - Select the **Service** as **Cherwell** as we are configuring the data source for Cherwell
   - Select the **Integration Type** as **REST**, since we will be integrating through REST APIs.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select **Timestamp** to view the present data with date and time.
   - Click **Next**.



Organization Details

Organization * MyCompany

Module * Incident Management

Service * Cherwell

Integration Type * REST API

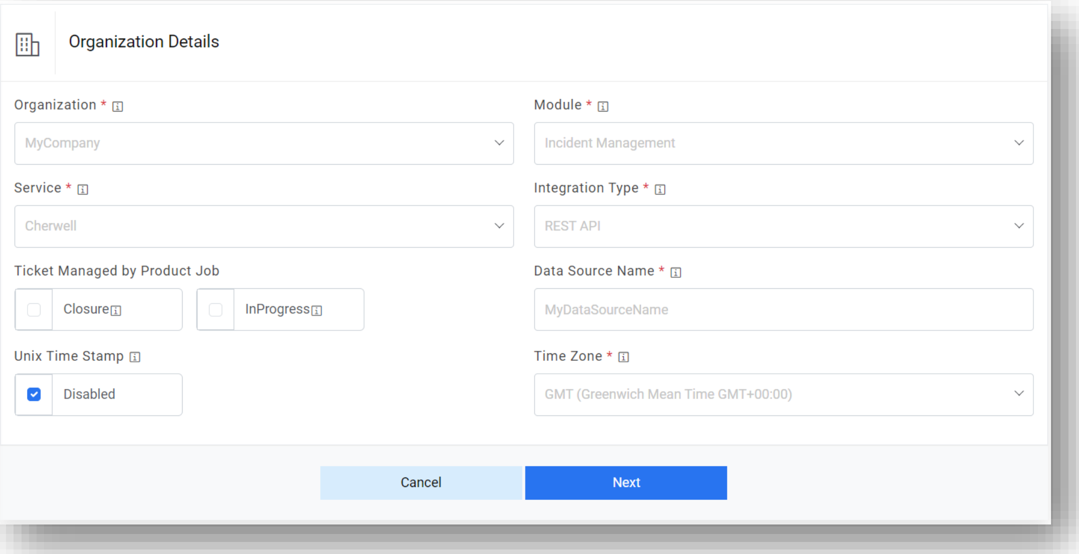
Ticket Managed by Product Job: □ Closure □ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp: ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 196 – Create Data Source (Cont.)



Organization Details

Organization * MyCompany

Module * Incident Management

Service * Cherwell

Integration Type * REST API

Ticket Managed by Product Job □ Closure □ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 197 – Create Data Source (Cont.)

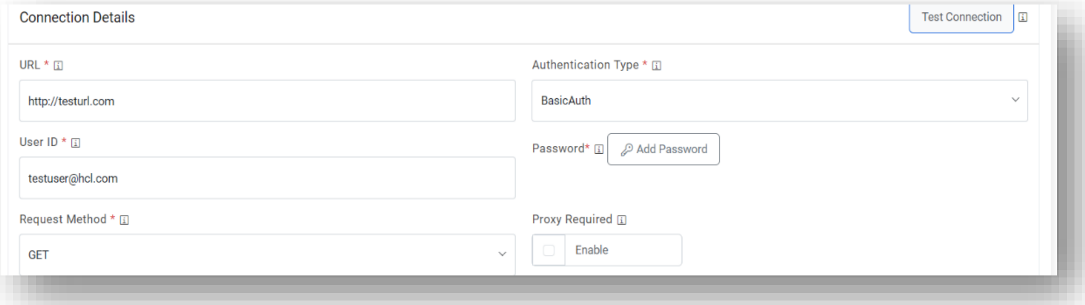
4. On the **Fetch Data** tab, type in the details as per the environment.
5. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL** – http://<iAutomate_API_URL>/iAutomateAPI/Request/GetIncidentTicketData/<Org_ID>?start_date>=#Start_Date#&end_date<=#End_Date#&
- Here, < iAutomate_API_URL > is the API URL of iAutomate where Push APIs are present and <Org_ID> is the OrgID for the organization for which you are creating the data source. It is available in Organization Master in Database.
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0

The user details that are entered here should be an API User

- Selection of **Basic / Windows** requires you to enter -
  - User Id
  - Password.
- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

URL * : http://testurl.com

Authentication Type * : BasicAuth

User ID * : testuser@hcl.com

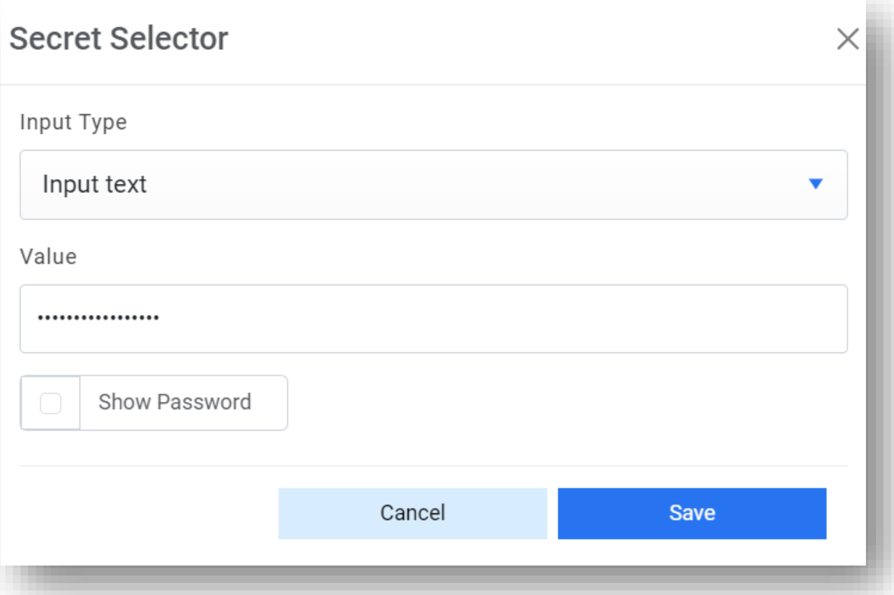
Password* : Add Password

Request Method * : GET

Proxy Required : □ Enable

Figure 198 – Create Data Source (Connection Details)

– **Password** – For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



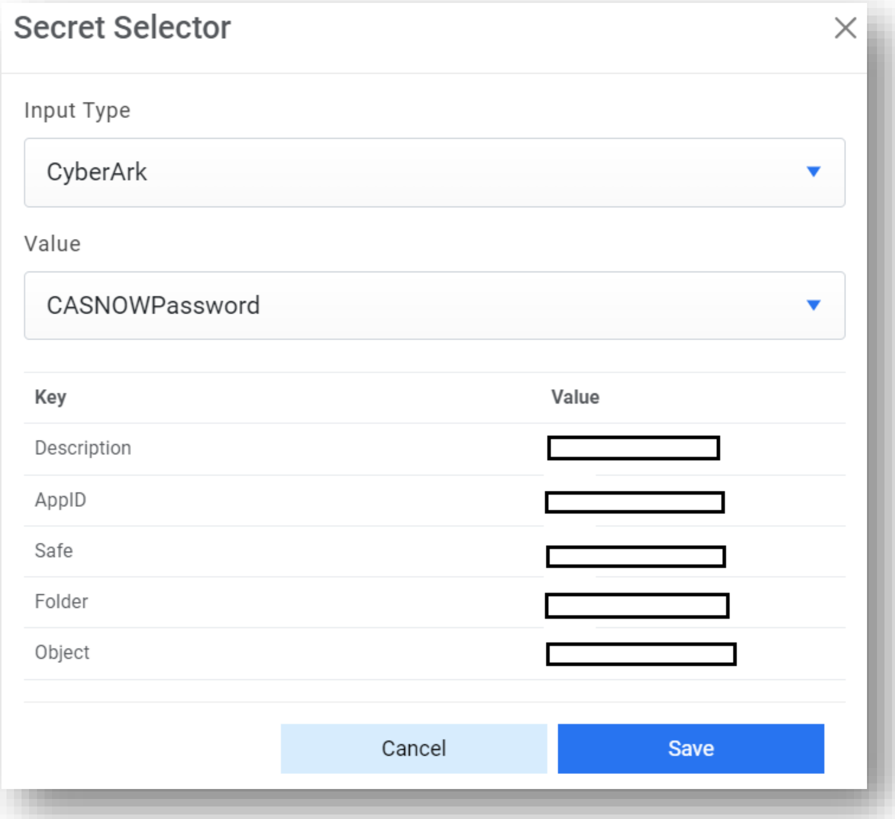
Secret Selector

Input Type: Input text

Value: •••••••••••••••••

□ Show Password

Figure 199 - Password in Plaintext



Secret Selector

Input Type

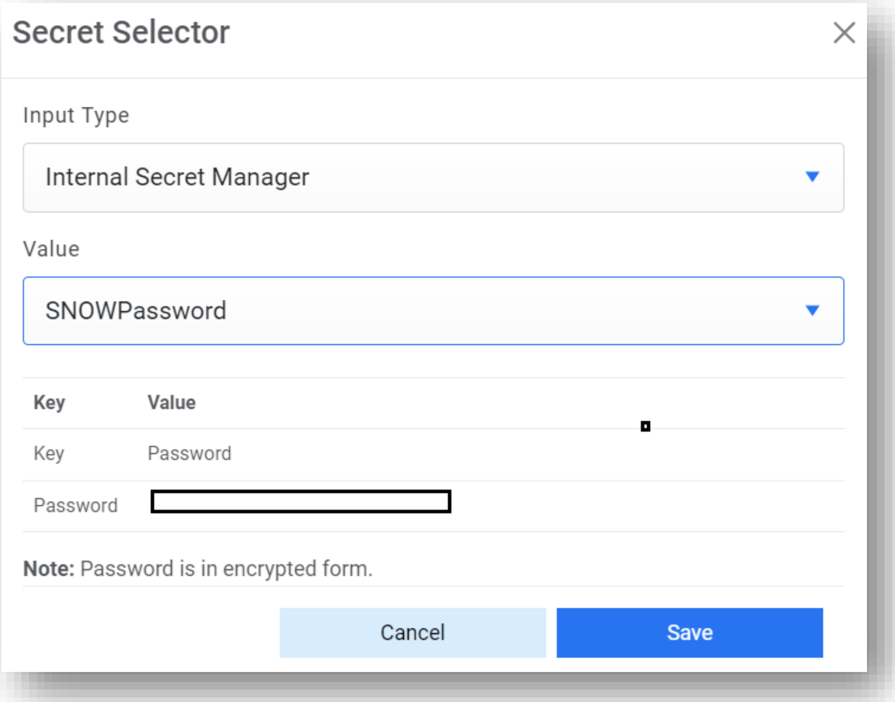
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | |
| AppID | |
| Safe | |
| Folder | |
| Object | |

Figure 200 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

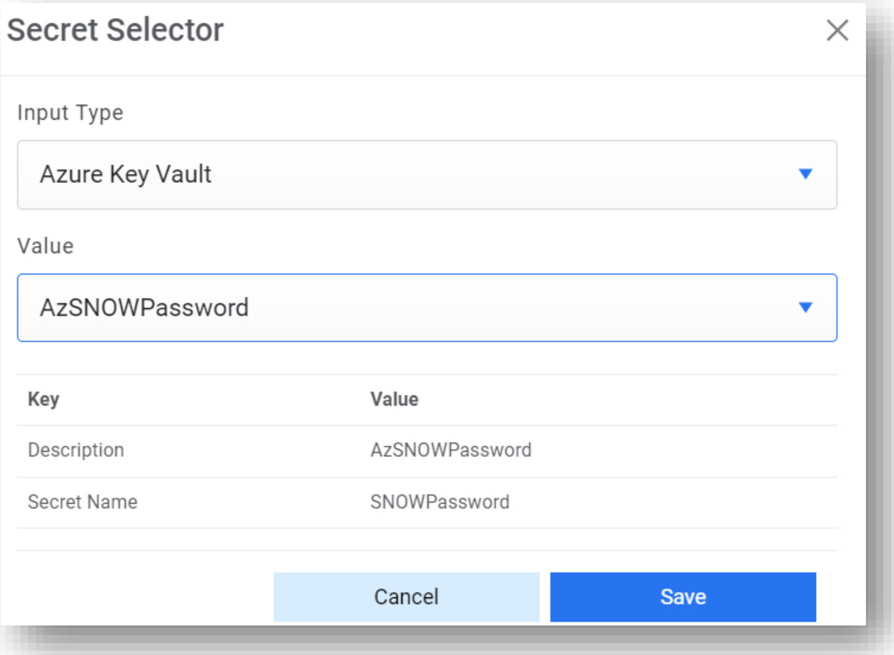
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 201 - Password from Secret Manager


Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

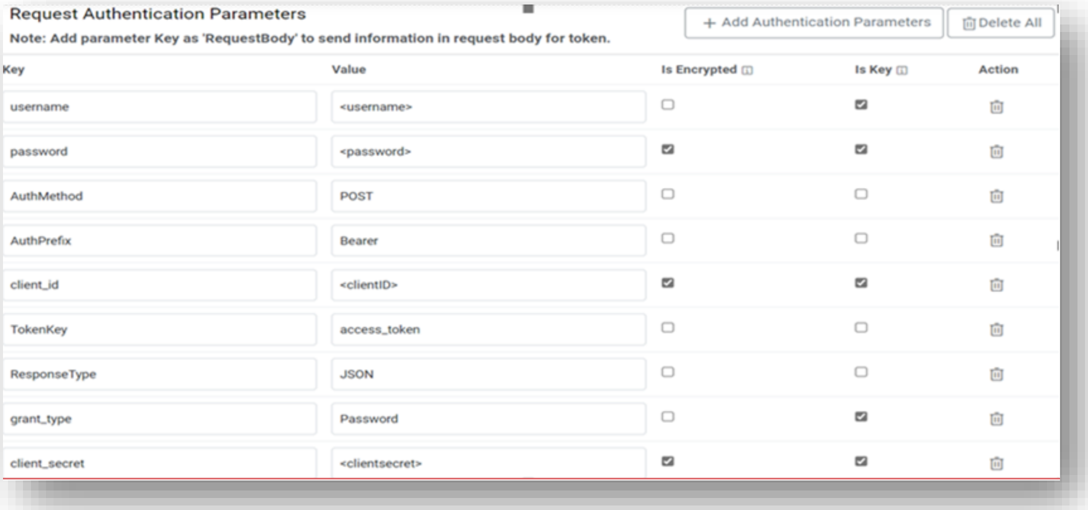
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 202 - Password from Azure Key Vault

- **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
- Based on the **Authentication Type**, add the parameters mentioned in the below table:

Table 31– Sample Authentication Parameters

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

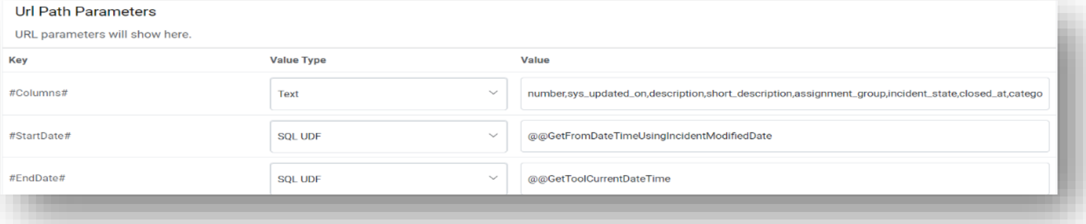
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | ☐ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | ☐ | ☐ | 🗑 |
| AuthPrefix | Bearer | ☐ | ☐ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | ☐ | ☐ | 🗑 |
| ResponseType | JSON | ☐ | ☐ | 🗑 |
| grant_type | Password | ☐ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 203 – Create Data Source (Request Authentication Parameters for OAuth2.0)

- **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #StartDate#

ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingIncidentPushStagingModifiedDate


Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,description,short_description,assignment_group,incident_state,closed_at,catego |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingIncidentModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 204– URL Path Parameters

- **Request Header Parameters –** Please enter the request header parameters as required.
- **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below:

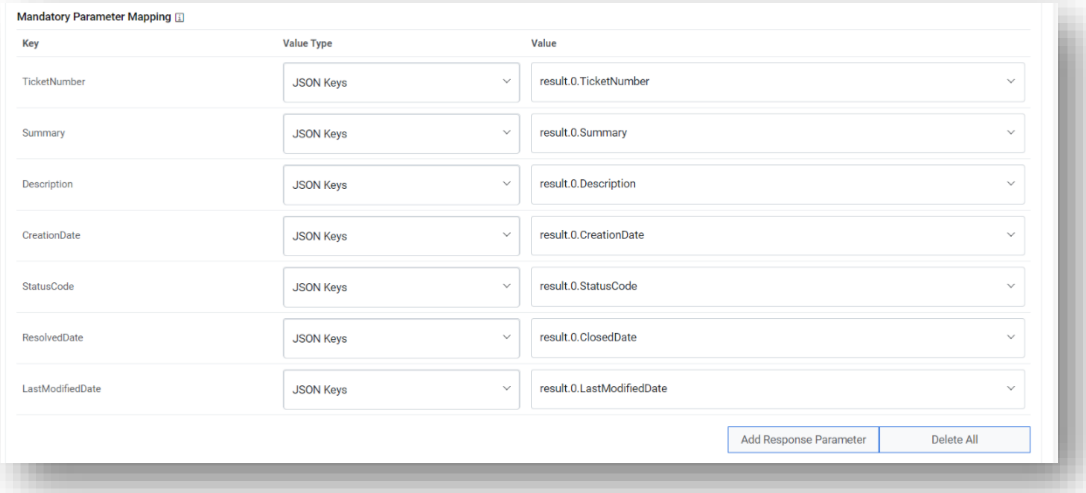
```
Response Body -

{"result": [{
```

```
            "TicketNumber": "INC0303860",

            "Summary": "testing",

            "Description": "testing data",

            "AssignedGroup": "xxxxxxxx",

            "StatusCode": "1",

            "CreationDate": "2020-05-06 12:06:05.000",

            "LastModifiedDate": "2020-05-06 12:06:05.000",

            "ClosedDate": "2020-05-06 12:26:05.000",

            "sys_id": "xxxxxxxx",

            "Col1": "",

            "Col2": "A",

            "Col3": "A",

            "Col4": "A",

            "Col5": "A"

        }]

}
```

6. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
7. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 32– Sample Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.TicketNumber |
| Summary | JSON.Keys | result.0.Summary |
| Description | JSON.Keys | result.0.Description |
| CreationDate | JSON.Keys | result.0.CreationDate |
| StatusCode | JSON.Keys | result.0.StatusCode |
| ResolvedDate | JSON.Keys | result.0.ClosedDate |
| LastModifiedDate | JSON.Keys | result.0.LastModifiedDate |



Mandatory Parameter Mapping

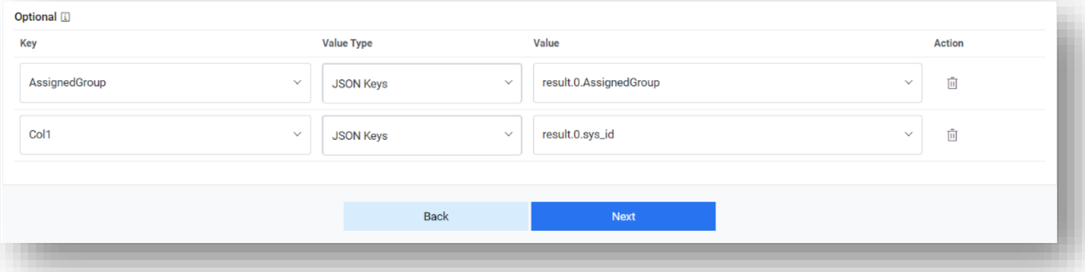
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.TicketNumber |
| Summary | JSON Keys | result.0.Summary |
| Description | JSON Keys | result.0.Description |
| CreationDate | JSON Keys | result.0.CreationDate |
| StatusCode | JSON Keys | result.0.StatusCode |
| ResolvedDate | JSON Keys | result.0.ClosedDate |
| LastModifiedDate | JSON Keys | result.0.LastModifiedDate |

Figure 205 – Mandatory Parameter Mapping

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 33– Sample Optional Parameters

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | result.0.AssignedGroup |
| Col1 | JSON.Keys | result.0.sys_id |



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | result.0.AssignedGroup | 🗑 |
| Col1 | JSON Keys | result.0.sys_id | 🗑 |

Figure 206 – Optional Parameter Mapping

8. Click **Next** to proceed to Release Rules.
9. On **Release Rules** tab, type in the details as per the requirement.
10. In the **Connection Details** section, enter the following details:

– **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.

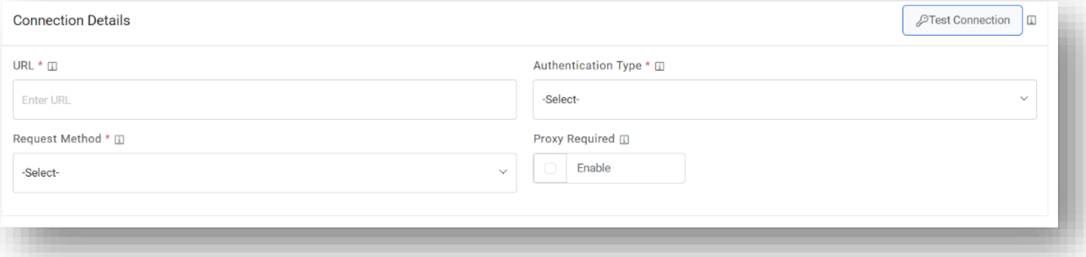
– **Sample URL** - *https://<url>.*

– **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.

– **Request Method** – Select Request Method as POST from the drop-down.

- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

Test Connection

URL *

Enter URL

Authentication Type *

-Select-

Request Method *

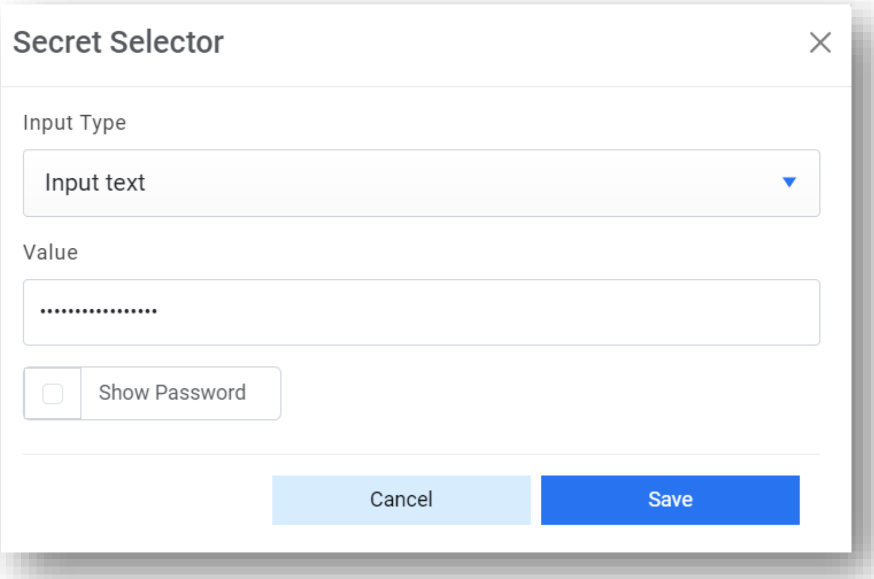
-Select-

Proxy Required

□ Enable

Figure 207 – Release Rules (Connection Details)

- **Password**- For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



Secret Selector

Input Type

Input text

Value

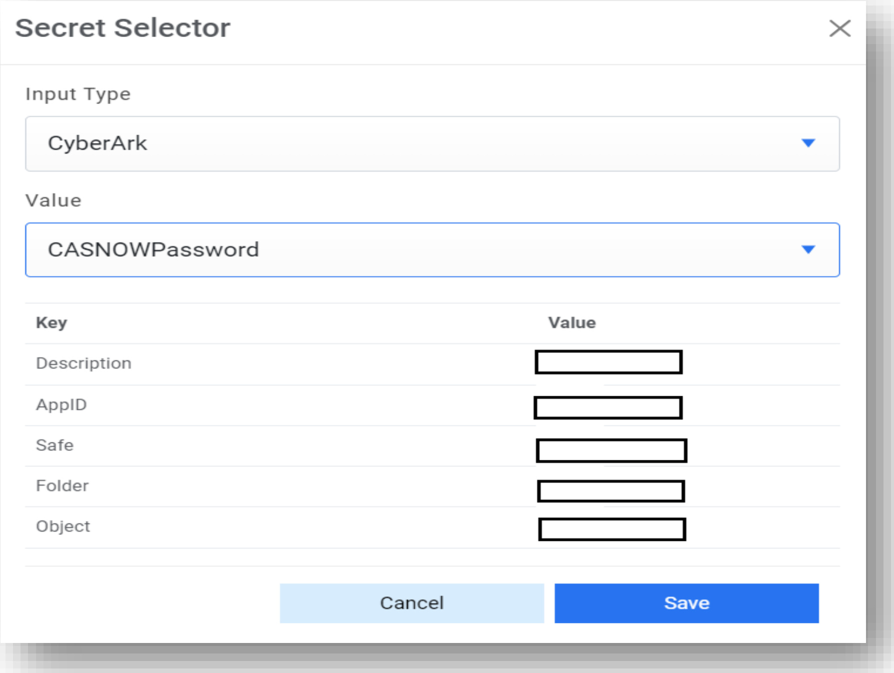
•••••••••••••••••

□ Show Password

Cancel

Save

Figure 208 - Password in plaintext



Secret Selector

Input Type

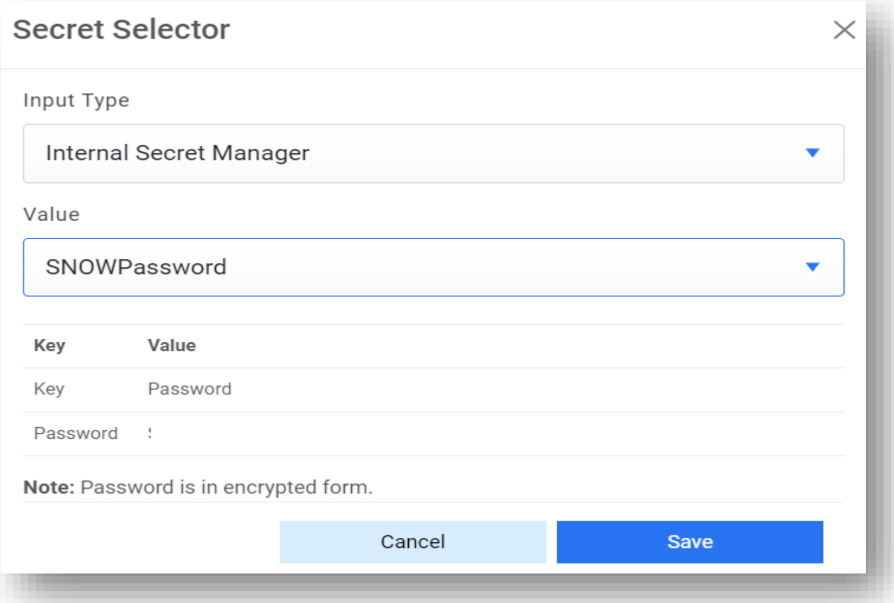
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 209 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

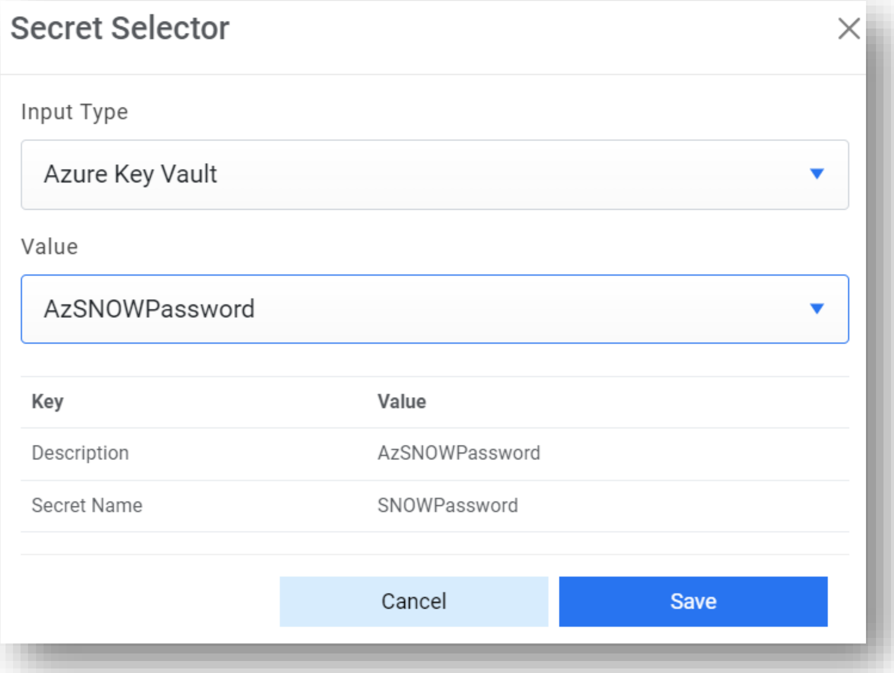
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | [redacted] |

**Note:** Password is in encrypted form.

Figure 210 - Password from Secret Manager



Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

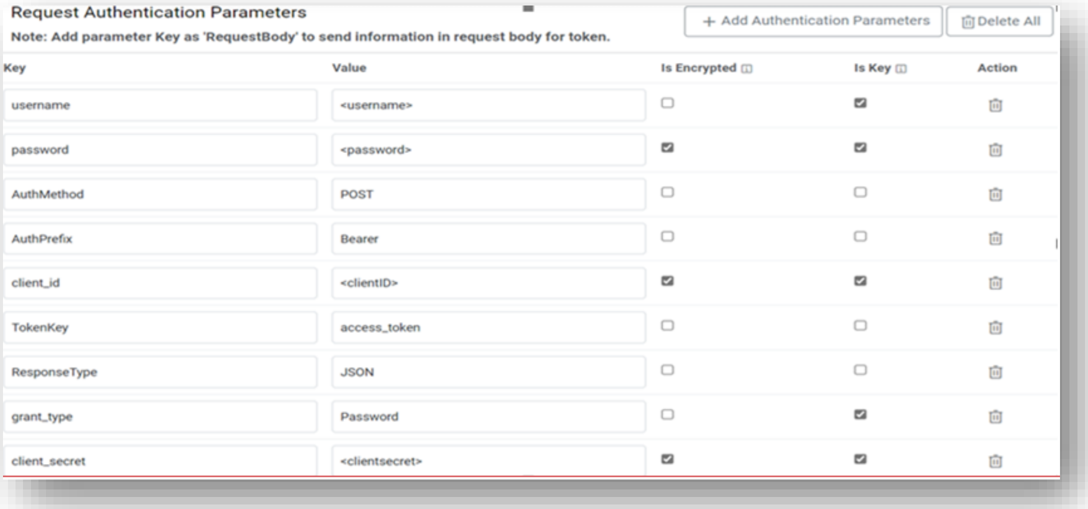
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 211 - Password from Azure Key Vault

11. **Request Authentication Parameters -** If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
12. Based on the **Authentication Type**, add the parameters mentioned in the below table:

Table 34 – Sample Authentication Parameters

| Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|
| grant_type | password | N | Y |
| username | <username> | N | Y |
| Password | <password> | Y | Y |
| client_id | <client_id> | N | Y |
| AuthPrefix | Bearer | N | N |
| AuthMethod | POST | N | N |
| ResponseType | JSON | N | N |
| TokenKey | access_token | N | N |



Request Authentication Parameters

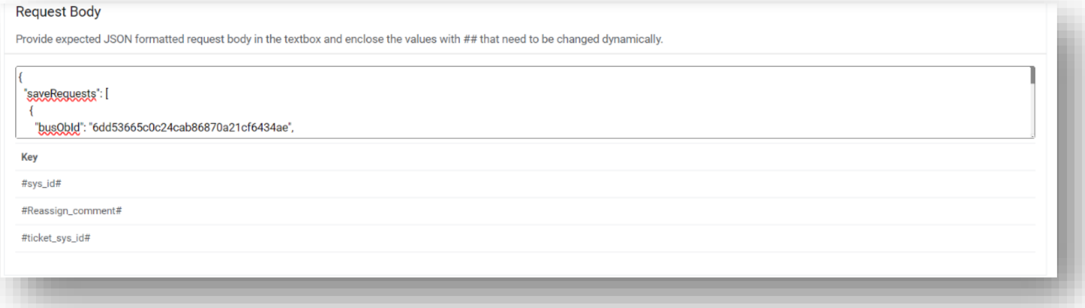
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | □ | □ | 🗑 |
| AuthPrefix | Bearer | □ | □ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | □ | □ | 🗑 |
| ResponseType | JSON | □ | □ | 🗑 |
| grant_type | Password | □ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 212 – Create Data Source (Request Authentication Parameters)

13. **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –
{
  "saveRequests": [
    {
      "busObId": "6dd53665c0c24cab86870a21cf6434ae",
      "busObPublicId": null,
      "busObRecId": "#sys_id#",
      "cacheKey": null,
      "cacheScope": "Tenant",
      "fields": [

      {
      "dirty": true,
      "displayName": null,
      "fieldId": "9339fc404e8d5299b7a7c64de79ab81a1c1ff4306c",
      "html": null,
      "name": null,
      "value": "Service Desk"
      },
      {
      "dirty": true,
      "displayName": null,
      "fieldId": "9339fc404e4c93350bf5be446fb13d693b0bb7f219",
      "html": null,
      "name": null,
      "value": ""
      },
      {
      "dirty": true,
      "displayName": null,
      "fieldId": "5eb3234ae1344c64a19819eda437f18d",
      "html": null,
      "name": null,
      "value": "Assigned"
      }

      ],
      "persist": true
    },
   {
      "busObId": "934d8181ba9d3a6a506d7643e1bc71f70fa9b47412",
      "busObPublicId": null,
      "busObRecId": null,
      "cacheKey": null,

      "cacheScope": "Tenant",

      "fields": [

        {

      "dirty": true,

      "displayName": null,

      "fieldId": "9341223bbcef1e2b8dfa6048a2bb4be1e94bad60ac",

      "html": null,

      "name": null,

      "value": "#Reassign_comment#"

    },

   {

      "dirty": true,

      "displayName": null,

      "fieldId": "9341222c4b89e253dd22b64d1fb16d0008bef6971f",

      "html": null,

      "name": null,

      "value": "#ticket_sys_id#"

    }

      ],

      "persist": true

    }

  ],

  "stopOnError": true}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

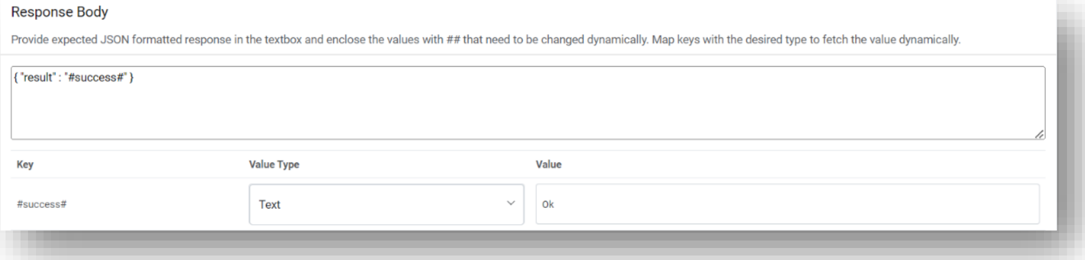
```
{
 "saveRequests": [
  {
   "busObId": "6dd53665c0c24cab86870a21cf6434ae",
```

| Key |
|---|
| #sys_id# |
| #Reassign_comment# |
| #ticket_sys_id# |

Figure 213 – Release Rules (Request Body)

14. **Response Body –** In this section, please enter the response body in JSON format. A sample request is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | Ok |

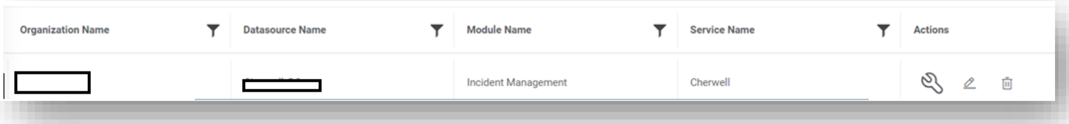
Figure 214 – Release Rules (Response Body)

15. **Response Key Value** mapping can be done as per the below table.

Table 35– Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

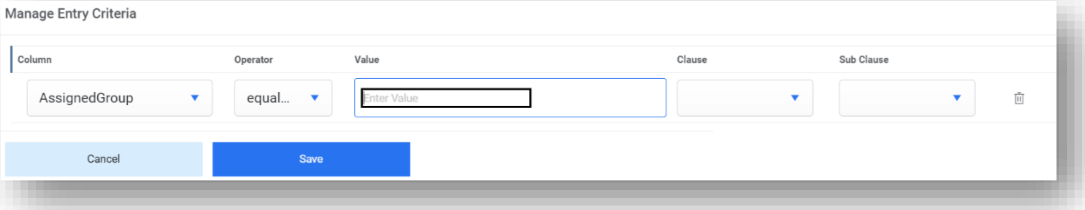
16. Click **Save** to add the data source.
17. To bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps:
    - Go to Action tab and click Manage Data Sources.
    - On the **Data Sources** tab, click 🔧 next to the data source user wants to manage. **Manage Entry Criteria** screen appears.



| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| [redacted] | [redacted] | Incident Management | Cherwell | |

Figure 215 – Manage Entry Criteria

- Select 'AssignedGroup' for the **Column field and** 'equals to' for the **Operator** field.
- Enter the sys_id of the assignment group in Cherwell in the **Value** field.
- **Clause** and **Sub-Clause** fields can also be added based on requirement.

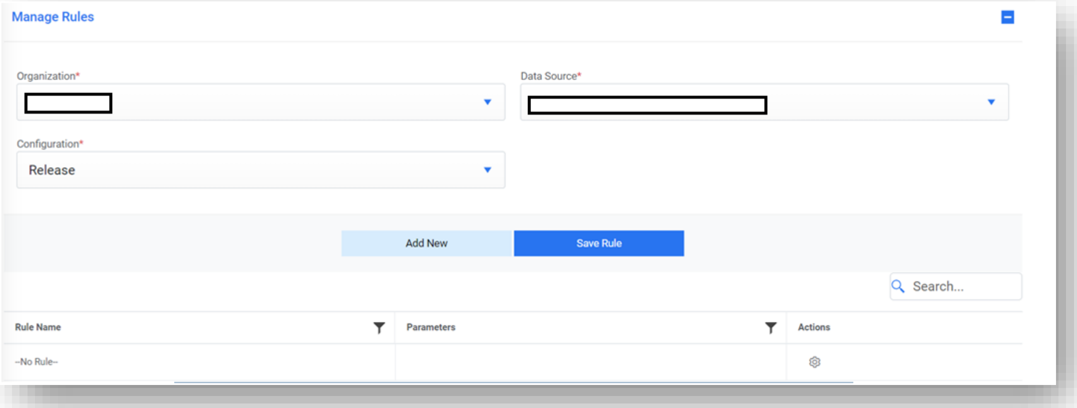


Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | Enter Value [redacted] | | |

Figure 216 – Manage Entry Criteria (Cont.)

18. Click **Save**.
19. To configure the Release rules for the data source created earlier, perform the below steps:
    - Go to Runbooks -> Manage Rules.
    - Select the **Organization** and the data source created from **Data Source** dropdown.



Manage Rules

Organization*: [redacted]

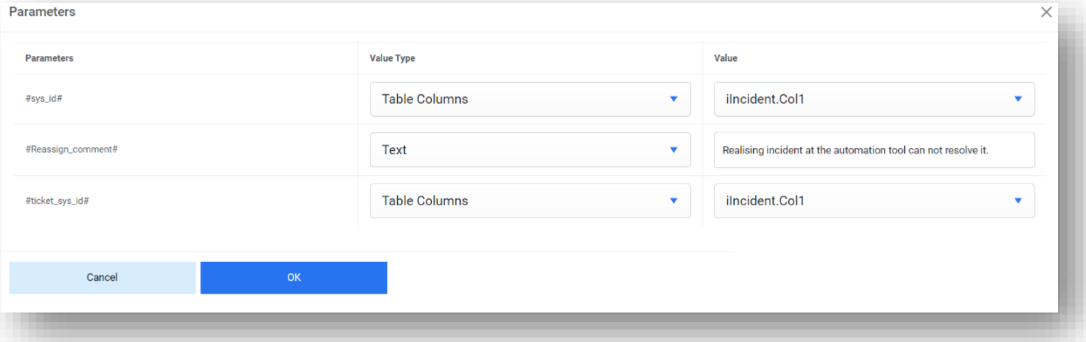
Data Source*: [redacted]

Configuration*: Release

| Rule Name | Parameters | Actions |
|---|---|---|
| –No Rule– | | ⚙ |

Figure 217 – Manage Release Rules

- Click on ⚙ corresponding to **–No Rule–**
- Map the parameters #sys_id# to the column in which sys_id was mapped while performing the mandatory parameter mapping while data source creation.
- Mention the reason for releasing ticket in #reassign_comments#.
- Map #ticket_sys_id# again to the column in which sys_id was mapped while performing the mandatory parameter mapping while data source creation.

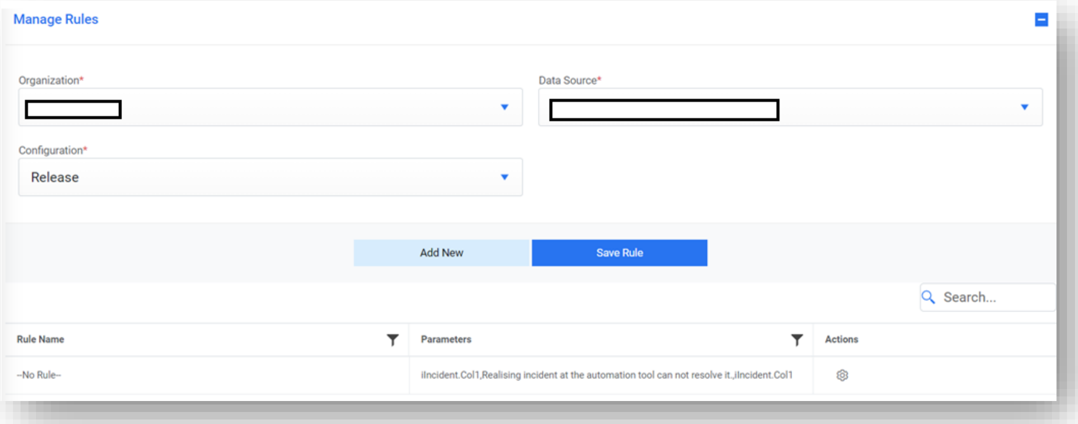


Parameters

| Parameters | Value Type | Value |
|---|---|---|
| #sys_id# | Table Columns | iIncident.Col1 |
| #Reassign_comment# | Text | Realising incident at the automation tool can not resolve it. |
| #ticket_sys_id# | Table Columns | iIncident.Col1 |

Figure 218 – Manage Release Rules (cont.)

20. Click **OK**.



Manage Rules

Organization*: [redacted]

Data Source*: [redacted]

Configuration*: Release

| Rule Name | Parameters | Actions |
|---|---|---|
| --No Rule-- | iIncident.Col1,Realising incident at the automation tool can not resolve it.,iIncident.Col1 | ⚙ |

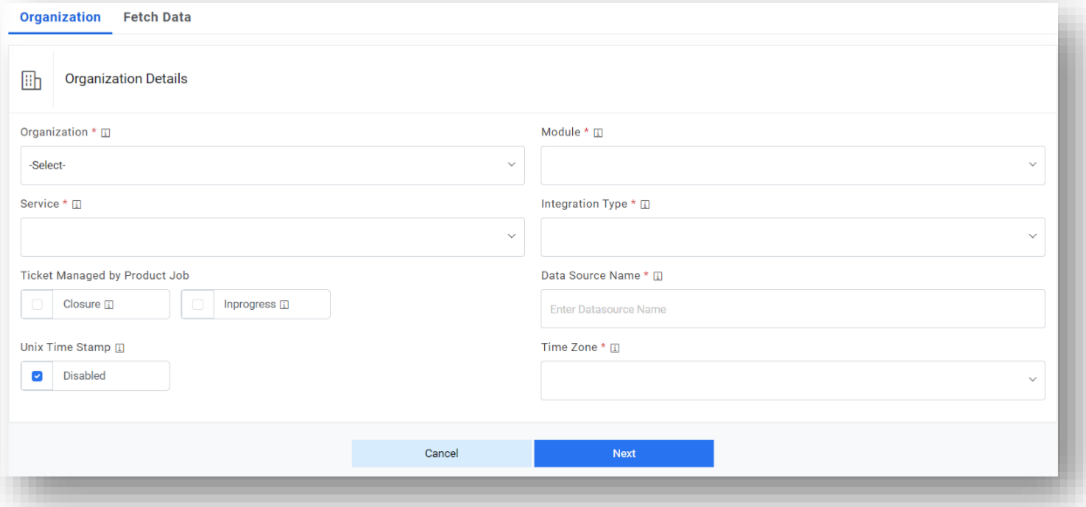
Figure 219 – Manage Release Rules (cont.)

21. Click Save Rule.

### 4.4.2 Service Request Task Management

To create a data source for Service Request Task Management, perform the following steps:

1. On the left menu bar, click Configuration -> Manage Data Sources.
2. The **Create Data Source** page appears with the following tabs:
    - Organization
    - Fetch Data
    - Release Rules



Organization | Fetch Data

Organization Details

Organization * : -Select-

Module * :

Service * :

Integration Type * :

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name * : Enter Datasource Name

Unix Time Stamp: ☑ Disabled

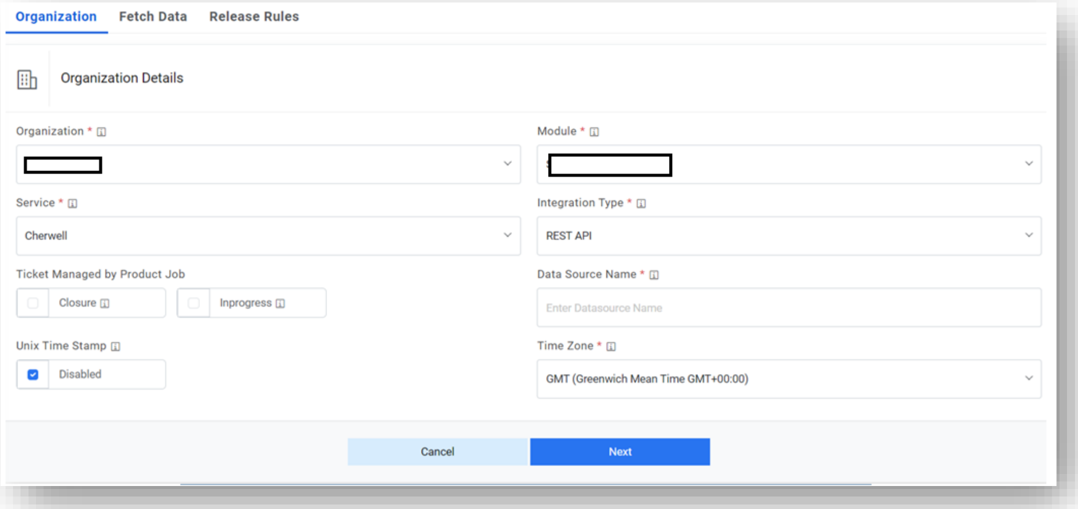
Time Zone * :

Cancel | Next

Figure 220 – Create Data Source

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab,
   - Select the **Organization Name** from the dropdown.
   - In the **Module** field, select **Service Request Task,** since we are configuring this data source for pulling the service request task tickets.
   - In the **Service** field, select **Cherwell Tool** as we are configuring the data source for Cherwell
   - In the **Integration Type** field, select **REST**, since we will be integrating through REST APIs.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select **Timestamp** to view the present data with date and time.
   - Click **Next**.



Organization | Fetch Data | Release Rules

Organization Details

Organization * : [redacted]

Module * : [redacted]

Service * : Cherwell

Integration Type * : REST API

Ticket Managed by Product Job: □ Closure □ Inprogress

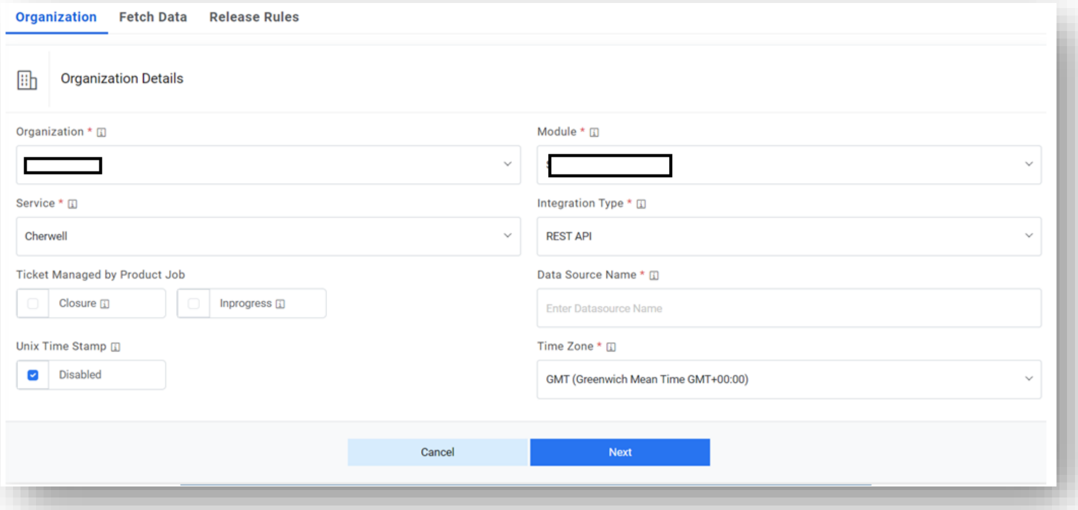
Data Source Name * : Enter Datasource Name

Unix Time Stamp: ☑ Disabled

Time Zone * : GMT (Greenwich Mean Time GMT+00:00)

Cancel | Next

Figure 221 – Create Data Source (Cont.)



Organization Details

Organization * [redacted]

Module * [redacted]

Service * Cherwell

Integration Type * REST API

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name * Enter Datasource Name

Unix Time Stamp: ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 222 – Create Data Source (Cont.)

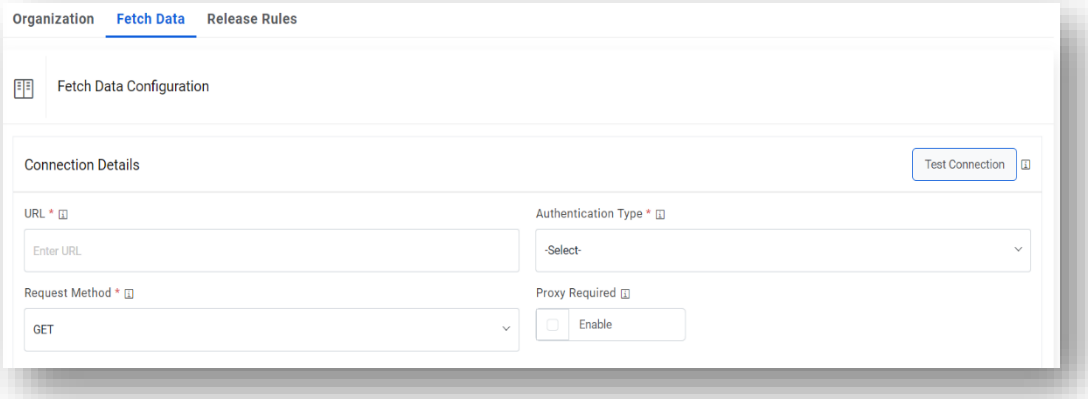
4. On the **Fetch Data** tab, populate the details as per the environment.
5. In the **Connection Details** section enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool
- Sample URL – http://<iAutomate_API_URL>/iAutomateAPI/Request/GetSRTicketData/<Org_ID>?start_date>=#Start_Date#&end_date<=#End_Date#&
- Here, < iAutomate_API_URL > is the API URL of iAutomate where Push APIs are present and <Org_ID> is the OrgID for the organization for which you are creating the data source. It is available in Organization Master in Database.
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0

The user details that are entered here should be an API User

- Selection of **Basic / Windows** requires you to enter –
  - User Id
  - Password
- Selection of **OAuth 2.0** requires you to enter –
  - User Id
  - Password
  - Authentication URL
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.


Organization | Fetch Data | Release Rules

Fetch Data Configuration

Connection Details

Test Connection

URL *

Enter URL

Authentication Type *

-Select-

Request Method *

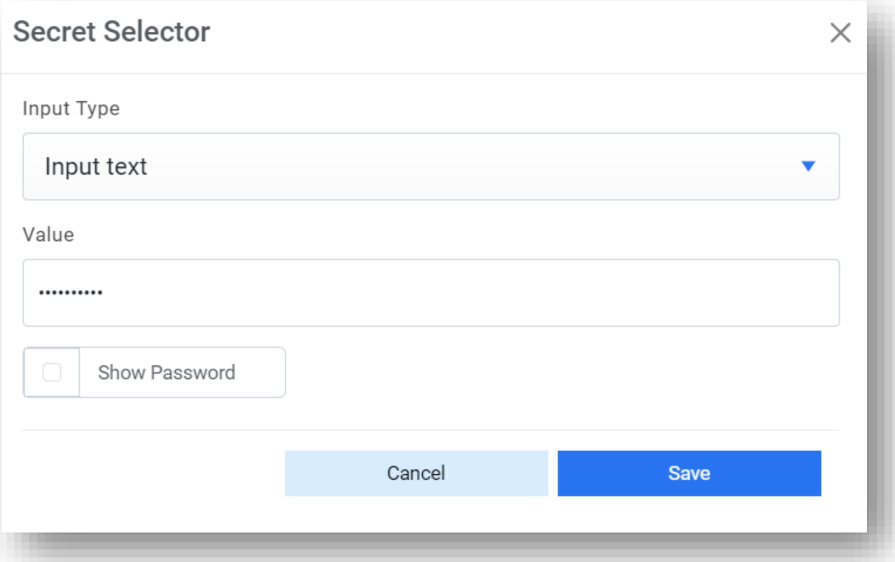
GET

Proxy Required

□ Enable

Figure 223 – Create Data Source (Connection Details)

– For **Password**, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.


Secret Selector

Input Type

Input text

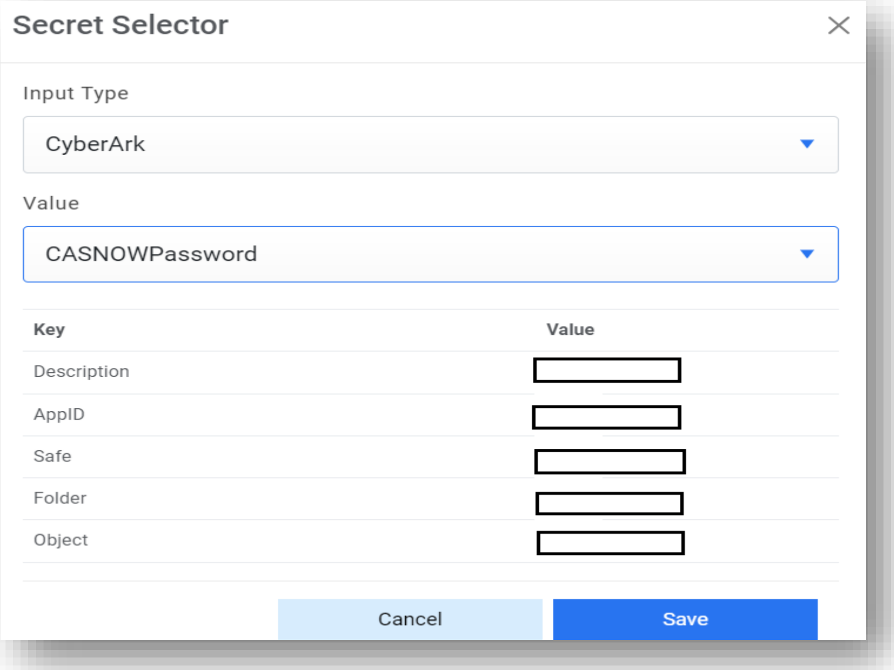
Value

••••••••••

□ Show Password

Cancel | Save

Figure 224 - Password in Plaintext

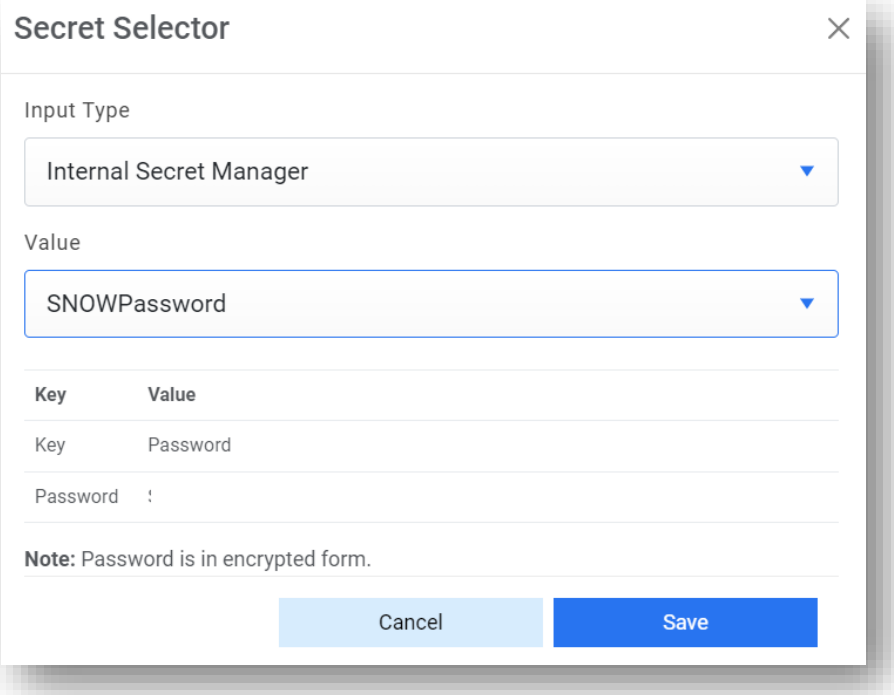


Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 225 - Password from Key Vault (CyberArk)



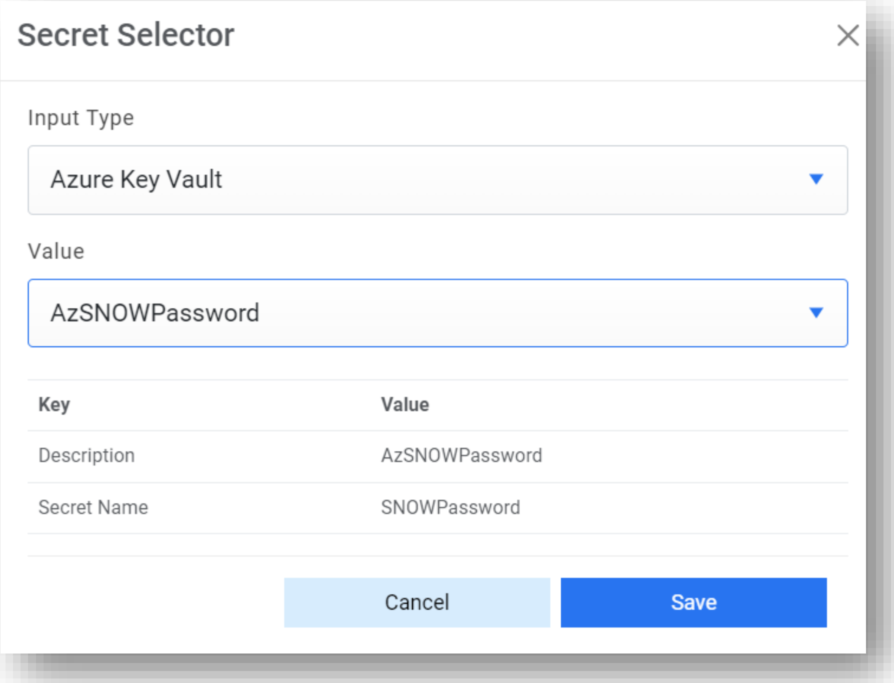
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | ! |

**Note:** Password is in encrypted form.

Figure 226 - Password from Secret Manager



Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

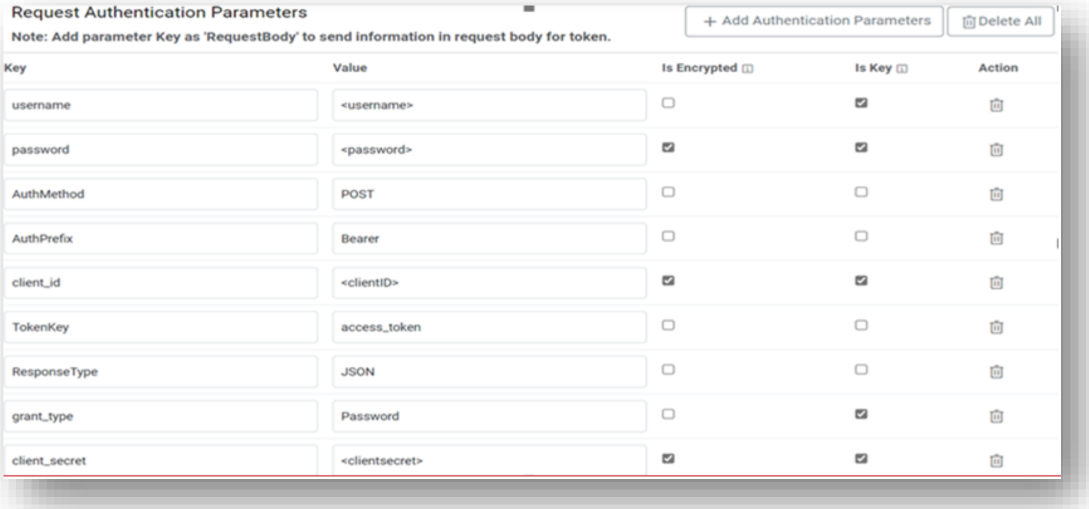
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 227 - Password from Azure Key Vault

6. **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
7. Based on the **Authentication Type**, add the parameters mentioned in the below table:

Table 36 – Sample Authentication Parameters

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

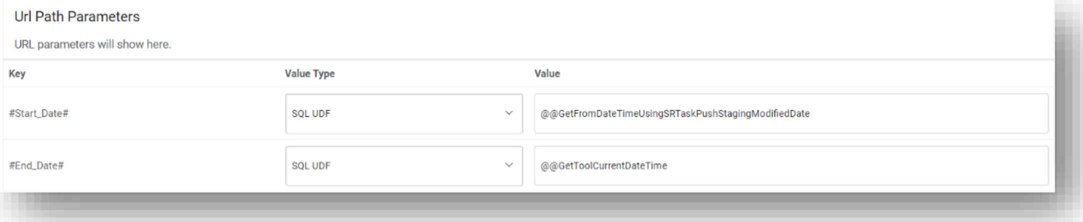
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | |
| password | <password> | ☑ | ☑ | |
| AuthMethod | POST | □ | □ | |
| AuthPrefix | Bearer | □ | □ | |
| client_id | <clientID> | ☑ | ☑ | |
| TokenKey | access_token | □ | □ | |
| ResponseType | JSON | □ | □ | |
| grant_type | Password | □ | ☑ | |
| client_secret | <clientsecret> | ☑ | ☑ | |

Figure 228 – Create Data Source (Request Authentication Parameters for OAuth2.0)

8. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #StartDate#

ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingSRTaskPushStagingModifiedDate



Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Start_Date# | SQL UDF | @@GetFromDateTimeUsingSRTaskPushStagingModifiedDate |
| #End_Date# | SQL UDF | @@GetToolCurrentDateTime |

Figure 229– URL Path Parameters

9. **Request Header Parameters –** Please enter the request header parameters as required.
10. **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below:

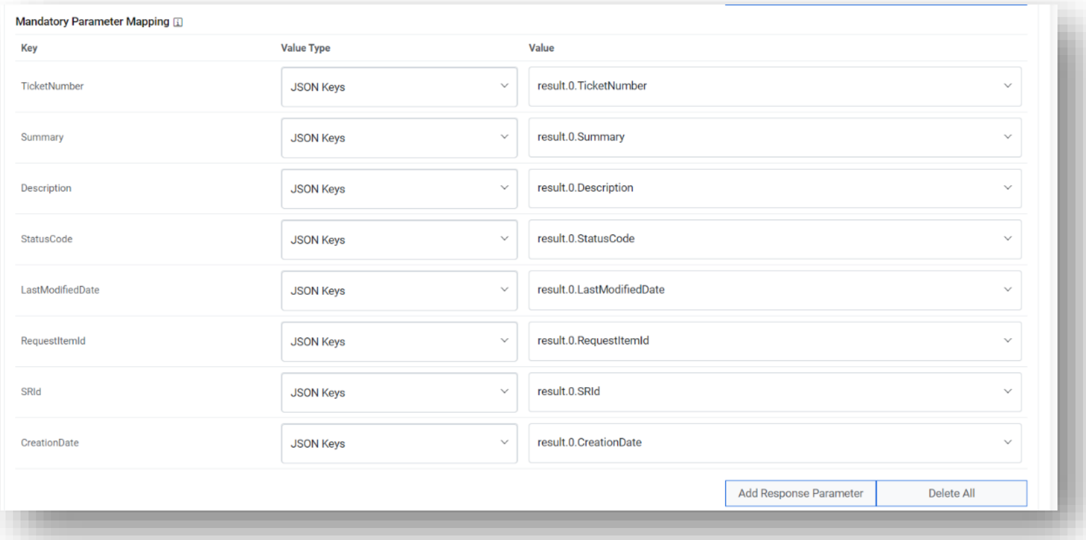
```
Response Body –

{"result": [{

            "TicketNumber": "SRTask0303863",

            "Summary": "testing",

            "Description": "testing data",

            "RequestItemId": "12345",

            "SRId": "2b535ab3dbc988506d7550d3dc96190e",

            "AssignedGroup": "",

            "StatusCode": "1",

            "CreationDate": "2020-05-07 05:06:05.000",

            "LastModifiedDate": "2020-05-07 05:54:54.000",

            "sys_id": "",

            "Col1": "",

            "Col2": "",

            "Col3": "",

            "Col4": "",

            "Col5": "",

            "iAutomate_CreatedDateInGMT":            "2020-05-08
09:14:24.903",

            "iAutomate_UpdatedDateInGMT":            "2020-05-08
09:14:24.903"

        }

]}
```

11. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
12. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 37– Sample Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.TicketNumber |
| Summary | JSON.Keys | result.0.Summary |

| | | |
|---|---|---|
| Description | JSON.Keys | result.0.Description |
| StatusCode | JSON.Keys | result.0.StatusCode |
| LastModifiedDate | JSON.Keys | result.0.LastModifiedDate |
| RequestItemId | JSON.Keys | result.0.RequestItemId |
| SRId | JSON.Keys | result.0.SRId |
| CreationDate | JSON.Keys | result.0.CreationDate |



Mandatory Parameter Mapping

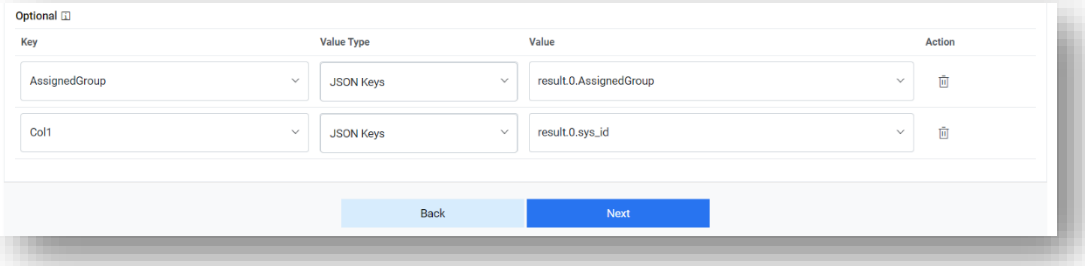
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.TicketNumber |
| Summary | JSON Keys | result.0.Summary |
| Description | JSON Keys | result.0.Description |
| StatusCode | JSON Keys | result.0.StatusCode |
| LastModifiedDate | JSON Keys | result.0.LastModifiedDate |
| RequestItemId | JSON Keys | result.0.RequestItemId |
| SRId | JSON Keys | result.0.SRId |
| CreationDate | JSON Keys | result.0.CreationDate |

Figure 230 – Mandatory Parameter Mapping

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 38– Sample Optional Parameters

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | result.0.AssignedGroup |
| Col1 | JSON.Keys | result.0.sys_id |



Optional

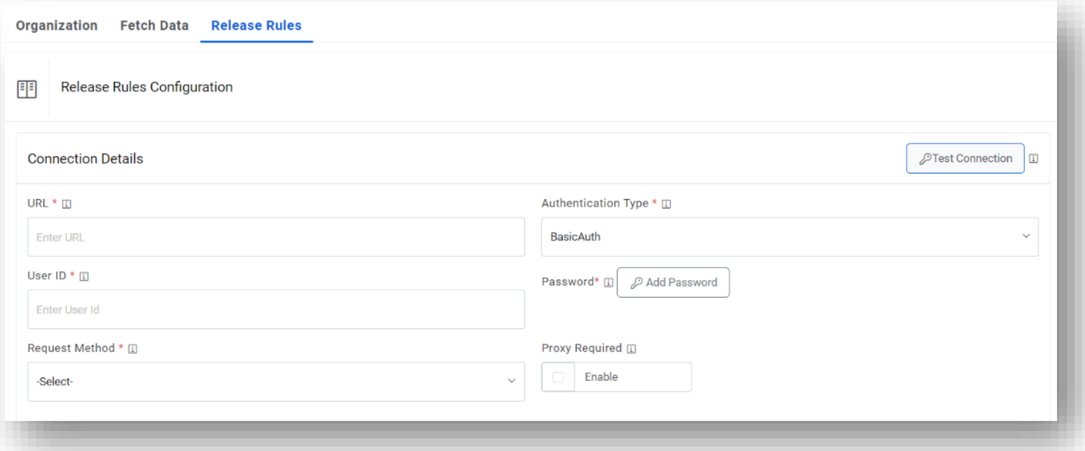
| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | result.0.AssignedGroup | |
| Col1 | JSON Keys | result.0.sys_id | |

Figure 231 – Optional Parameter Mapping

13. Click Next to proceed to Release Rules.
14. On **Release Rules** tab, type in the details as per the requirement.
15. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- **Sample URL** - *https://<url>.*
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- **Request Method** – Select Request Method as POST from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Release Rules Configuration

Connection Details

URL *: Enter URL

Authentication Type *: BasicAuth

User ID *: Enter User Id

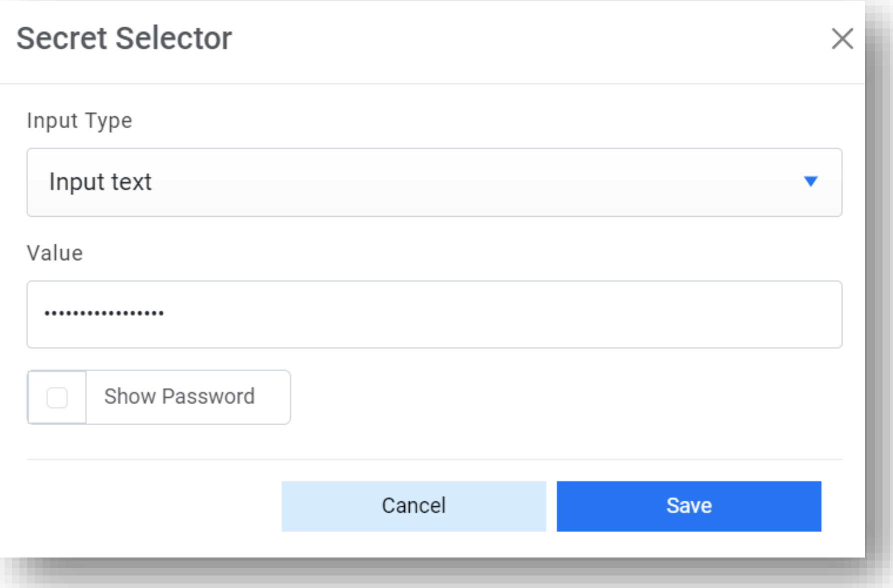
Password*

Request Method *: -Select-

Proxy Required: □ Enable

Figure 232 – Release Rules (Connection Details)

- For **Password**, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



Secret Selector

Input Type

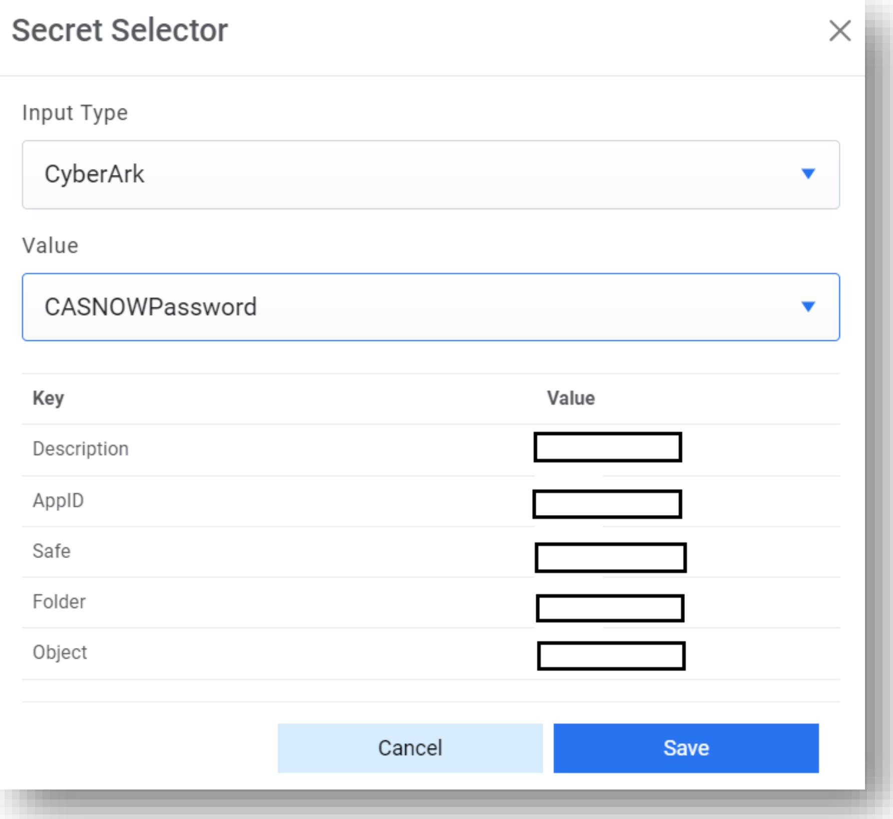
Input text

Value

•••••••••••••••••

☐ Show Password

Figure 233 - Password in plaintext



Secret Selector

Input Type

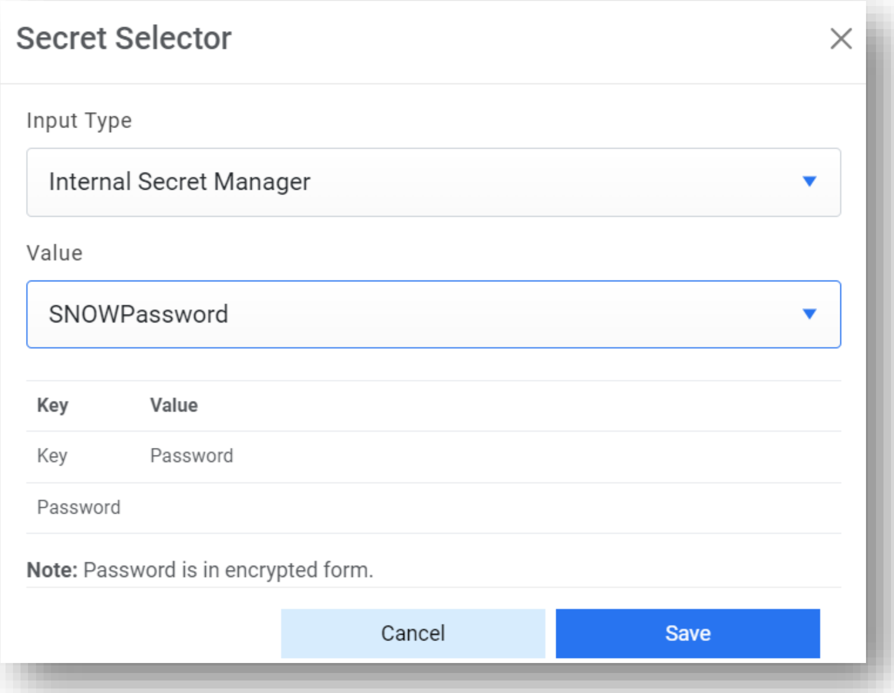
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | |
| AppID | |
| Safe | |
| Folder | |
| Object | |

Figure 234 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

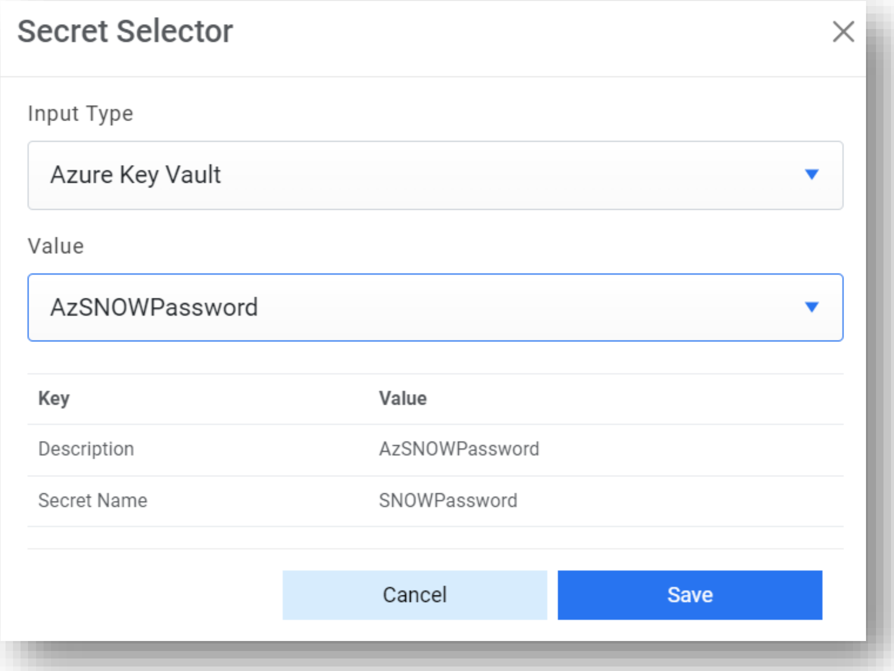
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 235 – Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

AzSNOWPassword

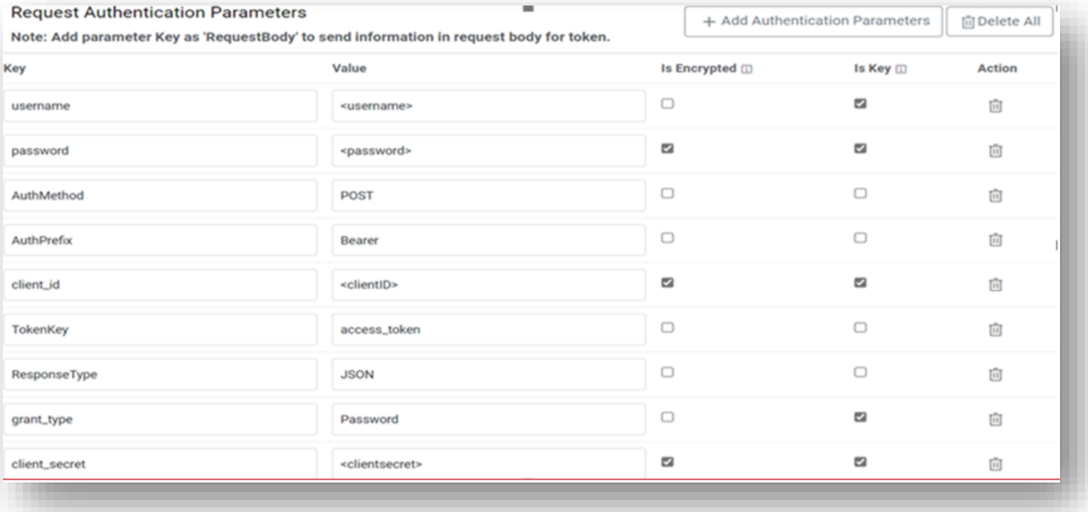
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 236 – Password from Azure Key Vault

16. **Request Authentication Parameters** - If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
17. Based on the **Authentication Type,** add the parameters mentioned in the below table:

Table 39 – Sample Authentication Parameters

| Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|
| grant_type | password | N | Y |
| username | <username> | N | Y |
| Password | <password> | Y | Y |
| client_id | <client_id> | N | Y |
| AuthPrefix | Bearer | N | N |
| AuthMethod | POST | N | N |
| ResponseType | JSON | N | N |
| TokenKey | access_token | N | N |



Request Authentication Parameters

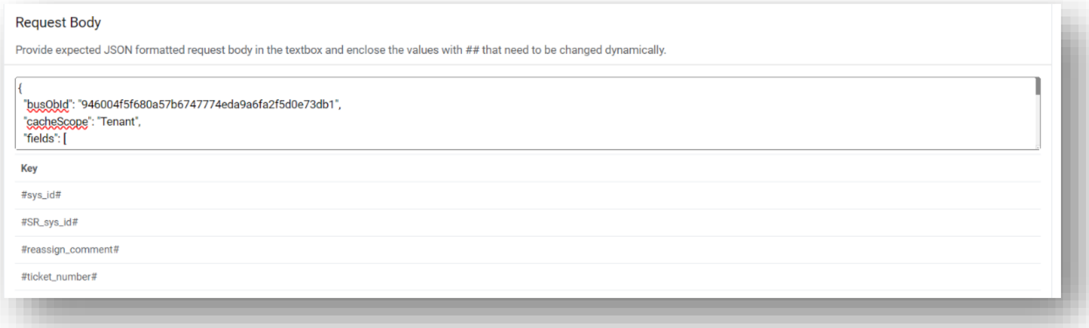
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | ☐ | ☑ | |
| password | <password> | ☑ | ☑ | |
| AuthMethod | POST | ☐ | ☐ | |
| AuthPrefix | Bearer | ☐ | ☐ | |
| client_id | <clientID> | ☑ | ☑ | |
| TokenKey | access_token | ☐ | ☐ | |
| ResponseType | JSON | ☐ | ☐ | |
| grant_type | Password | ☐ | ☑ | |
| client_secret | <clientsecret> | ☑ | ☑ | |

Figure 237 – Create Data Source (Request Authentication Parameters)

18. **Request Body -** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –

{

  "busObId": "946004f5f680a57b6747774eda9a6fa2f5d0e73db1",

  "cacheScope": "Tenant",

  "fields": [

    {

      "dirty": true,

      "displayName": "Task RecID",

      "fieldId": "946005353974025498ed1d4068936d72c8992d015c",

      "value": "#sys_id#"

    },

    {
      "dirty": true,
      "displayName": "Parent RecID",
      "fieldId": "9460053dd53d9888efddc34d3db0360cc5be25f567",
      "value": "#SR_sys_id#"
    },
    {
      "dirty": true,
      "displayName": "Journal Details",
      "fieldId": "946005008899c5f5c31caa43c99083519668f0ff33",
      "value": "#reassign_comment#"
    },
{
      "dirty": true,
      "displayName": "Ticket Number",
      "fieldId": "94602e208e8947bff420df4016b30962152556d5e2",
      "value": "#ticket_number#"
    },
    {
      "dirty": true,
      "displayName": "Assignment Team",
      "fieldId": "946005013472134fdc1b0649a685d41a4c73f6e179",
      "value": "Service Desk"
    },
    {
      "dirty": true,
      "displayName": "Status",
      "fieldId": "946004ff47672c8cda67da43a1945ce56f2f617855",
      "value": "New"
    },
    {
      "dirty": true,
      "displayName": "Task Type",
      "fieldId": "946004feb10853e55a192849c780773b2133028cc0",
      "value": "SR Task"
    },
    {
      "dirty": true,
      "displayName": "Reassigning",
      "fieldId": "946005a199ecde0a9cf0b748bb94e4040c2007540f",
      "value": "True"
    }
  ],
  "persist": true
}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

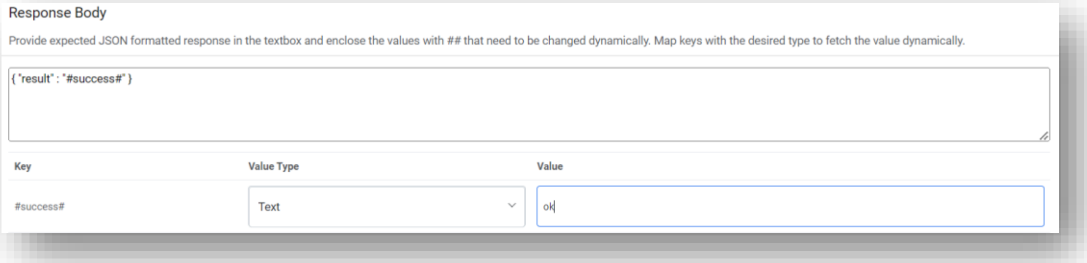
```
{
 "busObId": "946004f5f680a57b6747774eda9a6fa2f5d0e73db1",
 "cacheScope": "Tenant",
 "fields": [
```

| Key |
|---|
| #sys_id# |
| #SR_sys_id# |
| #reassign_comment# |
| #ticket_number# |

Figure 238 – Release Rules (Request Body)

19. **Response Body –** In this section, please enter the response body in JSON format. A sample request is mentioned below:

```
Response Body –
{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

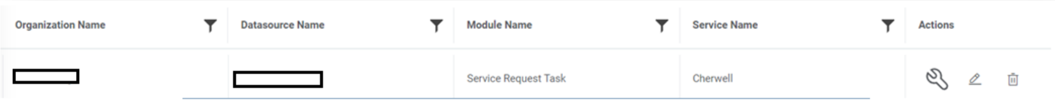
Figure 239 – Release Rules (Response Body)

20. **Response Key Value** mapping can be done as per the below table:

Table 40 – Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

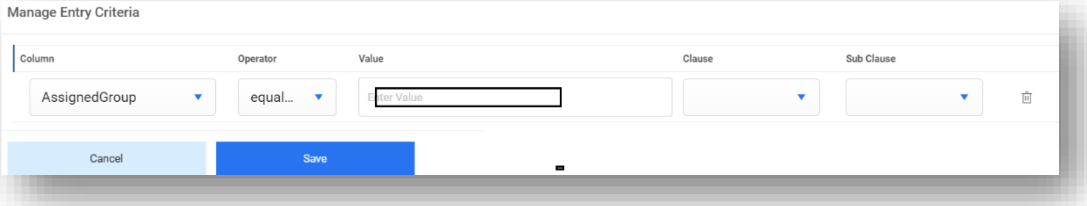
21. Click **Save** to add the data source.
22. To bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps:
    - Go to Configuration and click Manage Data Sources.
    - On the **Data Sources** tab, click 🔧 next to the data source user wants to manage. **Manage Entry Criteria** screen appears.



| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| [redacted] | [redacted] | Service Request Task | Cherwell | |

Figure 240 – Manage Entry Criteria

    - Select 'AssignedGroup' for the **Column** field and 'equals to' for the **Operator** field.
    - Enter the sys_id of the assignment group in Cherwell in the **Value** field.
    - **Clause** and **Sub-Clause** fields can also be added based on requirement.

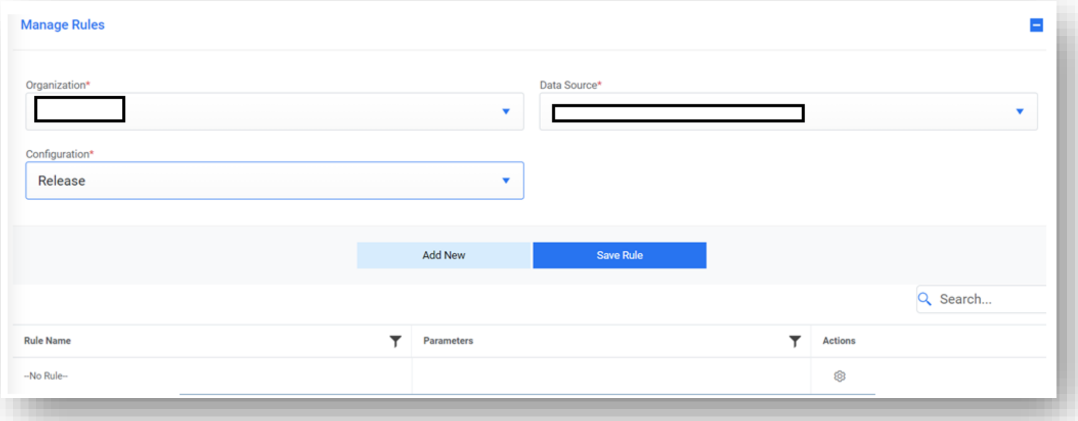


Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | [redacted] | | |

Figure 241 – Manage Entry Criteria (Cont.)

    - Click **Save**.
23. To configure the Release rules for the data source created earlier, perform the below steps:
    - Go to Runbooks and click Manage Rules.
    - Select the **Organization** and the data source created from **Data Source** dropdown.



Manage Rules

Organization*: [redacted]

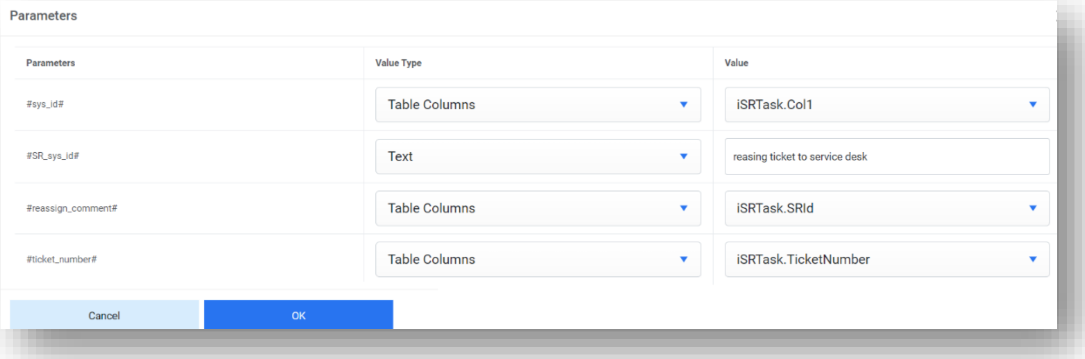
Data Source*: [redacted]

Configuration*: Release

| Rule Name | Parameters | Actions |
|---|---|---|
| --No Rule-- | | ⚙ |

Figure 242 – Manage Release Rules

- Click on ⚙ corresponding to **–No Rule–**.
- Map the parameters #sys_id# to the column in which sys_id was mapped while performing the mandatory parameter mapping while data source creation.
- Mention the reason for releasing ticket in #reassign_comments#.
- Map # SR_sys_id # again to the column in which SRId was mapped while performing the mandatory parameter mapping while data source creation.

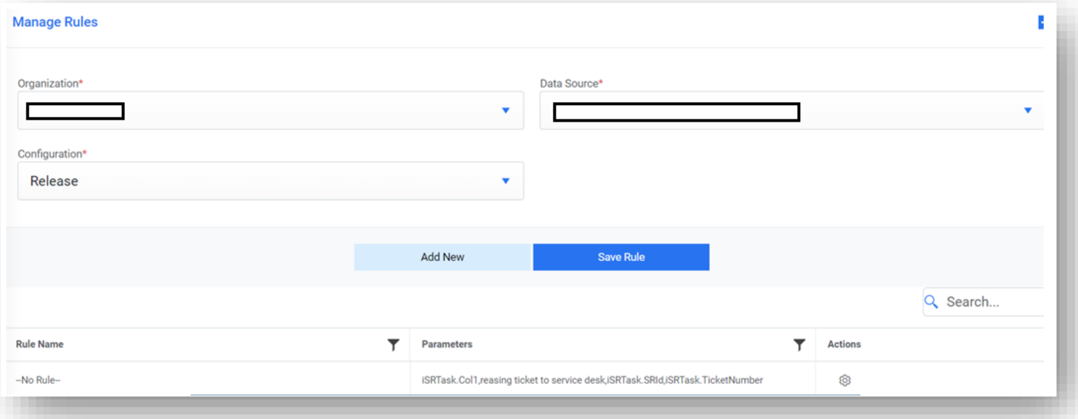


Parameters

| Parameters | Value Type | Value |
|---|---|---|
| #sys_id# | Table Columns | iSRTask.Col1 |
| #SR_sys_id# | Text | reasing ticket to service desk |
| #reassign_comment# | Table Columns | iSRTask.SRId |
| #ticket_number# | Table Columns | iSRTask.TicketNumber |

Figure 243 – Manage Release Rules (Cont.)

24. Click **OK**.



Manage Rules

Organization* [redacted]

Data Source* [redacted]

Configuration* Release

| Rule Name | Parameters | Actions |
|---|---|---|
| --No Rule-- | iSRTask.Col1,reasing ticket to service desk,iSRTask.SRId,iSRTask.TicketNumber | ⚙ |

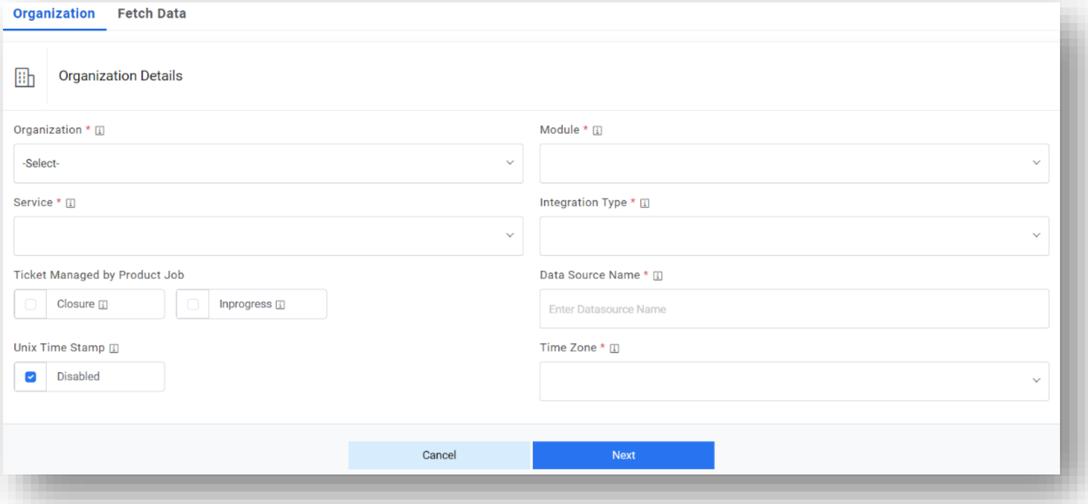
Figure 244 – Manage Release Rules (Cont.)

25. Click Save Rule.

### 4.4.3 Change Request Task Management

To create a data source for Change Request Task Management, perform the following steps:

1. On the left menu bar, click Configuration -> Manage Data Sources.
2. The **Create Data Source** page appears with the following tabs:
    - Organization
    - Fetch Data
    - Release Rules



Organization Details

Organization * -Select-

Module *

Service *

Integration Type *

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name * Enter Datasource Name

Unix Time Stamp: ☑ Disabled

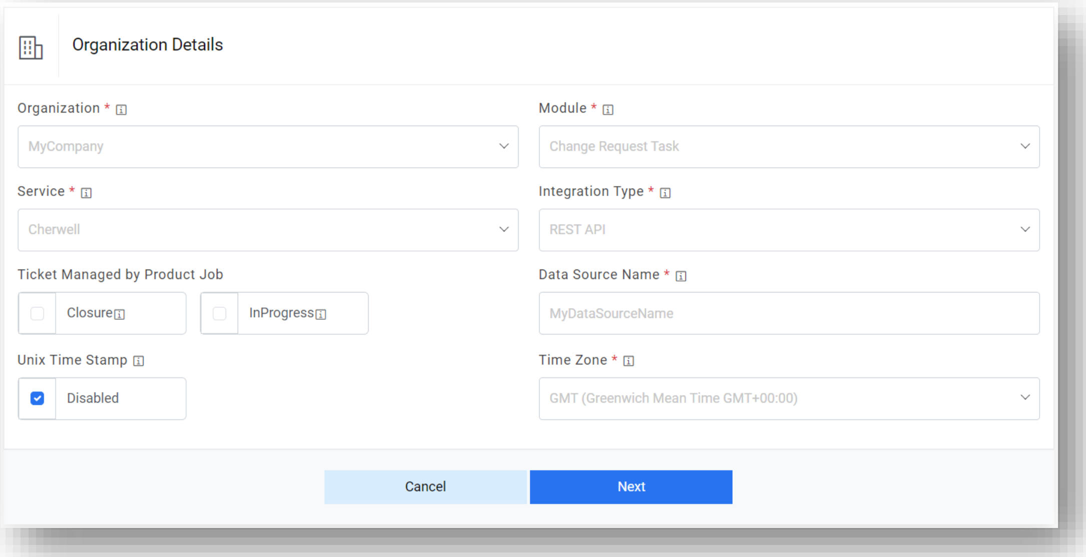
Time Zone *

Figure 245 – Create Data Source

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab,
    - Select the **Organization Name** from the dropdown.

- Select the **Module** as **Change Request Task** since we are configuring this data source for pulling the change request task tickets.
- Select the **Service** as **Cherwell Tool** as we are configuring the data source for Cherwell.
- Select the **Integration Type** as **REST**, since we will be integrating through REST APIs.
- Select the **Timezone** to specify the time zone of the selected data source.
- Select **Timestamp** to view the present data with date and time.
- Click **Next**.



Organization Details

Organization * MyCompany

Module * Change Request Task

Service * Cherwell

Integration Type * REST API

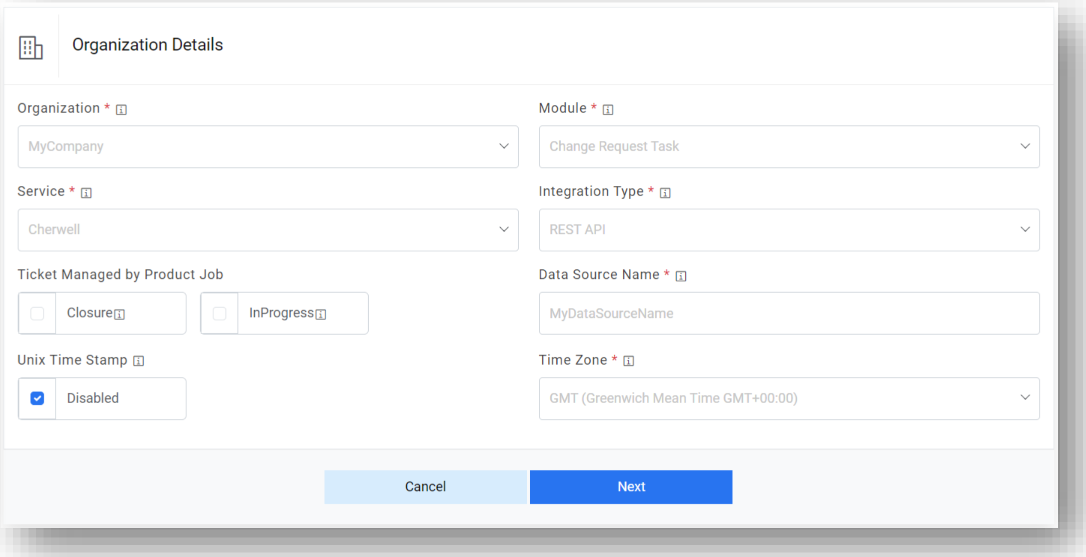
Ticket Managed by Product Job □ Closure □ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 246 – Create Data Source (Cont.)



Organization Details

Organization * MyCompany

Module * Change Request Task

Service * Cherwell

Integration Type * REST API

Ticket Managed by Product Job □ Closure □ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 247 – Create Data Source (Cont.)

4. On the **Fetch Data** tab, type in the details as per the environment.
5. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL**– http://<iAutomate_API_URL>/iAutomateAPI/Request/GetChangeTicketData/<Org_ID>?start_date>=#Start_Date#&end_date<=#End_Date#&
- Here, < iAutomate_API_URL > is the API URL of iAutomate where Push APIs are present and <Org_ID> is the OrgID for the organization for which you are creating the data source. It is available in Organization Master in Database.
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0

The user details that are entered here should be an API User

- Selection of **Basic / Windows** requires you to enter -
  - User Id
  - Password.
- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL
- **Request Method -** Select GET, POST or PUT as Request Method as per the configured URL
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.

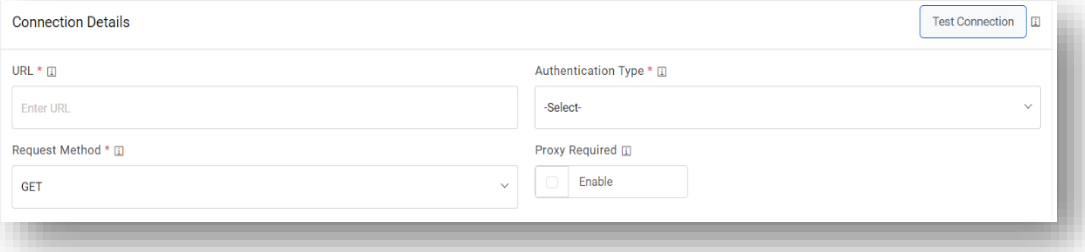
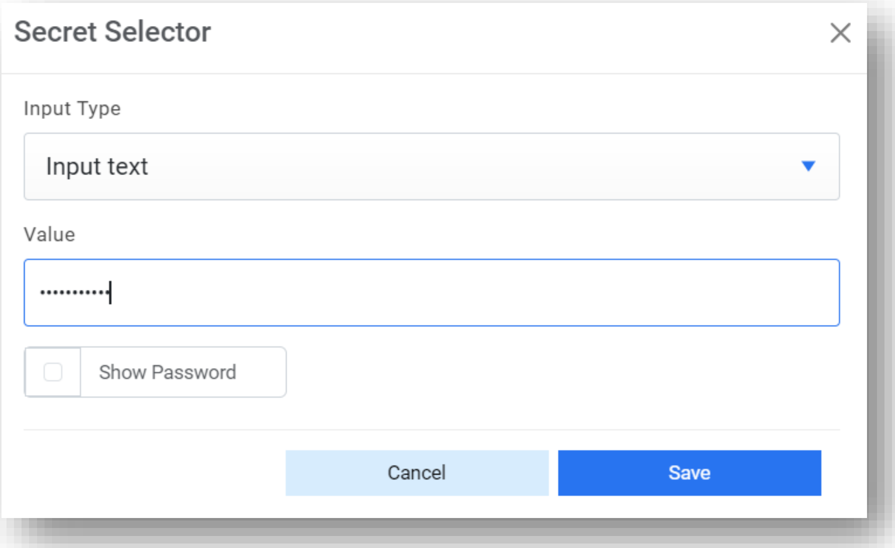


Figure 248 – Create Data Source (Connection Details)

- For **Password**, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



Secret Selector

Input Type

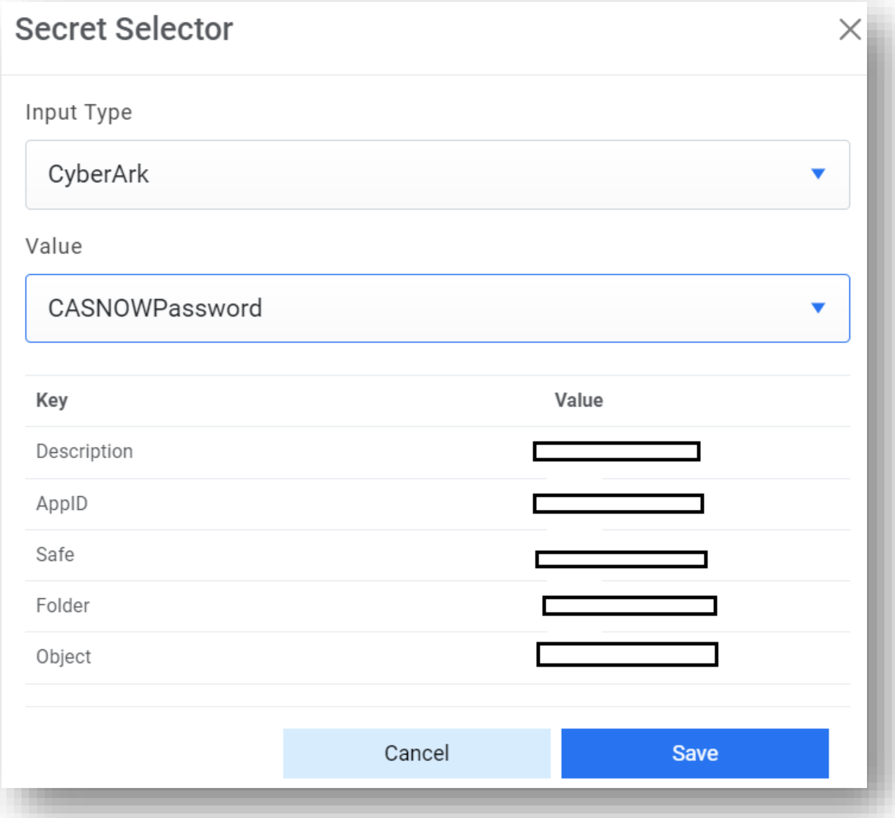
Input text

Value

••••••••••

□ Show Password

Figure 249 - Password in Plaintext



Secret Selector

Input Type

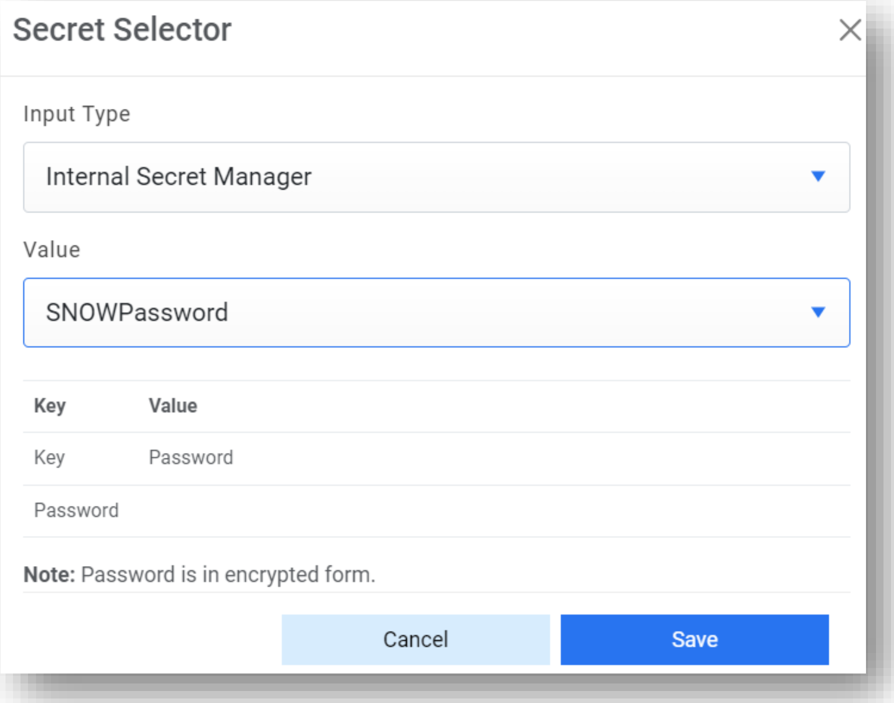
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 250 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

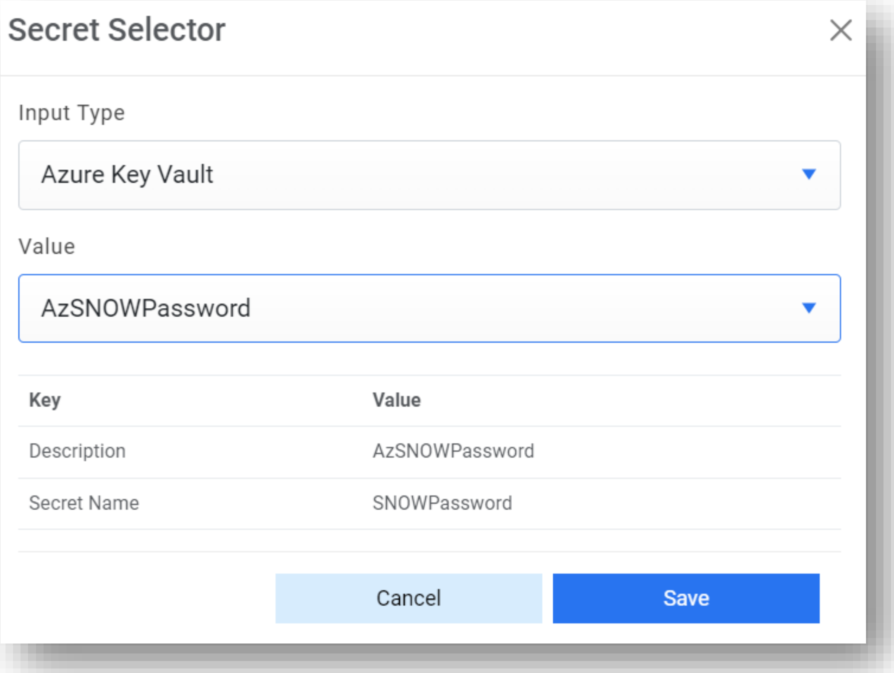
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 251 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

AzSNOWPassword

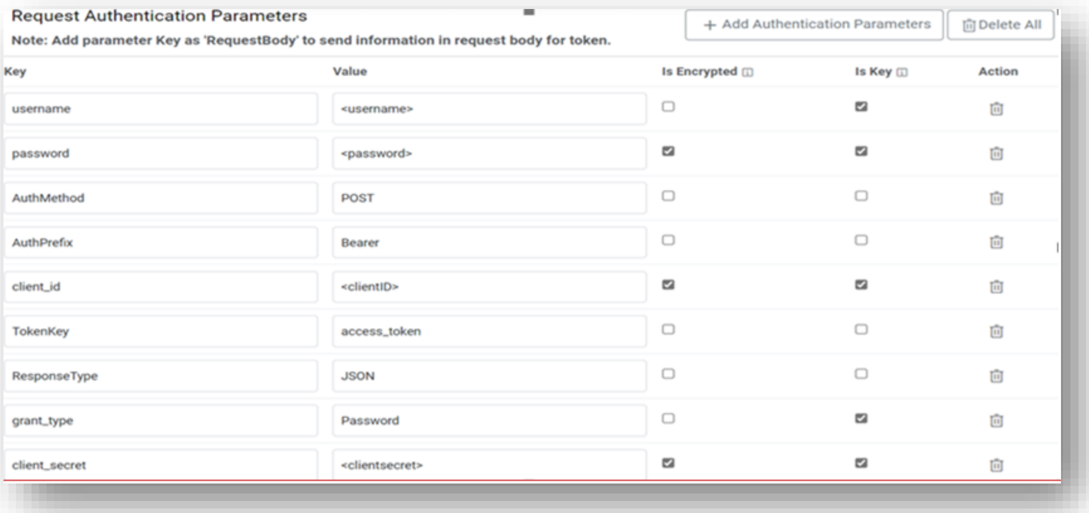
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 252 - Password from Azure Key Vault

6. **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
7. Based on the **Authentication Type,** add the parameters mentioned in the below table.

Table 41– Sample Authentication Parameters

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

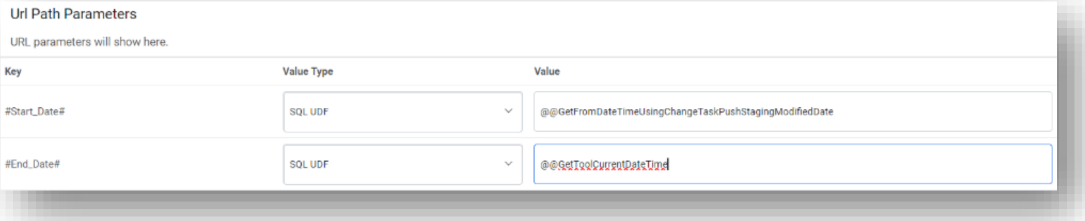
| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | □ | □ | 🗑 |
| AuthPrefix | Bearer | □ | □ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | □ | □ | 🗑 |
| ResponseType | JSON | □ | □ | 🗑 |
| grant_type | Password | □ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 253 – Create Data Source (Request Authentication Parameters for OAuth2.0)

8. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs.

```
Key: #StartDate#

ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingChangeTaskPushStagingModifiedDate
```

```
Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Start_Date# | SQL UDF | @@GetFromDateTimeUsingChangeTaskPushStagingModifiedDate |
| #End_Date# | SQL UDF | @@GetToolCurrentDateTime |

Figure 254– URL Path Parameters

9. **Request Header Parameters –** Please enter the request header parameters as required.
10. **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below:

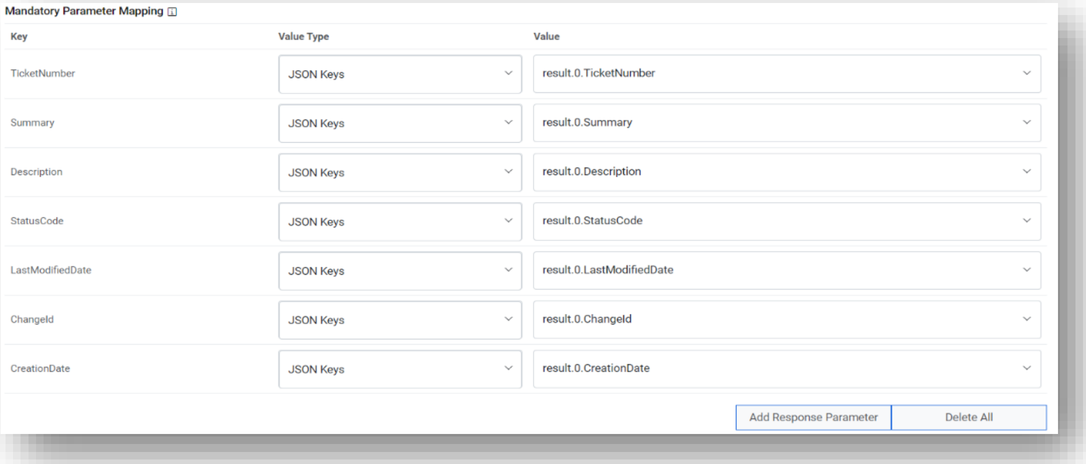
```
Response Body –

{

    "result": [

        {

            "TicketNumber": "12662",

            "Summary": "Test Task",

            "Description": "Test Task",

            "AssignedGroup": "xxxxxxxx",

            "ChangeId": "xxxxxxxx",

            "StatusCode": "1",

            "LastModifiedDate": "2020-05-13 05:11:47.000",

            "sys_id": "xxxxxxxx",

            "CreationDate": "2020-05-13 05:08:10.000",

            "Col1": "",

            "Col2": "",

            "Col3": "",

            "Col4": "",

            "Col5": "",

            "iAutomate_CreatedDateInGMT":            "2020-05-13
05:29:47.987",

            "iAutomate_UpdatedDateInGMT":            "2020-05-13
05:29:47.987"
```

```
            }

        ]

}
```

11. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
12. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 42– Sample Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.TicketNumber |
| Summary | JSON.Keys | result.0.Summary |
| Description | JSON.Keys | result.0.Description |
| StatusCode | JSON.Keys | result.0.StatusCode |
| LastModifiedDate | JSON.Keys | result.0.LastModifiedDate |
| ChangeId | JSON.Keys | result.0.ChangeId |
| CreationDate | JSON.Keys | result.0.CreationDate |



Mandatory Parameter Mapping

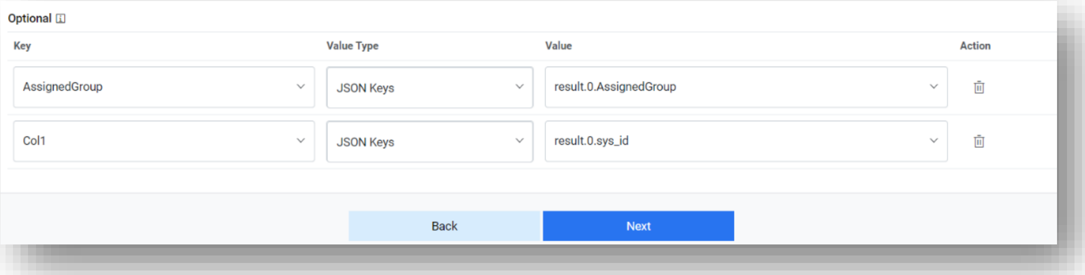
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.TicketNumber |
| Summary | JSON Keys | result.0.Summary |
| Description | JSON Keys | result.0.Description |
| StatusCode | JSON Keys | result.0.StatusCode |
| LastModifiedDate | JSON Keys | result.0.LastModifiedDate |
| ChangeId | JSON Keys | result.0.ChangeId |
| CreationDate | JSON Keys | result.0.CreationDate |

Figure 255 – Mandatory Parameter Mapping

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 43– Sample Optional Parameters

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | result.0.AssignedGroup |
| Col1 | JSON.Keys | result.0.sys_id |



Optional

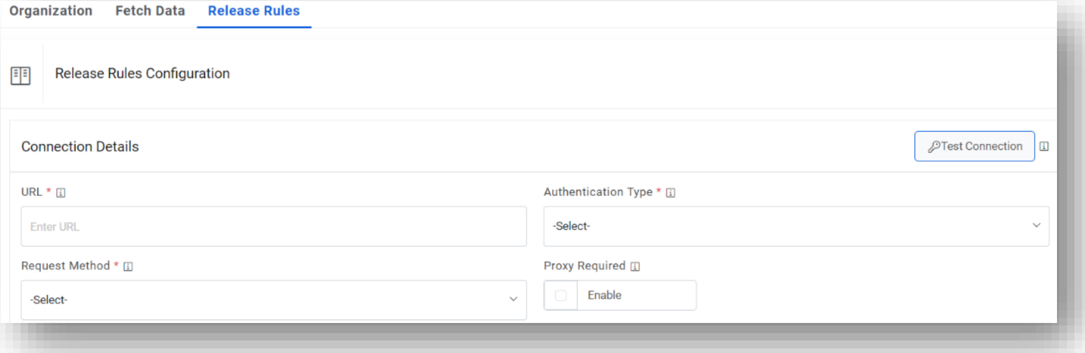
| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | result.0.AssignedGroup | |
| Col1 | JSON Keys | result.0.sys_id | |

Figure 256 – Optional Parameter Mapping

13. Click Next to proceed to Release Rules Configuration.
14. On **Release Rules** tab, type in the details as per the requirement.
15. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>. cherwellondemand.com/CherwellAPI/api/V1/sample
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- **Request Method** – Select Request Method as POST from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



URL * Enter URL

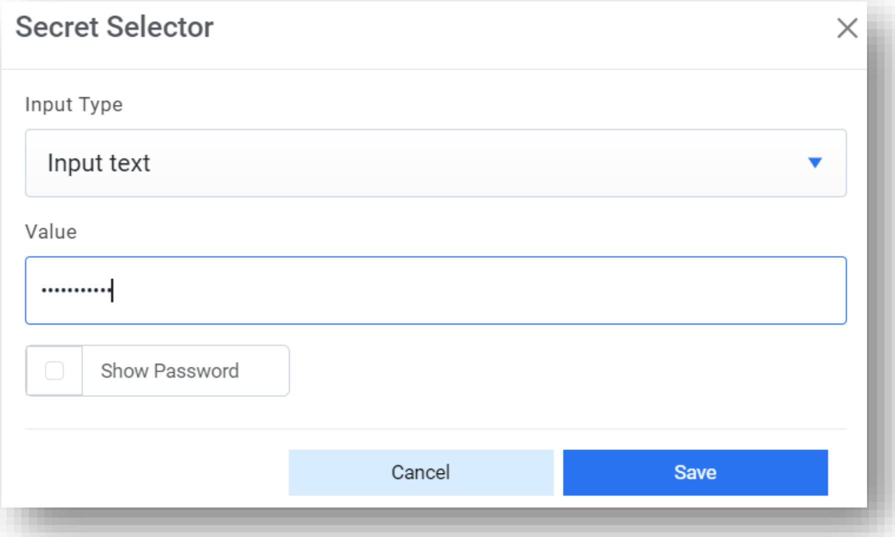
Authentication Type * -Select-

Request Method * -Select-

Proxy Required □ Enable

Figure 257 – Release Rules (Connection Details)

- For **Password**, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



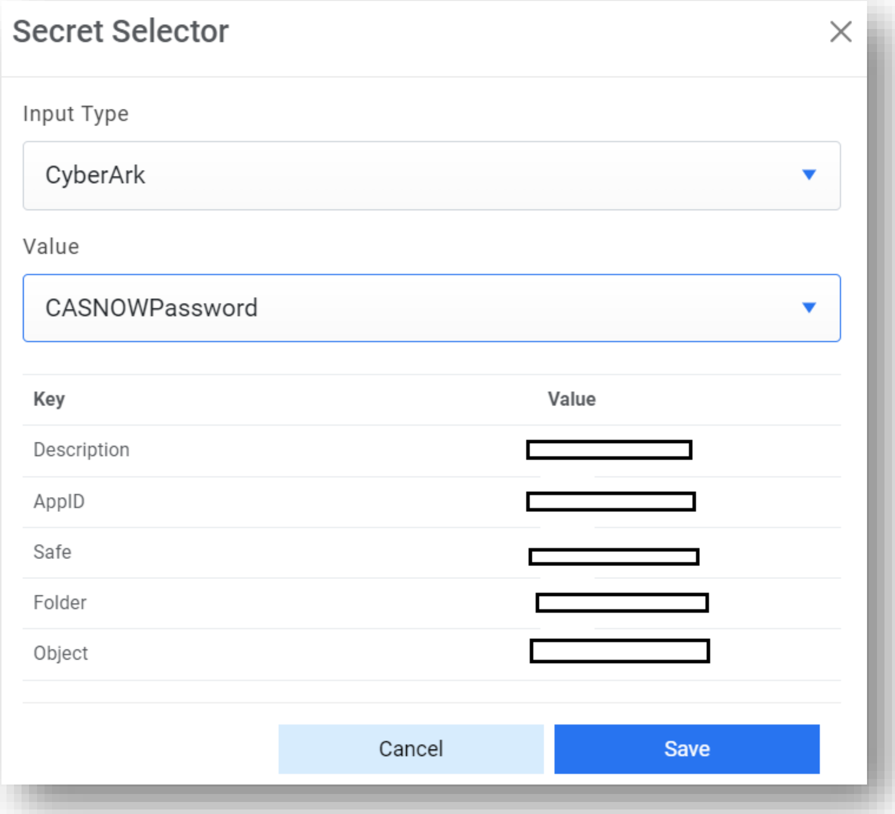
Secret Selector

Input Type: Input text

Value: ••••••••••

☐ Show Password

Figure 258 - Password in plaintext



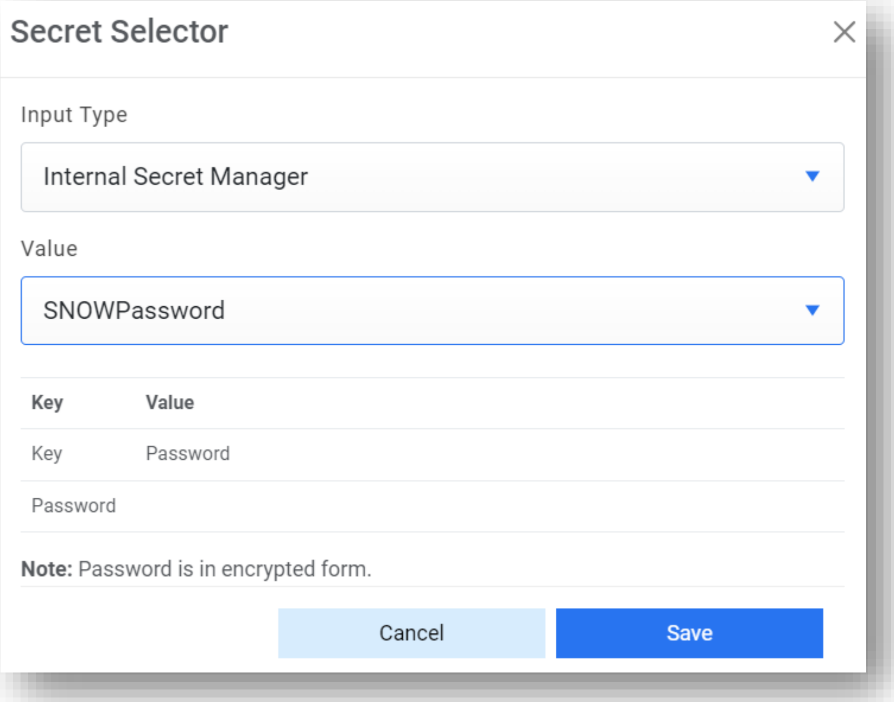
Secret Selector

Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 259 - Password from Key Vault (CyberArk)



## Secret Selector

Input Type

Internal Secret Manager

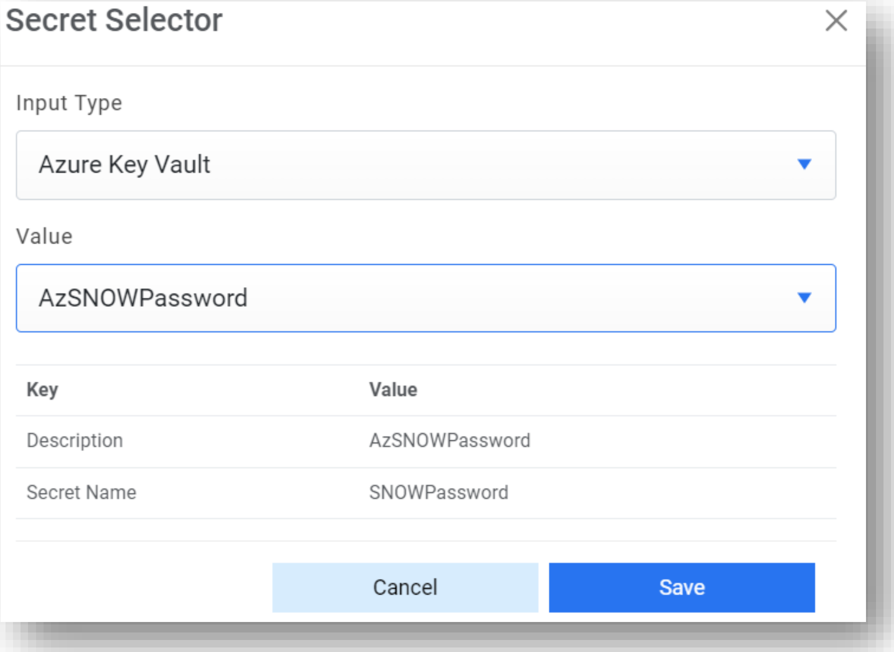
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 260 - Password from Secret Manager



## Secret Selector

Input Type

Azure Key Vault

Value

AzSNOWPassword

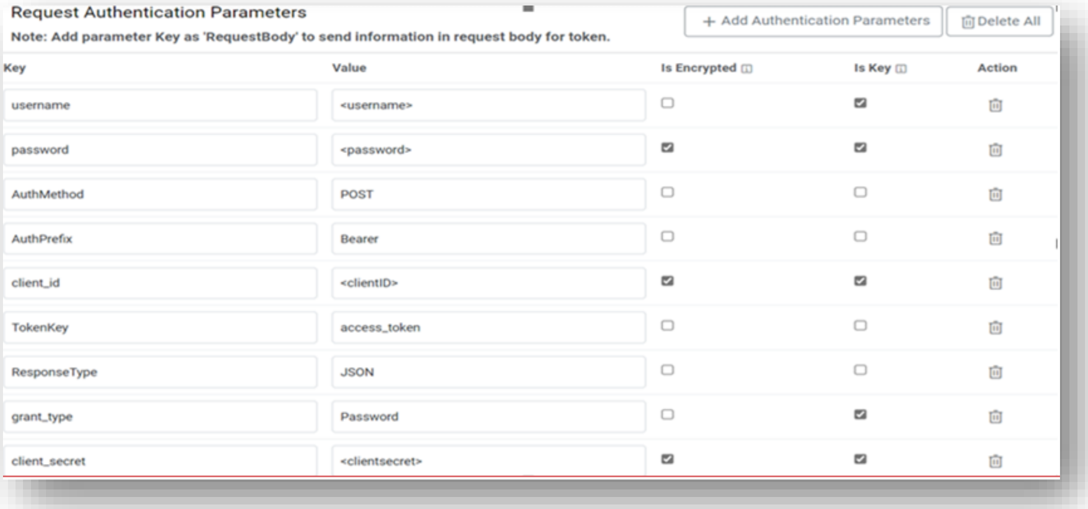
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 261 - Password from Azure Key Vault

16. **Request Authentication Parameters** - If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
17. Based on the Authentication Type, add the parameters mentioned in the below table:

Table 44– Sample Authentication Parameters

| Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|
| grant_type | password | N | Y |
| username | <username> | N | Y |
| Password | <password> | Y | Y |
| client_id | <client_id> | N | Y |
| AuthPrefix | Bearer | N | N |
| AuthMethod | POST | N | N |
| ResponseType | JSON | N | N |
| TokenKey | access_token | N | N |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

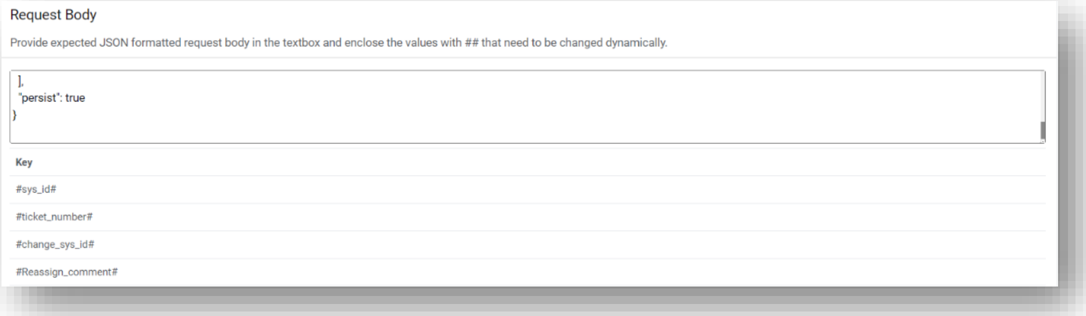
| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | ☐ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | ☐ | ☐ | 🗑 |
| AuthPrefix | Bearer | ☐ | ☐ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | ☐ | ☐ | 🗑 |
| ResponseType | JSON | ☐ | ☐ | 🗑 |
| grant_type | Password | ☐ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 262 – Create Data Source (Request Authentication Parameters)

18. **Request Body -** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –

{

  "busObId": "946004f5f680a57b6747774eda9a6fa2f5d0e73db1",

  "cacheScope": "Tenant",

  "fields": [

    {

      "dirty": true,

      "displayName": "Task RecID",

      "fieldId": "946005353974025498ed1d4068936d72c8992d015c",

      "value": "#sys_id#"

    },

 {
      "dirty": true,
      "displayName": "Ticket Number",
      "fieldId": "94602e208e8947bff420df4016b30962152556d5e2",
      "value": "#ticket_number#"
    },
    {
      "dirty": true,
      "displayName": "Parent RecID",
      "fieldId": "9460053dd53d9888efddc34d3db0360cc5be25f567",
      "value": "#change_sys_id#"
    },
    {
      "dirty": true,
      "displayName": "Journal Details",
      "fieldId": "946005008899c5f5c31caa43c99083519668f0ff33",
      "value": "#Reassign_comment#"
    },
    {
      "dirty": true,
      "displayName": "Assignment Team",
      "fieldId": "946005013472134fdc1b0649a685d41a4c73f6e179",
      "value": "GBP Change Management"
    },
    {
      "dirty": true,
      "displayName": "Status",
      "fieldId": "946004ff47672c8cda67da43a1945ce56f2f617855",
      "value": "Acknowledged"
    },
    {
```

```
   "dirty": true,

   "displayName": "Task Type",

   "fieldId": "946004feb10853e55a192849c780773b2133028cc0",

   "value": "Change Task"

  },

  {

   "dirty": true,

   "displayName": "Reassigning",

   "fieldId": "946005a199ecde0a9cf0b748bb94e4040c2007540f",

   "value": "True"

  }

 ],

 "persist": true

}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

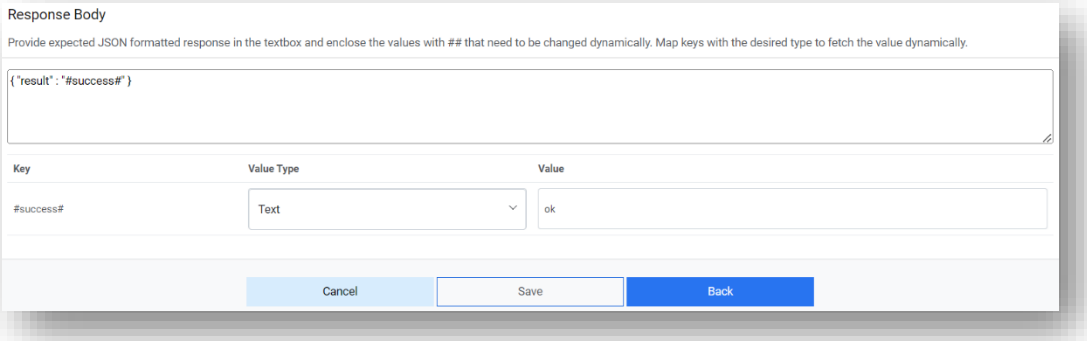
```
],
"persist": true
}
```

| Key |
|---|
| #sys_id# |
| #ticket_number# |
| #change_sys_id# |
| #Reassign_comment# |

Figure 263 – Release Rules (Request Body)

19. **Response Body –** In this section, please enter the response body in JSON format. A sample request is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

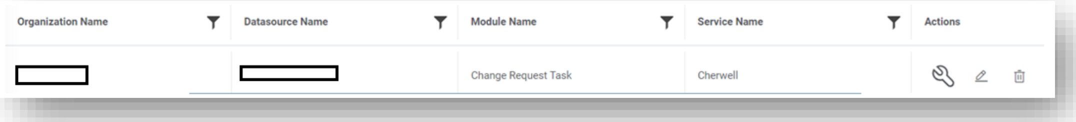
Figure 264 – Release Rules (Response Body)

20. **Response Key Value** mapping can be done as per the below table:

Table 45 – Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

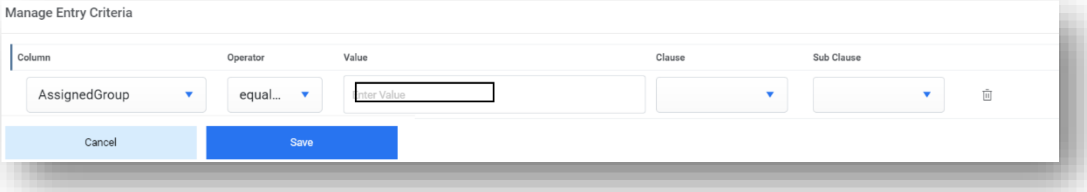
21. Click **Submit** to add the data source.
22. To bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps:
    - Go to Actions tab and click Manage Data Sources.
    - On the **Data Sources** tab, click 🔧 next to the data source user wants to manage. **Manage Entry Criteria** screen appears.



| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| [redacted] | [redacted] | Change Request Task | Cherwell | |

Figure 265 – Manage Entry Criteria

- Select 'AssignedGroup' for the **Column** field and 'equals to' for the **Operator** field.
- Enter the sys_id of the assignment group in Cherwell in the **Value** field.
- **Clause** and **Sub-Clause** fields can also be added based on requirement.



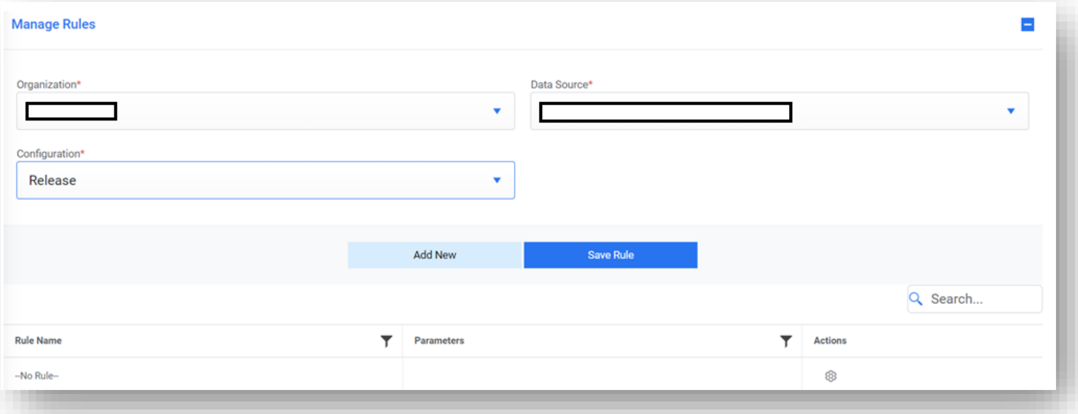
Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | Enter Value | | |

Figure 266 – Manage Entry Criteria (Cont.)

23. Click **Save**.
24. To configure the Release rules for the data source created earlier, perform the below steps:

- Go to **Runbooks** and click Manage Rules.
- Select the **Organization** and the data source created from **Data Source** dropdown.



Manage Rules

Organization* [redacted]

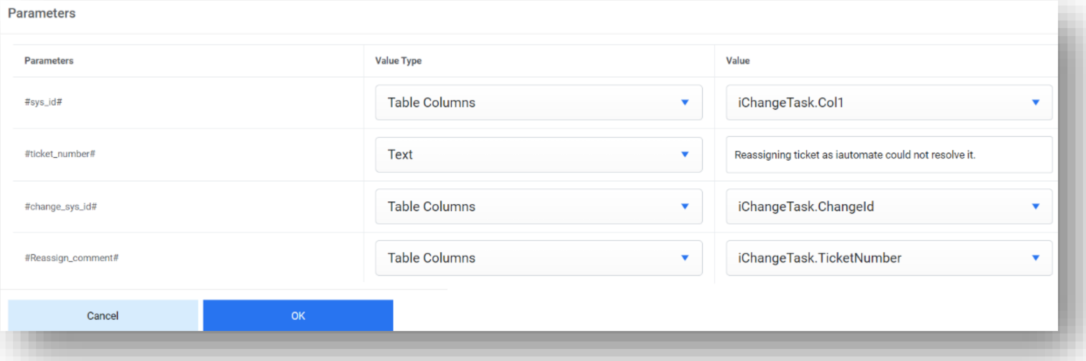
Data Source* [redacted]

Configuration* Release

| Rule Name | Parameters | Actions |
|---|---|---|
| --No Rule-- | | ⚙ |

Figure 267 – Manage Release Rules

- Click on ⚙ corresponding to **–No Rule—**.
- Map the parameters #sys_id# to the column in which sys_id was mapped while performing the mandatory parameter mapping while data source creation.
- Mention the reason for releasing ticket in #reassign_comments#.
- Map #change_sys_id # again to the column in which ChangeId was mapped while performing the mandatory parameter mapping while data source creation.

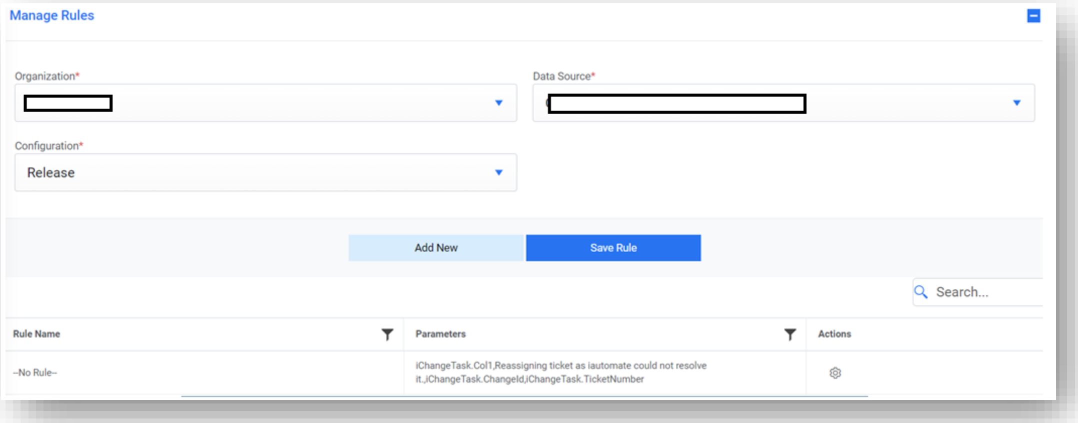


Parameters

| Parameters | Value Type | Value |
|---|---|---|
| #sys_id# | Table Columns | iChangeTask.Col1 |
| #ticket_number# | Text | Reassigning ticket as iautomate could not resolve it. |
| #change_sys_id# | Table Columns | iChangeTask.ChangeId |
| #Reassign_comment# | Table Columns | iChangeTask.TicketNumber |

Figure 268 – Manage Release Rules (Cont.)

25. Click **OK**.



Manage Rules

Organization* [redacted]

Data Source* [redacted]

Configuration* Release

| Rule Name | Parameters | Actions |
|---|---|---|
| --No Rule-- | iChangeTask.Col1,Reassigning ticket as iautomate could not resolve it.,iChangeTask.ChangeId,iChangeTask.TicketNumber | ⚙ |

Figure 269 – Manage Release Rules (Cont.)

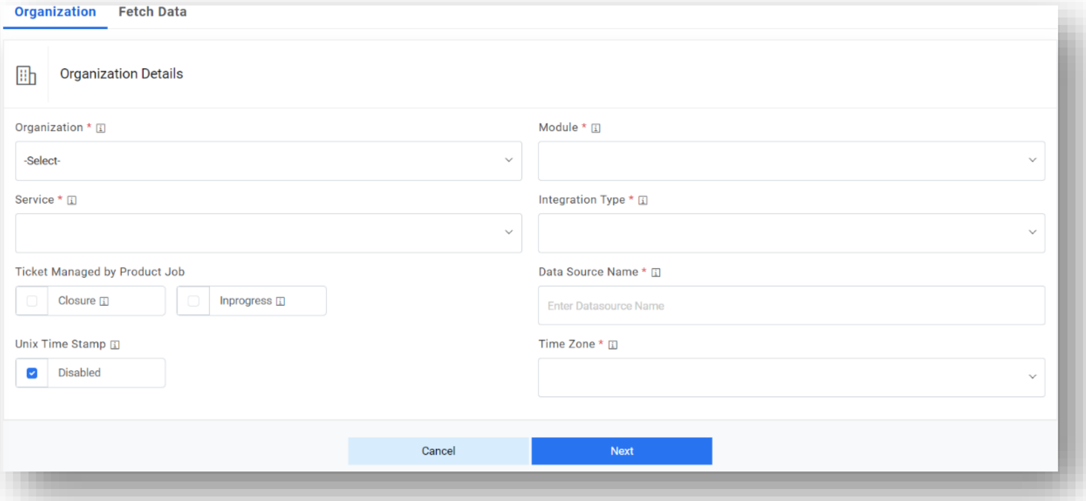
26. Click Save Rule.

## 4.5 Integration with BMC Remedyforce

### 4.5.1 Incident Management

To create a data source for Incident Management, perform the following steps:

1. On the left menu bar, click **Configuration** -> **Manage Data Sources**.
2. The **Create Data Source** page appears with the following tabs:
   - Organization
   - Fetch Data
   - Manage Rules



Organization Details

Organization * -Select-

Module *

Service *

Integration Type *

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name * Enter Datasource Name

Unix Time Stamp: ☑ Disabled

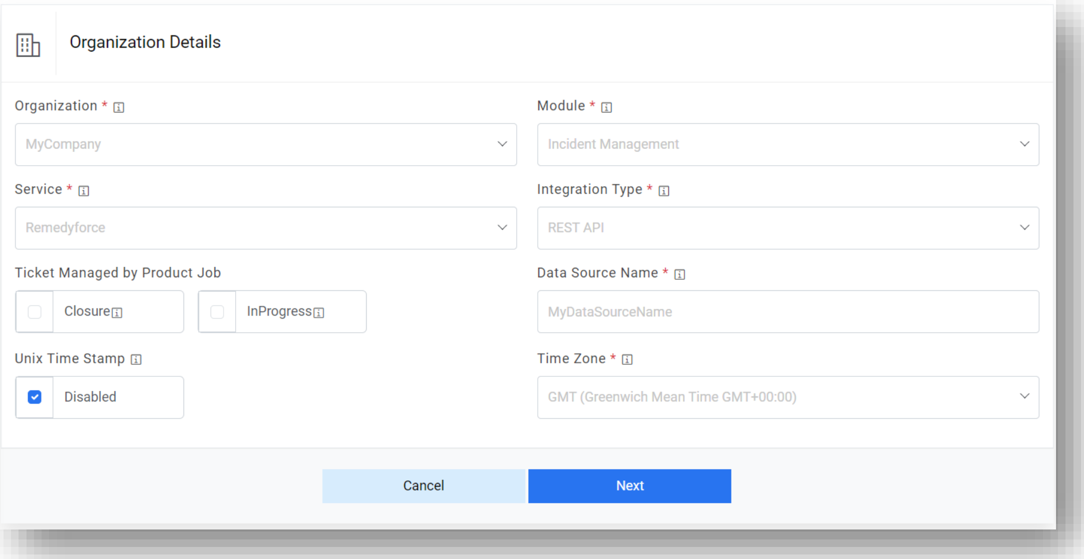
Time Zone *

Figure 270 - Create Data Source

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab,
   - Select the **Organization Name** from the dropdown.

- Select the **Module** as **Incident Management,** since we are configuring this data source for pulling the incident tickets.
- Select the **Service** as **Remedyforce Tool** as we are configuring the data source for BMC Remedyforce.
- Select the **Integration Type** as **REST,** since we will be integrating through REST APIs.
- Select the **Timezone** to specify the time zone of the selected data source.
- Select **Timestamp** to view the present data with date and time.
- Click **Next**.



Organization Details

Organization * MyCompany

Module * Incident Management

Service * Remedyforce

Integration Type * REST API

Ticket Managed by Product Job: □ Closure □ InProgress

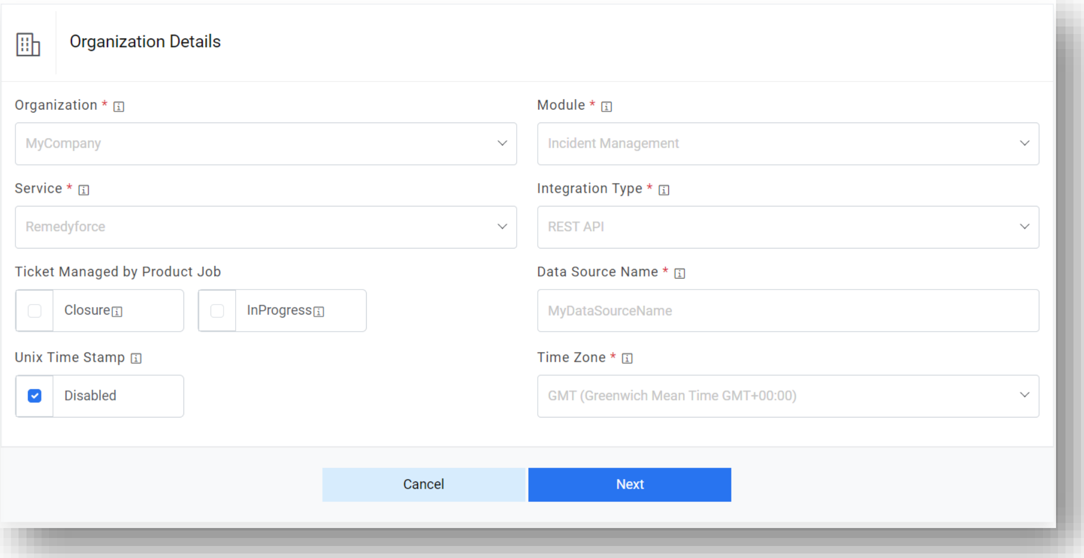
Data Source Name * MyDataSourceName

Unix Time Stamp: ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Cancel | Next

Figure 271 - Create Data Source (Cont.)



Organization Details

Organization * MyCompany

Module * Incident Management

Service * Remedyforce

Integration Type * REST API

Ticket Managed by Product Job: □ Closure □ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp: ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

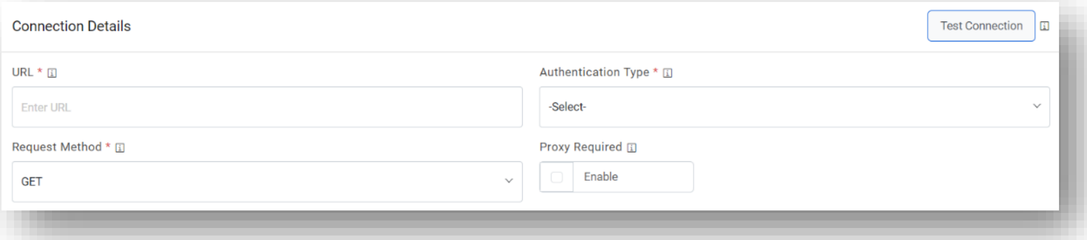
Cancel | Next

Figure 272 - Create Data Source (Cont.)

4. On the **Fetch Data** tab, type in the details as per the environment.
5. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>?q=SELECT+#Fields#+from+BMCServiceDesk__Incident__c+WHERE+BMCServiceDesk__queueName__c+=+'#AssignmentGroup#'+AND+BMCServiceDesk__Status_ID__c+IN+(#State#)
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0
- Selection of **Basic / Windows** requires you to enter -
  - User Id
  - Password.
- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

Test Connection

URL *

Enter URL

Authentication Type *

-Select-

Request Method *

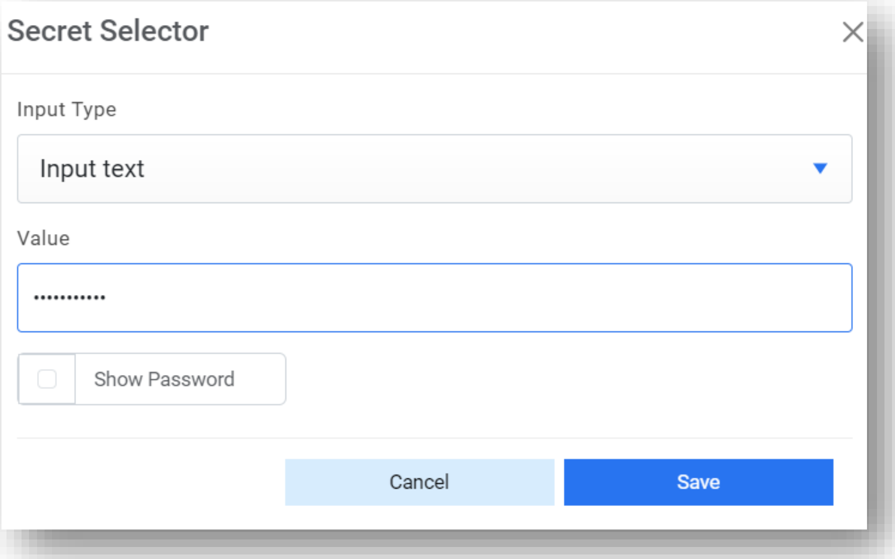
GET

Proxy Required

□ Enable

Figure 273 – Create Data Source (Connection Details)

- For **Password**, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



Secret Selector

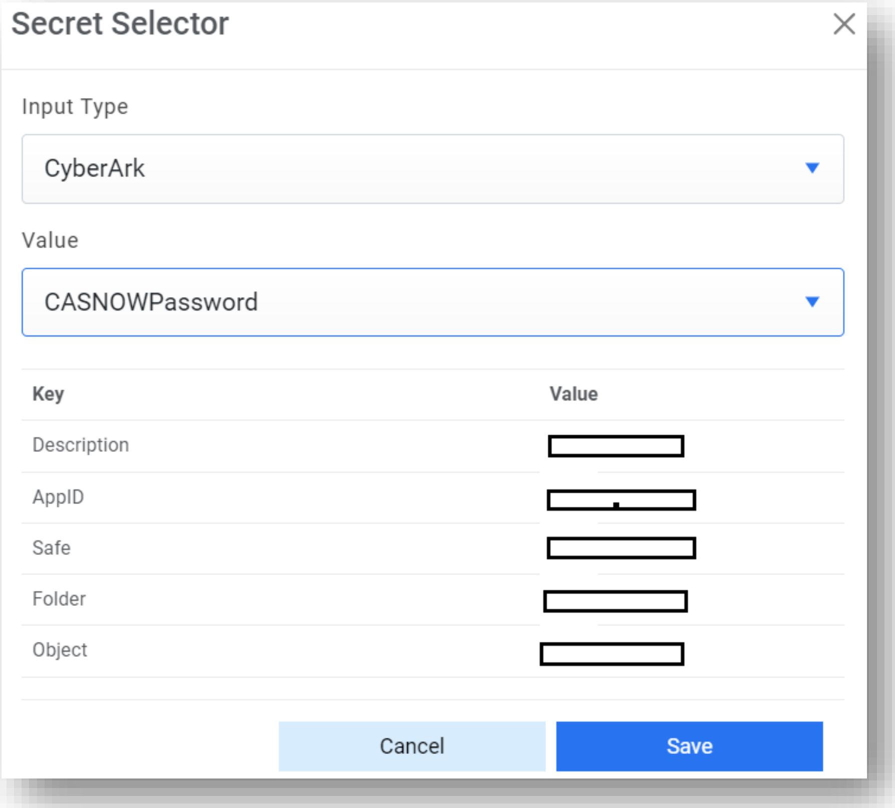
Input Type

Input text

Value

•••••••••••

Figure 274 - Password in plaintext



Secret Selector

Input Type

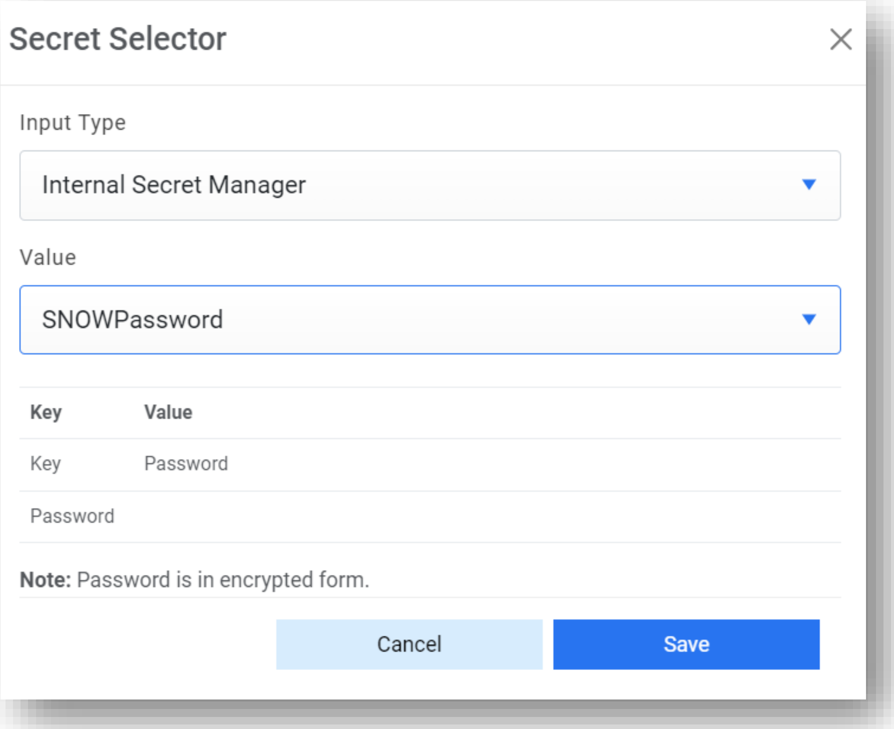
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 275 - Password from Key Vault (CyberArk)


Secret Selector

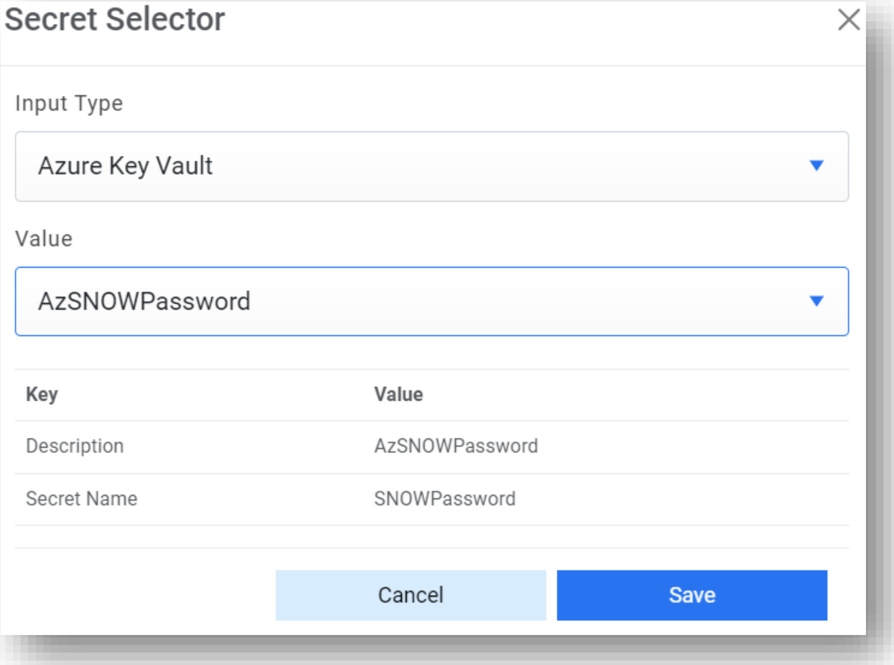
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 276 - Password from Secret Manager


Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

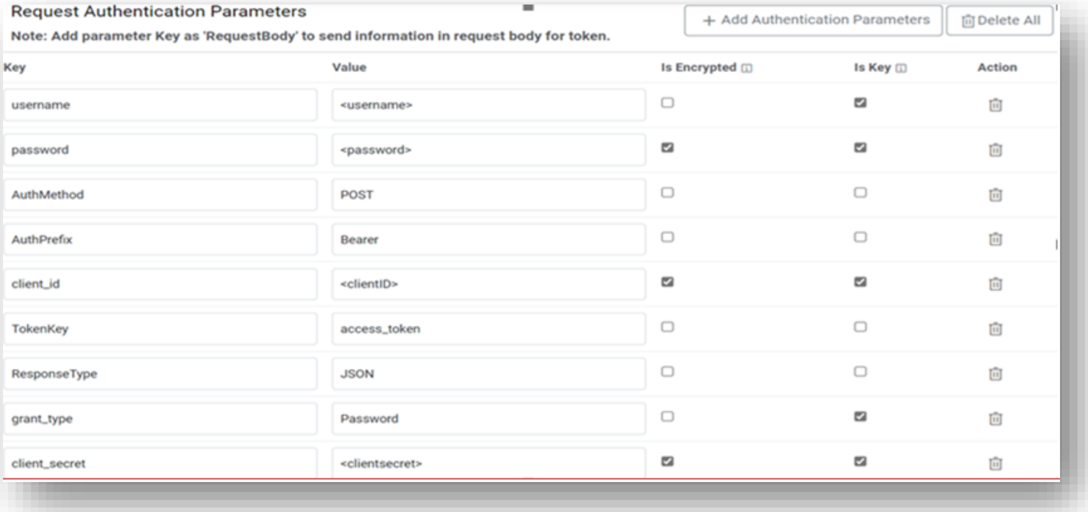
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 277 - Password from Azure Key Vault

6. **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
7. Based on the **Authentication Type,** add the parameters mentioned in the below table:

Table 46– Sample Authentication Parameters

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

Note: Add parameter Key as 'RequestBody' to send information in request body for token.

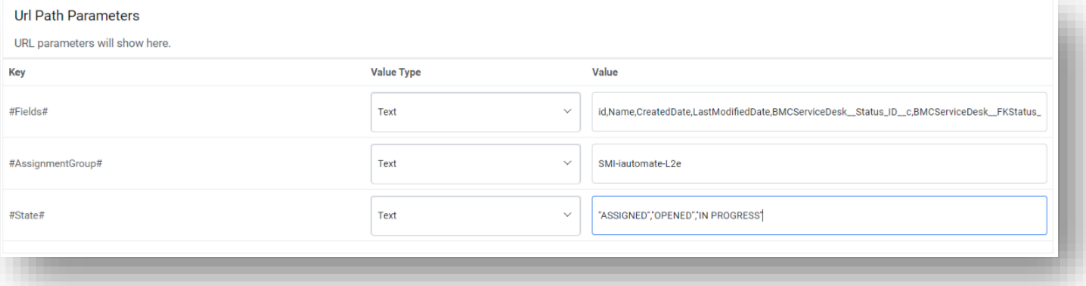
| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | ☐ | ☑ | |
| password | <password> | ☑ | ☑ | |
| AuthMethod | POST | ☐ | ☐ | |
| AuthPrefix | Bearer | ☐ | ☐ | |
| client_id | <clientID> | ☑ | ☑ | |
| TokenKey | access_token | ☐ | ☐ | |
| ResponseType | JSON | ☐ | ☐ | |
| grant_type | Password | ☐ | ☑ | |
| client_secret | <clientsecret> | ☑ | ☑ | |

Figure 278 – Create Data Source (Request Authentication Parameters)

8. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Fields#

ValueType: Text

Value:

id,Name,CreatedDate,LastModifiedDate,BMCServiceDesk__Status_ID_
_c,BMCServiceDesk__FKStatus__c,BMCServiceDesk__shortDescription
__c,BMCServiceDesk__incidentDescription__c,BMCServiceDesk__queu
eName__c,OwnerID


Note – These columns are mandatory. User can add more columns if
more data is required to be fetched from ITSM tool.


Key: #AssignmentGroup#
```

```
ValueType: Text

VALUE: SMI-iautomate-L2e



Key: #State#

ValueType: Text

VALUE: ''ASSIGNED'',''OPENED'',''IN PROGRESS''
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Fields# | Text | id,Name,CreatedDate,LastModifiedDate,BMCServiceDesk__Status_ID__c,BMCServiceDesk__FKStatus_ |
| #AssignmentGroup# | Text | SMI-iautomate-L2e |
| #State# | Text | "ASSIGNED","OPENED","IN PROGRESS" |

Figure 279 – URL Path Parameters (BMC Remedy – Incident Management)

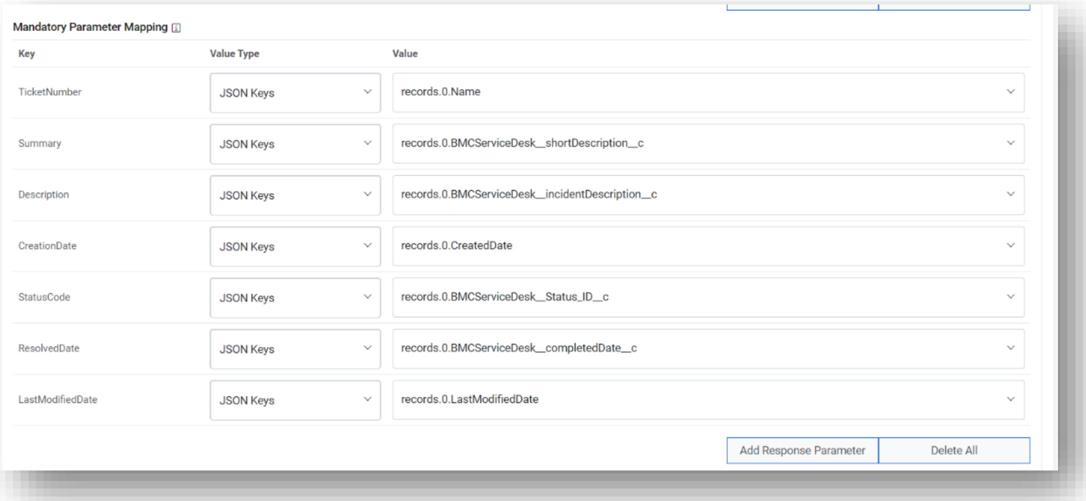
9. **Request Header Parameters –** Please enter the request header parameters as required.
10. **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below:

```
Response Body – {

    "totalSize": 1,

    "done": true,

    "records": [

        {

            "attributes": {

                "type": "BMCServiceDesk__Incident__c",

                "url":
"/services/data/v45.0/sobjects/BMCServiceDesk__Incident__c/a1T3
H0000008bssUAA"

            },

            "Id": "a1T3H0000008bssUAA",

            "Name": "00238924",

            "CreatedDate": "2020-07-14T14:48:04.000+0000",

            "LastModifiedDate": "2020-07-20T11:28:24.000+0000",
            "BMCServiceDesk__completedDate__c":        "2020-07-20T10:28:14.000+0000",

            "BMCServiceDesk__Status_ID__c": "CLOSED",

            "BMCServiceDesk__FKStatus__c": "a2958000000NzamAAC",

            "BMCServiceDesk__shortDescription__c":  "Test  Ticket for iAutomate",

            "BMCServiceDesk__incidentDescription__c":      "Test Ticket for iAutomate",

            "BMCServiceDesk__queueName__c": "SMI-iautomate-L2e",

            "OwnerId": "00G3H000000W37OUAS"

        }

    ]

}
```

11. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
12. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below.

Table 47– Sample Mandatory Mapping Parameters

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | records.0.Name |
| Summary | JSON.Keys | records.0.BMCServiceDesk__shortDescription__c |
| Description | JSON.Keys | records.0.BMCServiceDesk__incidentDescription__c |
| CreationDate | JSON.Keys | records.0.CreatedDate |
| StatusCode | JSON.Keys | records.0.BMCServiceDesk__Status_ID__c |
| ResolvedDate | JSON.Keys | records.0.BMCServiceDesk__completedDate__c |
| LastModifiedDate | JSON.Keys | records.0.LastModifiedDate |



Mandatory Parameter Mapping

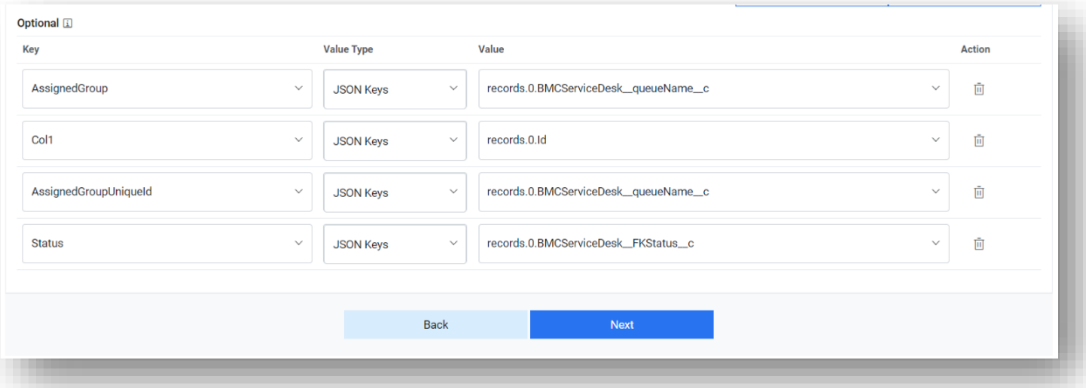
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | records.0.Name |
| Summary | JSON Keys | records.0.BMCServiceDesk__shortDescription__c |
| Description | JSON Keys | records.0.BMCServiceDesk__incidentDescription__c |
| CreationDate | JSON Keys | records.0.CreatedDate |
| StatusCode | JSON Keys | records.0.BMCServiceDesk__Status_ID__c |
| ResolvedDate | JSON Keys | records.0.BMCServiceDesk__completedDate__c |
| LastModifiedDate | JSON Keys | records.0.LastModifiedDate |

Figure 280 – Mandatory Parameter Mapping

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 48– Sample Optional Mapping Parameters

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | records.0.BMCServiceDesk__queueName__c |
| Col1 | JSON.Keys | records.0.id |
| AssignedGroupUniqueId | JSON.Keys | records.0.BMCServiceDesk__queueName__c |
| Status | JSON.Keys | records.0.BMCServiceDesk__FKStatus__c |



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | records.0.BMCServiceDesk__queueName__c | 🗑 |
| Col1 | JSON Keys | records.0.Id | 🗑 |
| AssignedGroupUniqueId | JSON Keys | records.0.BMCServiceDesk__queueName__c | 🗑 |
| Status | JSON Keys | records.0.BMCServiceDesk__FKStatus__c | 🗑 |

Figure 281 – Optional Parameter Mapping

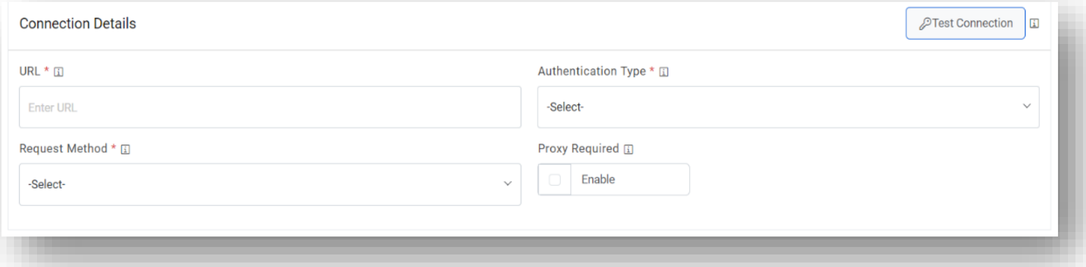
13. Click Next to proceed to Release Rules.
14. On **Release Rules** tab, type in the details as per the requirement.
15. In the **Connection Details** section, enter the following details:

– **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.

– **Sample URL** – http://my_host/#TicketID#

- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- Request Method – Select Request Method as PUT from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.


Connection Details

URL *

Enter URL

Request Method *

-Select-

Authentication Type *

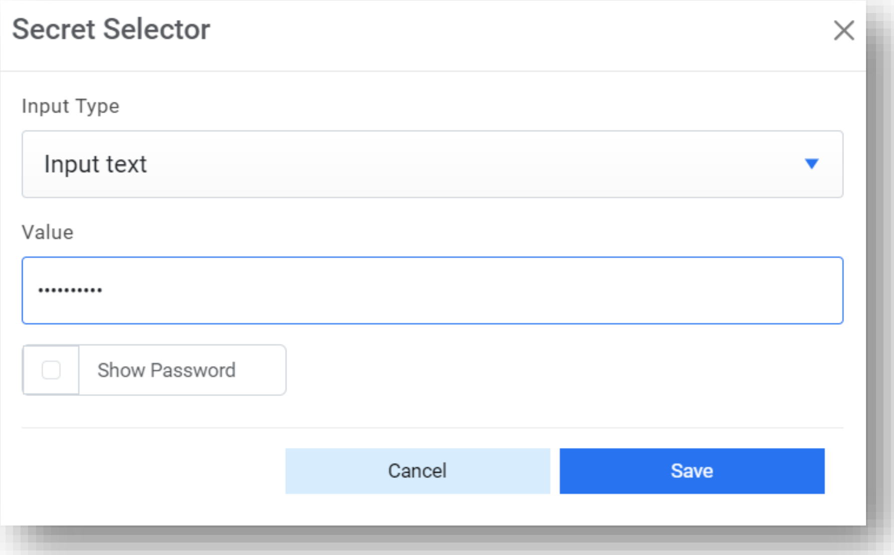
-Select-

Proxy Required

□ Enable

Figure 282 – Test Connection

- For **Password**, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.


Secret Selector

Input Type

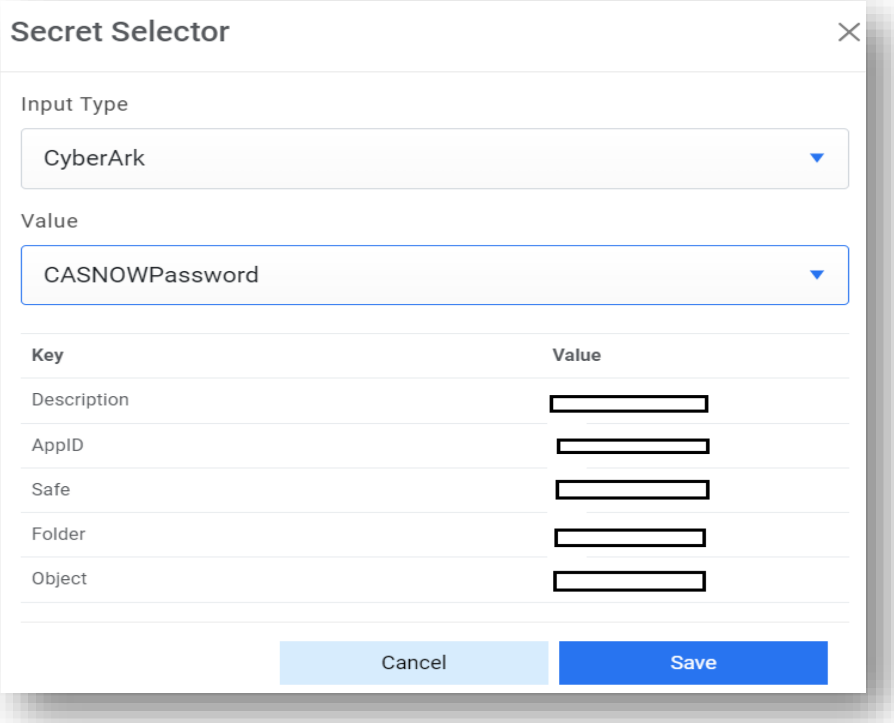
Input text

Value

••••••••••

□ Show Password

Figure 283 - Password in plaintext

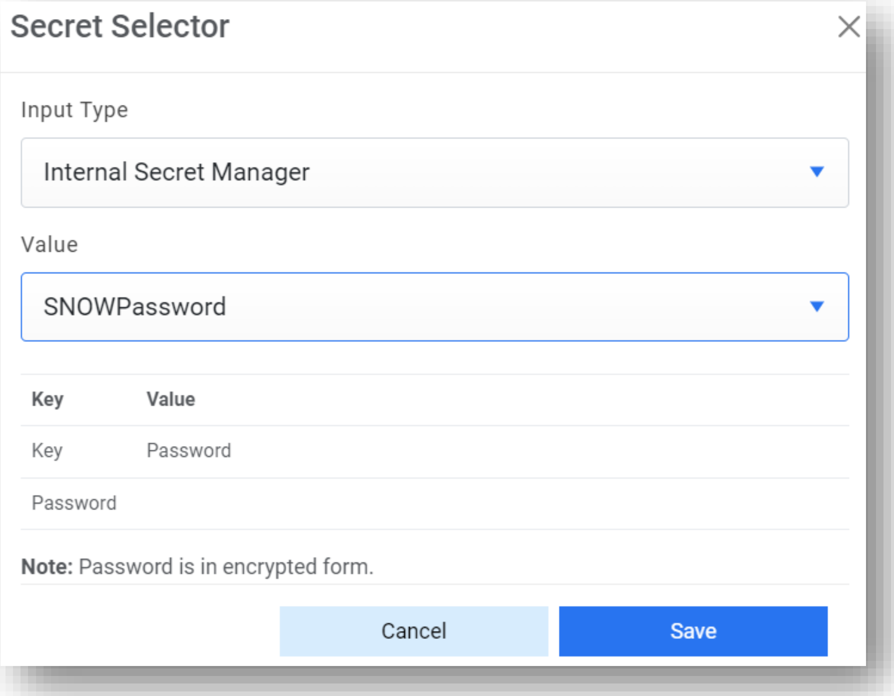


Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | [redacted] |
| AppID | [redacted] |
| Safe | [redacted] |
| Folder | [redacted] |
| Object | [redacted] |

Figure 284 - Password from Key Vault (CyberArk)



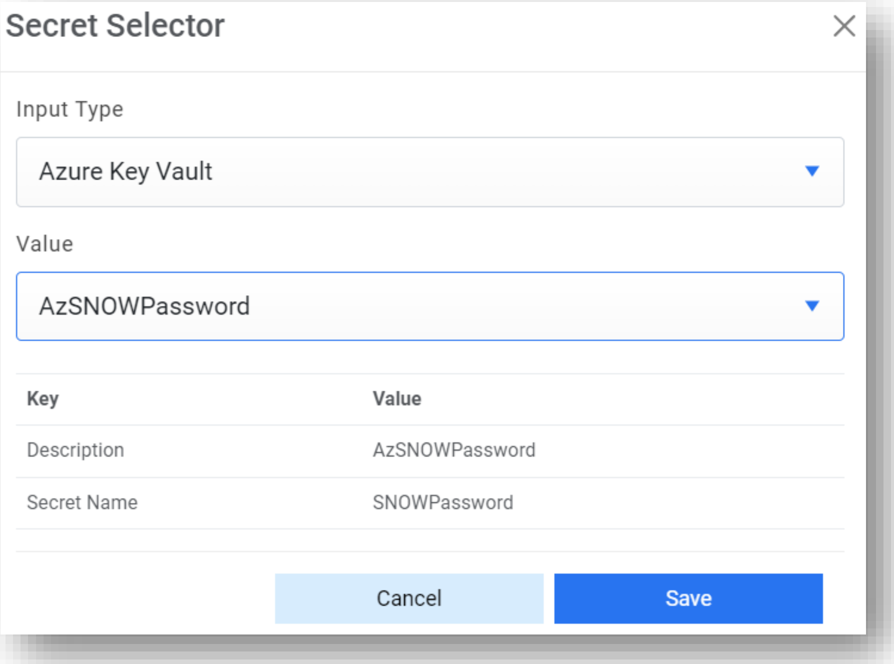
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 285 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

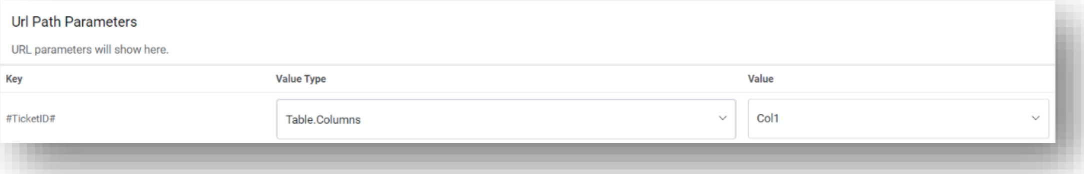
AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 286 - Password from Azure Key Vault

16. **URL Path Parameters** – Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #TicketId#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
“Col1”
```



Url Path Parameters

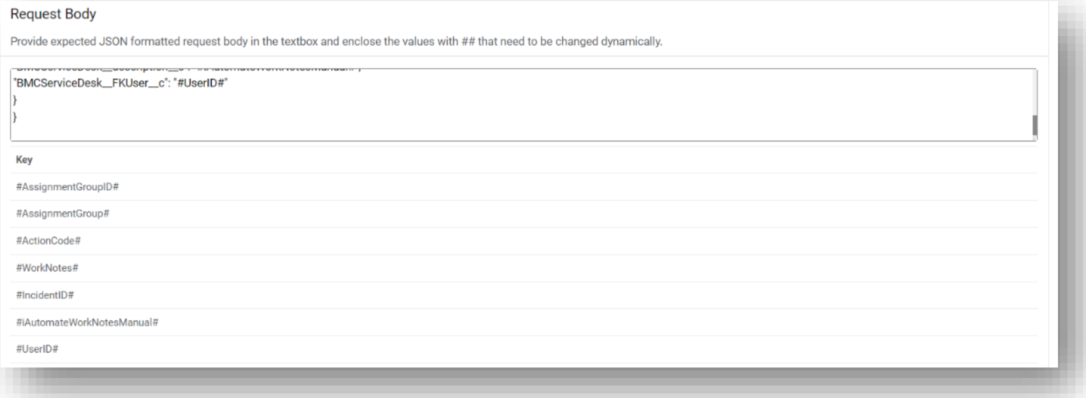
URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #TicketID# | Table.Columns | Col1 |

Figure 287 – Release Rules (URL Path Parameters)

17. **Request Header Parameters –** Please enter the request header parameters as required.
18. **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body – {

"grptransfer": {

"OwnerId": "#AssignmentGroupID#",

"BMCServiceDesk__queueName__c": "#AssignmentGroup#"
},

"workorder": {

"BMCServiceDesk__FKAction__c": "#ActionCode#",

"BMCServiceDesk__note__c": "#WorkNotes#",

"BMCServiceDesk__FKIncident__c": "#IncidentID#",

"BMCServiceDesk__description__c": "#iAutomateWorkNotesManual#",

"BMCServiceDesk__FKUser__c": "#UserID#"

}

}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

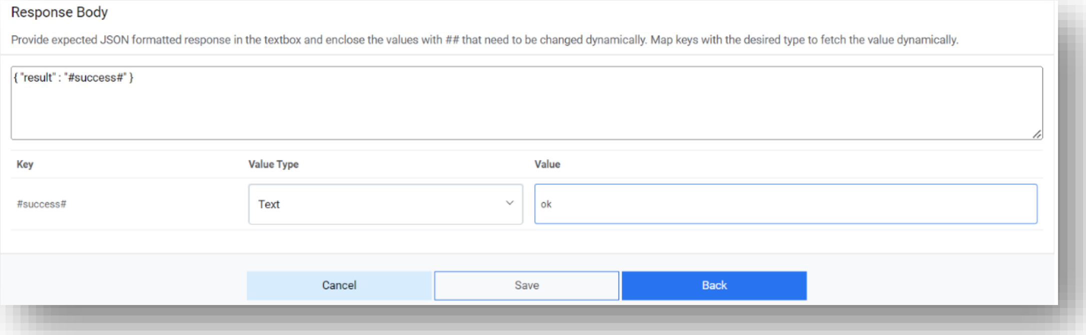
```
"BMCServiceDesk__FKUser__c": "#UserID#"
}
}
```

| Key |
|---|
| #AssignmentGroupID# |
| #AssignmentGroup# |
| #ActionCode# |
| #WorkNotes# |
| #IncidentID# |
| #iAutomateWorkNotesManual# |
| #UserID# |

Figure 288 – Release Rules (Request Body)

19. **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below.

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

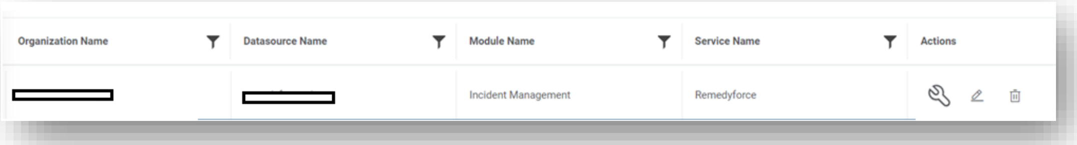
Figure 289 – Release Rules (Response Body)

20. **Response Key Value** mapping can be done as per the below table.

Table 49– Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | Success |

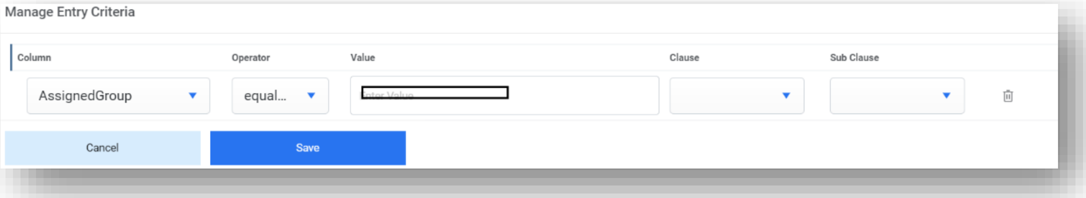
21. Click **Save** to add the data source.
22. In order to bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps:
    - Go to Action tab and click Manage Data Sources.
    - On the **Data Sources** tab, click 🔧 next to the data source user wants to manage. **Manage Entry Criteria** screen appears.



| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| [redacted] | [redacted] | Incident Management | Remedyforce | |

Figure 290 – Manage Entry Criteria

    - Select 'AssignedGroup' for the **Column** field and 'equals to' for the **Operator** field.
    - Enter the sys_id of the assignment group in Remedyforce in the **Value** field.
    - **Clause** and **Sub-Clause** fields can also be added based on requirement.



Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | [redacted] | | |

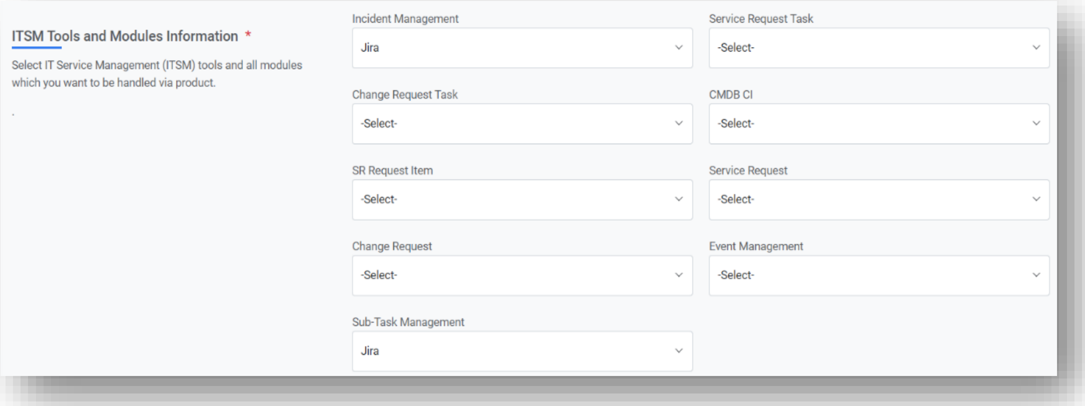
Figure 291 – Manage Entry Criteria (Cont.)

23. Click **Save**.

## 4.6 Integration with JIRA

### 4.6.1 Incident Management

For Integration of Jira ITSM tool with iAutomate, perform the following steps:



ITSM Tools and Modules Information *

Select IT Service Management (ITSM) tools and all modules which you want to be handled via product.

Incident Management: Jira

Service Request Task: -Select-

Change Request Task: -Select-

CMDB CI: -Select-

SR Request Item: -Select-

Service Request: -Select-

Change Request: -Select-

Event Management: -Select-

Sub-Task Management: Jira

Figure 292 - Integration with Jira ITSM Tool

- Create Data Source
- Fetch Data

– **URL**: URL>/rest/api/2/search?fields=#columns#&jql=issuetype=Incident AND status=Open ANDupdated >= "#start_date#" AND updated <= "#end_date#" ORDER BY updated DESC

– Authentication Type: Basic

– Request Method: GET

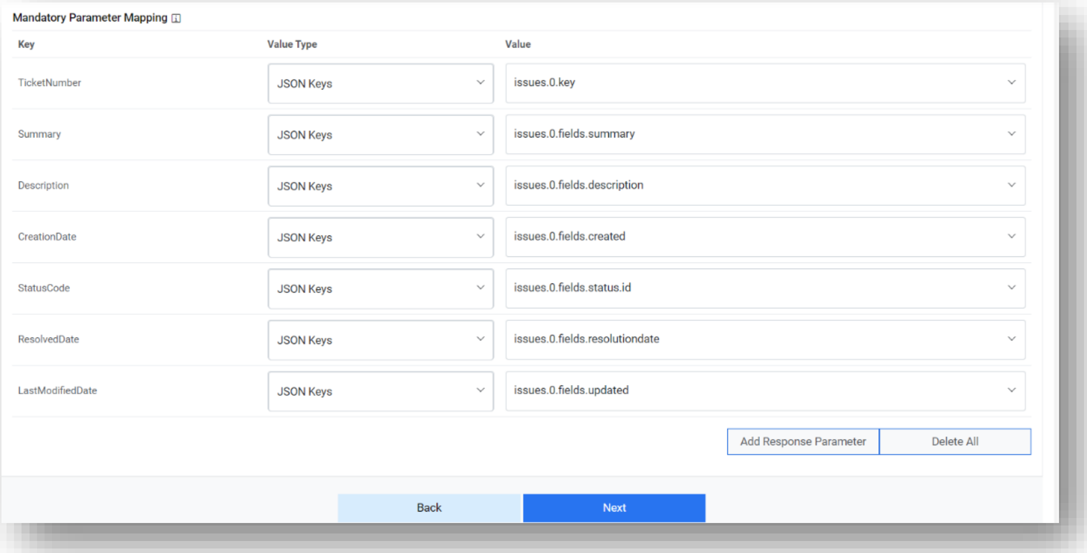
– URL Path Parameters

| Key | Value Type | Value |
|---|---|---|
| #columns# | Text | key,description,summary,created,updated,<br>status,assignee,resolutiondate |
| #start_date# | SQL UDF | @@GetFromDateTimeUsingIncidentModifiedDate |
| #end_date# | SQL UDF | @@GetToolCurrentDateTime |

– **Response Body:**

```
{
  "expand": "schema,names",
  "startAt": 0,
  "maxResults": 50,
  "total": 3,
  "issues": [{
      "expand":
"operations,versionedRepresentations,editmeta,changelog,rendere
dFields",
      "id": "10102",
      "self":
"http://<ipaddress>:<port>/rest/api/2/issue/10102",
     "key": "IT-48",
     "fields": {
        "summary": "REST ye merry gentlemen. Rest in peace",
                       "resolutiondate": "2021-05-
05T13:17:10.000+0530",
        "created": "2021-05-05T13:17:10.000+0530",
        "description": "Creating of an issue using project keys
and issue type names using the REST API",
        "assignee": null,
        "updated": "2021-05-05T13:17:10.000+0530",
        "status": {
           "self":
"http://<ipaddress>:<port>/rest/api/2/status/1",
           "description": "The issue is open and ready for the
assignee to start work on it.",
           "iconUrl":
"http://<ipaddress>:<port>/images/icons/statuses/open.png",
           "name": "Open",
           "id": "1",
           "statusCategory": {
              "self":
"http://<ipaddress>:<port>/rest/api/2/statuscategory/2",
              "id": 2,
              "key": "new",
              "colorName": "blue-gray",
              "name": "To Do"
           }
        }

     }
  }]
}
```

- **Mandatory Parameter Mapping:**

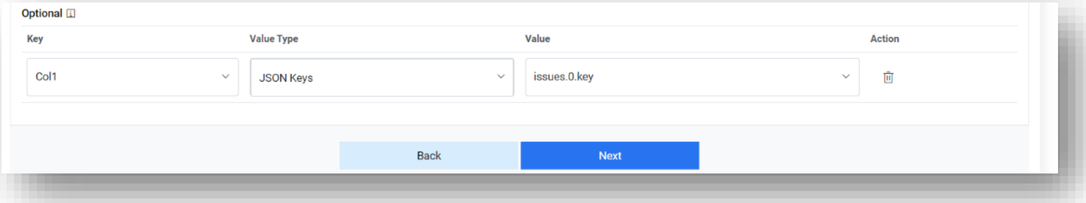


Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | issues.0.key |
| Summary | JSON Keys | issues.0.fields.summary |
| Description | JSON Keys | issues.0.fields.description |
| CreationDate | JSON Keys | issues.0.fields.created |
| StatusCode | JSON Keys | issues.0.fields.status.id |
| ResolvedDate | JSON Keys | issues.0.fields.resolutiondate |
| LastModifiedDate | JSON Keys | issues.0.fields.updated |

Figure 293 - Mandatory Parameter Mapping

- **Optional:**



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| Col1 | JSON Keys | issues.0.key | |

Figure 294 - Optional

- **Release Rule:**

  For release, Jira has 3 different APIs to change the assignee, to add a comment and to add worklog. So, we are using iAutomate's Custom Script API to update all 3 operations with one single API.

To create Custom API, go to Manage Custom Script Section.

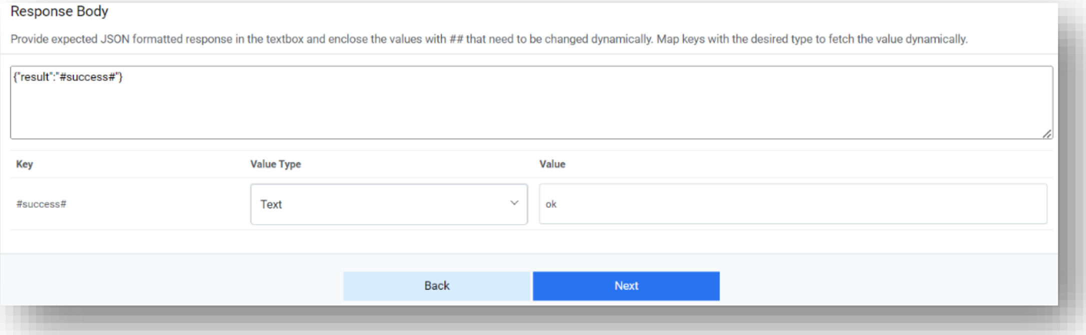
- URL: http://<ipaddress>:<port>/rest/api/2/issue/#key#/assignee
- Authentication Type: Basic
- **UserId**: myuser@hcl.com
- **Password**: ********
- Request Method: POST

**Request Body:**

```
{

    "key": "#ticketId#",

    "URL": "http://<ipaddress>:<port>/rest/api/2/issue/",

    "assignee_name": "#assignee_name#",

    "release_comment":"Ticket_released_from_iAutomate"
}
```

**Response Body:**

```
{"result":"#success#"}
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{"result":"#success#"}
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

Figure 295 - Response Body

- **Close Rules:**
  - URL: http://<ipaddress>:<port>/rest/api/2/issue/#key#/transitions
  - Authentication Type: Basic
  - Request Method: POST
  - URL Path Parameters:

| Key | Value Type | Value |
|---|---|---|
| #key# | Table.Columns | Col1 |

**Request Body:**

```
{
    "update": {
        "comment": [
            {
                "add": {
                    "body": "#worknote#"
                }
            }
        ]
    },
    "transition": {
        "id": "#statuscode#"
    }
}
```

**Response Body:**

```
{“result" : "ok" }
```

– **InProgress Rules:**

- URL: http://<ipaddress>:<port>/rest/api/2/issue/#sysid#/transitions
- Authentication Type: Basic
- Request Method: POST
- URL Path Parameters:

| Key | Value Type | Value |
|---|---|---|
| #key# | Table.Columns | Col1 |

**Request Body:**

```
{
    "update": {
        "comment": [
            {
                "add": {
                    "body": "#worknote#"
                }
            }
        ]
    },
    "transition": {
        "id": "#statuscode#"
     }
}
```

**Response Body:**

```
{ "result" : "ok" }
```

**JsResponseConverter:** After successful creation of data source,

1. Go to CollectIncident job under menu **Configuration** -> **Manage Jobs**.

2. Click on ⚙ icon. A popup will be opened.
3. Go to parameter tab and search for '**JsResponseConverter**' in the end. Replace its value with below string:

```
if(json.issues){for(var
result=[],i=0;i<json.issues.length;i++)result.push(json.issues[
i]);customJobject.dataCollectorNode.data.issues=result}
```

- **Manage Rules:**

For each of the release, close, and in-progress rules are defined as follows:

- **Release Rules:**

| Parameter | Value Type | Value |
|---|---|---|
| #assignee_name# | Text | Assignee_user |
| #ticketId# | Table.Columns | Col1 |

- **Close Rules:**

| Parameter | Value Type | Value |
|---|---|---|
| #worknote# | Text | Ticket closed from iAutomate |
| #ticketId# | Text | 91 |

- **In Progress Rules:**

| Parameter | Value Type | Value |
|---|---|---|
| #worknote# | Text | Ticket marked to in progress |
| #ticketId# | Text | 31 |

- **Manage Custom Script:**

To use multiple Jira APIs that are being used while releasing an incident, you need a python script that contains the calling of all required APIs.

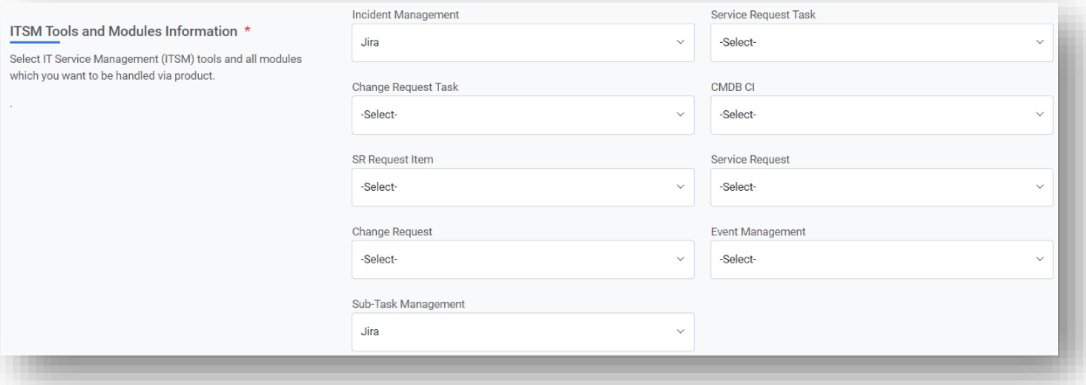
1. For that go to page Advance Configuration -> Script→ Manage Custom ScriptRBA.
2. Select **Input Mode** as Manual, **Script Language** as Python, enter the name of script in the **Script Name** textbox.
3. Enter **Tags** (if needed) and paste the content below in the **Script Text** textbox.

```
import json

import requests

import sys
```

```
try:
 ##url     =     "http://<ipaddress>:<port>/rest/api/2/issue/IT-
90/assignee"  //update assignee
## Mandory


 resp = json.loads(sys.argv[2])
 url = resp["URL"] + resp["key"] + "/assignee"


 payload = json.dumps({
    "name":  resp["assignee_name"]
  })
 headers = {
    'Authorization': 'Basic QXNoaXNoTWlzaHJhOkluZGlhQDEyMw==',
    'Content-Type': 'application/json'
  }


 response   =   requests.request("PUT",   url,   headers=headers,
data=payload)


 print(response.text)


 import requests
 import json
 import sys


 ##url    =    "http://<ipadress>:<port>/rest/api/2/issue/IT-90"
//add comment
 resp = json.loads(sys.argv[2])
 url = resp["URL"] + resp["key"]
 payload = json.dumps({
   "update": {
     "comment": [
       {
         "add": {
           "body": resp["release_comment"]
         }
       }
     ]
   }
 })

 response = requests.request("PUT", url, headers=headers, data=payload)
 print(response.text)
 import requests
 import json
 import sys
 ##url = "http://<ipaddress>:<port>/rest/api/2/issue/IT-90/worklog"   //add worklog
## Mandory
 resp = json.loads(sys.argv[2])
 url = resp["URL"] + resp["key"]+"/worklog"
 payload = json.dumps({
    "comment": resp["release_comment"],
    "timeSpentSeconds": 6000
 })
 response = requests.request("POST", url, headers=headers, data=payload)
 print(response.text)
except Exception as e:
  message = {"Error": "Error in running Script, Error=>" + str(e)}
  message = json.dumps(message)
  code = 400

  print(str(message))
```

## 4.6.2 Sub-Task Management

For Integration of Jira ITSM Sub-Task with iAutomate tool, perform the following steps:



ITSM Tools and Modules Information *

Select IT Service Management (ITSM) tools and all modules which you want to be handled via product.

Incident Management: Jira

Service Request Task: -Select-

Change Request Task: -Select-

CMDB CI: -Select-

SR Request Item: -Select-

Service Request: -Select-

Change Request: -Select-

Event Management: -Select-

Sub-Task Management: Jira

Figure 296 – Integration of Jira ITSM Sub-Task

- Create Data Source
- Fetch Data Configuration

– **Sample URL**: http://<JIRA_URL>/rest/api/2/search?fields=#columns#&jql=issuetype="Sub-task" AND status=Open AND updated >= "#start_date#" AND updated <= "#end_date#" ORDER BY updated

– Authentication Type: Basic

– Request Method: GET

– URL Path Parameters:

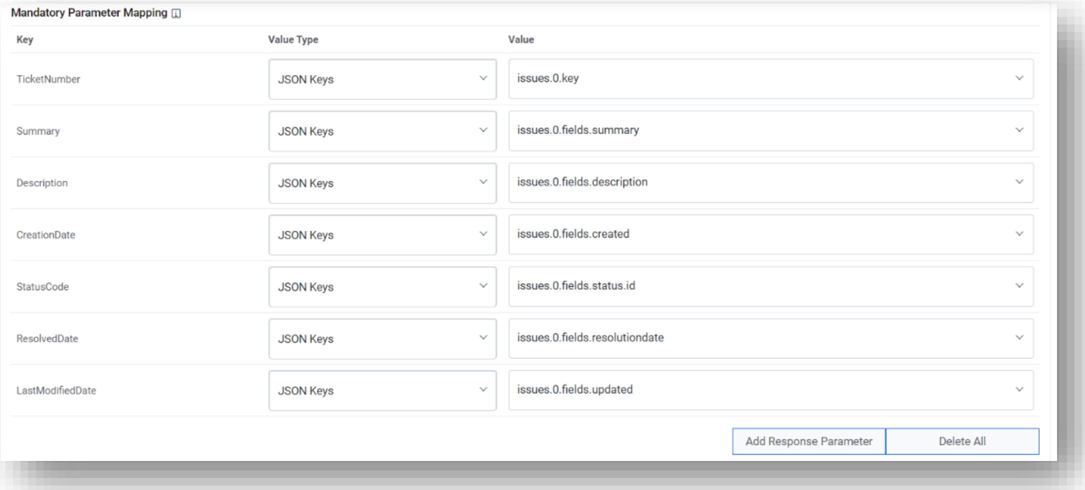
| Key | Value Type | Value |
|---|---|---|
| #columns# | Text | key,description,summary,created,updated,status,assignee,resolutiondate, issuetype |
| #start_date# | SQL UDF | @@GetFromDateTimeUsingTaskModifiedDate_Jira |
| #end_date# | SQL UDF | @@GetToolCurrentDateTime_Jira |

**Response Body:**

```
{

   "expand": "schema,names",

   "startAt": 0,

   "maxResults": 50,

   "total": 3,

   "issues": [{
       "expand":
 "operations,versionedRepresentations,editmeta,changelog,rendere
 dFields",
       "id": "10102",
       "self":
 "http://<ipaddress>:<port>/rest/api/2/issue/10102",
       "key": "IT-48",
       "fields": {
          "summary": "REST ye merry gentlemen. Rest in peace",
 "resolutiondate":"2021-05-05T13:17:10.000+0530",
          "created": "2021-05-05T13:17:10.000+0530",
          "description": "Creating of an issue using project keys
 and issue type names using the REST API",
          "assignee": null,
          "updated": "2021-05-05T13:17:10.000+0530",
          "status": {
             "self":                                    "http://
 <ipaddress>:<port>/rest/api/2/status/1",
             "description": "The issue is open and ready for the
 assignee to start work on it.",
             "iconUrl":
 "http://<ipaddress>:<port>/images/icons/statuses/open.png",
             "name": "Open",
             "id": "1",
             "statusCategory": {
                "self":
 "http://<ipaddress>:<port>/rest/api/2/statuscategory/2",
                "id": 2,
                "key": "new",
                "colorName": "blue-gray",
                "name": "To Do"
             }
```

```
                  }






            }

      } ]

}
```

- Mandatory Parameter Mapping:

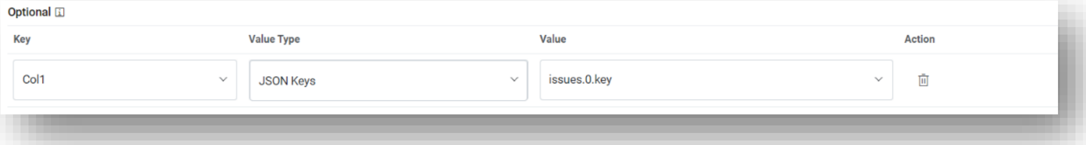


Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | issues.0.key |
| Summary | JSON Keys | issues.0.fields.summary |
| Description | JSON Keys | issues.0.fields.description |
| CreationDate | JSON Keys | issues.0.fields.created |
| StatusCode | JSON Keys | issues.0.fields.status.id |
| ResolvedDate | JSON Keys | issues.0.fields.resolutiondate |
| LastModifiedDate | JSON Keys | issues.0.fields.updated |

Figure 297 - Mandatory Parameter Mapping

- Optional:



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| Col1 | JSON Keys | issues.0.key | 🗑 |

Figure 298 - Optional

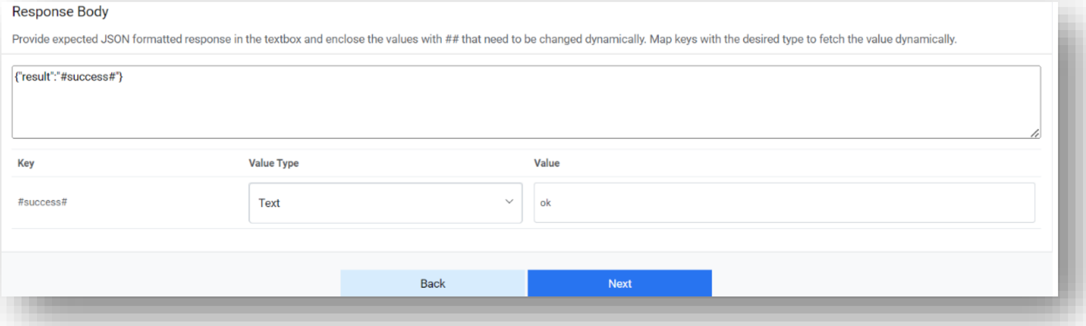
– **Release Rule:**

For release, Jira has 3 different APIs to change the assignee, to add a comment and to add worklog. So, we are using iAutomate’s Custom Script API to update all 3 operations with a single API.

- URL: http://<ipaddress>: <port>/rest/api/2/issue/#key#/assignee
- Authentication Type: Basic
- **UserId**: <ApiUser@hcl.com>
- **Password**: <user_password>
- Request Method: POST

Request Body:

```
{
    "key": "#ticketId#",
    "URL": "http://<ipaddress>:<port>/rest/api/2/issue/",
    "assignee_name": "#assignee_name#",
    "release_comment":"Ticket released from iAutomate"
             }
      Response Body:
{"result":"#success#"}
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{"result":"#success#"}
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

Figure 299 - Response Body

- **Close Rules:**
  - URL: http://10.1.152.20:8080/rest/api/2/issue/#key#/transitions
  - Authentication Type: Basic
  - Request Method: POST
  - URL Path Parameters:

| Key | Value Type | Value |
|---|---|---|
| #key# | Table.Columns | Col1 |

**Request Body:**

```
{
    "update": {
        "comment": [
            {
                "add": {
                    "body": "#worknote#"
                }
            }
        ]

    },

    "transition": {

        "id": "#statuscode#"

    }

}

Response Body: { "result" : "ok" }
```

– **InProgress Rules:**

- **URL**: http://<ipaddress>:<port>/rest/api/2/issue/#sysid#/transitions
- Authentication Type: Basic
- Request Method: POST
- URL Path Parameters:

| Key | Value Type | Value |
|---|---|---|
| #key# | Table.Columns | Col1 |

**Request Body:**

```
{

    "update": {

        "comment": [

            {

                "add": {

                    "body": "#worknote#"

                }

            }

        ]

    },

    "transition": {

        "id": "#statuscode#"

      }

}
```

**Response Body:**

```
{ "result" : "ok" }
```

**JsResponseConverter**: After successful creation of data source,

1. Go to CollectIncident job under menu **Configuration** -> **Manage Jobs**.
2. Click on ⚙ icon. A popup will be opened.
3. Go to parameter tab and search for 'JsResponseConverter' in the end.
4. Replace its value with below string:

```
if(json.issues){for(var
result=[],i=0;i<json.issues.length;i++)result.push(json.issues[
i]);customJobject.dataCollectorNode.data.issues=result}
```

- **Manage Rules**

For each of the release, close and inprogress, rules will be defined as follows:

**Release Rules:**

| Parameter | Value Type | Value |
|---|---|---|
| #assignee_name# | Text | <Assignee_user> |
| #ticketId# | Table.Columns | Col1 |

**Close Rules:**

| Parameter | Value Type | Value |
|---|---|---|
| #worknote# | Text | Ticket resolved from iAutomate |
| #statuscode# | Text | 61 |

**In Progress Rules:**

| Parameter | Value Type | Value |
|---|---|---|
| #worknote# | Text | Ticket marked to in progress |
| #statuscode# | Text | 11 |

- **Manage Custom Script:**

To use multiple Jira APIs that are being used while releasing an incident, we need a python script that contains the calling of all required APIs.

1. For that go to page Advance Configuration ->Script -> Manage Custom Script -> Create Script.
2. Select Manual as Input Mode, Python as Script Language, enter the name of script in the Script Name textbox.
3. Enter tags if needed and paste below content as it is in **Script Text** textbox.

```
import json

import requests

import sys
```

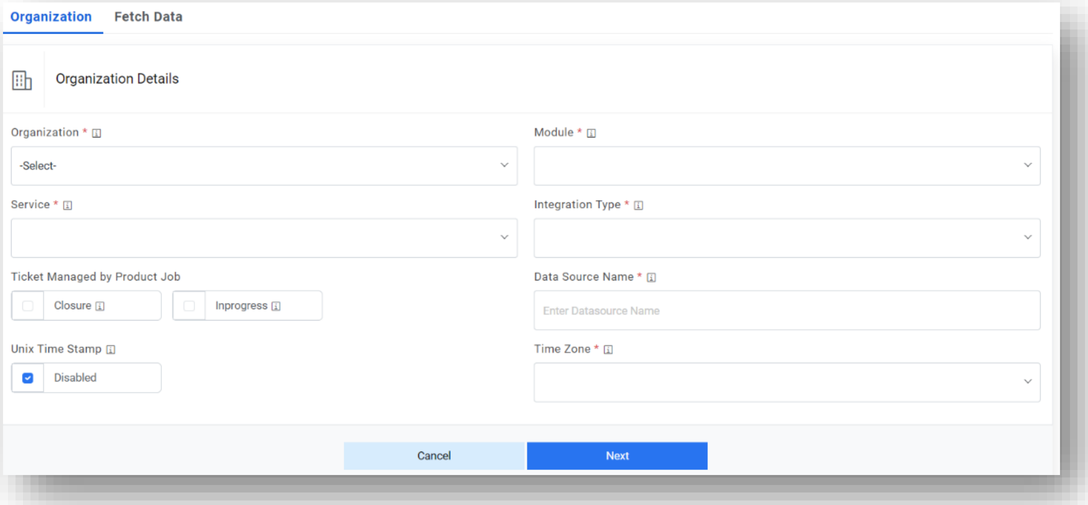
```
try:
 ##url   =   "http://   <ipaddress>:<port>/rest/api/2/issue/IT-
90/assignee"  //update assignee
## Mandory

 resp = json.loads(sys.argv[2])
 url = resp["URL"] + resp["key"] + "/assignee"

 payload = json.dumps({
    "name":  resp["assignee_name"]
  })
 headers = {
    'Authorization': 'Basic QXNoaXNoTWlzaHJhOkluZGlhQDEyMw==',
    'Content-Type': 'application/json'
  }

 response   =   requests.request("PUT",   url,   headers=headers,
data=payload)

 print(response.text)

 import requests
 import json
 import sys

 ##url   =   "http://<ipaddress>:<port>/rest/api/2/issue/IT-90"
//add comment
 resp = json.loads(sys.argv[2])
 url = resp["URL"] + resp["key"]
 payload = json.dumps({
   "update": {
     "comment": [
       {
         "add": {
           "body": resp["release_comment"]
         }
       }
     ]
   }
 })


 response = requests.request("PUT", url, headers=headers,
data=payload)


 print(response.text)
 import requests
 import json
 import sys


 ##url = "http://<ipaddress>:<port>/rest/api/2/issue/IT-
90/worklog"   //add worklog
## Mandory
 resp = json.loads(sys.argv[2])
 url = resp["URL"] + resp["key"]+"/worklog"
 payload = json.dumps({
    "comment": resp["release_comment"],
    "timeSpentSeconds": 6000
 })
 response   =   requests.request("POST",   url,   headers=headers,
data=payload)


 print(response.text)


except Exception as e:
  message = {"Error": "Error in running Script, Error=>" + str(e)}
  message = json.dumps(message)
  code = 400
  print(str(message))
```

## 4.7 Integration with ServiceXchange

### 4.7.1 Incident Management

To create data source for Incident Management, perform the following steps.

1. On the left menu bar, click Configuration-> Manage Data Sources.
2. The **Create Data Source** page appears with the following tabs:
    - Organization
    - Fetch Data
    - Release Rules
    - Close Rules
    - InProgress Rules



Organization *: -Select-

Module *:

Service *:

Integration Type *:

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name *: Enter Datasource Name

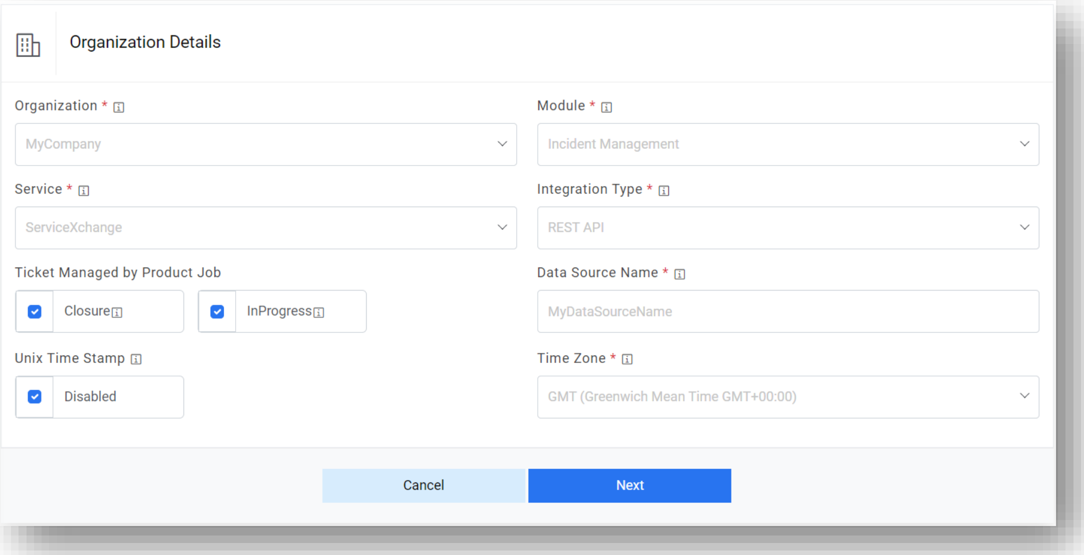
Unix Time Stamp: ☑ Disabled

Time Zone *:

Figure 300 - Create Data Source

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

3. On the **Organization** tab,
   - Select the Organization Name from the dropdown.
   - Select the Module as Incident Management, since we are configuring this data source for pulling the incident tickets.
   - Select the Service as SX Tool as we are configuring the data source for Cherwell
   - Select the Integration Type as REST, since we will be integrating through REST APIs.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select **Timestamp** to view the present data with date and time.
   - Click Next.



Organization Details

Organization * MyCompany

Module * Incident Management

Service * ServiceXchange

Integration Type * REST API

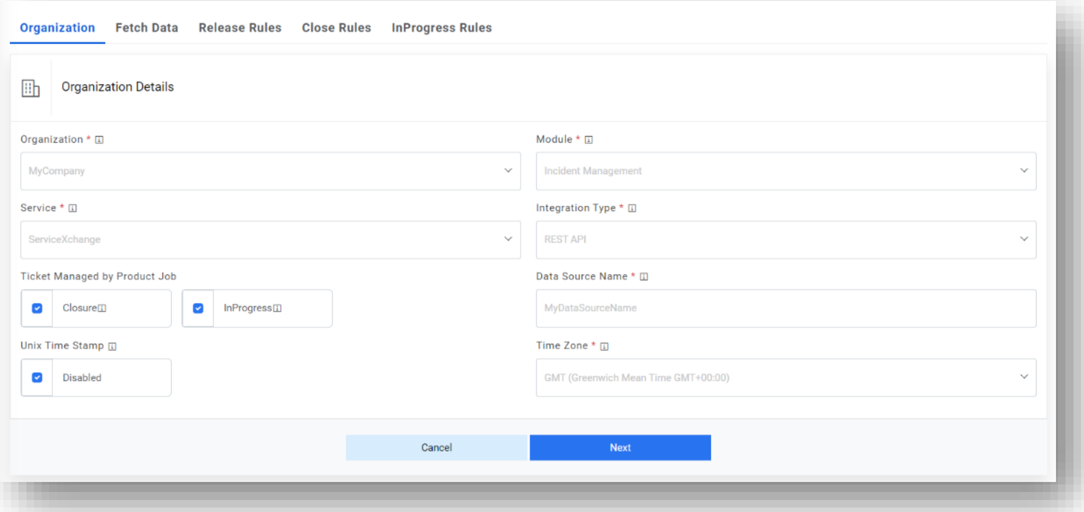
Ticket Managed by Product Job ☑ Closure ☑ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 301 - Create Data Source (Cont.)



Organization Details

Organization * MyCompany

Module * Incident Management

Service * ServiceXchange

Integration Type * REST API

Ticket Managed by Product Job ☑ Closure ☑ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 302 - Create Data Source (Cont.)

4. On the Fetch Data tab, type in the details as per the environment.

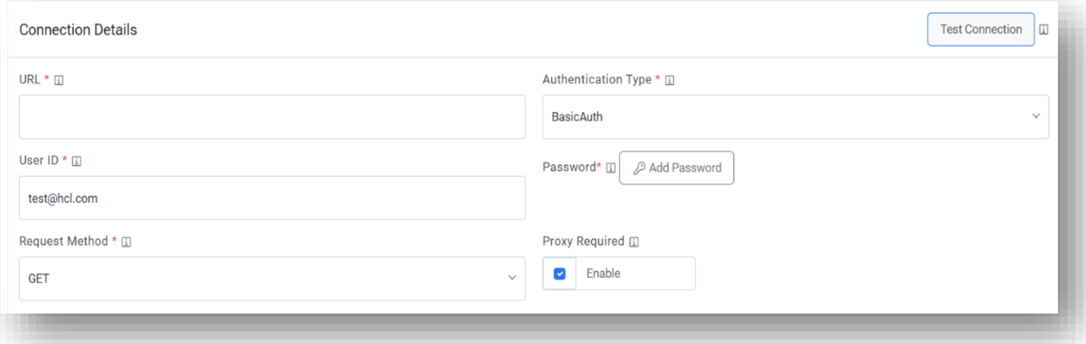
5. In the Connection Details section, enter the following details:

- **URL** – Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- Sample URL– http://<iAutomate_API_URL>/iAutomateAPI/Request/GetIncidentTicketData/<Org_ID>?ModuleId=1&start_date>=#Start_Date#&end_date<=#End_Date#&
- Here, < iAutomate_API_URL > is the API URL of iAutomate where Push APIs are present and <Org_ID> is the OrgID for the organization for which you are creating the data source. It is available in Organization Master in Database.
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0

The user details that are entered here should be an API User

- Selection of **Basic / Windows** requires you to enter -
  - User Id
  - Password.
- Selection of **OAuth 2.0** requires you to enter -
  - User Id
  - Password
  - Authentication URL
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



URL *

Authentication Type * BasicAuth

User ID * test@hcl.com

Password*

Request Method * GET

Proxy Required ☑ Enable

Figure 303 - Create Data Source (Connection Details)

- **Password** – For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select

Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.

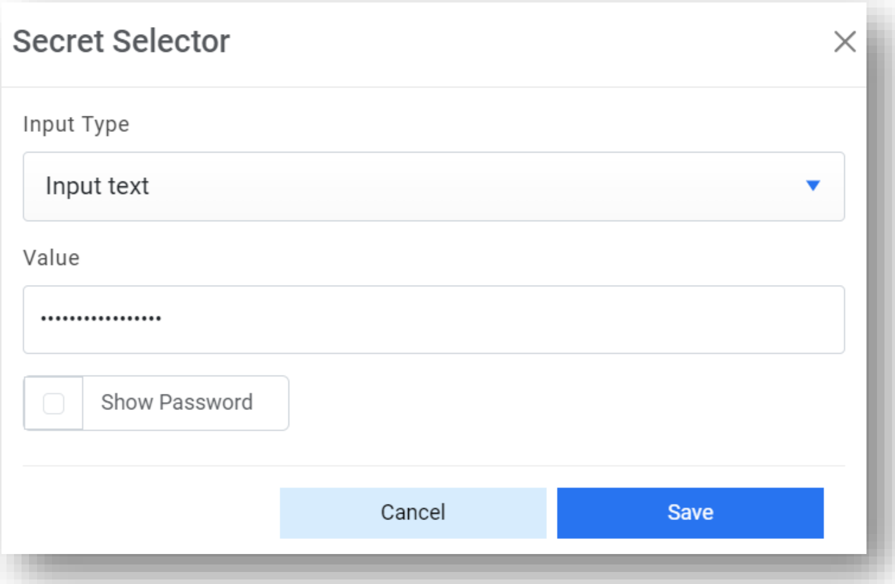
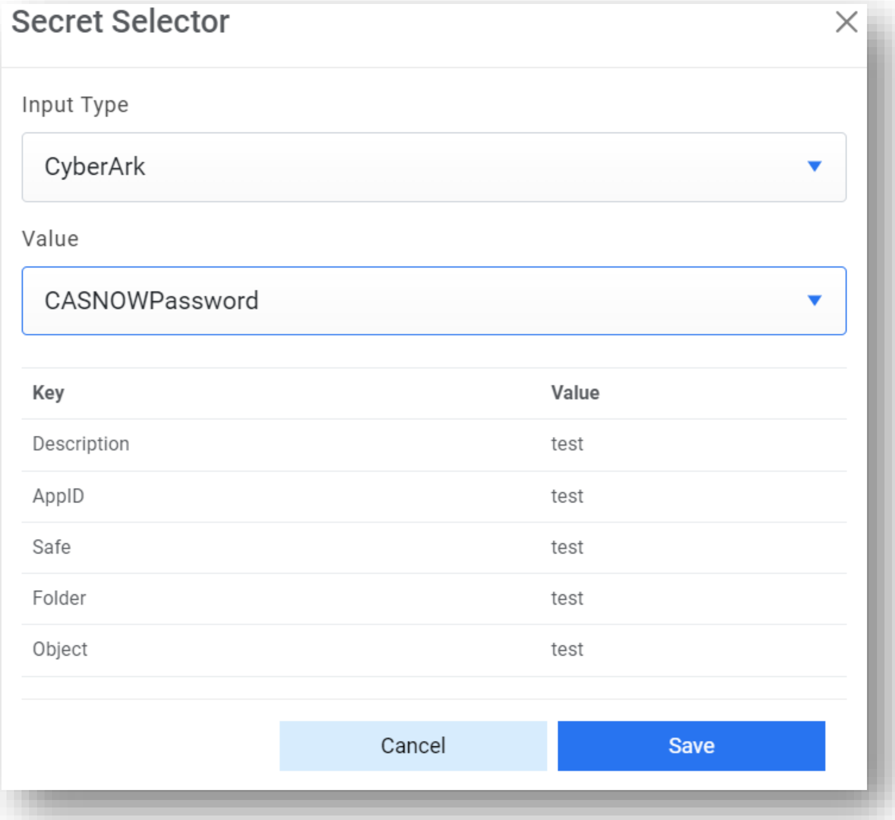
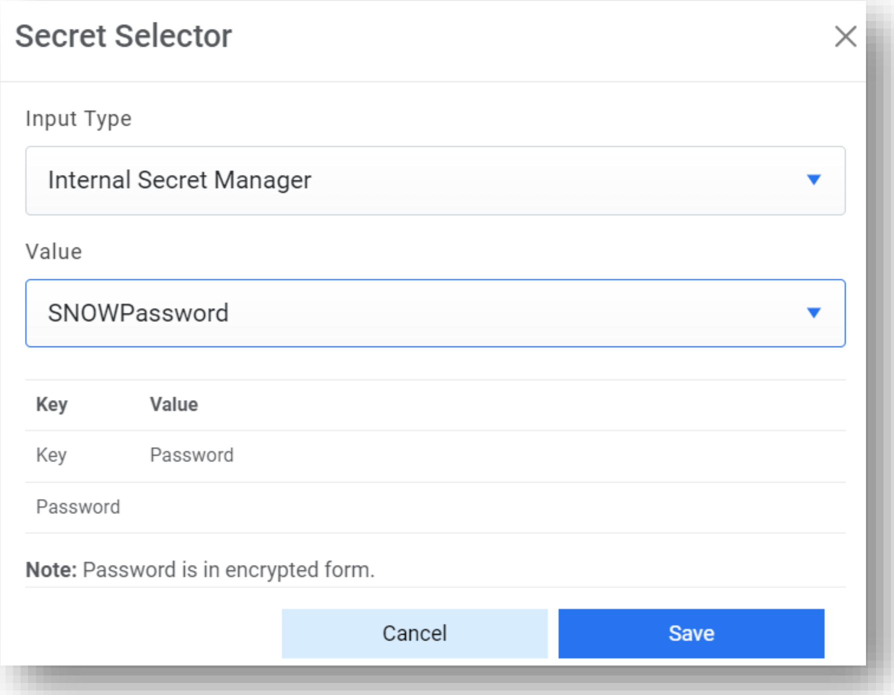


Figure 304 - Password in plaintext



| Key | Value |
|---|---|
| Description | test |
| AppID | test |
| Safe | test |
| Folder | test |
| Object | test |

Figure 305 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

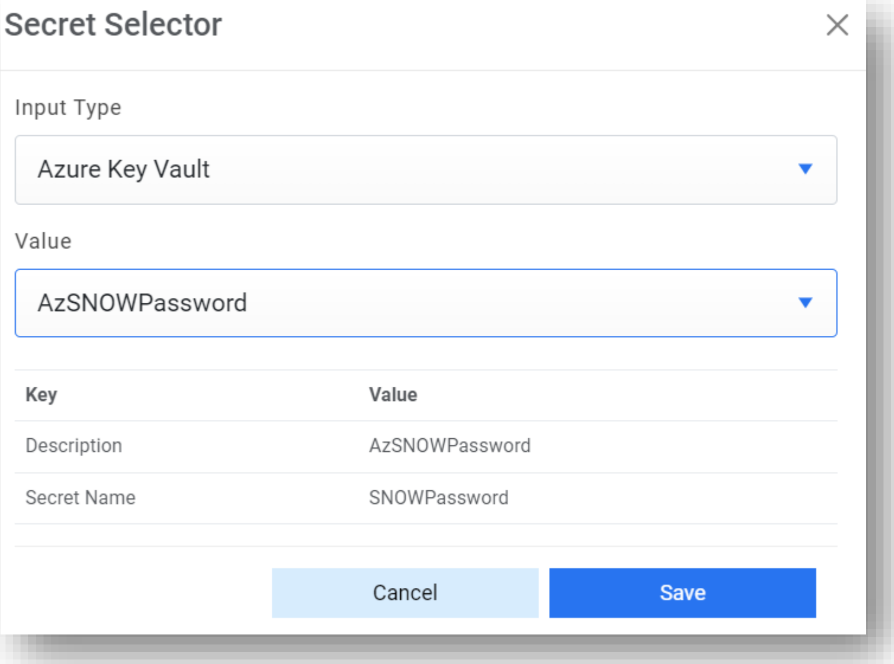
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 306 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

AzSNOWPassword

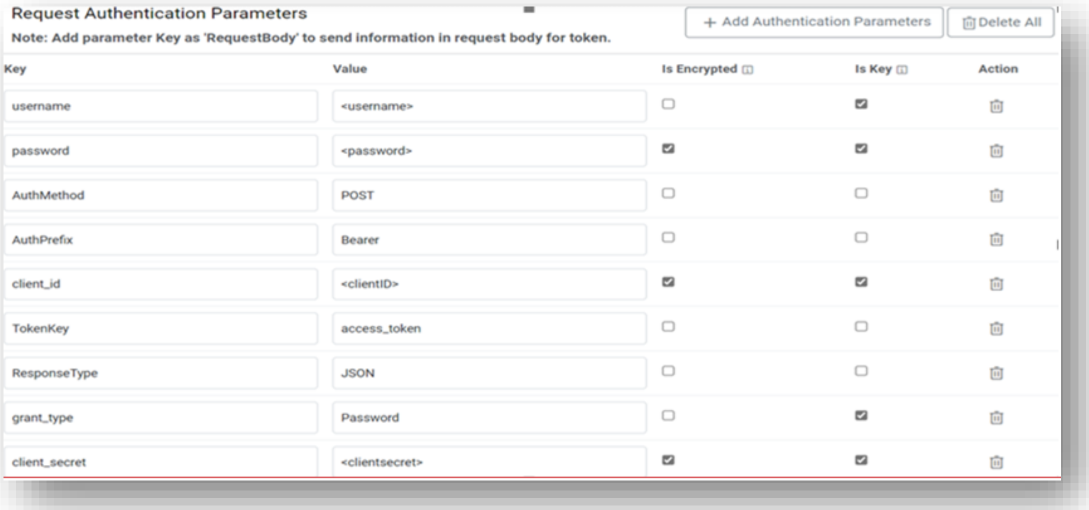
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 307 - Password from Azure Key Vault

6. **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab. Based on the **Authentication Type**, add the parameters mentioned in the below table.

Table 50– Sample Authentication Parameters

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | username | <username> | NO | YES |
| OAuth2.0 | password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

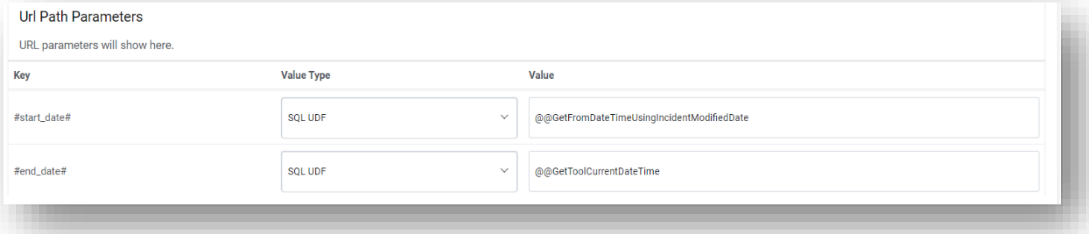
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | □ | □ | 🗑 |
| AuthPrefix | Bearer | □ | □ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | □ | □ | 🗑 |
| ResponseType | JSON | □ | □ | 🗑 |
| grant_type | Password | □ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 308 - Create Data Source (Request Authentication Parameters for OAuth2.0)

7. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

| Key | Value Type | Value |
|---|---|---|
| #start_date# | SQL UDF | @@GetFromDateTimeUsingIncidentModifiedDate_ServiceXchange |
| #end_date# | SQL UDF | @@GetToolCurrentDateTime_ServiceXchange |



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #start_date# | SQL UDF | @@GetFromDateTimeUsingIncidentModifiedDate |
| #end_date# | SQL UDF | @@GetToolCurrentDateTime |

Figure 309– URL Path Parameters

8. **Request Header Parameters –** Please enter the request header parameters as required.

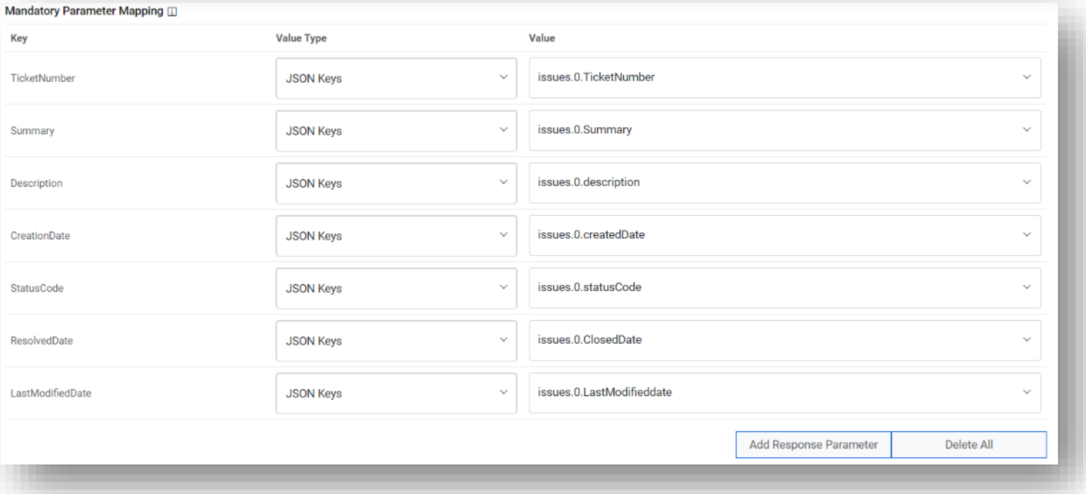
**Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below:

```
Response Body –
{
    "statusCode": 200,
    "status": "Success",
    "message": null,
    "result": [
        {
            "TicketNumber": "INC0303869",
            "Summary": "testing",
            "Description": "testing data",
            "AssignedGroup": "xxxxxxxx",
            "StatusCode": "1",
            "CreationDate": "2022-09-23 09:26:52.000",
            "LastModifiedDate": "2022-09-23 09:26:52.000",
            "ClosedDate": "2022-09-22 06:24:52.000",
            "sys_id": "xxxxxxxx",
            "Col1": "",
            "Col2": "",
            "Col3": "",
            "Col4": "",
            "Col5": "",
            "Col6": "",
            "Col7": "",
            "Col8": "",
            "Col9": "",
            "Col10": "",
            "iAutomate_CreatedDateInGMT":"2022-09-23
09:27:22.773",
              "iAutomate_UpdatedDateInGMT":              "2022-09-23
09:27:22.773"

         }

     ]

}
```

9. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
10. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 51– Sample Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.TicketNumber |
| Summary | JSON.Keys | result.0.Summary |
| Description | JSON.Keys | result.0.Description |
| CreationDate | JSON.Keys | result.0.CreationDate |
| StatusCode | JSON.Keys | result.0.StatusCode |
| ResolvedDate | JSON.Keys | result.0.ClosedDate |
| LastModifiedDate | JSON.Keys | result.0.LastModifiedDate |



Mandatory Parameter Mapping

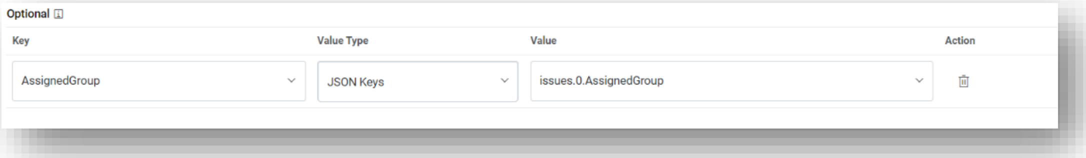
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | issues.0.TicketNumber |
| Summary | JSON Keys | issues.0.Summary |
| Description | JSON Keys | issues.0.description |
| CreationDate | JSON Keys | issues.0.createdDate |
| StatusCode | JSON Keys | issues.0.statusCode |
| ResolvedDate | JSON Keys | issues.0.ClosedDate |
| LastModifiedDate | JSON Keys | issues.0.LastModifieddate |

Figure 310 - Mandatory Parameter Mapping

If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 52 - Sample Optional Parameters

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | result.0.AssignedGroup |



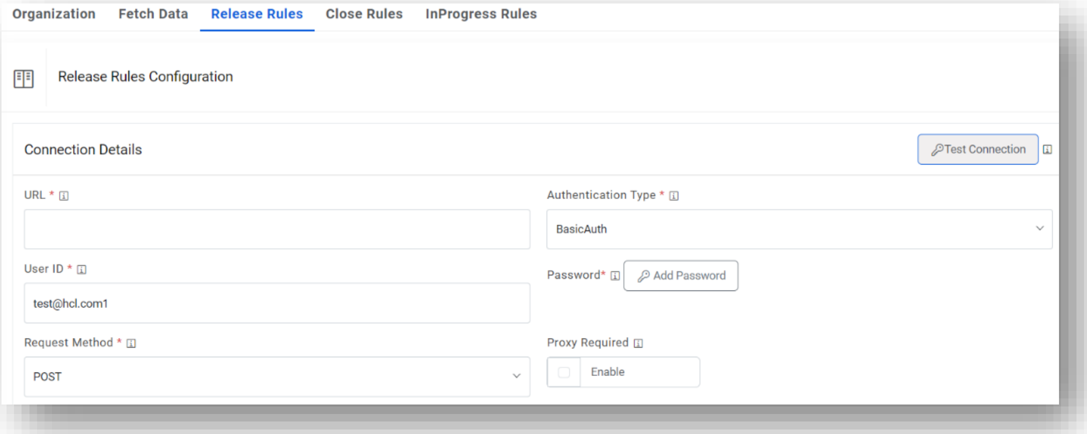
Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | issues.0.AssignedGroup | 🗑 |

Figure 311 - Optional Parameter Mapping

11. Click Next to proceed to Release Rules.
12. On **Release Rules** tab, type in the details as per the requirement.
13. In the **Connection Details** section, enter the following details:

- **URL** – Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- Sample URL - https://inboundBoomiDevCHN1.dryicehcl.com/ws/simple/updateIncidentInSX
- **Authentication Type** – Please enter the information in line with the Authentication type configured for fetching data configuration previously. For e.g., Basic.
- **Request Method** – Select Request Method as POST from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.



URL *

Authentication Type * BasicAuth

User ID * test@hcl.com1

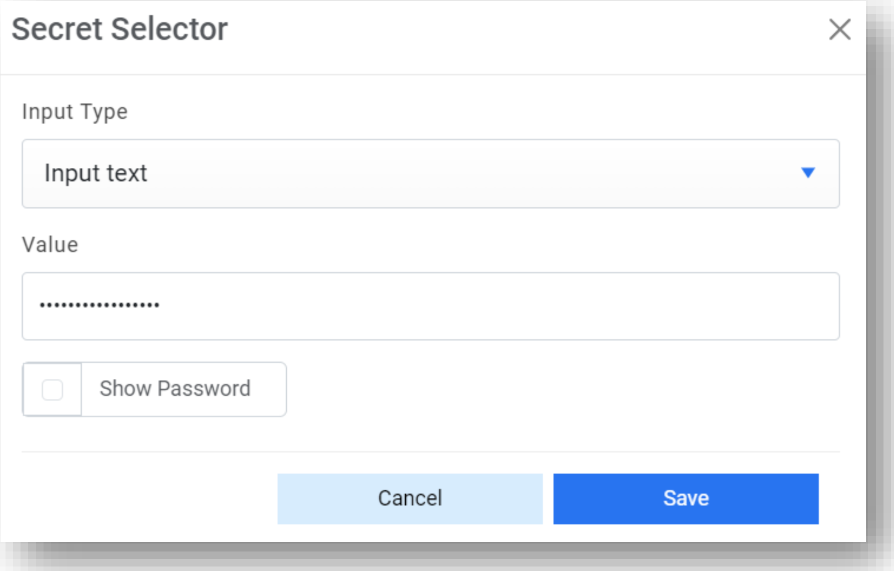
Password*

Request Method * POST

Proxy Required □ Enable

Figure 312 - Release Rules (Connection Details)

- **Password** - For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



Secret Selector

Input Type

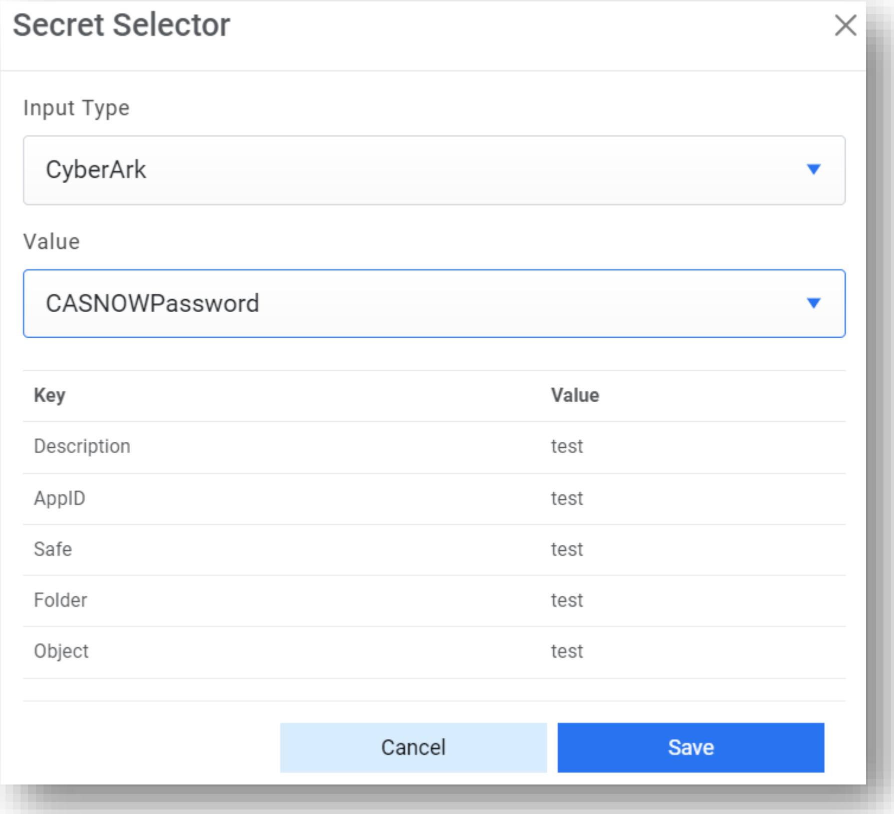
Input text

Value

•••••••••••••••••

□ Show Password

Figure 313 - Password in Plaintext



Secret Selector

Input Type

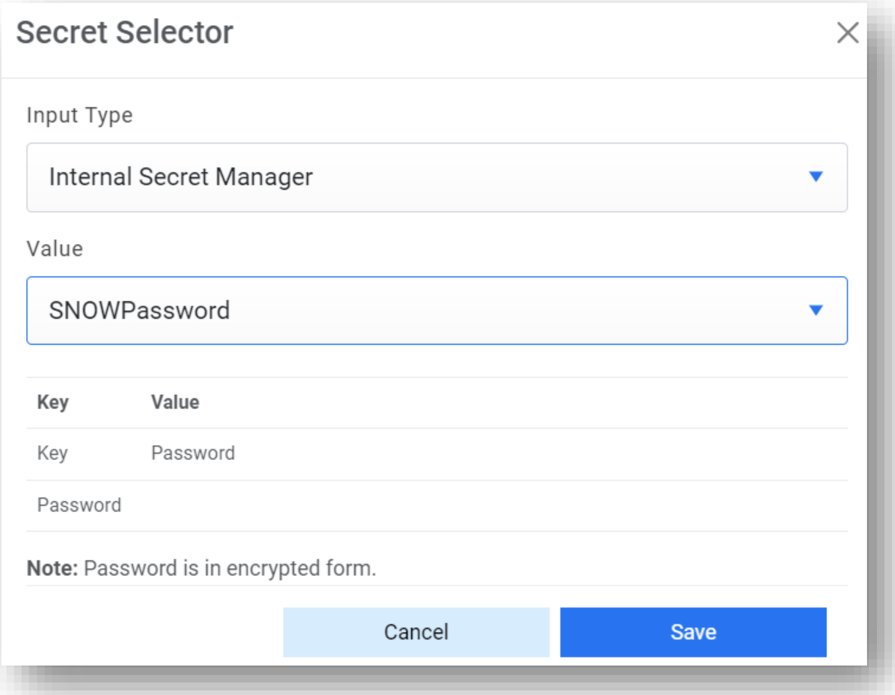
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | test |
| AppID | test |
| Safe | test |
| Folder | test |
| Object | test |

Figure 314 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

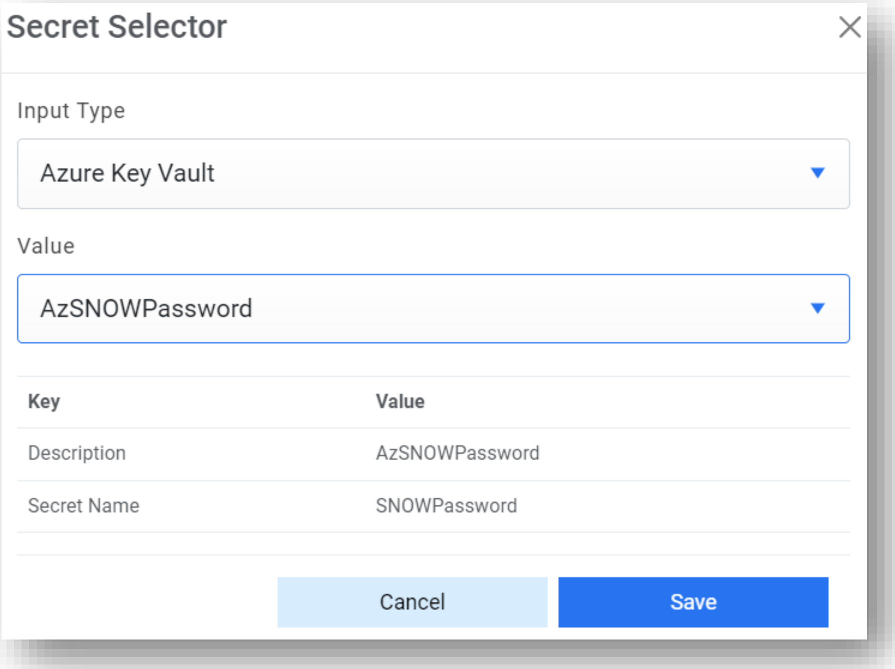
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 315 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

AzSNOWPassword

| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

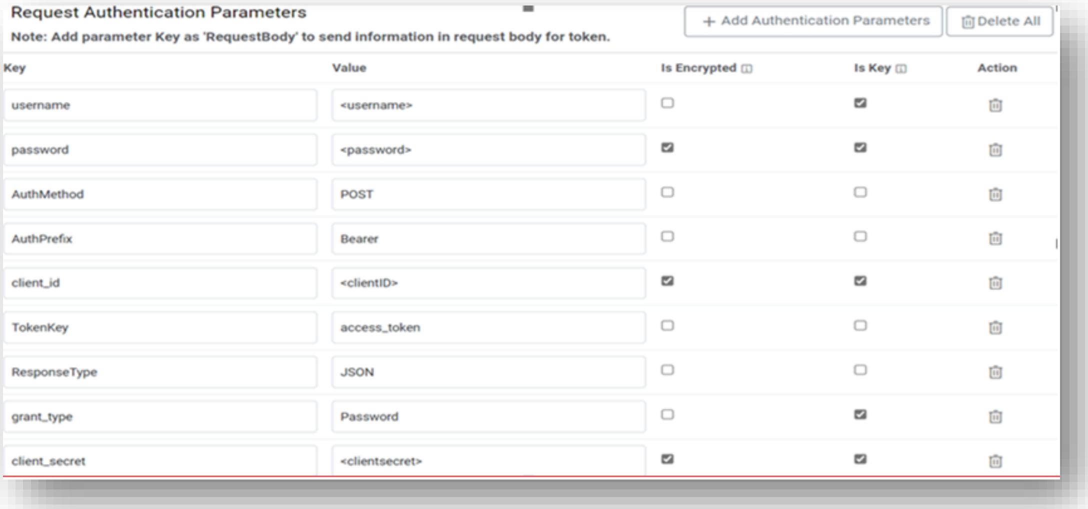
Figure 316 - Password from Azure Key Vault

14. **Request Authentication Parameters -** If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
15. Based on the **Authentication Type,** add the parameters mentioned in the below table

Table 53 - Sample Authentication Parameters

| Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|
| grant_type | password | N | Y |

|  |  |  |  |
|---|---|---|---|
| username | <username> | N | Y |
| Password | <password> | Y | Y |
| client_id | <client_id> | N | Y |
| AuthPrefix | Bearer | N | N |
| AuthMethod | POST | N | N |
| ResponseType | JSON | N | N |
| TokenKey | access_token | N | N |



Request Authentication Parameters

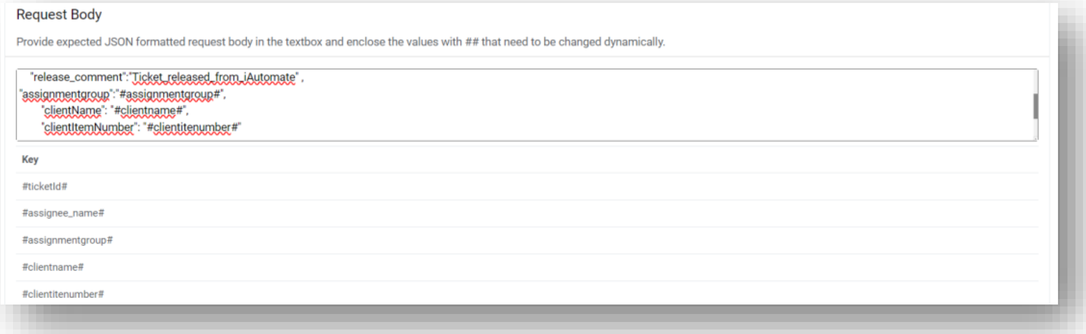
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | □ | □ | 🗑 |
| AuthPrefix | Bearer | □ | □ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | □ | □ | 🗑 |
| ResponseType | JSON | □ | □ | 🗑 |
| grant_type | Password | □ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 317 - Create Data Source (Request Authentication Parameters)

16. **Request Body -** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body -
{
        "ticketnumber": "#ticket#",
        "status": "#status#",
        "worknote": "#worknote#",
        "assignmentgroup":"#assignmentgroup#",
        "clientName": "#clientname#",
        "clientItemNumber": "#clientitenumber#"
}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

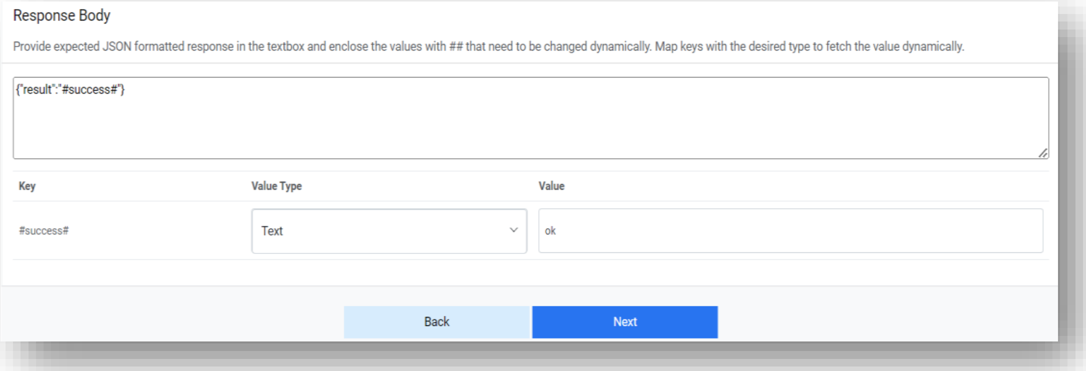
```
   "release_comment":"Ticket_released_from_iAutomate" ,
"assignmentgroup":"#assignmentgroup#",
      "clientName": "#clientname#",
      "clientItemNumber": "#clientitenumber#"
```

| Key |
|---|
| #ticketId# |
| #assignee_name# |
| #assignmentgroup# |
| #clientname# |
| #clientitenumber# |

Figure 318 – Request Body

17. **Response Body –** In this section, please enter the response body in JSON format. A sample request is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{"result":"#success#"}
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

Figure 319 - Release Rules (Response Body)

18. **Response Key Value** mapping can be done as per the below table.

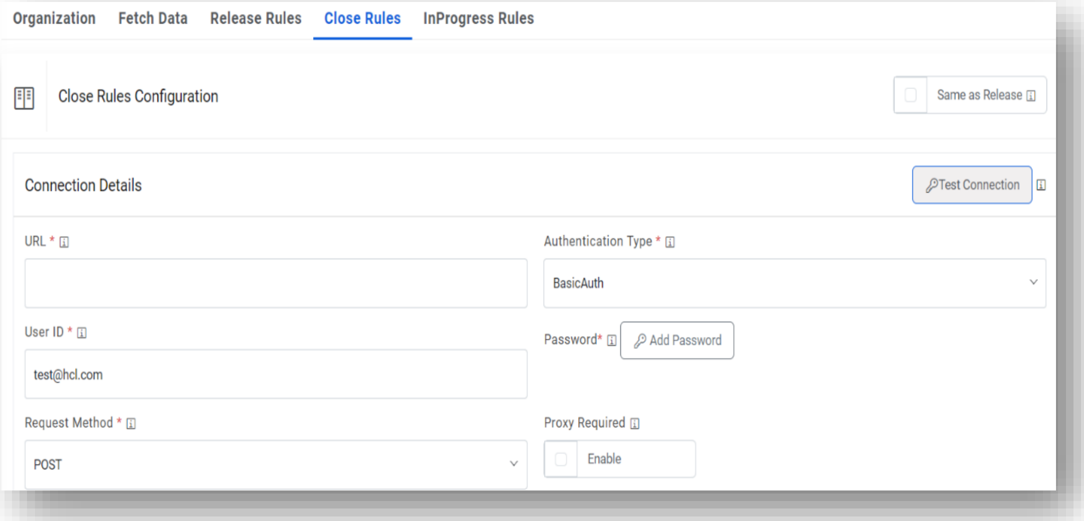
Table 54 - Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

19. Click **Save** to add the data source.
20. On **Close Rules** tab, type in the details as per the requirement.
21. In the **Connection Details** section, enter the following details:

- Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- Sample URL - https://<myhost>
- **Authentication Type** – Please enter the information in line with the Authentication type configured for fetching data configuration previously. For e.g., Basic.
- **Request Method** – Select Request Method as POST from the drop-down.

- **Proxy Required** – Check Proxy Required, if the environment needs access to content from data sources outside the firewall.



Close Rules Configuration

□ Same as Release

Connection Details

URL *

Authentication Type * BasicAuth

User ID * test@hcl.com

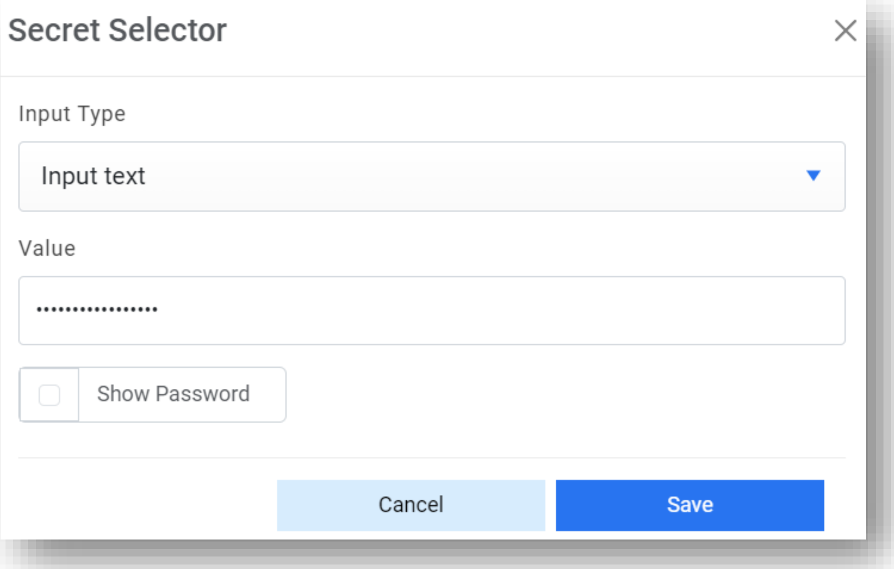
Password*

Request Method * POST

Proxy Required □ Enable

Figure 320- Close Rules (Connection Details)

- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.
- **Password** – For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.

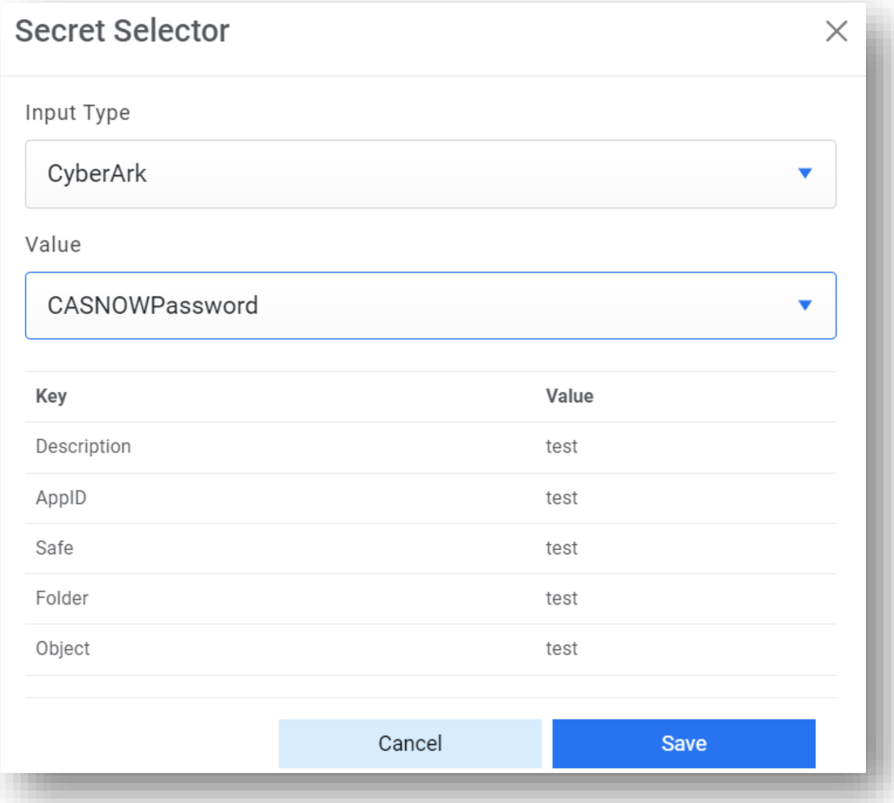

Secret Selector

Input Type: Input text

Value: ••••••••••••••••

□ Show Password

Figure 321 - Password in Plaintext



Secret Selector

Input Type

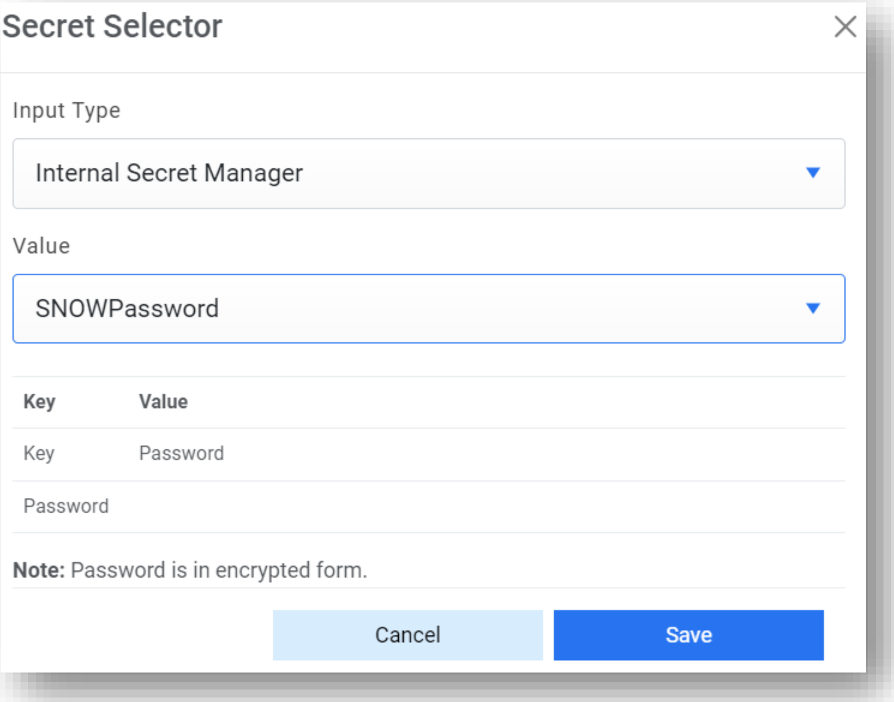
CyberArk

Value

CASNOWPassword

| Key | Value |
|---|---|
| Description | test |
| AppID | test |
| Safe | test |
| Folder | test |
| Object | test |

Figure 322 - Password from Key Vault (CyberArk)



Secret Selector

Input Type

Internal Secret Manager

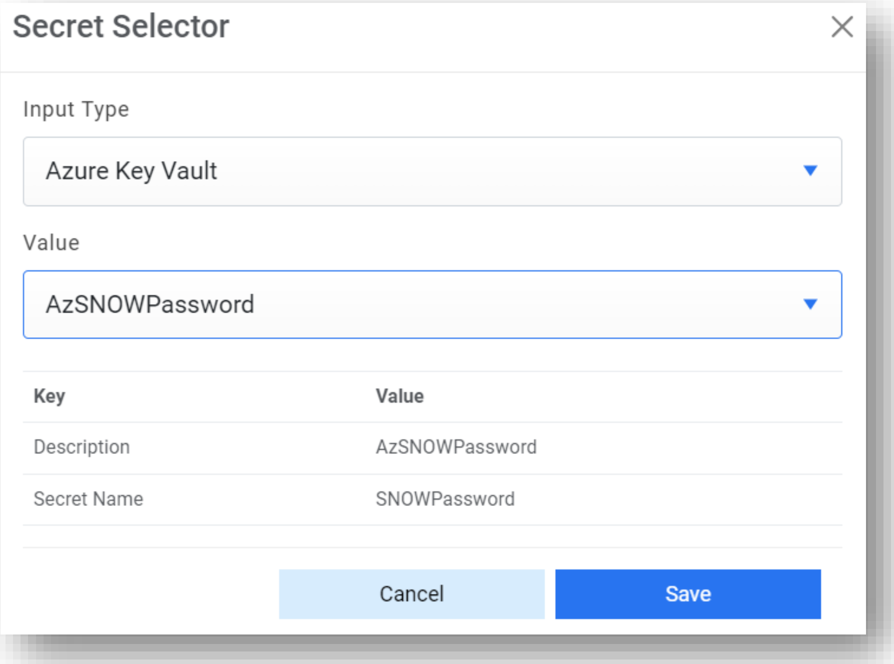
Value

SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 323 - Password from Secret Manager



Secret Selector

Input Type

Azure Key Vault

Value

AzSNOWPassword

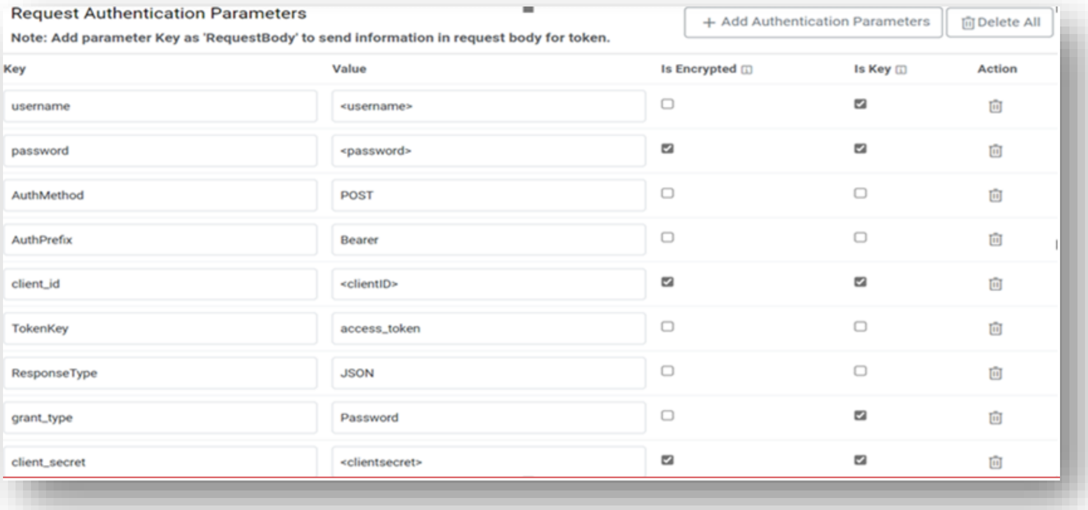
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 324 - Password from Azure Key Vault

22. **Request Authentication Parameters -** If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
23. Based on the **Authentication Type**, add the parameters mentioned in the below table

Table 55 – Sample Authentication Parameters

| Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|
| grant_type | password | N | Y |
| username | <username> | N | Y |
| Password | <password> | Y | Y |
| client_id | <client_id> | N | Y |
| AuthPrefix | Bearer | N | N |
| AuthMethod | POST | N | N |
| ResponseType | JSON | N | N |
| TokenKey | access_token | N | N |



Request Authentication Parameters

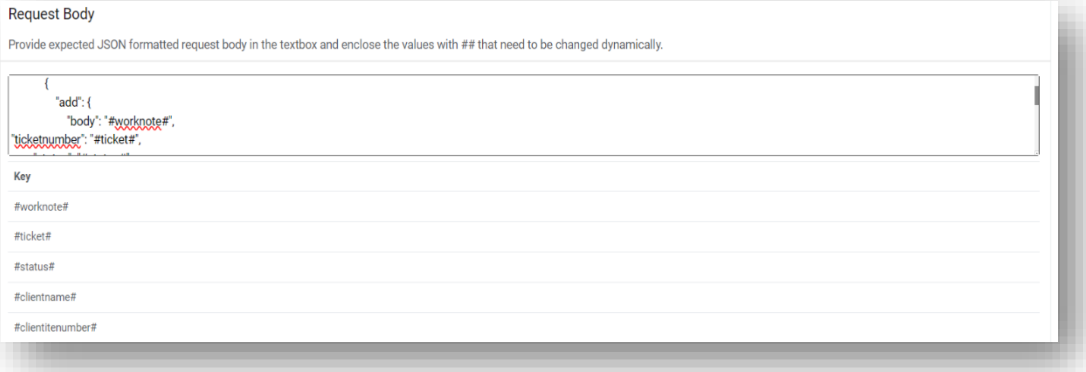
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | □ | □ | 🗑 |
| AuthPrefix | Bearer | □ | □ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | □ | □ | 🗑 |
| ResponseType | JSON | □ | □ | 🗑 |
| grant_type | Password | □ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 325 - Create Data Source (Request Authentication Parameters)

24. **Request Body -** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –

{

         "ticketnumber": "#ticket#",

         "status": "#status#",

         "worknote": "#worknote#",

         "clientName": "#clientname#",

         "clientItemNumber": "#clientitenumber#"

     }
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

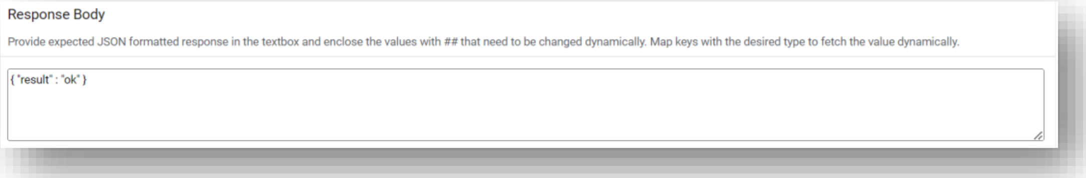
```
      {
          "add": {
              "body": "#worknote#",
"ticketnumber": "#ticket#",
```

| Key |
|---|
| #worknote# |
| #ticket# |
| #status# |
| #clientname# |
| #clientitenumber# |

Figure 326 - Close Rules (Request Body)

25. **Response Body –** In this section, please enter the response body in JSON format. A sample request is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "ok" }
```

Figure 327 - Close Rules (Response Body)

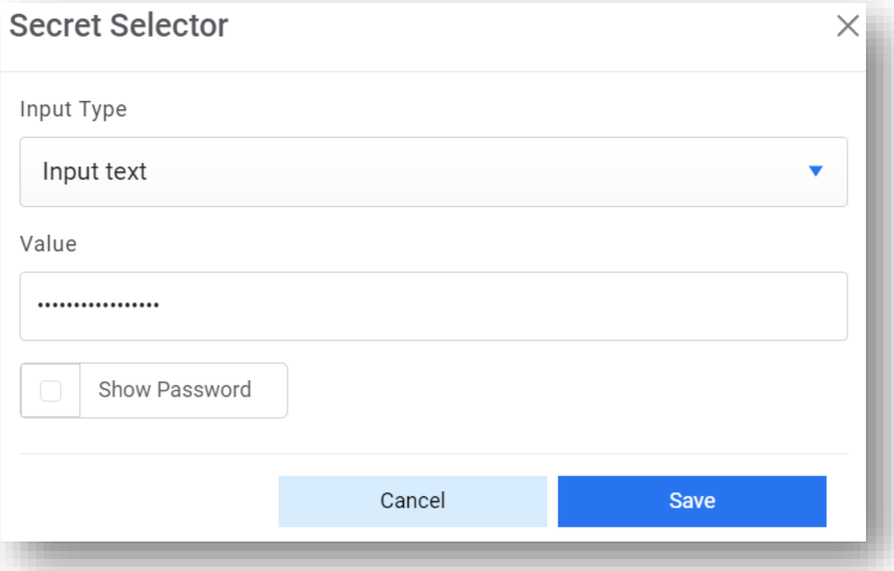
26. **Response Key Value** mapping can be done as per the below table.

Table 56 - Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

27. Click **Save** to add the data source.
28. On the **InProgress Rules** tab, type in the details as per the requirement.
29. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<ipaddress>:<port>
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously. For e.g., **Basic**.
- **Request Method** – Select Request Method as POST from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.
- **Password** - For password, click on icon next to it. If the password is available in plaintext, then select Input type as Input Text and enter the password in Value field. Else if it is available in Azure Key Vault then select Input Type as Azure Key Vault and then select any of the configured details from the value field. Else if it is available in any Key Vault such as CyberArk or Secret Manager then select Input Type as CyberArk or Secret Manager respectively and then select any of the configured details from the value field.



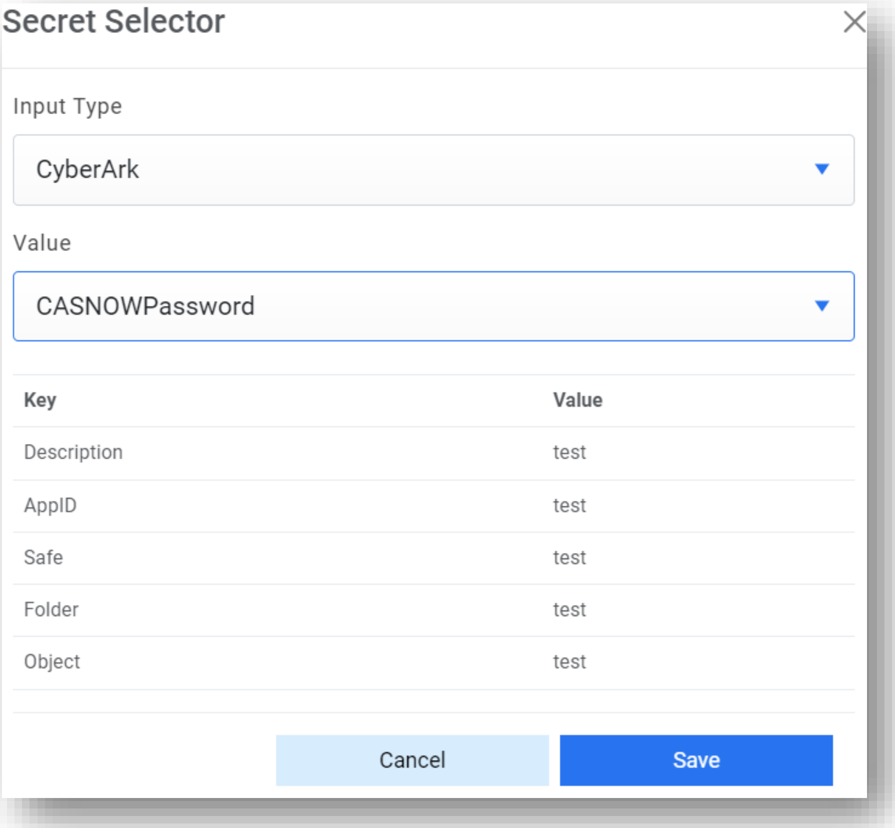
Secret Selector

Input Type: Input text

Value: ••••••••••••••••

□ Show Password

Figure 328 - Password in Plaintext



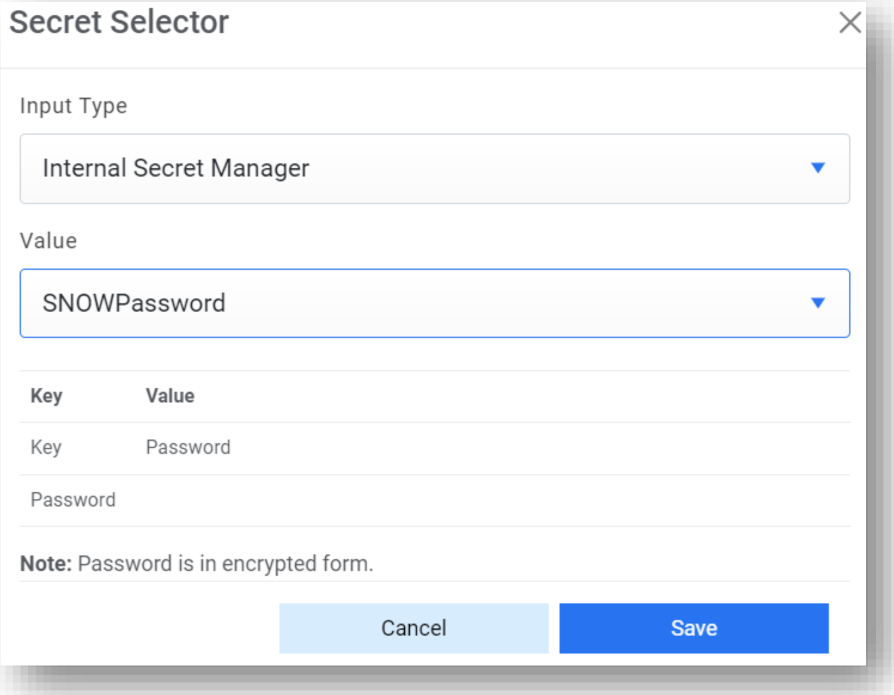
Secret Selector

Input Type: CyberArk

Value: CASNOWPassword

| Key | Value |
|---|---|
| Description | test |
| AppID | test |
| Safe | test |
| Folder | test |
| Object | test |

Figure 329 - Password from Key Vault (CyberArk)



Secret Selector

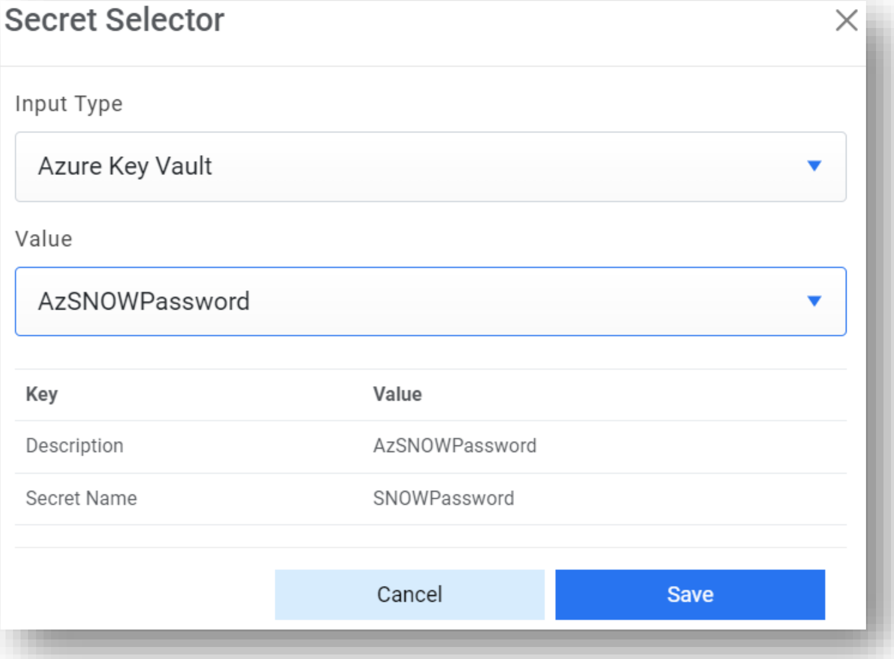
Input Type: Internal Secret Manager

Value: SNOWPassword

| Key | Value |
|---|---|
| Key | Password |
| Password | |

**Note:** Password is in encrypted form.

Figure 330 - Password from Secret Manager



Secret Selector

Input Type: Azure Key Vault

Value: AzSNOWPassword

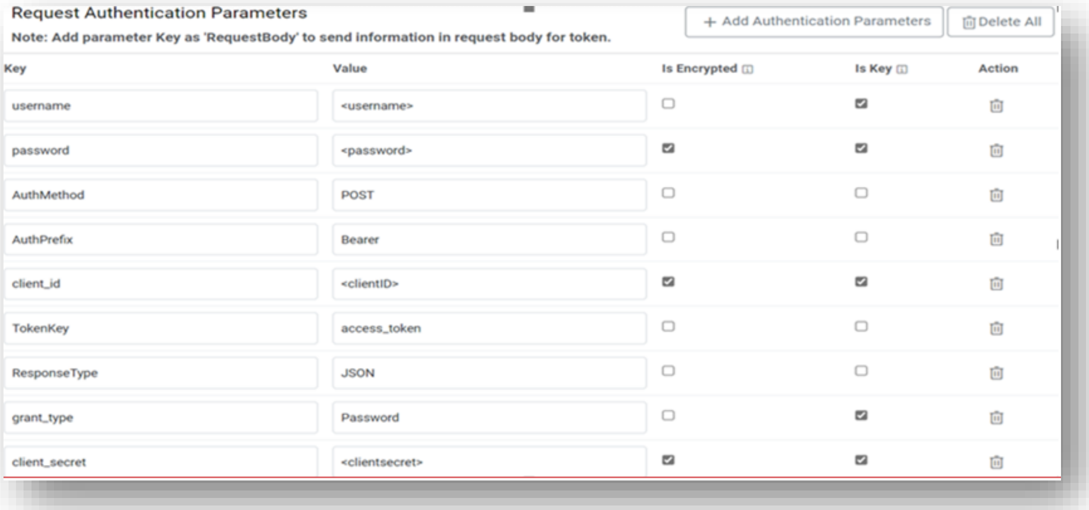
| Key | Value |
|---|---|
| Description | AzSNOWPassword |
| Secret Name | SNOWPassword |

Figure 331 - Password from Azure Key Vault

30. **Request Authentication Parameters -** If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
31. Based on the **Authentication Type,** add the parameters mentioned in the below table

Table 57– Sample Authentication Parameters

| Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|
| grant_type | password | N | Y |
| username | <username> | N | Y |
| Password | <password> | Y | Y |
| client_id | <client_id> | N | Y |
| AuthPrefix | Bearer | N | N |
| AuthMethod | POST | N | N |
| ResponseType | JSON | N | N |
| TokenKey | access_token | N | N |



Request Authentication Parameters

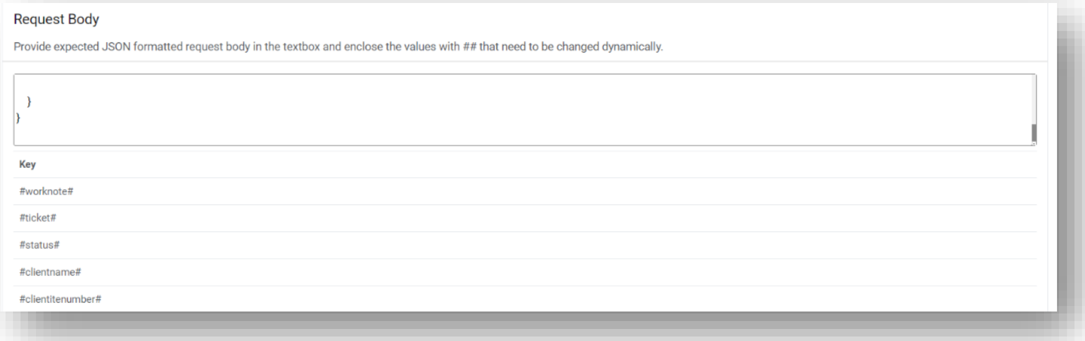
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | ☐ | ☑ | |
| password | <password> | ☑ | ☑ | |
| AuthMethod | POST | ☐ | ☐ | |
| AuthPrefix | Bearer | ☐ | ☐ | |
| client_id | <clientID> | ☑ | ☑ | |
| TokenKey | access_token | ☐ | ☐ | |
| ResponseType | JSON | ☐ | ☐ | |
| grant_type | Password | ☐ | ☑ | |
| client_secret | <clientsecret> | ☑ | ☑ | |

Figure 332 - Create Data Source (Request Authentication Parameters)

32. **Request Body -** In this section, please enter the request body in JSON format. A sample request is mentioned below:

```
Request Body –

{

        "ticketnumber": "#ticket#",

        "status": "#status#",

        "worknote": "#worknote#",

        "clientName": "#clientname#",

        "clientItemNumber": "#clientitemnumber#"

 }
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

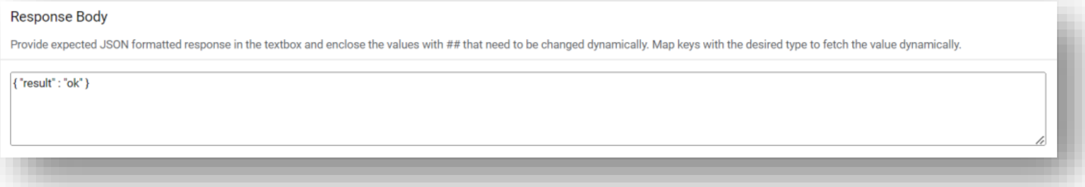
```
  }
}
```

| Key |
|---|
| #worknote# |
| #ticket# |
| #status# |
| #clientname# |
| #clientitenumber# |

Figure 333 - Response Body

33. **Response Body –** In this section, please enter the response body in JSON format. A sample request is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "ok" }
```

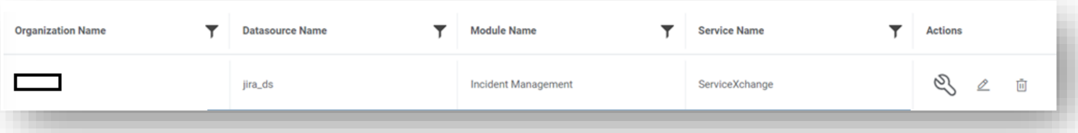
Figure 334 - InProgress Rules Configuration (Response Body)

34. **Response Key Value** mapping can be done as per the below table.

Table 58 - Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

35. Click **Save** to add the data source.
36. To bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps:
    - Go to Configuration and click Manage Data Sources.
    - On the **Data Sources** tab, click 🔧 next to the data source user wants to manage. **Manage Entry Criteria** screen appears.

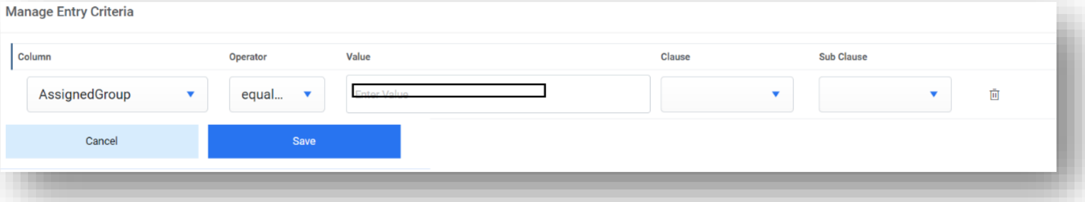


| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| | jira_ds | Incident Management | ServiceXchange | 🔧 ✎ 🗑 |

Figure 335 - Manage Entry Criteria

- Select ‘AssignedGroup’ for the **Column field and** ‘equals to’ for the **Operator** field.

- Enter the sys_id of the assignment group in SX in the **Value** field.
- **Clause** and **Sub-Clause** fields can also be added based on requirement.



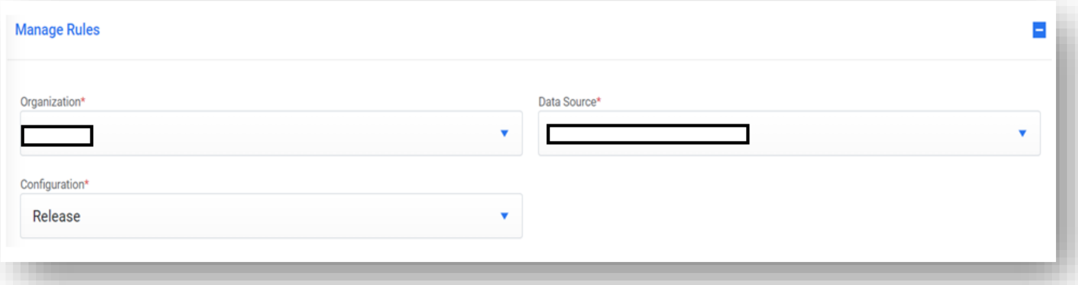
Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | [redacted] | | |

Figure 336 - Manage Entry Criteria (Cont.)

- Click **Save**.

37. To configure the rules for the data source created earlier, perform the below steps:
    - Go to Runbooks -> click Manage Rules.
    - Select the **Organization** and the data source created from **Data Source** dropdown.



Manage Rules

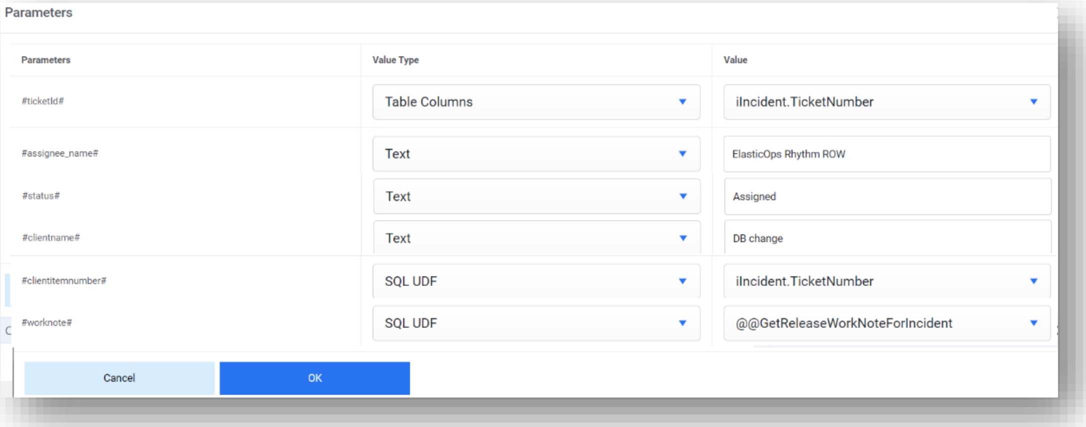
Organization*: [redacted]

Data Source*: [redacted]

Configuration*: Release

Figure 337 - Manage Rules

- Click on ⚙ corresponding to **–No Rule—**
- Map the parameter **#Assignmentgroup#** with **ElasticOps Rhythm ROW** as value and value Type is Text.
- Map the parameter **#ticket#** with **iIncident.TicketNumber** as value and value type is Table Columns.
- Map the parameter **#status#** with **Assigned** as value and text as Value Type.
- Map the parameter **#clientname#** with **DB Cheques** as value and text as Value Type.
- Map the parameter **#clientitemnumber#** with **iIncident.TicketNumber** as value and table column as Value Type.
- Map the parameter **#worknote#** with **@@GetReleaseWorkNoteForIncident** as Value and SQL UDF as Value Type.
- Click **OK**.



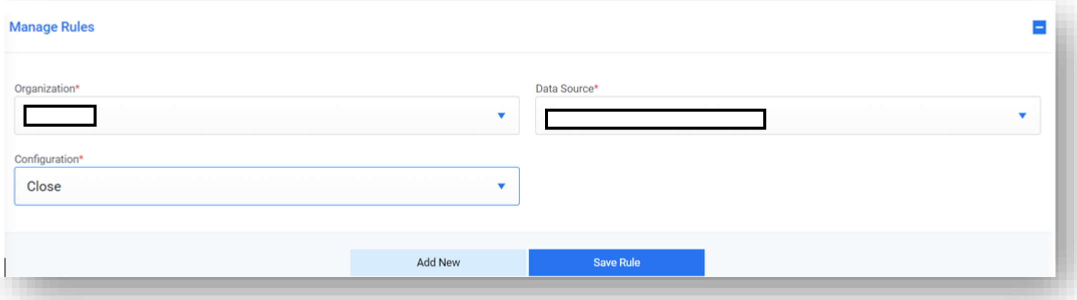
Parameters

| Parameters | Value Type | Value |
|---|---|---|
| #ticketId# | Table Columns | iIncident.TicketNumber |
| #assignee_name# | Text | ElasticOps Rhythm ROW |
| #status# | Text | Assigned |
| #clientname# | Text | DB change |
| #clientitemnumber# | SQL UDF | iIncident.TicketNumber |
| #worknote# | SQL UDF | @@GetReleaseWorkNoteForIncident |

Figure 338 - Manage Rules (Cont.)

- Click OK Rule.

38. To configure the **Close rules** for the data source created earlier, perform the below steps:
    - Go to select **Runbooks** -> click **Manage Rules**.
    - Select the **Organization** and the data source created from **Data Source** dropdown.



Manage Rules

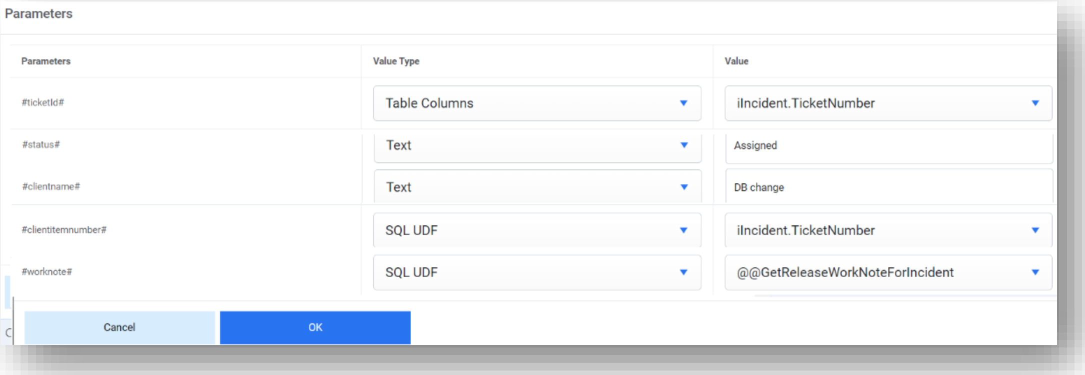
Organization*: [redacted]

Data Source*: [redacted]

Configuration*: Close

Figure 339 - Manage Rules

- Click on ⚙ corresponding to **–No Rule—**
- Map the parameter **#ticket#** with **iIncident.TicketNumber** as value and value type is Table Columns.
- Map the parameter **#status#** with **Fixed** as value and text as Value Type.
- Map **#worknote#** again to the value type as SQL UDF in which #worknote# was mapped with function **@@GetToolWorkNoteForIncident**.
- Map the parameter **#clientname#** with **DB Cheques** as value and text as Value Type.
- Map the parameter **#clientitemnumber#** with **iIncident.TicketNumber** as value and table column as Value Type



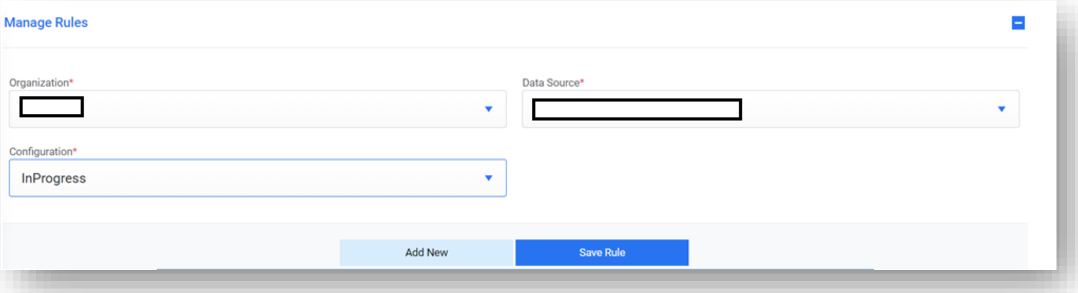
Parameters

| Parameters | Value Type | Value |
|---|---|---|
| #ticketId# | Table Columns | iIncident.TicketNumber |
| #status# | Text | Assigned |
| #clientname# | Text | DB change |
| #clientitemnumber# | SQL UDF | iIncident.TicketNumber |
| #worknote# | SQL UDF | @@GetReleaseWorkNoteForIncident |

Figure 340 - Manage Rules (Cont.)

- Click **OK**.
- Click Save Rule.

39. To configure the InProgress rules for the data source created earlier, perform the below steps:
    - Go to Runbooks -> click Manage Rules.
    - Select the **Organization** and the data source created from **Data Source** dropdown.



Manage Rules

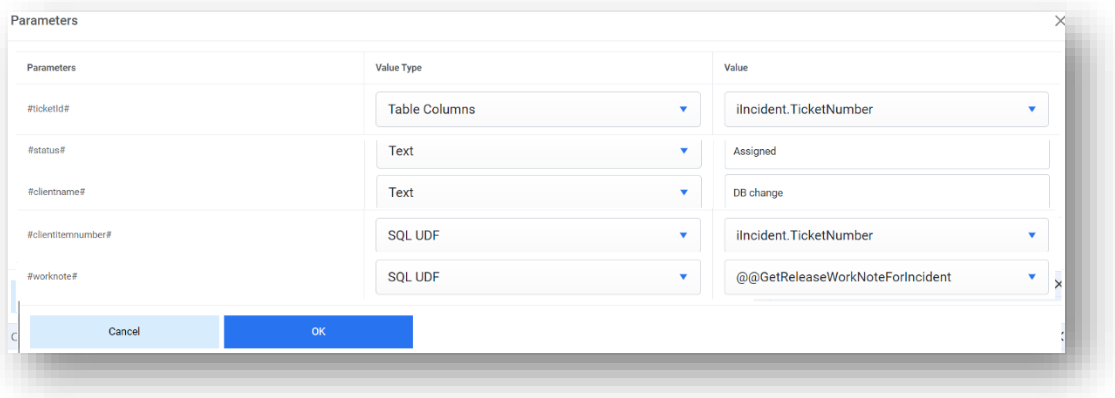
Organization*: [redacted]

Data Source*: [redacted]

Configuration*: InProgress

Figure 341 - Manage Release Rules

- Click on ⚙ corresponding to **–No Rule—**
- Map the parameter **#ticket#** with **iIncident.TicketNumber** as value and value type is Table Columns.
- Map the parameter **#status#** with **InProgress** as value and text as Value Type.
- Map the parameter **#worknote#** with **iAutomate is working on the ticket** as Value and text as Value Type.
- Map the parameter **#clientname#** with **DB Cheques** as value and text as Value Type.
- Map the parameter **#clientitemnumber#** with **iIncident.TicketNumber** as value and table column as Value Type.



| Parameters | Value Type | Value |
|---|---|---|
| #ticketId# | Table Columns | iIncident.TicketNumber |
| #status# | Text | Assigned |
| #clientname# | Text | DB change |
| #clientitemnumber# | SQL UDF | iIncident.TicketNumber |
| #worknote# | SQL UDF | @@GetReleaseWorkNoteForIncident |

Figure 342 - Manage Rules (Cont.)

- Click **OK**.
- Click Save Rule.

**Integration with Event Management Tools:**

Any Event Management tool acts as a data source for iAutomate from where it pulls the event or Probable Root Cause data and then performs appropriate actions for resolution. Thus, to enable integration with Event Management, it requires for a data source to be created as part of iAutomate configuration.

Before proceeding with the configuration related to Data Source creation, user has to ensure that an organization has been configured. If not done already, please refer to the Configuration Guide for the same and create the organization before proceeding ahead.

Please note that for integration with Event Management tool, while creating the organization, user needs to select the Event Management tool from the dropdown.

## 4.8 Integration with Moogsoft

### 4.8.1 Incident Management with ITSM (ServiceNow)

This scenario is applicable when the ITSM tools is available in the client environment and both event management & iAutomate is integrated with the ITSM, which acts as a system of record. The event data flows from event management tool to the ITSM leading to a ticket, based on the probable root cause. Upon ticket creation, iAutomate picks the ticket from the ITSM tool and performs the appropriate action for resolution.

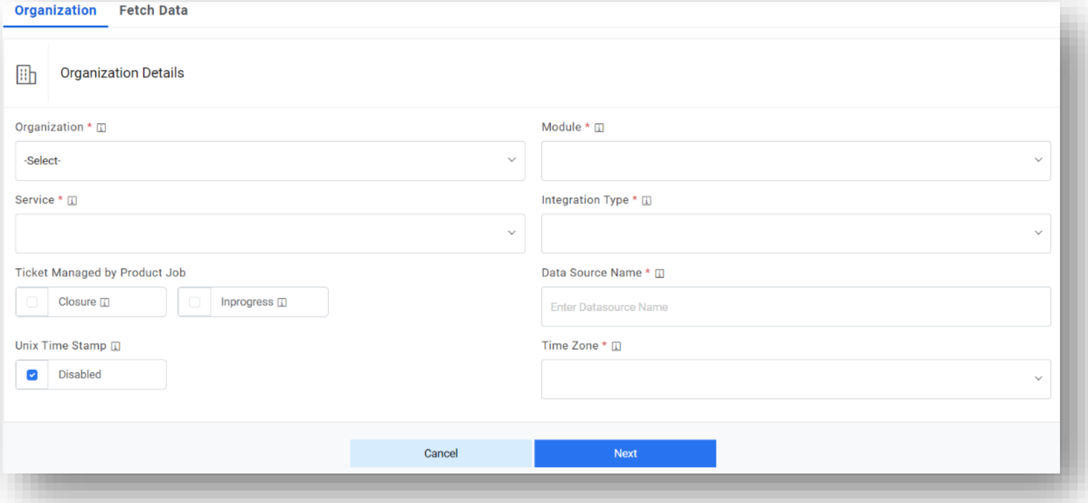
The user has the option to view the tickets and trigger the resolutions via Moogsoft as well as iAutomate console.

To create a data source, perform the following steps:

1. On the left menu bar, click **Configuration → Manage Data Sources**.
2. The **Create Data Source** page appears with the following tabs:
   - Organization
   - Fetch Data
3. Release Rules

– Close Rules (Optional – applicable only when the ticket closure status update is managed by iAutomate directly instead of RBA tool)

– InProgress Rules (Optional – applicable only when the ticket's in progress status updates is managed by iAutomate directly instead of RBA tool)

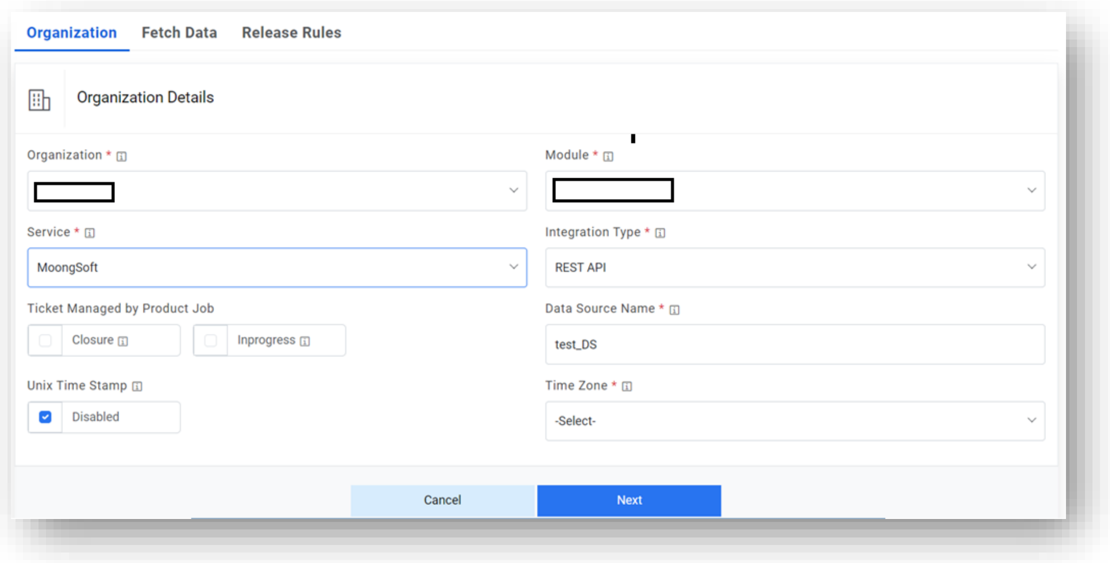


Unix Time Stamp: ☑ Disabled

Figure 343 – Create Data Source

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

4. On the **Organization** tab,
   - Select the **Organization Name** from the dropdown.
   - Select the **Module** as **Event Management,** since we are configuring this data source for pulling the event data.
   - Select the **Service** as **Moogsoft Tool** as we are configuring the data source for Moogsoft
   - Select the **Integration Type** as **REST**, since we will be integrating through REST APIs.
   - Check **Is ticket Closure Managed by iAutomate job** if you want iAutomate to manage the ticket closure updates instead of the RBA tool. In this scenario, an additional tab **Close Rules Configuration** will be activated for providing further details, steps for which are mentioned later.
   - Check "**Is ticket InProgress Managed by iAutomate job**" if you want iAutomate to manage the tickets in progress status updates instead of the RBA tool. In this scenario, an additional tab "**InProgress Rules Configuration**" will be activated for providing further details, steps for which are mentioned later.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select **Timestamp** to view the present data with date and time.
   - Click **Next**.



Organization Details

Organization *: [redacted]

Module *: [redacted]

Service *: MoongSoft

Integration Type *: REST API

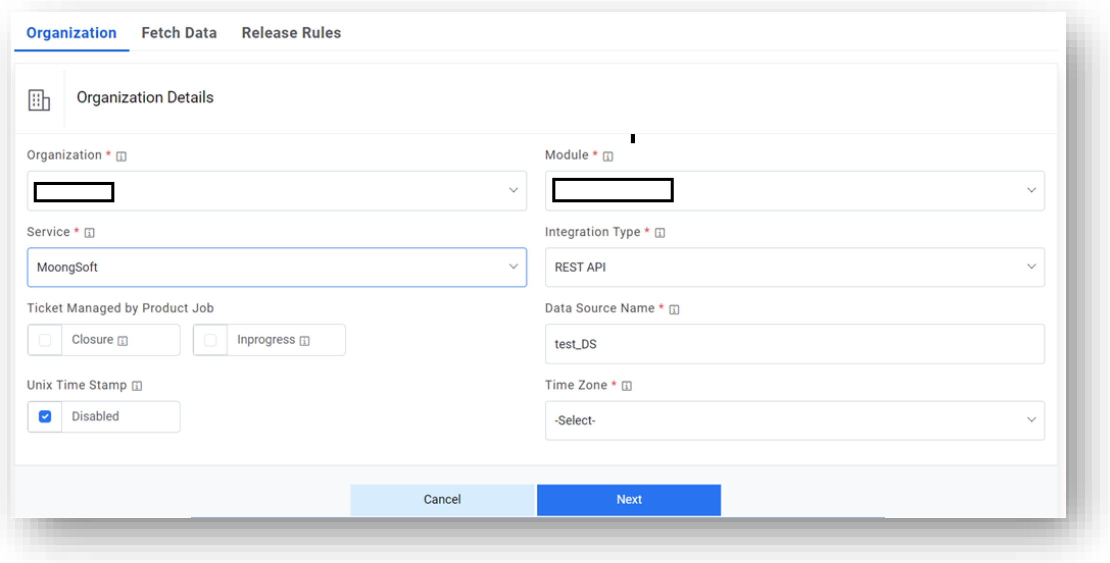
Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name *: test_DS

Unix Time Stamp: ☑ Disabled

Time Zone *: -Select-

Figure 344 – Create Data Source (Cont.)



Organization Details

Organization *: [redacted]

Module *: [redacted]

Service *: MoongSoft

Integration Type *: REST API

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name *: test_DS

Unix Time Stamp: ☑ Disabled

Time Zone *: -Select-

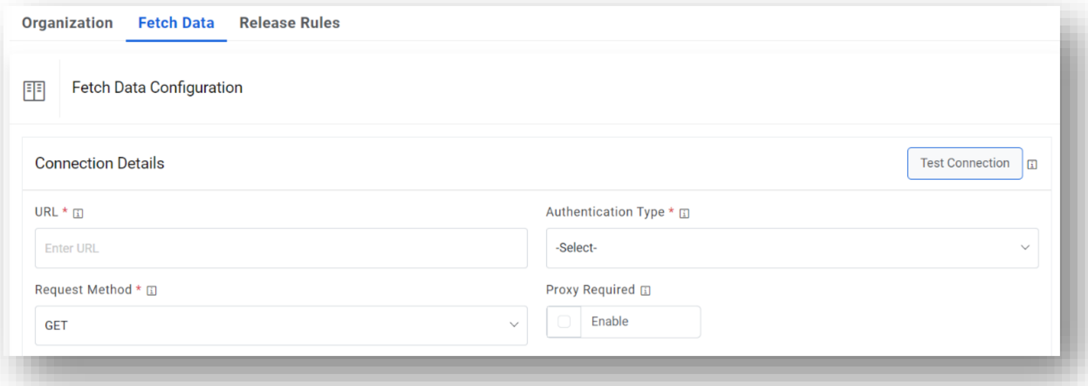
Figure 345 – Create Data Source (Cont.)

5. On the Fetch Data tab, type in the details as per the environment.
6. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL** – https://<url>?sysparm_fields=#Columns#&sysparm_query=sys_updated_on>=#StartDate#^sys_updated_on<=#EndDate#^ORDERBYsys_updated_on
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0
- Selection of **Basic / Windows** requires you to enter –
    - User Id

    - Password
- Selection of **OAuth 2.0** requires you to enter -
    - User Id
    - Password
    - Authentication URL
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Organization | Fetch Data | Release Rules

Fetch Data Configuration

Connection Details

Test Connection

URL * : Enter URL

Authentication Type * : -Select-

Request Method * : GET

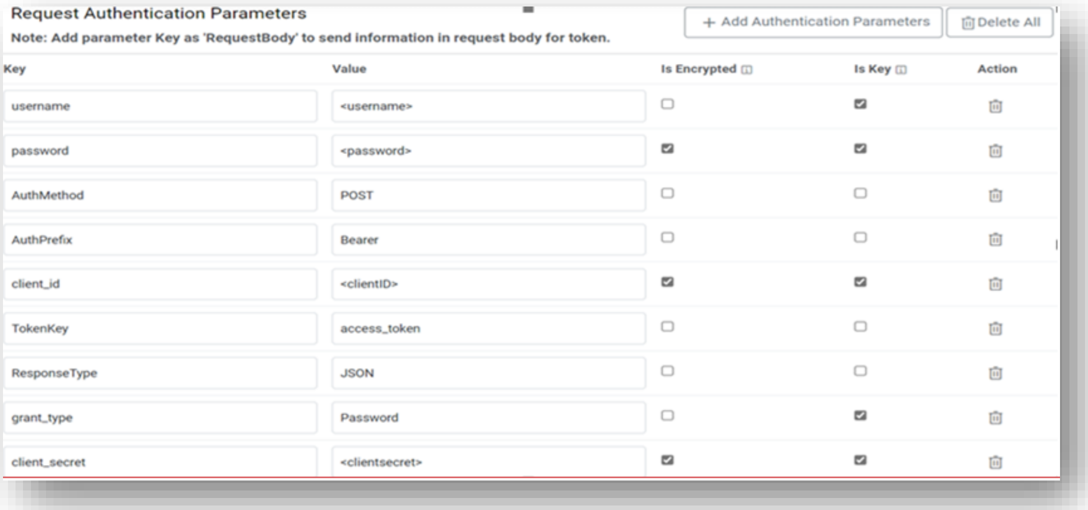
Proxy Required : □ Enable

Figure 346 – Create Data Source (Connection Details)

7. Request Authentication Parameters – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
8. Based on the Authentication Type, add the parameters mentioned in the below table.

Table 59– Sample Authentication Parameters

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | Username | <username> | NO | YES |
| OAuth2.0 | Password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

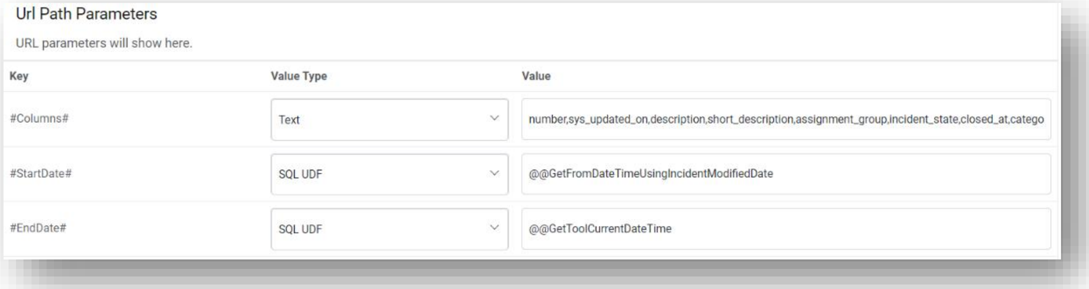
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | ☐ | ☑ | 🗑 |
| password | <password> | ☑ | ☑ | 🗑 |
| AuthMethod | POST | ☐ | ☐ | 🗑 |
| AuthPrefix | Bearer | ☐ | ☐ | 🗑 |
| client_id | <clientID> | ☑ | ☑ | 🗑 |
| TokenKey | access_token | ☐ | ☐ | 🗑 |
| ResponseType | JSON | ☐ | ☐ | 🗑 |
| grant_type | Password | ☐ | ☑ | 🗑 |
| client_secret | <clientsecret> | ☑ | ☑ | 🗑 |

Figure 347 – Create Data Source (Request Authentication Parameters for OAuth2.0)

9. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #Columns#

ValueType: Text

Value:

number,sys_updated_on,short_description,description,assignment_
group,incident_state,closed_at,category,dv_assigned_to,sys_id


Note – These columns are mandatory. User can add more columns if
more data is required to be fetched from ITSM tool.


Key: #StartDate#

ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingIncidentModifiedDate


Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #Columns# | Text | number,sys_updated_on,description,short_description,assignment_group,incident_state,closed_at,catego |
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingIncidentModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 348– URL Path Parameters

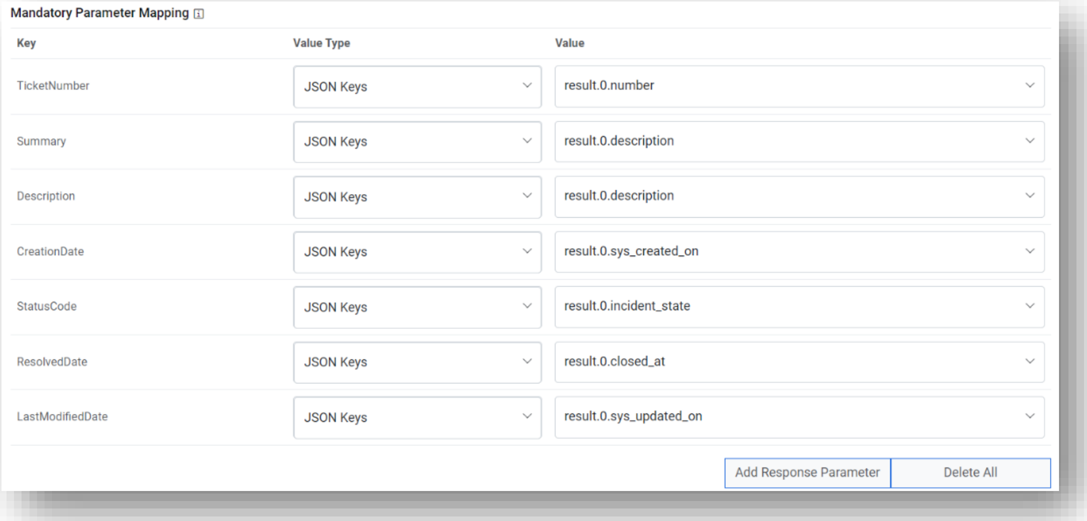
10. **Request Header Parameters** – Please enter the request header parameters as required.
11. **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below.

```
Response Body –

{   "result":  [{    "number":  "INC0079154",    "closed_at":  "",
"assignment_group":  {    "link":  "<https://my_host>",    "value":
"All user group"   },  "incident_state":  "6",   "sys_created_on":
"2017-12-22    06:59:03",           "description":    "Memory
Utilization:10.0.0.11",      "short_description":       "Memory
Utilization:localhost",        "sys_updated_on":    "2018-01-02
06:39:56",    "category":  "",    "priority":  "4",    "sys_id":
"xxxxxxxx"   }] }
```

12. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
13. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below.

Table 60– Sample Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.number |
| Summary | JSON.Keys | result.0.short_description |
| Description | JSON.Keys | result.0.description |
| CreationDate | JSON.Keys | result.0.sys_created_on |
| StatusCode | JSON.Keys | result.0.incident_state |
| ResolvedDate | JSON.Keys | result.0.closed_at |
| LastModifiedDate | JSON.Keys | result.0.sys_updated_on |



Mandatory Parameter Mapping

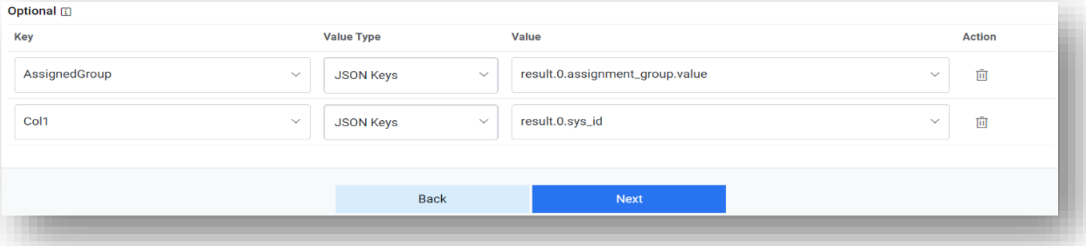
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.number |
| Summary | JSON Keys | result.0.description |
| Description | JSON Keys | result.0.description |
| CreationDate | JSON Keys | result.0.sys_created_on |
| StatusCode | JSON Keys | result.0.incident_state |
| ResolvedDate | JSON Keys | result.0.closed_at |
| LastModifiedDate | JSON Keys | result.0.sys_updated_on |

Figure 349 – Mandatory Parameter Mapping

14. If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 61– Sample Optional Parameters

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | result.0.assignment_group.value |
| Col1 | JSON.Keys | result.0.sys_id |



Optional

| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | result.0.assignment_group.value | 🗑 |
| Col1 | JSON Keys | result.0.sys_id | 🗑 |

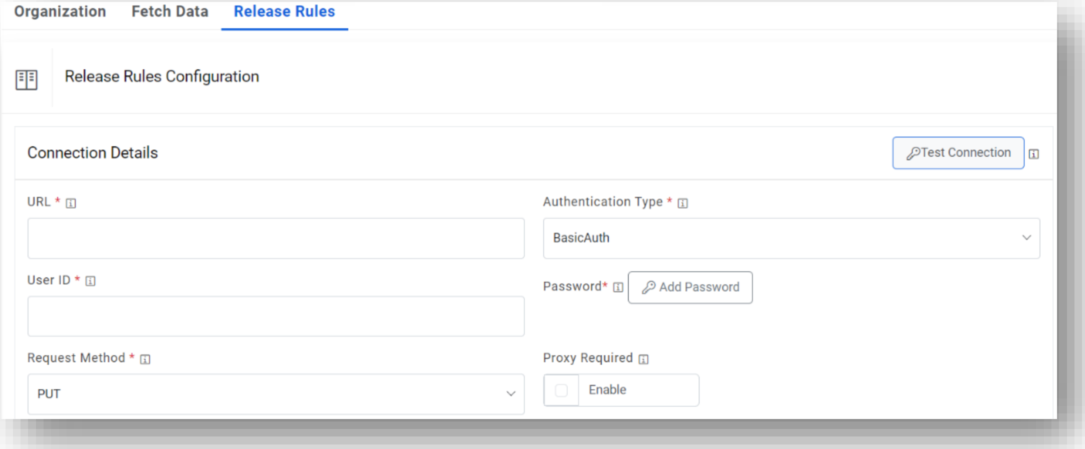
Figure 350 – Optional Parameter Mapping

15. Click Next to proceed to Release Rules.
16. On Release Rules tab, type in the details as per the requirement.
17. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>.service-now.com/api/now/table/incident/#incident#
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- **Request Method** – Select Request Method as PUT from the drop-down.

- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Organization | Fetch Data | Release Rules

Release Rules Configuration

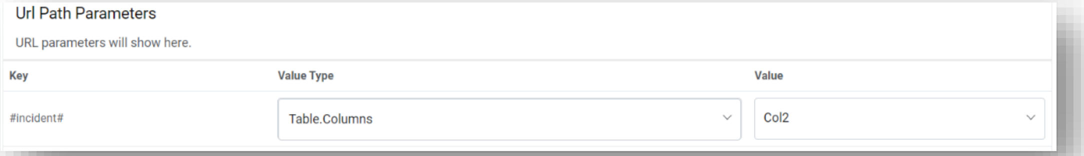
Connection Details

| | |
|---|---|
| URL * | Authentication Type *: BasicAuth |
| User ID * | Password* |
| Request Method *: PUT | Proxy Required: □ Enable |

Figure 351 – Release Rules (Connection Details)

18. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs.

```
Key: #incident#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
“Col2”
```



Url Path Parameters

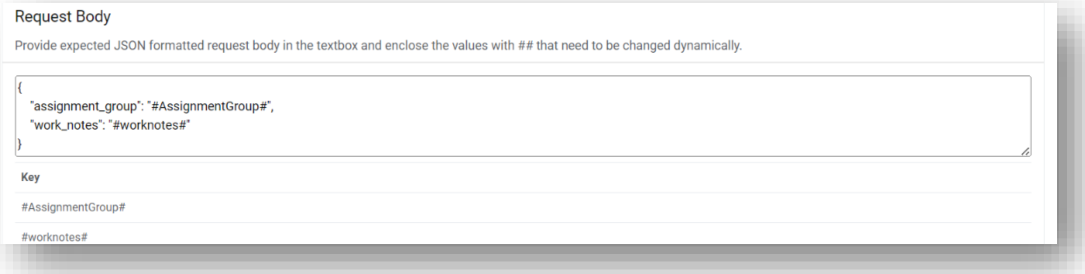
URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #incident# | Table.Columns | Col2 |

Figure 352 – Release Rules (URL Path Parameters)

19. **Request Header Parameters** – Please enter the request header parameters as required.
20. **Request Body** – In this section, please enter the request body in JSON format. A sample request is mentioned below.

```
Request Body –

{ "assignment_group" : "#AssignmentGroup#","work_notes" :
"#work_notes#" }
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

```
{
  "assignment_group": "#AssignmentGroup#",
  "work_notes": "#worknotes#"
}
```

Key

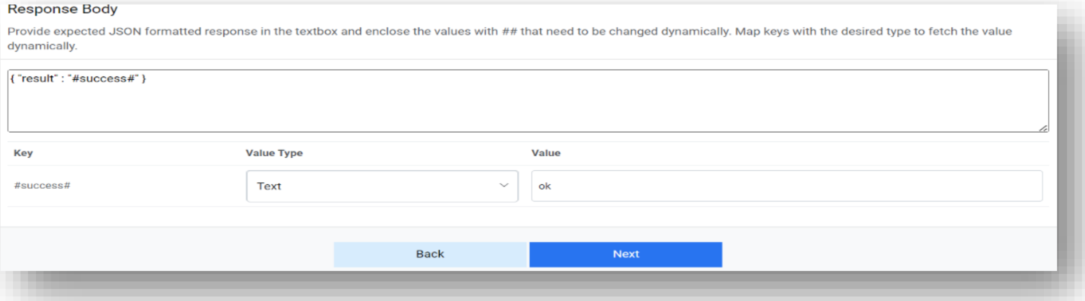
#AssignmentGroup#

#worknotes#

Figure 353 – Release Rules (Request Body)

21. **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below-

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

Figure 354 – Release Rules (Response Body)

22. **Response Key Value** mapping can be done as per the below table-

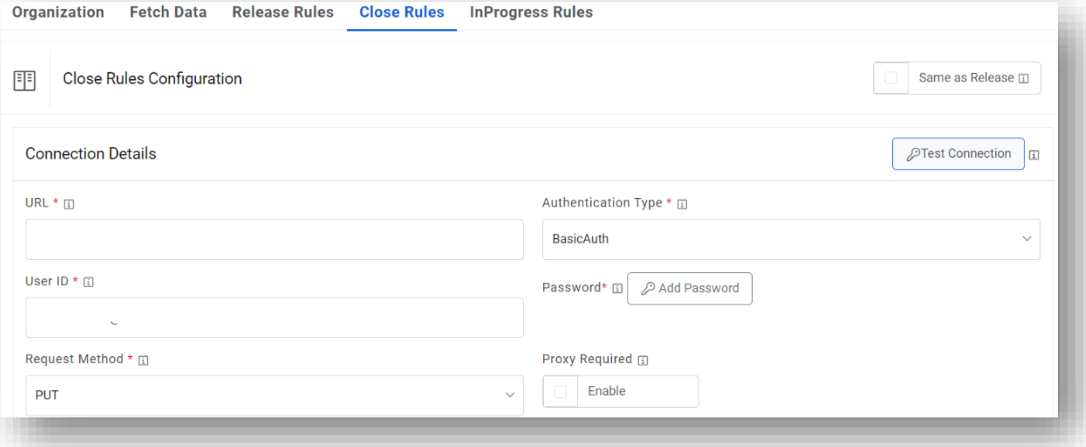
Table 62– Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

23. On the Close Rules tab, type in the details as per the requirement. Check Same as Release if similar configurations as mentioned in "Release Rules Configuration" are required, else proceed ahead
24. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and are dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>.service-now.com/api/now/table/incident/#incident#
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- Request Method – Select Request Method as PUT from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.


Organization | Fetch Data | Release Rules | Close Rules | InProgress Rules

Close Rules Configuration

□ Same as Release

Connection Details

Test Connection

URL *

Authentication Type *: BasicAuth

User ID *

Password* Add Password

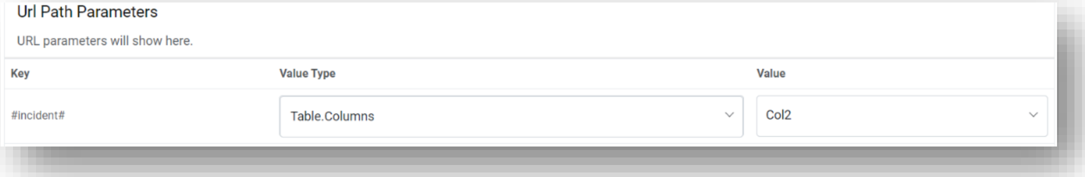
Request Method *: PUT

Proxy Required: □ Enable

Figure 355 – Close Rules Configuration (Connection Details)

25. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #incident#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
“Col2”
```


Url Path Parameters

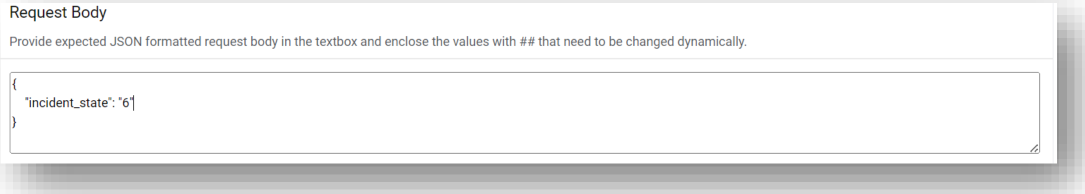
URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #incident# | Table.Columns | Col2 |

Figure 356 – Close Rules (URL Path Parameters)

26. Request Header Parameters – Please enter the request header parameters as required.
27. **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below -

```
Request Body -

{ "incident_state" : "6"} If you also want to add worknotes while
Close ticket, use json {"incident_state":"6", "work_notes":
"#Notes#"}
```



Request Body

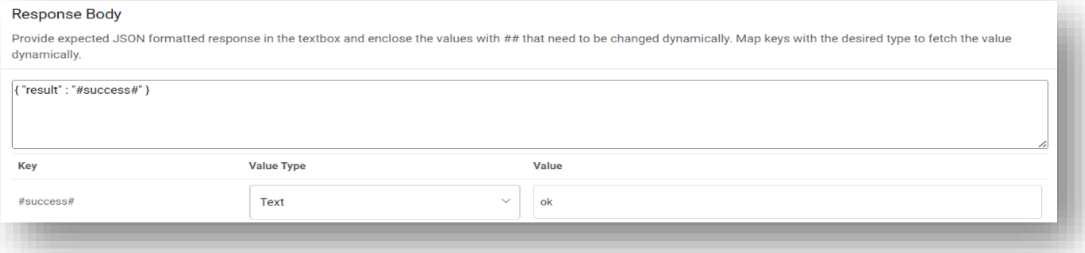
Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

```
{
   "incident_state": "6"
}
```

Figure 357 – Close Rules Configuration (Request Body)

28. **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below:

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

Figure 358 – Close Rules Configuration (Response Body)

29. **Response Key Value** mapping can be done as per the below table -

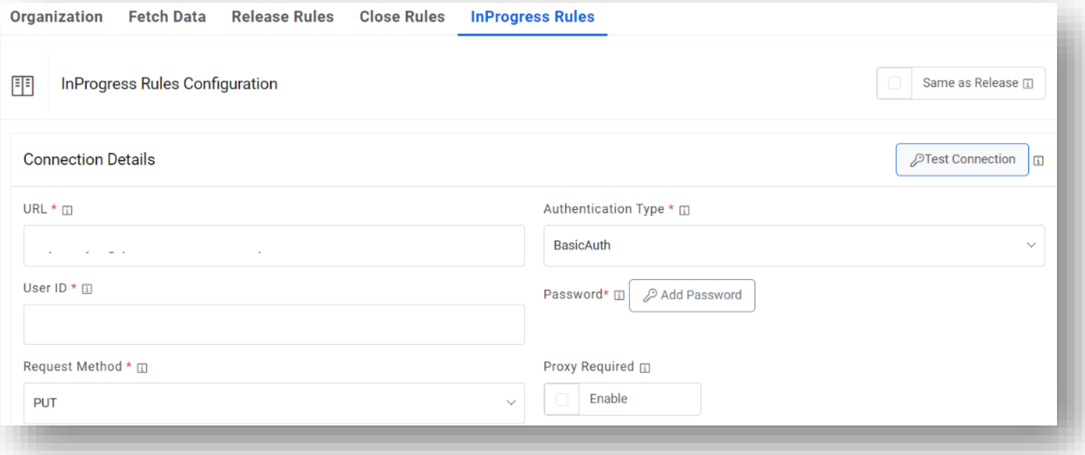
Table 63 – Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

30. On the InProgress Rules tab, type in the details as per the requirement. Check Same as Release if similar configurations as mentioned in "Release Rules Configuration" are required, else proceed ahead.
31. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and are dependent on the URL or API provided by the tool.
- **Sample URL** - https://<url>.service-now.com/api/now/table/incident/#incident#
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- Request Method – Select Request Method as PUT from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.


Organization | Fetch Data | Release Rules | Close Rules | InProgress Rules

InProgress Rules Configuration

□ Same as Release

Connection Details

Test Connection

URL *

Authentication Type *: BasicAuth

User ID *

Password* Add Password

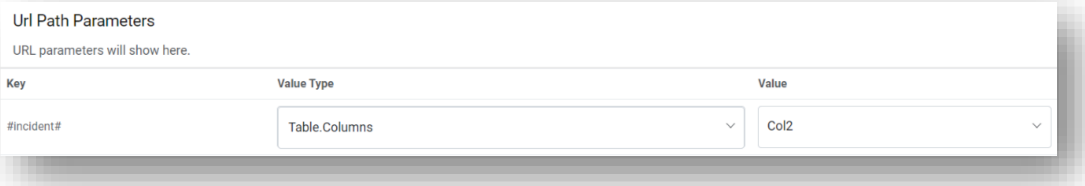
Request Method *: PUT

Proxy Required: □ Enable

Figure 359 – InProgress Rules Configuration (Connection Details)

32. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs.

```
Key: #incident#

ValueType: Table Columns

Value:

Select from dropdown that mapped to sys_id from previous screen
“Col2”
```


Url Path Parameters

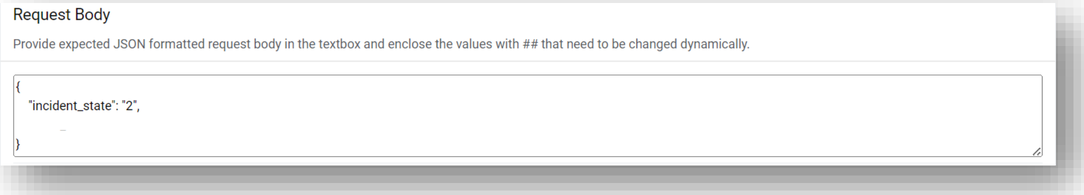
URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #incident# | Table.Columns | Col2 |

Figure 360 – InProgress Rules Configuration (URL Path Parameters)

33. **Request Header Parameters –** Please enter the request header parameters as required.
34. **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below -

```
Request Body –

{"incident_state" : "2"} If you also want to add worknotes while
inprogress ticket, use json {"incident_state":"2", "work_notes":
"#Notes#"}
```



Request Body

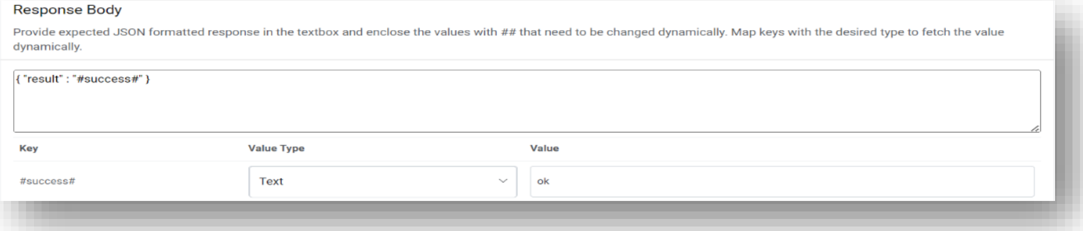
Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

```
{
  "incident_state": "2",
}
```

Figure 361 – InProgress Rules Configuration (Request Body)

35. **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below -

```
Response Body –

{ "result" : "#success#" }
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

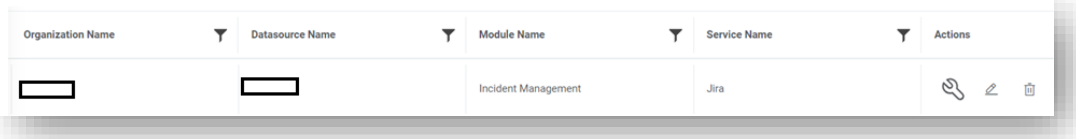
Figure 362 – InProgress Rules Configuration (Response Body)

36. **Response Key Value** mapping can be done as per the below table -

Table 64– Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

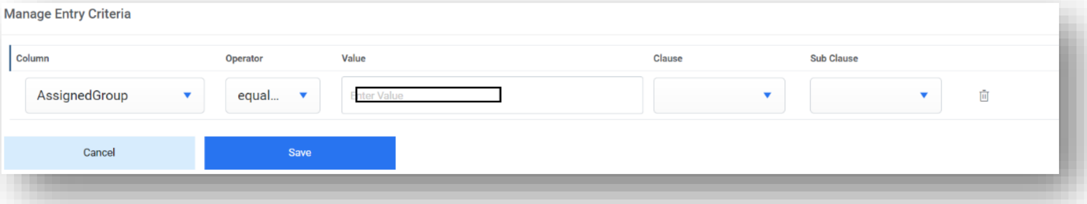
37. Click **Save** to add the data source.
38. To bring the tickets within iAutomate scope, a specific queue needs to be configured in the ITSM tool and same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps:
    - Go to Configuration and click Manage Data Sources.
    - On the **Data Sources** tab, click 🔧 next to the data source user wants to manage.
    - Manage Entry Criteria screen appears.



| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| | | Incident Management | Jira | |

Figure 363 – Manage Entry Criteria

- Select 'AssignedGroup' for the **Column** field and 'equals to' for the **Operator**.
- Enter the sys_id of the assignment group in ServiceNow in the **Value** field.
- **Clause** and **Sub-Clause** fields can also be added based on requirement.



Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | Enter Value | | |

Figure 364 – Manage Entry Criteria (Cont.)

39. Click **Save**.

### 4.8.2 Incident Management without ITSM (ServiceNow)

This scenario is applicable when the ITSM tools is not available in the client environment and event management tool and iAutomate are tightly integrated directly. The event data or the probable root cause identified flows to iAutomate which then performs the appropriate action for resolution.

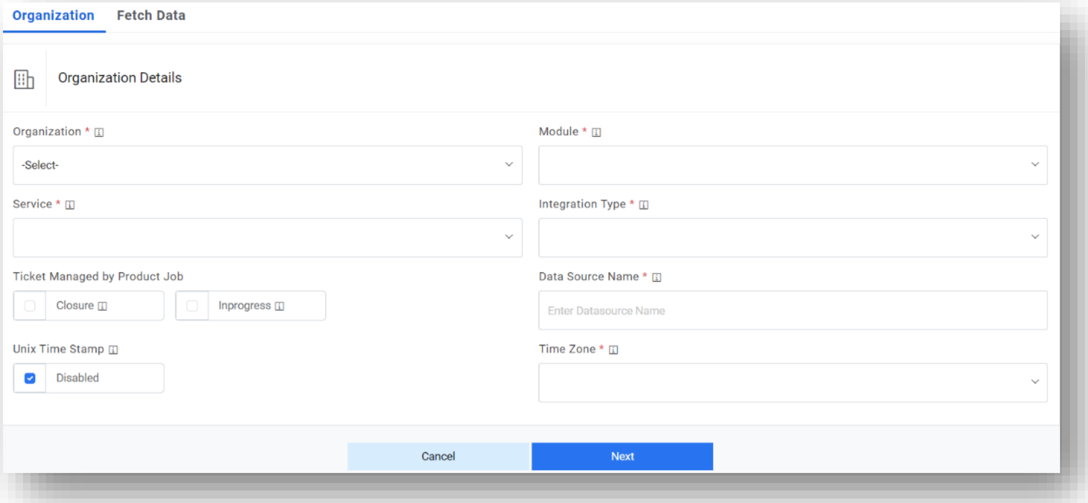
The user has the option to view the events and trigger the resolutions via Moogsoft as well as iAutomate console.

To create a data source, perform the following steps:

1. On the left menu bar, click **Configuration** -> **Manage Data Source**.
2. The **Create Data Source** page appears with the following tabs:
   - Organization
   - Fetch Data
3. **Release Rules**

– **Close Rules** (Optional – applicable only when the issue closure status update is managed by iAutomate directly instead of RBA tool)

– **InProgress Rules** (Optional – applicable only when the issue's in progress status updates is managed by iAutomate directly instead of RBA tool)



Organization Details

Organization * : -Select-

Module * :

Service * :

Integration Type * :

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name * : Enter Datasource Name

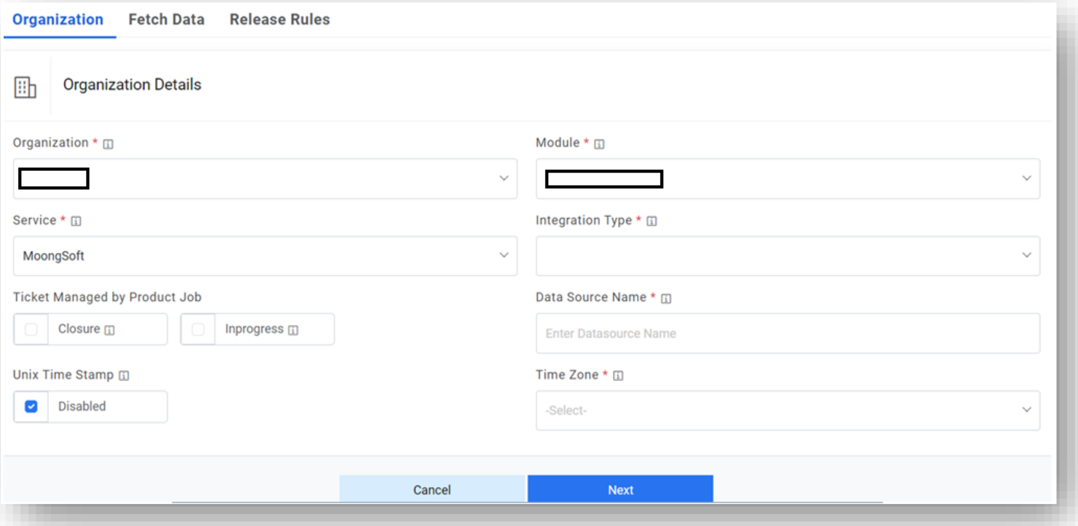
Unix Time Stamp: ☑ Disabled

Time Zone * :

Figure 365 – Create Data Source

Release Rules are only applicable for the following **Module** types- **Incident Management, Change Request Task and Service Request Task.** This tab will not be activated for other module types.

4. On the **Organization** tab,
   - Select the **Organization Name** from the dropdown.
   - Select the **Module** as **Event Management,** since we are configuring this data source for pulling the event data.
   - Select the **Service** as **Moogsoft Tool** as we are configuring the data source for Moogsoft
   - Select the **Integration Type** as **REST**, since we will be integrating through REST APIs.
   - Check **Is Ticket Closure Managed by iAutomate job** if you want iAutomate to manage the ticket closure updates instead of the RBA tool. In this scenario, an additional tab **Close Rules Configuration** will be activated for providing further details, steps for which are mentioned later.
   - Check "**Is ticket InProgress Managed by iAutomate job**" if you want iAutomate to manage the ticket's in progress status updates instead of the RBA tool. In this scenario, an additional tab "**InProgress Rules Configuration**" will be activated for providing further details, steps for which are mentioned later.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select **Timestamp** to view the present data with date and time.
   - Click **Next**.



Organization Details

Organization * [redacted]

Module * [redacted]

Service * MoongSoft

Integration Type *

Ticket Managed by Product Job

□ Closure □ Inprogress

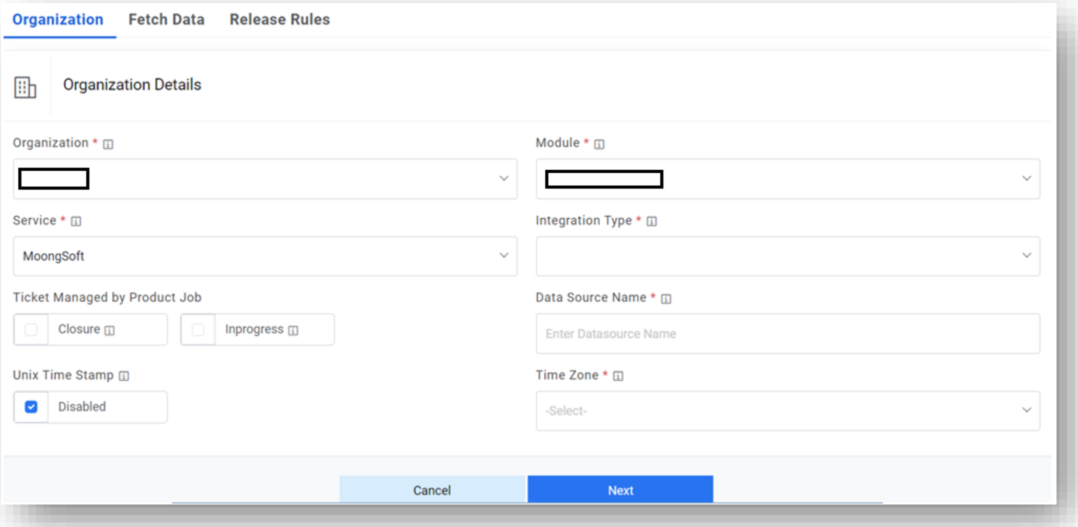
Data Source Name * Enter Datasource Name

Unix Time Stamp

☑ Disabled

Time Zone * -Select-

Figure 366 – Create Data Source (Cont.)



Organization Details

Organization * [redacted]

Module * [redacted]

Service * MoongSoft

Integration Type *

Ticket Managed by Product Job: □ Closure □ Inprogress

Data Source Name * Enter Datasource Name

Unix Time Stamp: ☑ Disabled

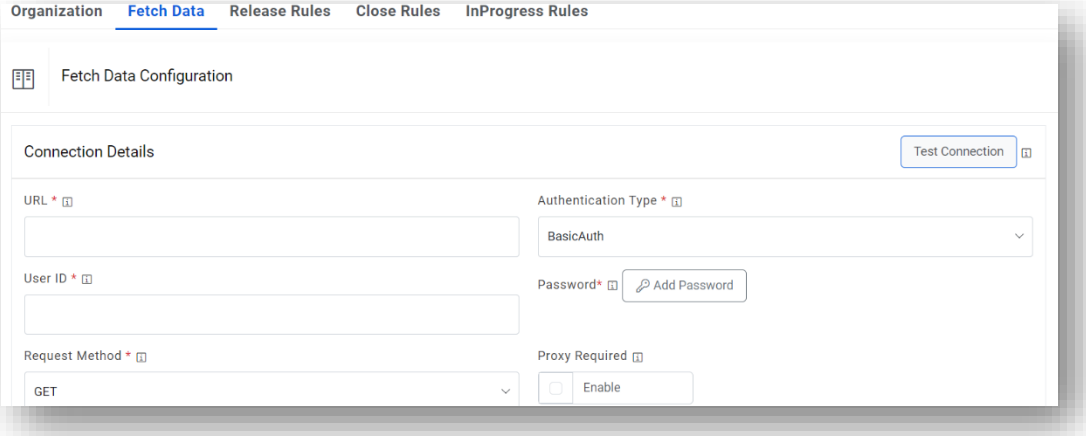
Time Zone * -Select-

Figure 367 – Create Data Source (Cont.)

5. On the **Fetch Data** tab, type in the details as per the environment.
6. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL** - http://<IP>:<PORT>/iAutomateAPI/Request/GetIncidentTicketData/11?start_date>=#startdate#&end_date<=#enddate#
- **Authentication Type** – Select one of the Authentication Types from Basic / Windows, OAuth 2.0
- Selection of **Basic / Windows** requires you to enter -
    - User Id
    - Password
- Selection of **OAuth 2.0** requires you to enter -
    - User Id
    - Password
    - Authentication URL
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Organization | Fetch Data | Release Rules | Close Rules | InProgress Rules

Fetch Data Configuration

Connection Details

Test Connection

URL *

Authentication Type *

BasicAuth

User ID *

Password* Add Password

Request Method *

GET

Proxy Required

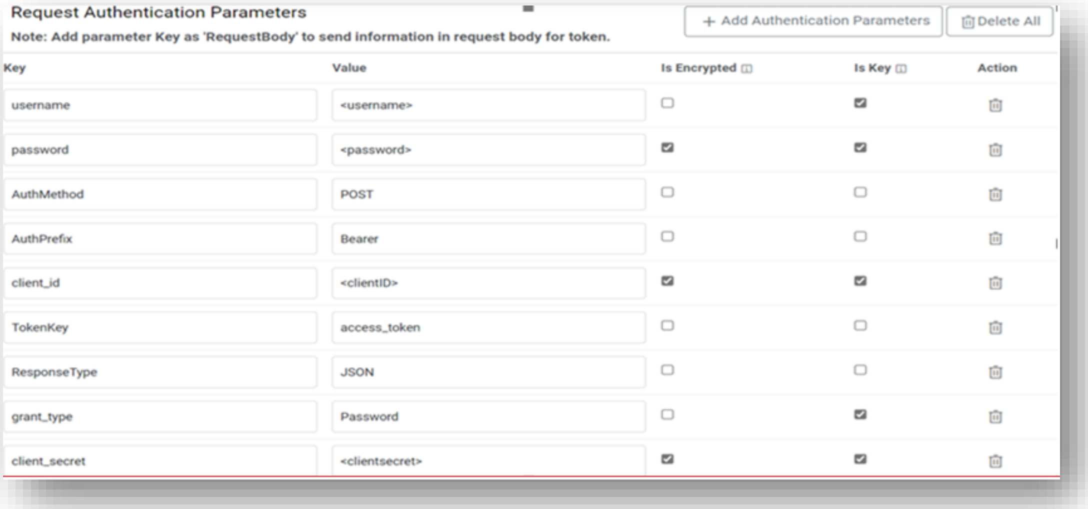
□ Enable

Figure 368 – Create Data Source (Connection Details)

7. **Request Authentication Parameters** – If the user has additional parameters, click Add Authentication Parameters under the Request Authentication Parameters tab.
8. Based on the **Authentication Type**, add the parameters mentioned in the below table.

Table 65– Sample Authentication Parameters

| Authentication Type | Key | Value | Is Encrypted? | Is Key? |
|---|---|---|---|---|
| OAuth2.0 | Username | <username> | NO | YES |
| OAuth2.0 | Password | <password> | YES | YES |
| OAuth2.0 | AuthMethod | POST | NO | NO |
| OAuth2.0 | AuthPrefix | Bearer | NO | NO |
| OAuth2.0 | client_id | <clientID> | YES | YES |
| OAuth2.0 | client_secret | <clientsecret> | YES | YES |
| OAuth2.0 | TokenKey | access_token | NO | NO |
| OAuth2.0 | ResponseType | JSON | NO | NO |
| OAuth2.0 | grant_type | Password | NO | YES |



Request Authentication Parameters

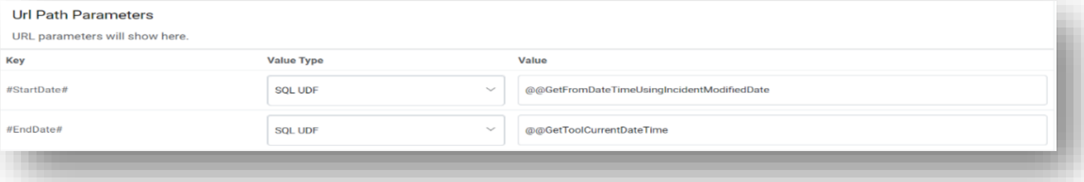
Note: Add parameter Key as 'RequestBody' to send information in request body for token.

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| username | <username> | □ | ☑ | |
| password | <password> | ☑ | ☑ | |
| AuthMethod | POST | □ | □ | |
| AuthPrefix | Bearer | □ | □ | |
| client_id | <clientID> | ☑ | ☑ | |
| TokenKey | access_token | □ | □ | |
| ResponseType | JSON | □ | □ | |
| grant_type | Password | □ | ☑ | |
| client_secret | <clientsecret> | ☑ | ☑ | |

Figure 369 – Create Data Source (Request Authentication Parameters for OAuth2.0)

9. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #StartDate#

ValueType: SQL UDF

VALUE: @@GetFromDateTimeUsingIncidentModifiedDate


Key: #EndDate#

ValueType: SQL UDF

VALUE: @@GetToolCurrentDateTime
```



Url Path Parameters

URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #StartDate# | SQL UDF | @@GetFromDateTimeUsingIncidentModifiedDate |
| #EndDate# | SQL UDF | @@GetToolCurrentDateTime |

Figure 370– URL Path Parameters

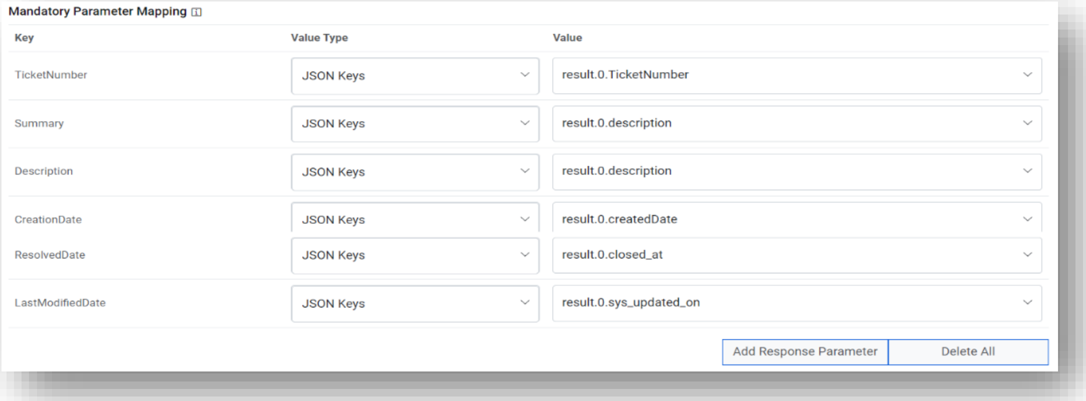
10. **Request Header Parameters –** Please enter the request header parameters as required.
11. **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below.

```
Response Body –

{ "result": [ { "TicketNumber": "xxxxxxxx", "Summary": "Restart
Spooler service on target server ", "Description": "Restart
Spooler service on target server", "AssignedGroup": "xxxxxxxx",
"StatusCode":  "1",  "CreationDate":  "2020-05-04  10:40:30.000",
"LastModifiedDate":  "2020-05-04  04:41:50.000",  "ClosedDate":
"2020-05-06  10:41:53.000",  "sys_id":  "xxxxxxxx",  "Col1":  "",
"Col2":   "",   "Col3":   "",   "Col4":   "",   "Col5":   "",
"iAutomate_CreatedDateInGMT":    "2020-05-04    05:25:36.350",
"iAutomate_UpdatedDateInGMT": "2020-05-04 05:25:36.350" } ] }
```

12. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
13. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below.

Table 66– Sample Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.TicketNumber |
| Summary | JSON.Keys | result.0.Summary |
| Description | JSON.Keys | result.0.Description |
| CreationDate | JSON.Keys | result.0.CreationDate |
| StatusCode | JSON.Keys | result.0.StatusCode |
| ResolvedDate | JSON.Keys | result.0.ClosedDate |
| LastModifiedDate | JSON.Keys | result.0.LastModifiedDate |



Mandatory Parameter Mapping

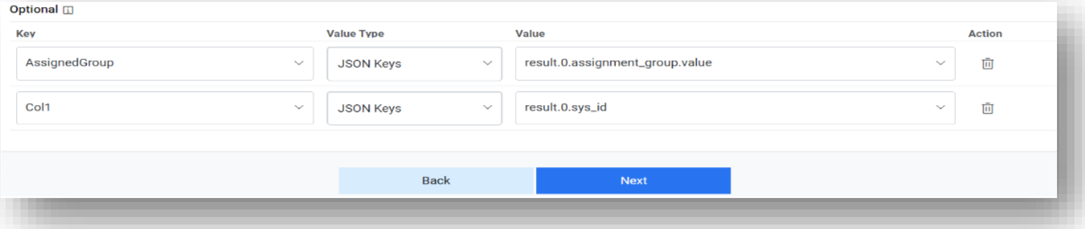
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.0.TicketNumber |
| Summary | JSON Keys | result.0.description |
| Description | JSON Keys | result.0.description |
| CreationDate | JSON Keys | result.0.createdDate |
| ResolvedDate | JSON Keys | result.0.closed_at |
| LastModifiedDate | JSON Keys | result.0.sys_updated_on |

Figure 371 – Mandatory Parameter Mapping

14. If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 67– Sample Optional Parameters

| Key | Value Type | Value |
|---|---|---|
| AssignedGroup | JSON.Keys | result.0. AssignedGroup |
| Col1 | JSON.Keys | result.0.sys_id |



| Key | Value Type | Value | Action |
|---|---|---|---|
| AssignedGroup | JSON Keys | result.0.assignment_group.value | |
| Col1 | JSON Keys | result.0.sys_id | |

Figure 372 – Optional Parameter Mapping

15. Click Next to proceed to Release Rules.
16. On **Release Rules** tab, type in the details as per the requirement.
17. In the **Connection Details** section, enter the following details:

– **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.

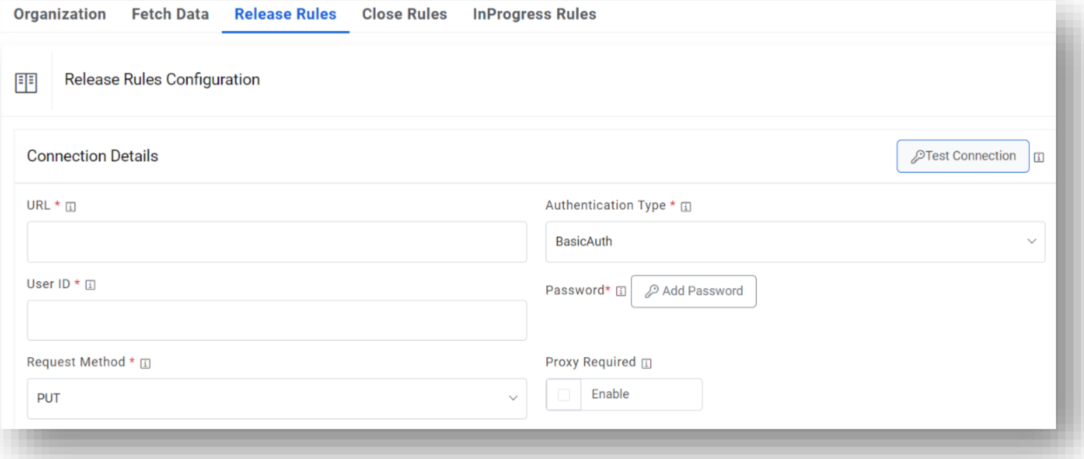
– **Sample URL** - https://<URL>/graze/v1/#value#

– **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.

– **Request Method** – Select Request Method as PUT from the drop-down.

– **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Release Rules Configuration

Connection Details

URL *

Authentication Type * BasicAuth

User ID *

Password*

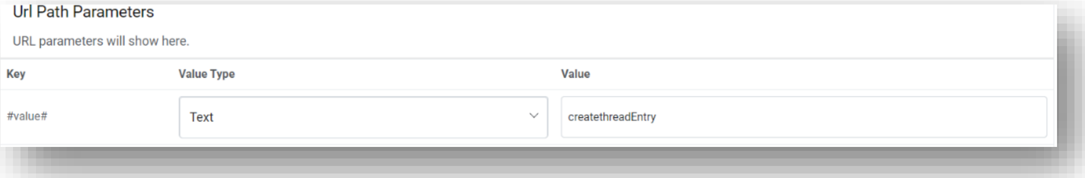
Request Method * PUT

Proxy Required □ Enable

Figure 373 – Release Rules Configuration (Connection Details)

18. **URL Path Parameters –** Based on the URL entered earlier, please map the values to the URL Path Parameters. E.g., for the URL entered earlier, please populate the below inputs:

```
Key: #value#

ValueType: Text

Value: createThreadEntry
```



Url Path Parameters

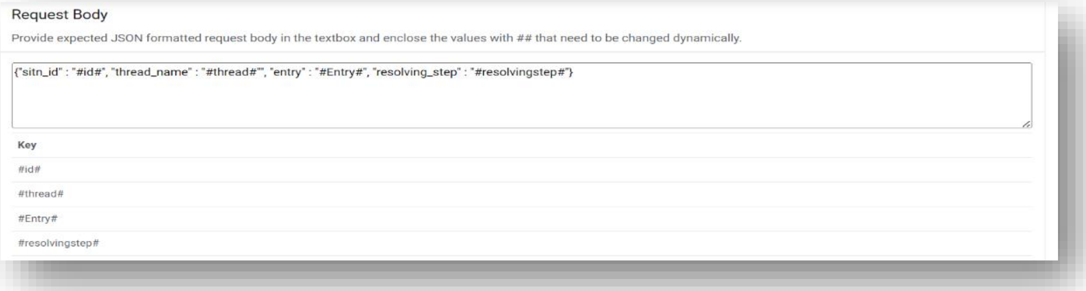
URL parameters will show here.

| Key | Value Type | Value |
|---|---|---|
| #value# | Text | createthreadEntry |

Figure 374 – Release Rules Configuration (URL Path Parameters)

19. **Request Header Parameters –** Please enter the request header parameters as required.
20. **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below.

```
Request Body –

{"sitn_id" : "#id#", "thread_name" : "#thread#"", "entry" :
"#Entry#", "resolving_step" : "#resolvingstep#"}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

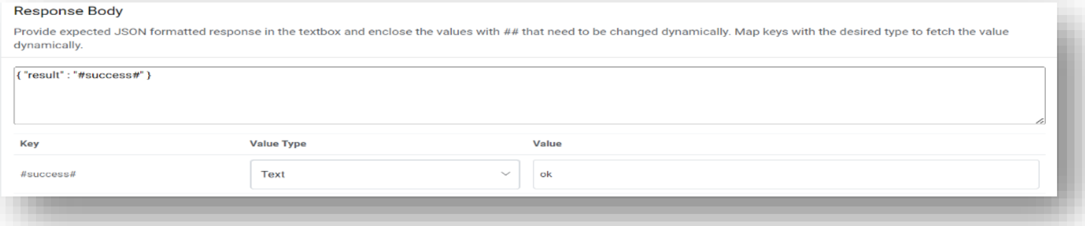
```
{"sitn_id" : "#id#", "thread_name" : "#thread#"", "entry" : "#Entry#", "resolving_step" : "#resolvingstep#"}
```

| Key |
|---|
| #id# |
| #thread# |
| #Entry# |
| #resolvingstep# |

Figure 375 – Release Rules Configuration (Request Body)

21. **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below -

```
Response Body –

{"result":"#success#"}
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
{ "result" : "#success#" }
```

| Key | Value Type | Value |
|---|---|---|
| #success# | Text | ok |

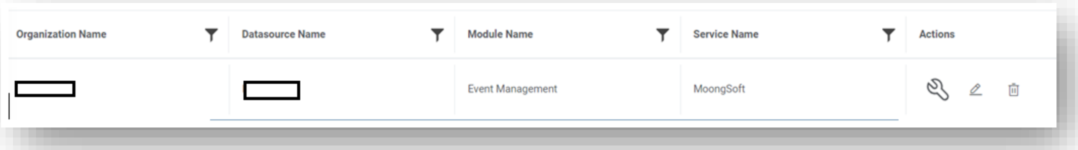
Figure 376 – Release Rules Configuration (Response Body)

22. **Response Key Value** mapping can be done as per the below table.

Table 68– Sample Response Key Value Mapping

| | | |
|---|---|---|
| #success# | Text | OK |

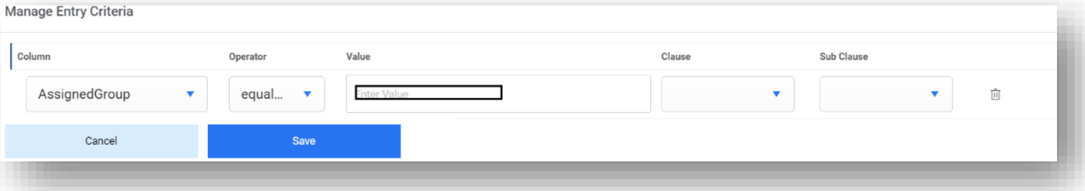
23. Click **Save** to add the data source.
24. To bring the tickets within iAutomate scope, a specific queue needs to be configured in the Event Management tool and same has to be configured in iAutomate. This is achieved through **Manage the Entry Criteria**. Please perform the below steps:
    - Go to Actions tab and click Manage Data Sources.
    - On the **Data Sources** tab, click 🔧 next to the data source user wants to manage. **Manage Entry Criteria** screen appears.



| Organization Name | Datasource Name | Module Name | Service Name | Actions |
|---|---|---|---|---|
| | | Event Management | MoongSoft | |

Figure 377 – Manage Entry Criteria

- Select 'AssignedGroup' for the **Column** field and 'equals to' for the **Operator** field.
- Enter the sys_id of the assignment group in Moogsoft in the **Value** field.
- **Clause** and **Sub-Clause** fields can also be added based on requirement.



Manage Entry Criteria

| Column | Operator | Value | Clause | Sub Clause |
|---|---|---|---|---|
| AssignedGroup | equal... | Enter Value | | |

Figure 378 – Manage Entry Criteria (Cont.)

25. Click **Save**.

## 4.9 Integration with Zenoss

This scenario is applicable when the ITSM tools is not available in the client environment and event management tool and iAutomate are tightly integrated directly. The event data or the probable root cause identified flows to iAutomate which then performs the appropriate action for resolution.

To create a data source, perform the following steps:

1. On the left menu bar, click Configuration -> Manage Data Source.
2. The **Create Data Source** page appears with the following tabs:
    - Organization
    - Fetch Data
3. **Release Rules**

- Close Rules (Optional – applicable only when the issue closure status update is managed by iAutomate directly instead of RBA tool)
- InProgress Rules (Optional – applicable only when the issues in progress status updates is managed by iAutomate directly instead of RBA tool)

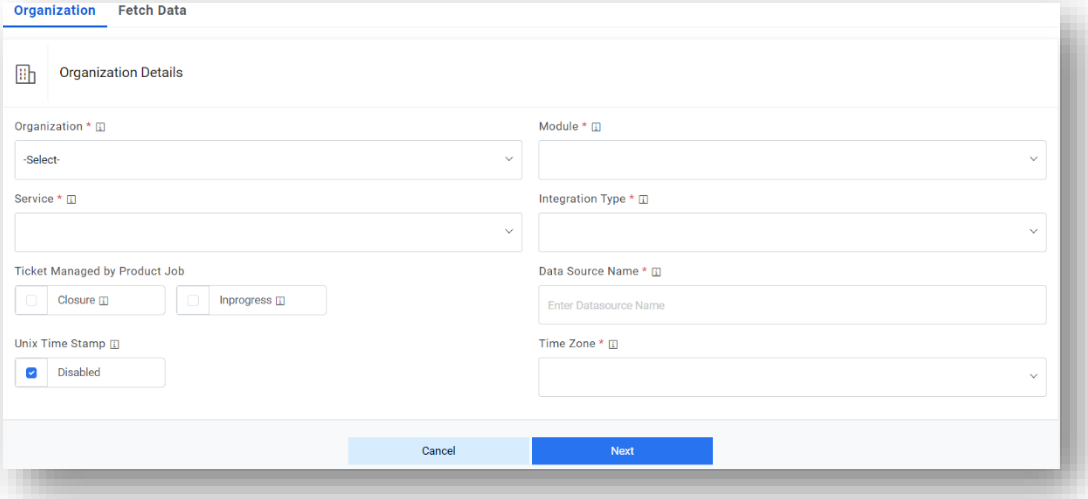
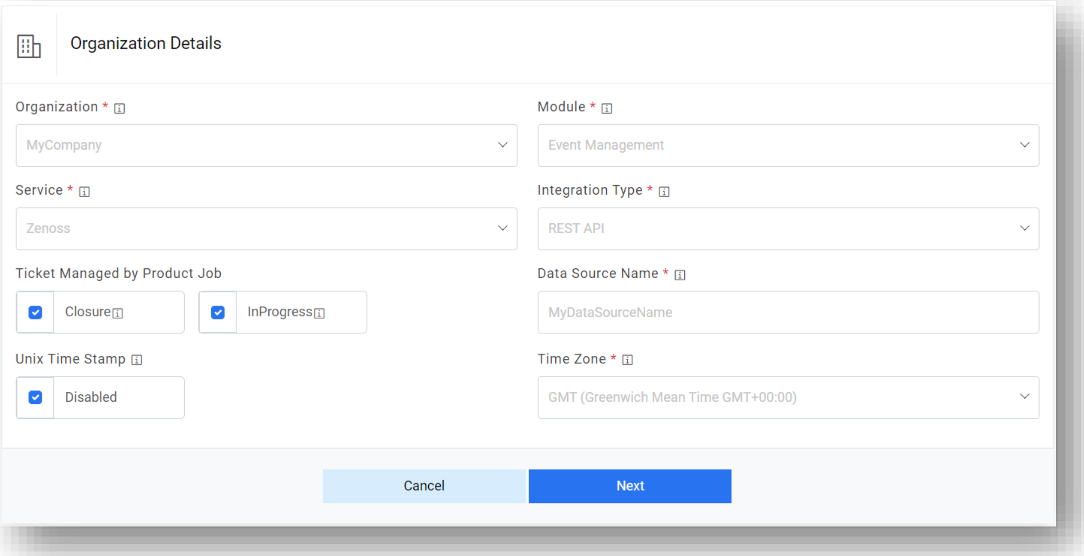


Figure 379 – Create Data Source

4. On the **Organization** tab,
   - Select the **Organization Name** from the dropdown.
   - Select the **Module** as **Event Management,** since we are configuring this data source for pulling the event data.
   - Select the **Service** as **Zenoss Tool** as we are configuring the data source for Zenoss
   - Select the **Integration Type** as **REST**, since we will be integrating through REST APIs.
   - Check **Is Ticket Closure Managed by iAutomate job** if you want iAutomate to manage the issue closure updates instead of the RBA tool. In this scenario, an additional tab **Close Rules Configuration** will be activated for providing further details, steps for which are mentioned later.
   - Check "**Is ticket InProgress Managed by iAutomate job**" if you want iAutomate to manage the issues in progress status updates instead of the RBA tool. In this scenario, an additional tab "**InProgress Rules Configuration**" will be activated for providing further details, steps for which are mentioned later.
   - Select the **Timezone** to specify the time zone of the selected data source.
   - Select **Timestamp** to view the present data with date and time.
   - Click **Next**.



Organization Details

Organization * MyCompany

Module * Event Management

Service * Zenoss

Integration Type * REST API

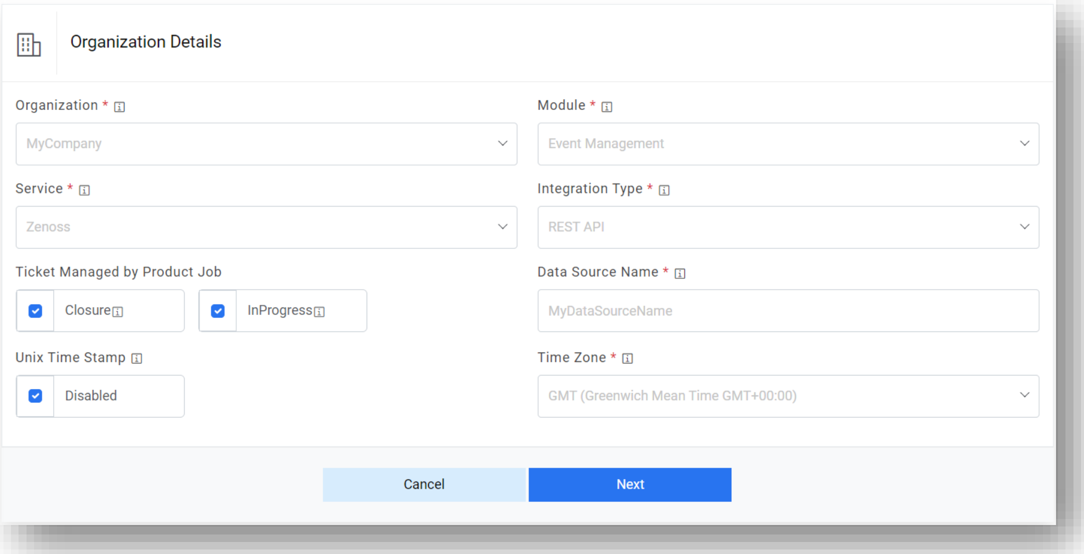
Ticket Managed by Product Job ☑ Closure ☑ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

Figure 380 – Create Data Source (Cont.)



Organization Details

Organization * MyCompany

Module * Event Management

Service * Zenoss

Integration Type * REST API

Ticket Managed by Product Job ☑ Closure ☑ InProgress

Data Source Name * MyDataSourceName

Unix Time Stamp ☑ Disabled

Time Zone * GMT (Greenwich Mean Time GMT+00:00)

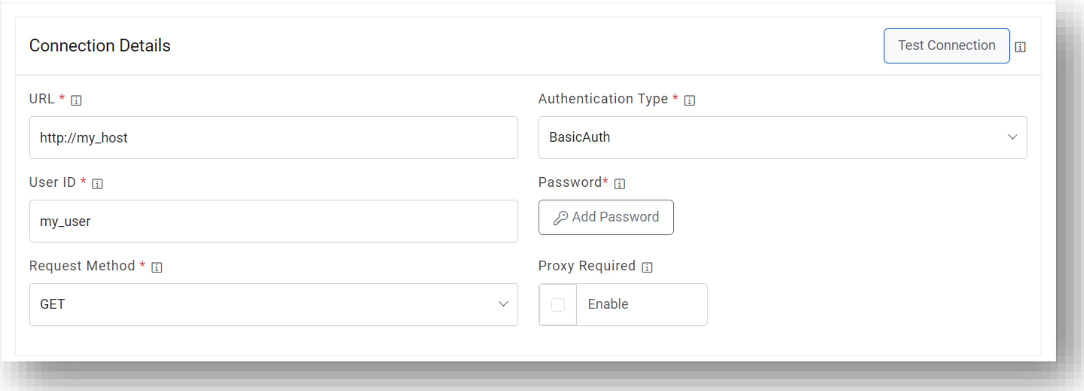
Figure 381 – Create Data Source (Cont.)

5. On the Fetch Data tab, type in the details as per the environment.
6. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL which contains the placeholders that display the parameters based on the applied clause such as the number of records to be fetched, query type, date on which the data is fetched, and the order by and so on. It is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<zenossURL>/
- **Authentication Type** - Select one of the Authentication Types from NoAuth / Basic / Windows
- Selection of **Basic / Windows** requires you to enter -
  - User Id
  - Password
- **Request Method -** Select Request Method as **POST** from the drop-down.

– **Proxy Required -** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Connection Details

URL * http://my_host

Authentication Type * BasicAuth

User ID * my_user

Password*

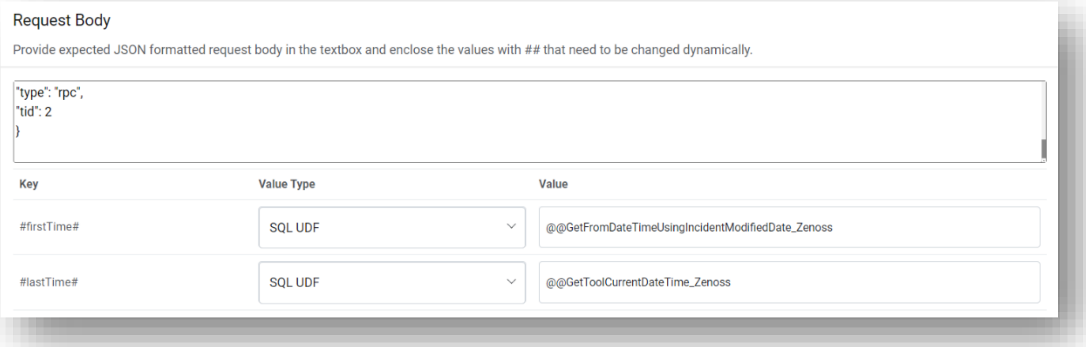
Request Method * GET

Proxy Required □ Enable

Figure 382 – Create Data Source (Connection Details)

7. **Request Header Parameters –** Please enter the request header parameters as required.
8. **Request Body -** As request method selected earlier is **POST**, please enter the body of URL. A sample response is mentioned below -

```
Request Body –

{"action": "EventsRouter","method": "query",

"data": [{

"keys":    ["evid",    "summary",    "eventState",    "severity",
"eventClass",   "ownerid",   "firstTime",   "lastTime",   "count",
"eventClassKey", "message"],

"params": {

"eventState": [0, 1], "severity": [5],

"excludeNonActionables": false,

"firstTime": "#firstTime# TO #lastTime#","eventClass": []},

"limit": 200,

"sort": "firstTime",

"dir": "ASC",

"start": 0,

"uid": "/cz0/zport/dmd"

}],

"type": "rpc",
"tid": 2

}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

```
"type": "rpc",
"tid": 2
}
```

| Key | Value Type | Value |
|---|---|---|
| #firstTime# | SQL UDF | @@GetFromDateTimeUsingIncidentModifiedDate_Zenoss |
| #lastTime# | SQL UDF | @@GetToolCurrentDateTime_Zenoss |

Figure 383 – Create Data Source (Connection Details)

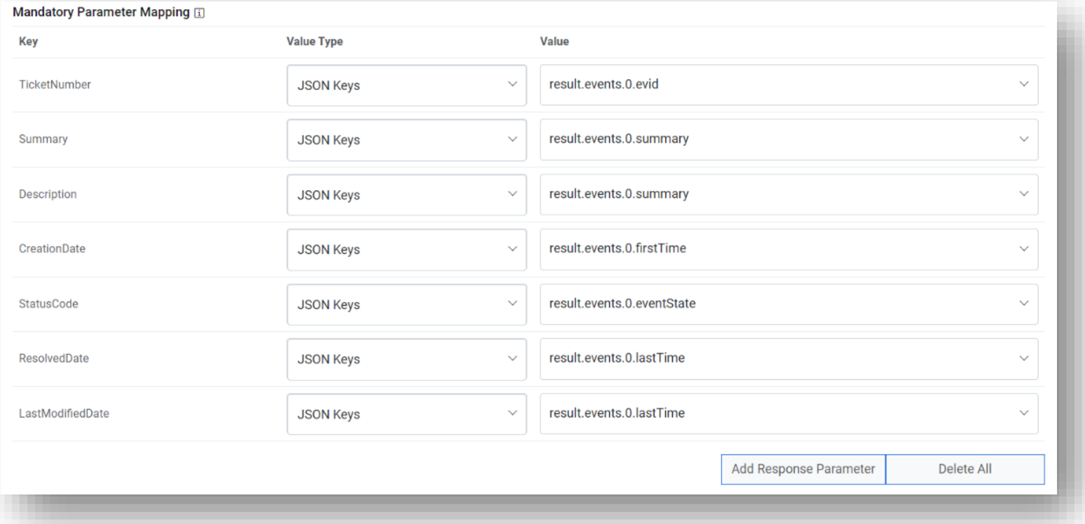
9. **Response Body –** In this section, please enter the output of URL query for one of the incidents in JSON format. A sample response is mentioned below -

```
Response Body –

{

"result": {

        "totalCount": 1,

        "events": [

            {

                "count": 1,

                "firstTime": 1600874287.072,

                "severity": 5,

                "evid": "0242ac11-000c-b913-11ea-fdaffba5ea6f",

                "eventClassKey": "",

                "summary": "xxxxxxxx | manageIP: xxxxxxxx",

                "eventState": "New",

                "ownerid": null,

                "eventClass": {

                    "text": "/App",

                    "uid": "/zport/dmd/Events/App"

                },

                "lastTime": 1600874287.072,
                "message": "xxxxxxxx"

            }

        ],

        "success": true,

        "asof": 1601266658.118566

    }

}
```

10. After entering the response, click **Extract Keys** to add the parameters in the **Mandatory Parameter Mapping** section.
11. **Mandatory Parameter Mapping –** Please map the mandatory parameters to the respective values as mentioned in the screenshot below:

Table 69– Sample Mandatory Parameter Mapping

| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON.Keys | result.0.evid |
| Summary | JSON.Keys | result.events.0.summary |
| Description | JSON.Keys | result.events.0.message |
| CreationDate | JSON.Keys | result.events.0.firstTime |
| StatusCode | JSON.Keys | result.events.0.eventState |
| ResolvedDate | JSON.Keys | result.events.0.lastTime |
| LastModifiedDate | JSON.Keys | result.events.0.lastTime |



Mandatory Parameter Mapping

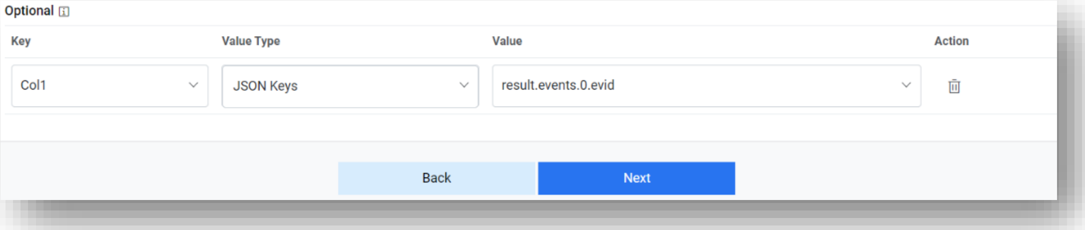
| Key | Value Type | Value |
|---|---|---|
| TicketNumber | JSON Keys | result.events.0.evid |
| Summary | JSON Keys | result.events.0.summary |
| Description | JSON Keys | result.events.0.summary |
| CreationDate | JSON Keys | result.events.0.firstTime |
| StatusCode | JSON Keys | result.events.0.eventState |
| ResolvedDate | JSON Keys | result.events.0.lastTime |
| LastModifiedDate | JSON Keys | result.events.0.lastTime |

Figure 384 – Mandatory Parameter Mapping

12. If you need to add **Optional** parameters, click **Add Response Parameter** to add more parameters. For our purpose, we will be adding a couple of extra parameters, as mentioned below, as we need them in the later section.

Table 70– Sample Optional Parameters

| Key | Value Type | Value |
|---|---|---|
| Col1 | JSON.Keys | result.0.evid |



Optional

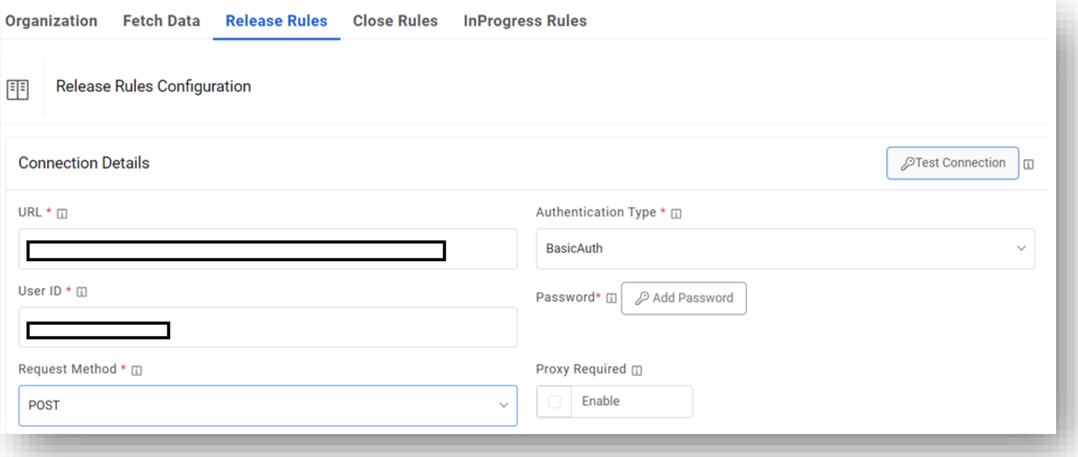
| Key | Value Type | Value | Action |
|---|---|---|---|
| Col1 | JSON Keys | result.events.0.evid | 🗑 |

Figure 385 – Optional Parameter Mapping

13. Click Next to proceed to Release Rules.
14. On **Release Rules** tab, type in the details as per the requirement.
15. In the **Connection Details** section, enter the following details:

- **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.
- **Sample URL** - https://<zenossurl>
- **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.
- **Request Method** – Select Request Method as POST from the drop-down.
- **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



URL * [redacted]

Authentication Type * BasicAuth

User ID * [redacted]

Password*

Request Method * POST

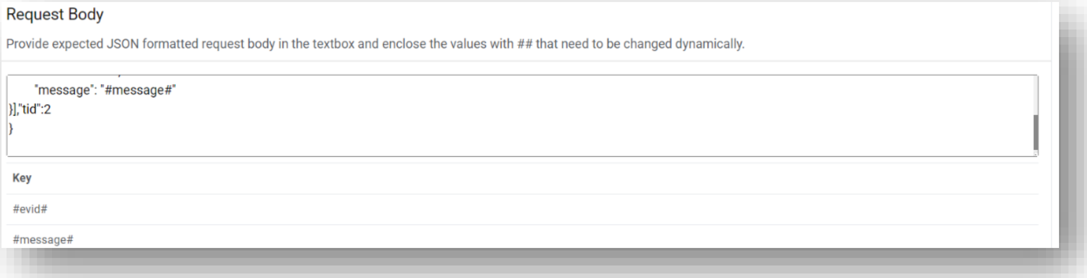
Proxy Required □ Enable

Figure 386 – Release Rules Configuration (Connection Details)

16. **Request Header Parameters –** Please enter the request header parameters as required.

17. **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below -

```
Request Body –

{

    "action": "EventsRouter",

    "method": "write_log",

    "data": [{

        "evid": "#evid#",

        "message": "#message#"

    }],"tid":2

}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

```
    "message": "#message#"
}],"tid":2
}
```

Key

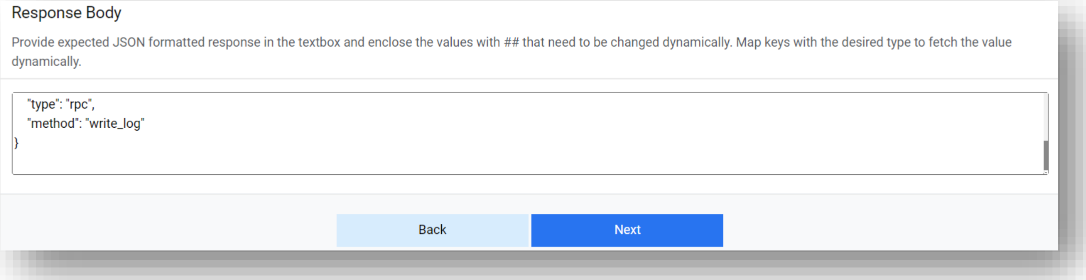
#evid#

#message#

Figure 387 – Release Rules (Request Body)

18. **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below -

```
Response Body –

{

    "uuid": "xxxxxxxx",

    "action": "EventsRouter",

    "result": {

        "success": true

    },

    "tid": 2,

    "type": "rpc",

    "method": "write_log"

}
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
    "type": "rpc",
    "method": "write_log"
}
```

Figure 388 – Release Rules Configuration (Response Body)

19. On **Close Rules** tab, type in the details as per the requirement. Check **Same as Release** if similar configurations as mentioned in "Release Rules Configuration" are required, else proceed ahead.
20. In the **Connection Details** section, enter the following details:

– **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.

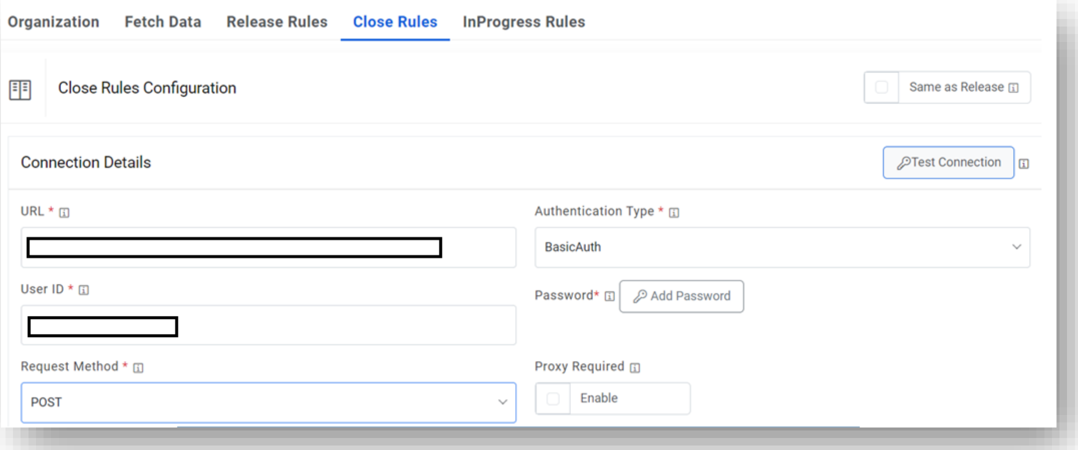
– **Sample URL** - https://<url>

– **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.

– Request Method – Select Request Method as POST from the drop-down.

– **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



Close Rules Configuration

□ Same as Release

Connection Details

URL *: [redacted]

Authentication Type *: BasicAuth

User ID *: [redacted]

Password*: Add Password

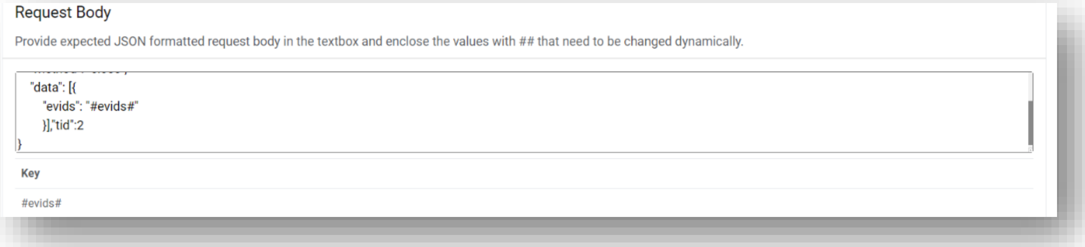
Request Method *: POST

Proxy Required: □ Enable

Figure 389 – Release Rules Configuration (Connection Details)

21. **Request Header Parameters –** Please enter the request header parameters as required.
22. **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below.

```
Request Body -

{
    "action": "EventsRouter",

    "method": "close",

    "data": [{

        "evids": "#evids#"

        }],"tid":2

}
```



Request Body

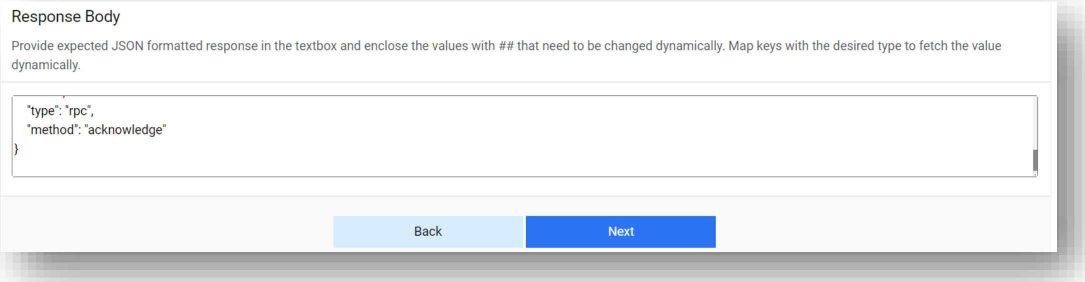
Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

```
 "data": [{
   "evids": "#evids#"
   }],"tid":2
}
```

Figure 390 – Release Rules (Request Body)

23. **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below -

```
Response Body –

{

    "uuid": "xxxxxxxx",

    "action": "EventsRouter",

    "result": {

        "data": {

            "updated": 31,

            "total": 3670

        },

        "success": true

    },

    "tid": 2,

    "type": "rpc",

    "method": "acknowledge"

}
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
  "type": "rpc",
  "method": "acknowledge"
}
```

Figure 391 – Release Rules Configuration (Response Body)

24. On **InProgress Rules** tab, type in the details as per the requirement. Check **Same as Release** if similar configurations as mentioned in "Release Rules Configuration" are required, else proceed ahead.
25. In the **Connection Details** section, enter the following details:

– **URL –** Type the URL of the selected service type to release the ticket. It contains the placeholders that display the parameters based on the applied clause and is dependent on the URL or API provided by the tool.

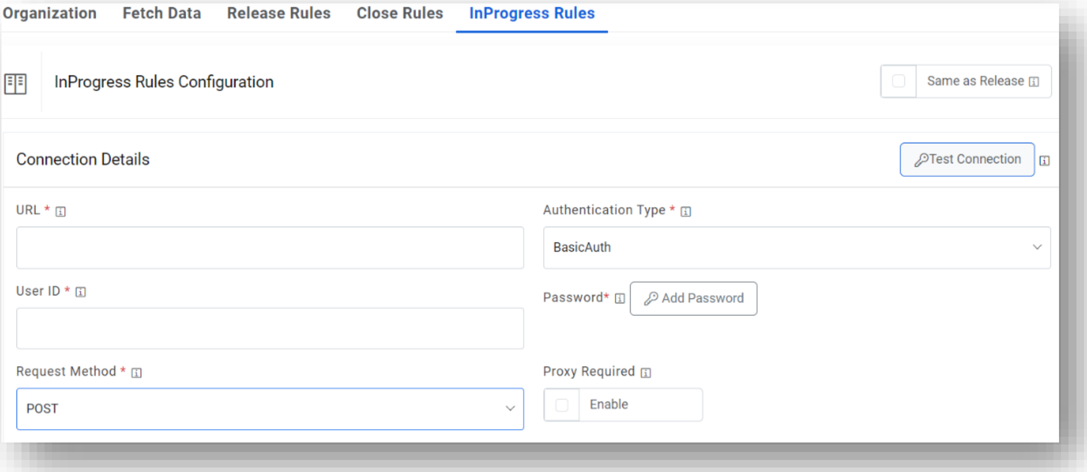
– **Sample URL** - https//<url>

– **Authentication Type –** Please enter the information in line with the Authentication type configured for fetching data configuration previously.

– **Request Method** – Select Request Method as POST from the drop-down.

– **Proxy Required –** Check **Proxy Required**, if the environment needs access to content from data sources outside the firewall.

Click on **Test Connection** to check accessibility of URL from service. Testing the connection is not mandatory, you can still create Data source.



InProgress Rules Configuration

□ Same as Release

Connection Details

URL *

Authentication Type * BasicAuth

User ID *

Password*

Request Method * POST

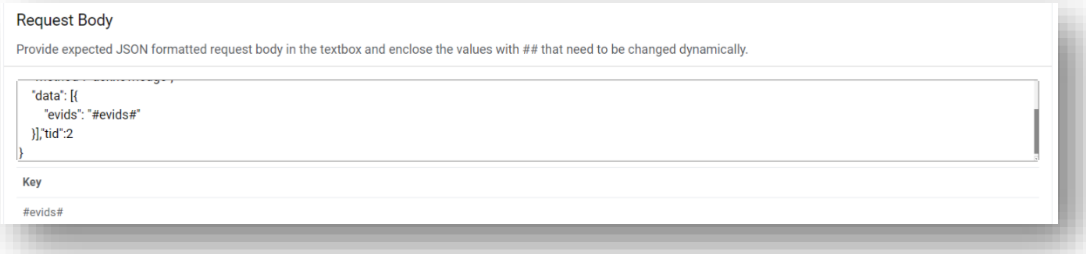
Proxy Required □ Enable

Figure 392 – Release Rules (Connection Details)

26. **Request Header Parameters –** Please enter the request header parameters as required.
27. **Request Body –** In this section, please enter the request body in JSON format. A sample request is mentioned below -

```
Request Body –
```

```
{
    "action": "EventsRouter",
    "method": "acknowledge",
    "data": [{
        "evids": "#evids#"
    }],"tid":2
}
```



Request Body

Provide expected JSON formatted request body in the textbox and enclose the values with ## that need to be changed dynamically.

```
  "data": [{
    "evids": "#evids#"
  }],"tid":2
}
```

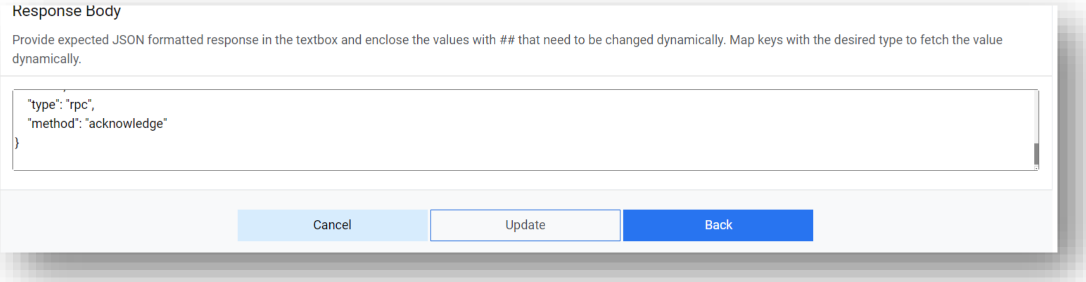
Key

#evids#

Figure 393 – Release Rules Configuration (Request Body)

28. **Response Body –** In this section, please enter the response body in JSON format. A sample response is mentioned below -

```
Response Body -
{
    "uuid": "xxxxxxxx",
    "action": "EventsRouter",
    "result": {
        "data": {
            "updated": 31,
            "total": 3670
        },
        "success": true
    },
    "tid": 2,
    "type": "rpc",
    "method": "acknowledge"
}
```



Response Body

Provide expected JSON formatted response in the textbox and enclose the values with ## that need to be changed dynamically. Map keys with the desired type to fetch the value dynamically.

```
  "type": "rpc",
  "method": "acknowledge"
}
```

Figure 394 – Release Rules Configuration (Response Body)

29. Click **Save** to add the data source.

# 5 Integration with RBA / Orchestrator Tools

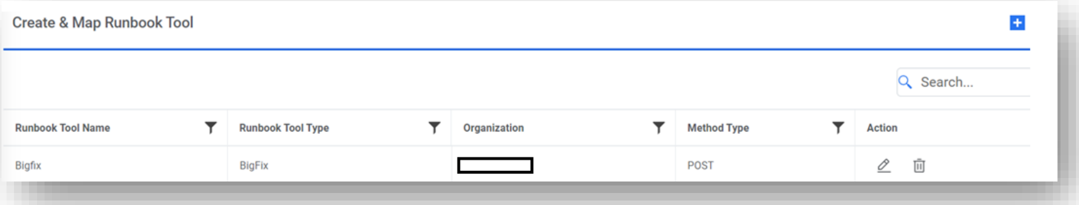
iAutomate leverages the services of a Runbook Automation (RBA) / Orchestrator tool to perform actions as defined in the runbooks a.k.a. workflows. Thus, to enable integration with RBA tool, you need to onboard a runbook automation tool through configuration.

Before proceeding with the configuration related to Data Source creation, user has to ensure that an organization has been configured. If not done already, please refer to the Configuration Guide for the same and create the organization before proceeding ahead.

## 5.1 Integration with Broadcom CA ITPAM

To manage / onboard Broadcom CA ITPAM as the RBA tool, perform the following steps:

1. On the left menu bar, click **Runbooks**, -> click **Manage Runbook Tool**. The **Manage Runbook Tool** appears.

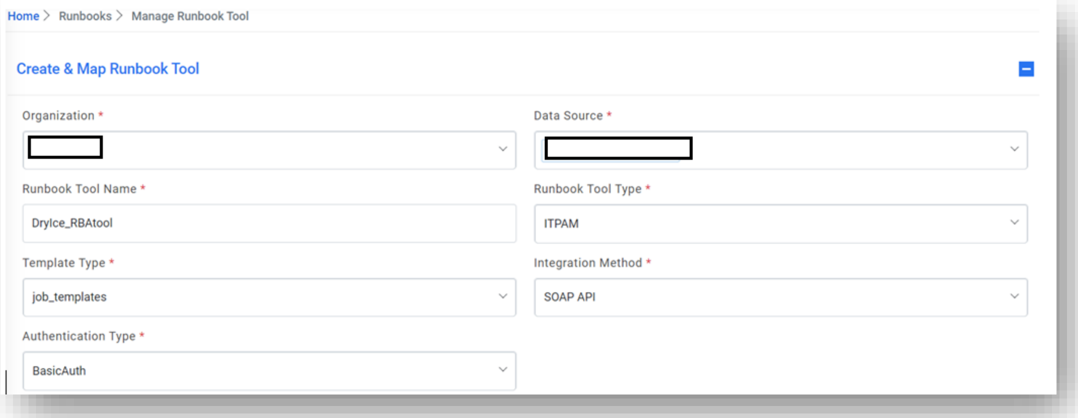


Create & Map Runbook Tool

| Runbook Tool Name | Runbook Tool Type | Organization | Method Type | Action |
|---|---|---|---|---|
| Bigfix | BigFix | [redacted] | POST | ✎ 🗑 |

Figure 395 - Manage Runbook Tool

2. It lists the available runbook tools in a tabular view and lets the user to add a new runbook tool using ⊞ **Add New** button. User can also edit or delete the existing runbook tools.
3. Click **Add New** to add a new tool or click ✎ to edit an existing runbook automation tool.
4. Select organization for which you need to create runbook tool in the **Organization Name** field.
5. Type the runbook tool name in the **Runbook Tool Name** field.
6. Select **ITPAM** from the **Runbook Tool Type** drop-down
7. Select **SOAP API** as the integration method for ITPAM for the **Integration Method** field.



Create & Map Runbook Tool

Organization * [redacted]

Data Source * [redacted]

Runbook Tool Name * DryIce_RBAtool

Runbook Tool Type * ITPAM

Template Type * job_templates

Integration Method * SOAP API

Authentication Type * BasicAuth

Figure 396 - Manage Runbook Tool (Cont.)

8. Select one of the **Authentication Type** from BasicAuth / WindowsAuth

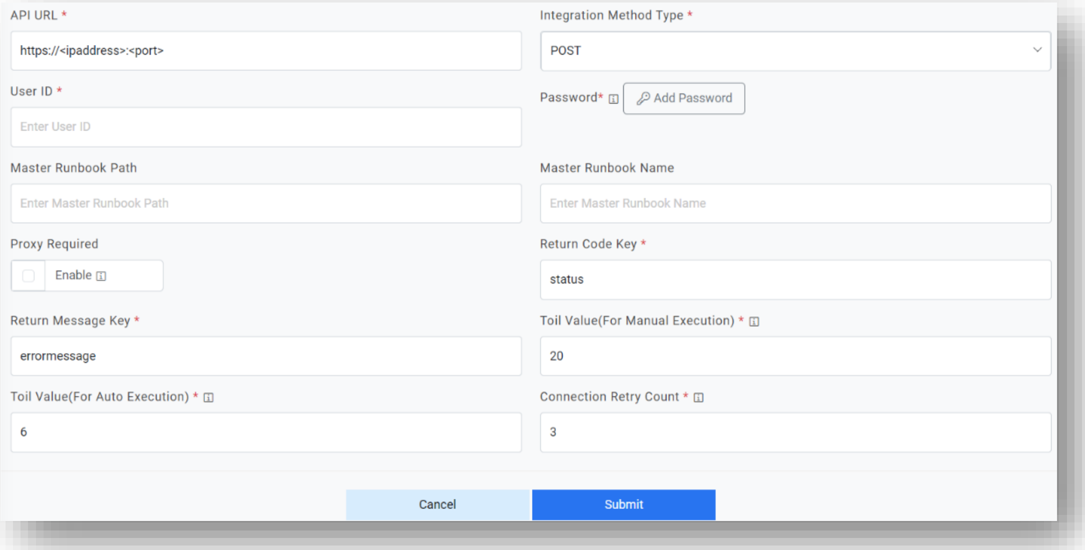
– Selection of from **BasicAuth / WindowsAuth** requires you to enter –

  - User Id
  - Password

9. Type the URL in the **API URL**. field.
10. **Sample URL** - http://<url>:<port>/itpam/soap
11. In the **Integration Method Type** field select 'POST'.
12. Type the username and password in the **User ID** and **Password** field to get access to API web services.
13. API URL, User ID, and Password are dependent on the selected integration method.
14. Specify the path to get the consolidated scripts for the execution of runbooks in the **Master Runbook Path** field. This will be provided by respective **Runbook Tool** teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

15. Select **Proxy Required**, if the environment needs access to content from servers outside a firewall.
16. Type the return code key in the **Return Code Key** field to identify the success or failure of runbook execution. E.g. status
17. Type the return message key in the **Return Message Key** field to display the success or failure of runbook execution. E.g. errormessage
18. Type the **Toil Value** (For Manual Execution) which implies the manual execution time of runbook under this tool (in minutes).
19. Type the **Toil Value** (For Auto Execution) which implies the auto execution time of runbook under this tool (in minutes), if available else it's a non-mandatory field.



API URL * https://<ipaddress>:<port>

Integration Method Type * POST

User ID * Enter User ID

Password* Add Password

Master Runbook Path Enter Master Runbook Path

Master Runbook Name Enter Master Runbook Name

Proxy Required □ Enable

Return Code Key * status

Return Message Key * errormessage

Toil Value(For Manual Execution) * 20

Toil Value(For Auto Execution) * 6

Connection Retry Count * 3

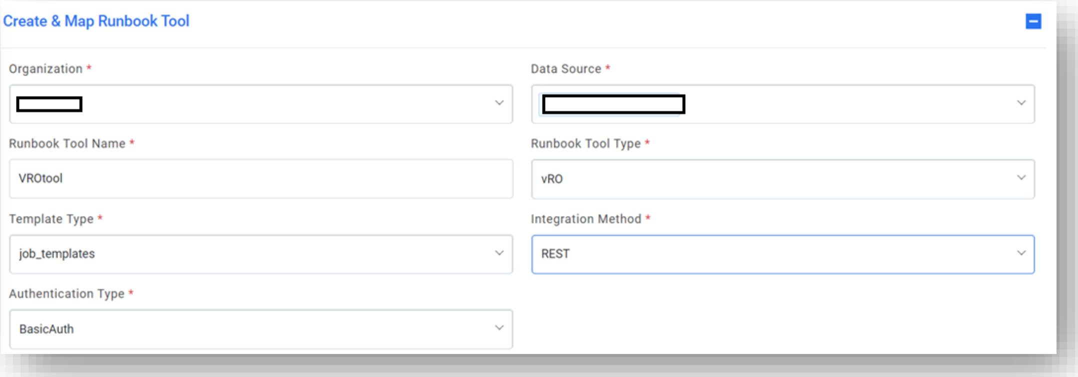
Figure 397 - Manage Runbook Tool (Cont.)

20. Click **Submit** / **Update** for adding a new tool or making changes to an existing tool. An appropriate success message will be displayed.

## 5.2 Integration with VMware vRealize Orchestrator (vRO)

To manage / onboard VMware vRO as the RBA tool, perform the following steps:

1. On the left menu bar, click **Runbooks**, and then click **Manage Runbook Tool**. The **Manage Runbook Tool** appears.
2. Click **Add New** to add a new tool or click ✎ to edit an existing runbook automation tool.
3. Select organization for which you need to create runbook tool in the **Organization Name** field.
4. Type the runbook tool name in the **Runbook Tool Name** field.
5. Select **vRO** from the **Runbook Tool Type** drop-down.
6. Select **REST** as the integration method for vRO for the **Integration Method** field.



Create & Map Runbook Tool

Organization * [redacted]

Data Source * [redacted]

Runbook Tool Name * VROtool

Runbook Tool Type * vRO

Template Type * job_templates

Integration Method * REST

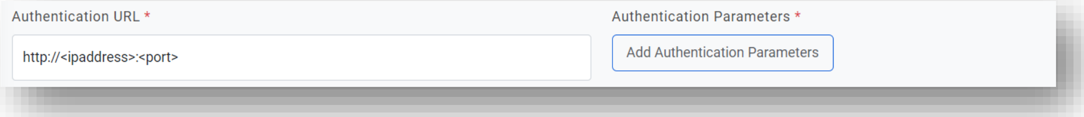
Authentication Type * BasicAuth

Figure 398 - Manage Runbook Tool (Cont.)

7. Select one of the Authentication Type from BasicAuth / Token Auth

– Selection of from **BasicAuth / Token Auth** requires you to enter –
  - User Id
  - Password

– Selection of from **Token Auth** requires you to enter –
  - Authentication URL

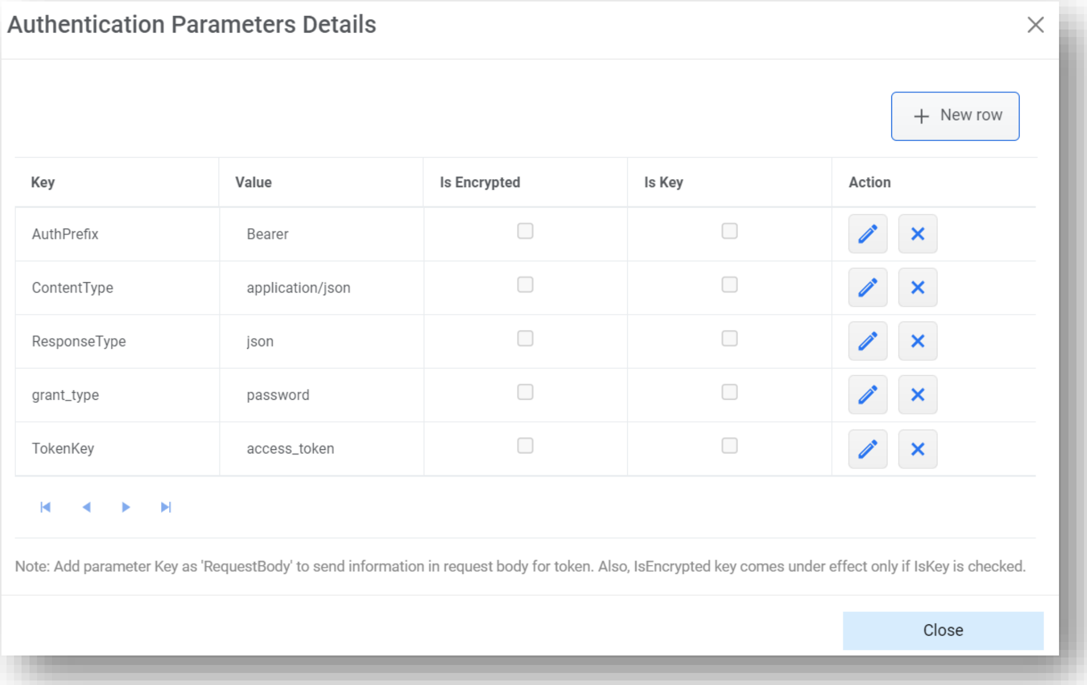


Authentication URL * http://<ipaddress>:<port>

Authentication Parameters *

Figure 399 - Manage Runbook Tool (Cont.)

8. Click **Add Authentication Parameters** to add more parameters, as depicted below.



Authentication Parameters Details

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| AuthPrefix | Bearer | ☐ | ☐ | ✎ ✕ |
| ContentType | application/json | ☐ | ☐ | ✎ ✕ |
| ResponseType | json | ☐ | ☐ | ✎ ✕ |
| grant_type | password | ☐ | ☐ | ✎ ✕ |
| TokenKey | access_token | ☐ | ☐ | ✎ ✕ |

Note: Add parameter Key as 'RequestBody' to send information in request body for token. Also, IsEncrypted key comes under effect only if IsKey is checked.

Figure 400 - Manage Runbook Tool (Cont.)

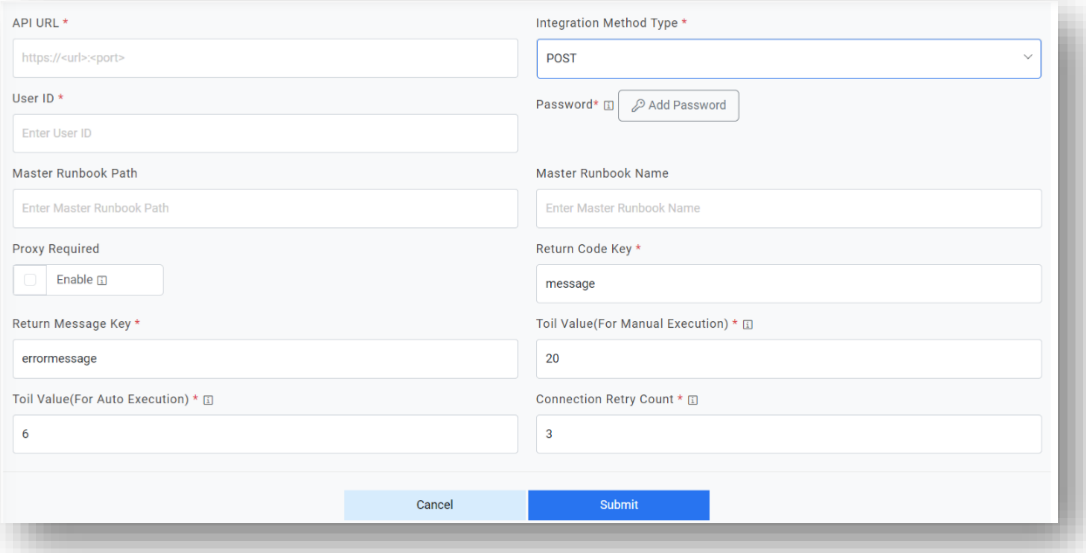
9. Type the URL in the **API URL**. field.
10. **Sample URL** - http://<url>:<port>/vco/api/workflows
11. In the Integration Method Type select 'POST'.
12. Type the username and password in the **User ID** and **Password** field to get access to API web services.

API URL, User ID, and Password are dependent on the selected integration method.

13. Specify the path to get the consolidated scripts for the execution of runbooks in the **Master Runbook Path** field. This will be provided by respective Runbook Tool teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

14. Select **Proxy Required**, if the environment needs access to content from servers outside a firewall.
15. Type the return code key in the **Return Code Key** field to identify the success or failure of runbook execution. E.g. ReturnCode.
16. Type the return message key in the **Return Message Key** field to display the success or failure of runbook execution. E.g. ReturnMessage.
17. Type the **Toil Value** (For Manual Execution) which implies the manual execution time of runbook under this tool (in minutes).
18. Type the **Toil Value** (For Auto Execution) which implies the auto execution time of runbook under this tool (in minutes), if available else it's a non-mandatory field.



API URL *

https://<url>:<port>

Integration Method Type *

POST

User ID *

Enter User ID

Password* Add Password

Master Runbook Path

Enter Master Runbook Path

Master Runbook Name

Enter Master Runbook Name

Proxy Required

□ Enable

Return Code Key *

message

Return Message Key *

errormessage

Toil Value(For Manual Execution) *

20

Toil Value(For Auto Execution) *

6

Connection Retry Count *

3

Cancel Submit

Figure 401 - Manage Runbook Tool (Cont.)

19. Click **Submit** / **Update** for adding a new tool or making changes to an existing tool. An appropriate success message will be displayed.

## 5.3 Integration with Ansible CLI

To manage / onboard Ansible CLI as the RBA tool, perform the following steps:

1. On the left menu bar, click **Runbooks**, and then click **Manage Runbook Tool**. The **Manage Runbook Tool** appears.
2. Click **Add New** to add a new tool or click ✎ to edit an existing runbook automation tool.
3. Select organization for which you need to create runbook tool in the **Organization Name** field.
4. Type the runbook tool name in the **Runbook Tool Name** field.
5. Select ANSIBLE CLI from the Runbook Tool Type drop-down
6. Select **CLI** as the integration method for Ansible CLI for the **Integration Method** field.
7. Type the IP Address in the **IP Address** field.
8. Sample IP Address – 10.0.0.0
9. In Integration Method Type select 'POST'.
10. Type the username and password in the **User ID** and **Password** field to get access to CLI host.

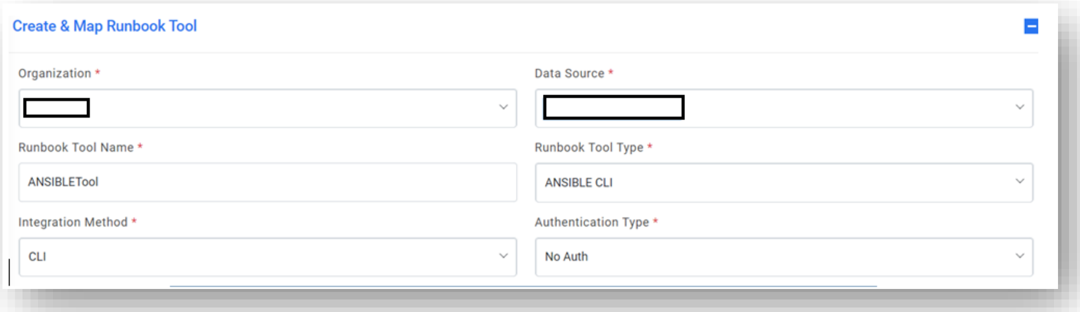
IP Address, User ID, and Password are dependent on the selected integration method.

11. Specify the path to get the consolidated scripts for the execution of runbooks in the **Master Runbook Path** field. This will be provided by respective **Runbook Tool** teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

12. Select **Proxy Required**, if the environment needs access to content from servers outside a firewall.
13. Type the return code key in the **Return Code Key** field to identify the success or failure of runbook execution.

14. Type the return message key in the **Return Message Key** field to display the success or failure of runbook execution.
15. Type the **Toil Value** (For Manual Execution) which implies the manual execution time of runbook under this tool (in minutes).
16. Type the **Toil Value** (For Auto Execution) which implies the auto execution time of runbook under this tool (in minutes), if available else it's a non-mandatory field.



Create & Map Runbook Tool

Organization * [redacted]

Data Source * [redacted]

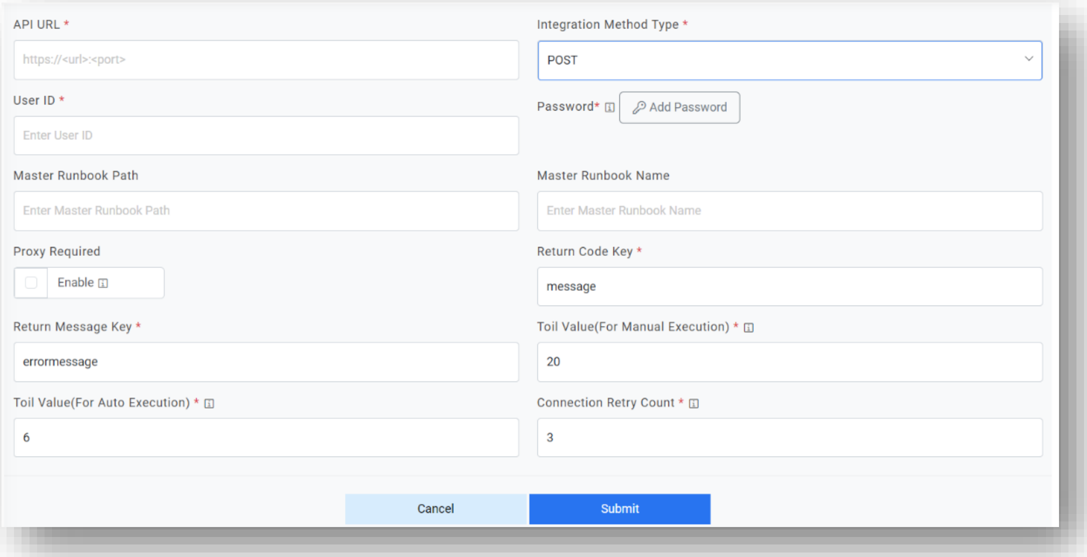
Runbook Tool Name * ANSIBLETool

Runbook Tool Type * ANSIBLE CLI

Integration Method * CLI

Authentication Type * No Auth

Figure 402 - Manage Runbook Tool (Cont.)



API URL * https://<url>:<port>

Integration Method Type * POST

User ID * Enter User ID

Password* Add Password

Master Runbook Path Enter Master Runbook Path

Master Runbook Name Enter Master Runbook Name

Proxy Required □ Enable

Return Code Key * message

Return Message Key * errormessage

Toil Value(For Manual Execution) * 20

Toil Value(For Auto Execution) * 6

Connection Retry Count * 3

Figure 403 - Manage Runbook Tool (Cont.)

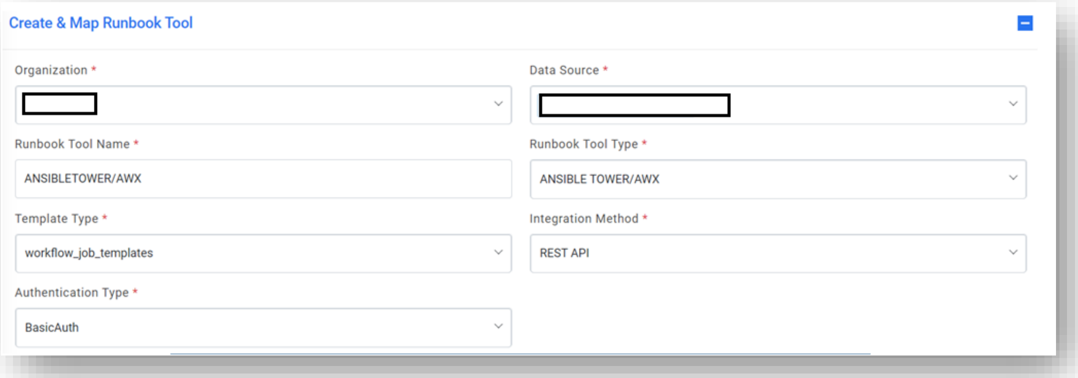
17. Click **Submit** / **Update** for adding a new tool or making changes to an existing tool. An appropriate success message will be displayed.

## 5.4 Integration with Ansible Tower / AWX

To manage / onboard Ansible Tower \ AWX as the RBA tool, perform the following steps:

1. On the main menu bar, click **Runbooks**, and then click **Manage Runbook Tool**. The **Manage Runbook Tool** appears.
2. Click **Add New** to add a new tool or click ✎ to edit an existing runbook automation tool.
3. Select an organization for which you need to create runbook tool in the **Organization Name** field.
4. Type the runbook tool name in the **Runbook Tool Name** field.

5. Select ANSIBLE TOWER\AWX from the Runbook Tool Type drop-down.
6. Select **REST** as the integration method for Ansible Tower \ AWX for the **Integration Method** field.



Create & Map Runbook Tool

Organization *: [redacted]

Data Source *: [redacted]

Runbook Tool Name *: ANSIBLETOWER/AWX

Runbook Tool Type *: ANSIBLE TOWER/AWX

Template Type *: workflow_job_templates

Integration Method *: REST API

Authentication Type *: BasicAuth

Figure 404 - Manage Runbook Tool (Cont.)

7. Select the **Authentication Type** from BasicAuth

– Selection of from **BasicAuth** requires you to enter:
    - User Id
    - Password

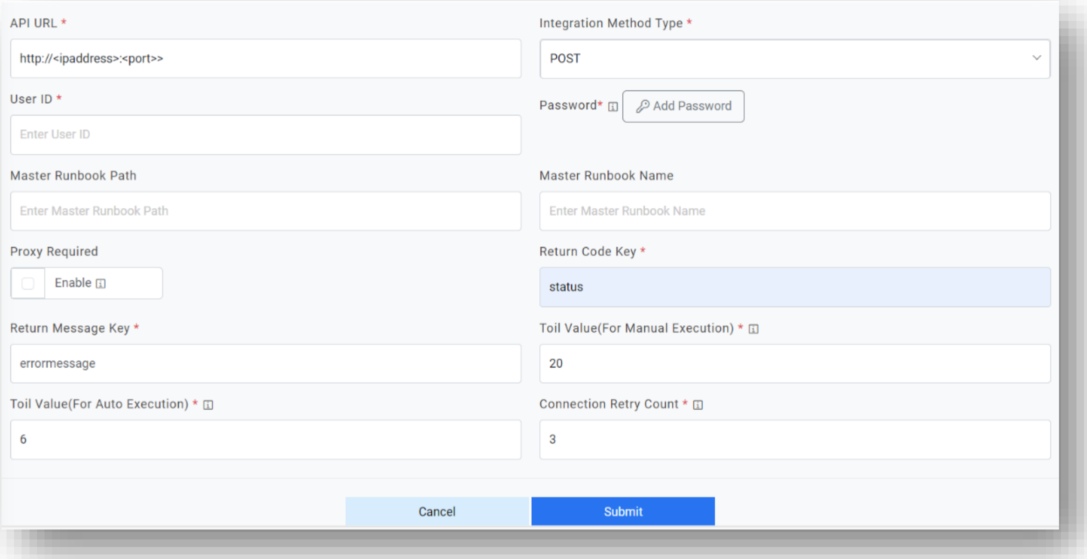
8. Type the URL in the **API URL** field.
9. **Sample URL** - https://<URL/IP>:<PORT>
10. In Integration Method Type select 'POST'.
11. Type the username and password in the **User ID** and **Password** field to get access to API web services.

API URL, User ID, and Password are dependent on the selected integration method

12. Specify the path to get the consolidated scripts for the execution of runbooks in the Master Runbook Path field. This will be provided by respective Runbook Tool teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

13. Select **Proxy Required**, if the environment needs access to content from servers outside a firewall.
14. Type the return code key in the **Return Code Key** field to identify the success or failure of runbook execution. E.g. status_codes.
15. Type the return message key in the **Return Message Key** field to display the success or failure of runbook execution. E.g. iautomate_success.
16. Type the Toil Value (For Manual Execution) which implies the manual execution time of runbook under this tool (in minutes).
17. Type the Toil Value (For Auto Execution) which implies the auto execution time of runbook under this tool (in minutes), if available else it's a non-mandatory field.



API URL *: http://<ipaddress>:<port>>

Integration Method Type *: POST

User ID *: Enter User ID

Password* 🛈 Add Password

Master Runbook Path: Enter Master Runbook Path

Master Runbook Name: Enter Master Runbook Name

Proxy Required: □ Enable 🛈

Return Code Key *: status

Return Message Key *: errormessage

Toil Value(For Manual Execution) * 🛈: 20

Toil Value(For Auto Execution) * 🛈: 6

Connection Retry Count * 🛈: 3

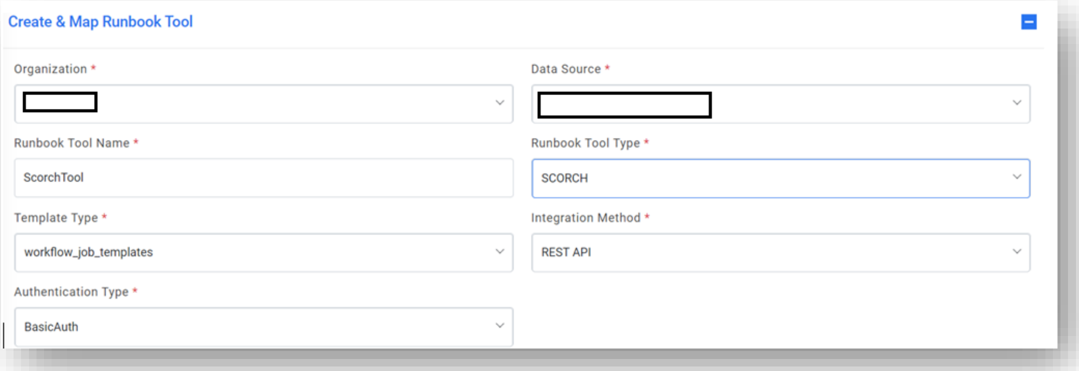
Figure 405 - Manage Runbook Tool (Cont.)

18. Click **Submit** / **Update** for adding a new tool or making changes to an existing tool. An appropriate success message will be displayed.

## 5.5 Integration with Microsoft System Orchestrator (MS SCORCH)

To manage / onboard Microsoft SCORCH as the RBA tool, perform the following steps:

1. On the main menu bar, click **Runbooks**, and then click **Manage Runbook Tool**. The **Manage Runbook Tool** appears.
2. Click **Add New** to add a new tool or click ✎ to edit an existing runbook automation tool.
3. Select organization for which you need to create runbook tool in the **Organization Name** field.
4. Type the runbook tool name in the **Runbook Tool Name** field.
5. Select **SCORCH** from the **Runbook Tool Type** drop-down
6. Select **REST** as the integration method for SCORCH for the **Integration Method** field.



Create & Map Runbook Tool

Organization *: [redacted]

Data Source *: [redacted]

Runbook Tool Name *: ScorchTool

Runbook Tool Type *: SCORCH

Template Type *: workflow_job_templates

Integration Method *: REST API

Authentication Type *: BasicAuth

Figure 406 - Manage Runbook Tool (Cont.)

7. Select one of the Authentication Type from BasicAuth, OAuth 2.0

– Selection of from **BasicAuth** requires you to enter –

  - User Id

    - Password

- Selection of from **OAuth 2.0** requires you to enter –
    - Authentication URL

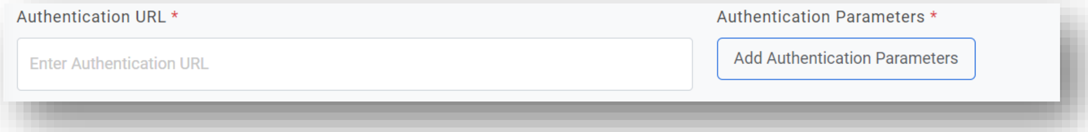
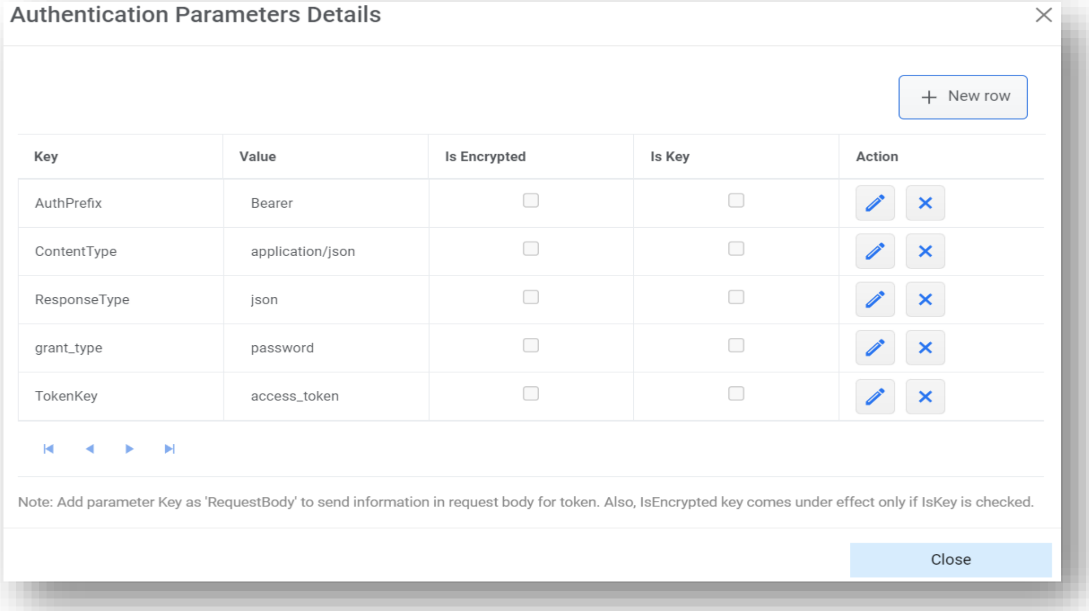


Figure 407 - Manage Runbook Tool (Cont.)

8. Click **Add Authentication Parameters** to add more parameters, as depicted below.



| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| AuthPrefix | Bearer | □ | □ | ✎ ✕ |
| ContentType | application/json | □ | □ | ✎ ✕ |
| ResponseType | json | □ | □ | ✎ ✕ |
| grant_type | password | □ | □ | ✎ ✕ |
| TokenKey | access_token | □ | □ | ✎ ✕ |

Note: Add parameter Key as 'RequestBody' to send information in request body for token. Also, IsEncrypted key comes under effect only if IsKey is checked.

Figure 408 - Manage Runbook Tool (Cont.)

9. Type the URL in the **API URL**. field.
10. **Sample URL** - http://<URL/IP>:<PORT>/Orchestrator2012/Orchestrator.svc/
11. In the **Integration Method Type** field select 'POST'.
12. Type the username and password in the **User ID** and **Password** field to get access to API web services.

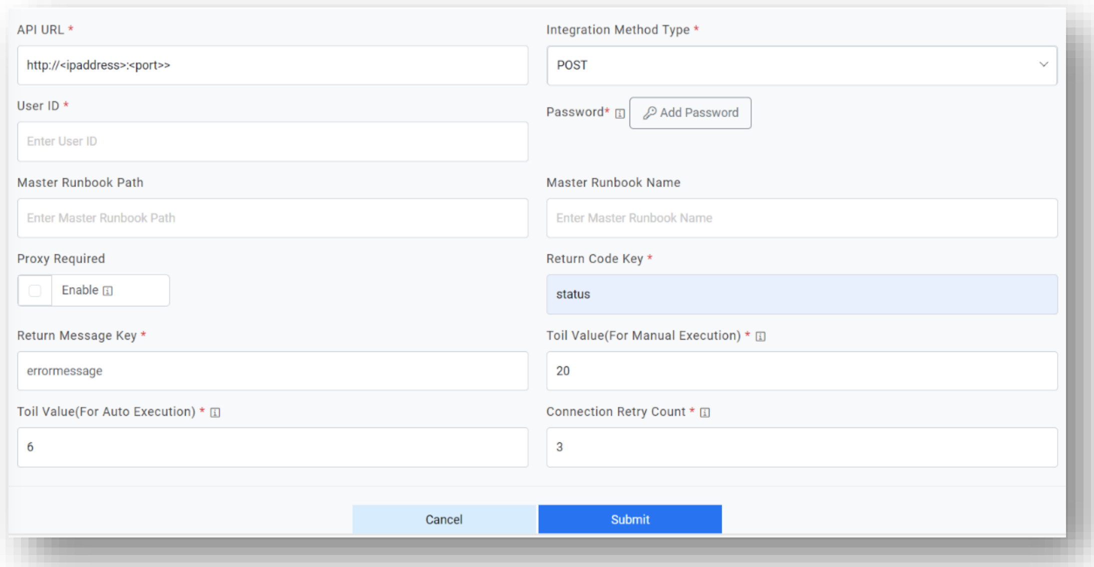
API URL, User ID, and Password are dependent on the selected integration method

13. Specify the path to get the consolidated scripts for the execution of runbooks in the **Master Runbook Path** field. This will be provided by respective **Runbook Tool** teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

14. Select **Proxy Required,** if the environment needs access to content from servers outside a firewall.
15. Type the return code key in the **Return Code Key** field to identify the success or failure of runbook execution. E.g. statuscode.
16. Type the return message key in the **Return Message Key** field to display the success or failure of runbook execution. E.g. message.

17. Type the Toil Value (For Manual Execution) which implies the manual execution time of runbook under this tool (in minutes).
18. Type the Toil Value (For Auto Execution) which implies the auto execution time of runbook under this tool (in minutes), if available else it's a non-mandatory field.



| | |
|---|---|
| API URL * | Integration Method Type * |
| http://<ipaddress>:<port>> | POST |
| User ID * | Password* Add Password |
| Enter User ID | |
| Master Runbook Path | Master Runbook Name |
| Enter Master Runbook Path | Enter Master Runbook Name |
| Proxy Required | Return Code Key * |
| □ Enable | status |
| Return Message Key * | Toil Value(For Manual Execution) * |
| errormessage | 20 |
| Toil Value(For Auto Execution) * | Connection Retry Count * |
| 6 | 3 |

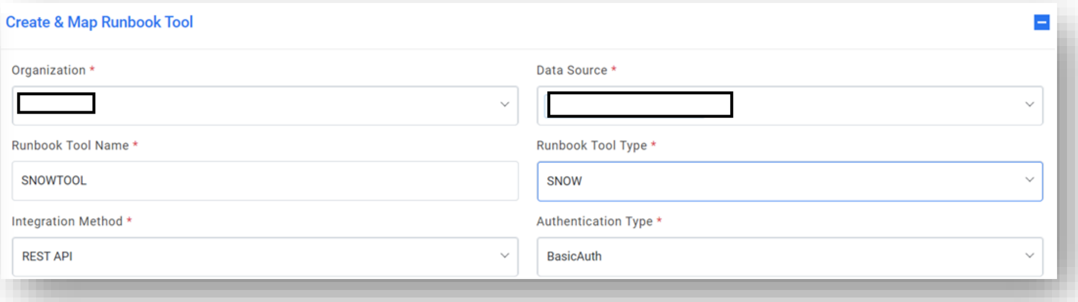
Figure 409 - Manage Runbook Tool (Cont.)

19. Click **Submit** / **Update** for adding a new tool or making changes to an existing tool. An appropriate success message will be displayed.

## 5.6 Integration with ServiceNow Orchestration

To manage / onboard ServiceNow Orchestration as the RBA tool, perform the following steps:

1. On the main menu bar, click **Runbooks**, and then click **Manage Runbook Tool**. The **Manage Runbook Tool** appears.
2. Click **Add New** to add a new tool or click ✎ to edit an existing runbook automation tool.
3. Select organization for which you need to create runbook tool in the **Organization Name** field.
4. Type the runbook tool name in the **Runbook Tool Name** field.
5. Select **SNOW** from the **Runbook Tool Type** drop-down
6. Select **REST API** as the integration method for ServiceNow Orchestration for the **Integration Method** field.



Create & Map Runbook Tool

Organization * [redacted]

Data Source * [redacted]

Runbook Tool Name * SNOWTOOL

Runbook Tool Type * SNOW

Integration Method * REST API

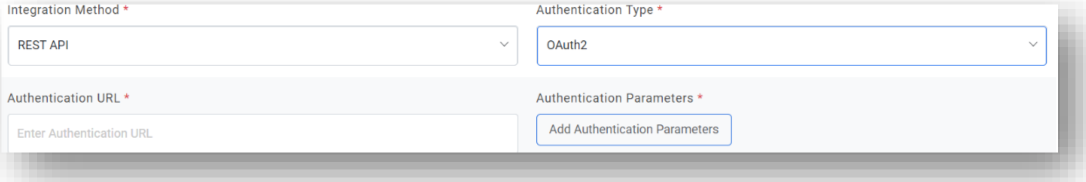
Authentication Type * BasicAuth

Figure 410 - Manage Runbook Tool (Cont.)

7. Select one of the Authentication Type from BasicAuth, OAuth 2.0

– Selection of from **BasicAuth** requires you to enter:
    - User Id
    - Password

– Selection of from **OAuth 2.0** requires you to enter:
    - Authentication URL



Integration Method * REST API

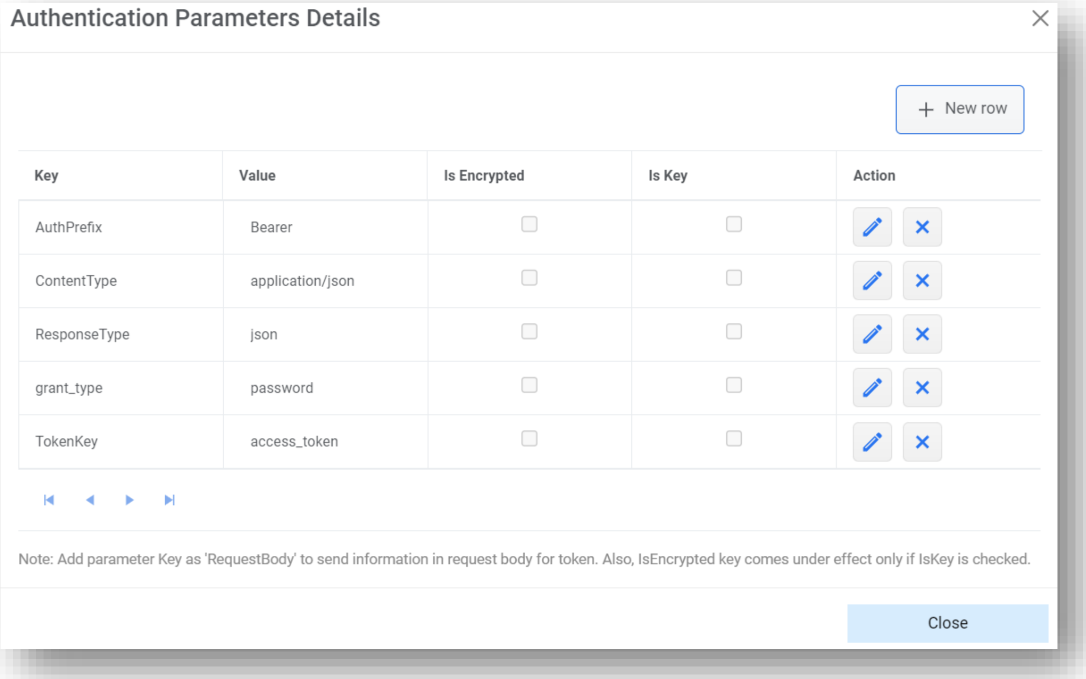
Authentication Type * OAuth2

Authentication URL * Enter Authentication URL

Authentication Parameters *

Figure 411 - Manage Runbook Tool (Cont.)

8. Click **Add Authentication Parameters** to add more parameters, as depicted below -



| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| AuthPrefix | Bearer | □ | □ | ✎ ✕ |
| ContentType | application/json | □ | □ | ✎ ✕ |
| ResponseType | json | □ | □ | ✎ ✕ |
| grant_type | password | □ | □ | ✎ ✕ |
| TokenKey | access_token | □ | □ | ✎ ✕ |

Note: Add parameter Key as 'RequestBody' to send information in request body for token. Also, IsEncrypted key comes under effect only if IsKey is checked.

Figure 412 - Manage Runbook Tool (Cont.)

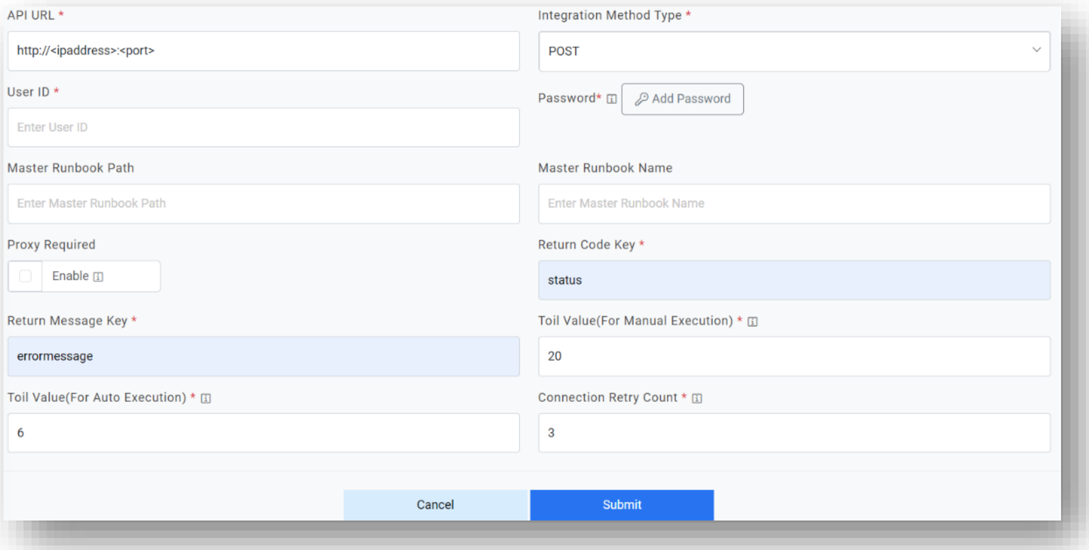
9. Type the URL in the API URL. field.
10. **Sample URL** - http://<URL/IP>:<PORT>/api/26803/run_book_center/
11. Select the Integration Method Type as POST
12. Type the username and password in the **User ID** and **Password** field to get access to API web services.

API URL, User ID, and Password are dependent on the selected integration method.

13. Specify the path to get the consolidated scripts for the execution of runbooks in the **Master Runbook Path** field. This will be provided by respective **Runbook Tool** teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

14. Select **Proxy Required**, if the environment needs access to content from servers outside a firewall.
15. Type the return code key in the **Return Code Key** field to identify the success or failure of runbook execution. E.g. ReturnCode
16. Type the return message key in the **Return Message Key** field to display the success or failure of runbook execution. E.g. ReturnMessage
17. Type the Toil Value (For Manual Execution) which implies the manual execution time of runbook under this tool (in minutes).
18. Type the Toil Value (For Auto Execution) which implies the auto execution time of runbook under this tool (in minutes), if available else it's a non-mandatory field.



API URL *
http://<ipaddress>:<port>

Integration Method Type *
POST

User ID *
Enter User ID

Password* Add Password

Master Runbook Path
Enter Master Runbook Path

Master Runbook Name
Enter Master Runbook Name

Proxy Required
□ Enable

Return Code Key *
status

Return Message Key *
errormessage

Toil Value(For Manual Execution) *
20

Toil Value(For Auto Execution) *
6

Connection Retry Count *
3

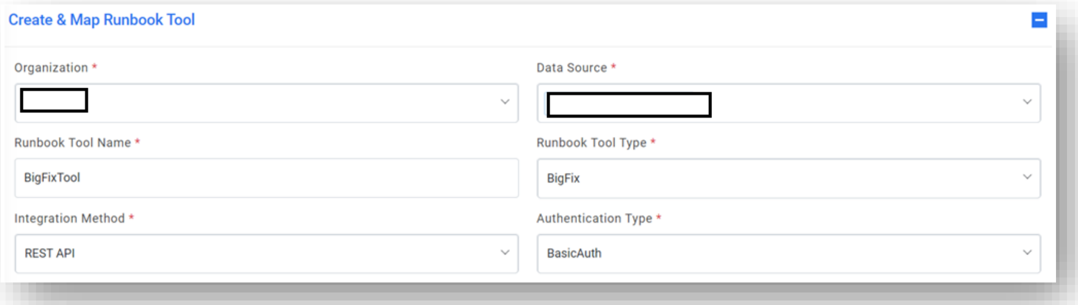
Figure 413 - Manage Runbook Tool (Cont.)

19. Click **Submit** / **Update** for adding a new tool or making changes to an existing tool. An appropriate success message will be displayed.

## 5.7 Integration with BigFiX

To manage / onboard BigFix as the RBA tool, perform the following steps:

1. On the main menu bar, click **Runbooks**, and then click **Manage Runbook Tool**. The **Manage Runbook Tool** appears.
2. Click **Add New** to add a new tool or click ✎ to edit an existing runbook automation tool.
3. Select organization for which you need to create runbook tool in the **Organization Name** field.
4. Type the runbook tool name in the **Runbook Tool Name** field.
5. Select **BigFix** from the **Runbook Tool Type** drop-down.
6. Select **REST** as the integration method for BigFix for the **Integration Method** field.



Create & Map Runbook Tool

Organization *

Data Source *

Runbook Tool Name *
BigFixTool

Runbook Tool Type *
BigFix

Integration Method *
REST API

Authentication Type *
BasicAuth

Figure 414 - Manage Runbook Tool (Cont.)

7. Select one of the Authentication Type from BasicAuth

– Selection of from **BasicAuth** requires you to enter:
   - User Id

- Password

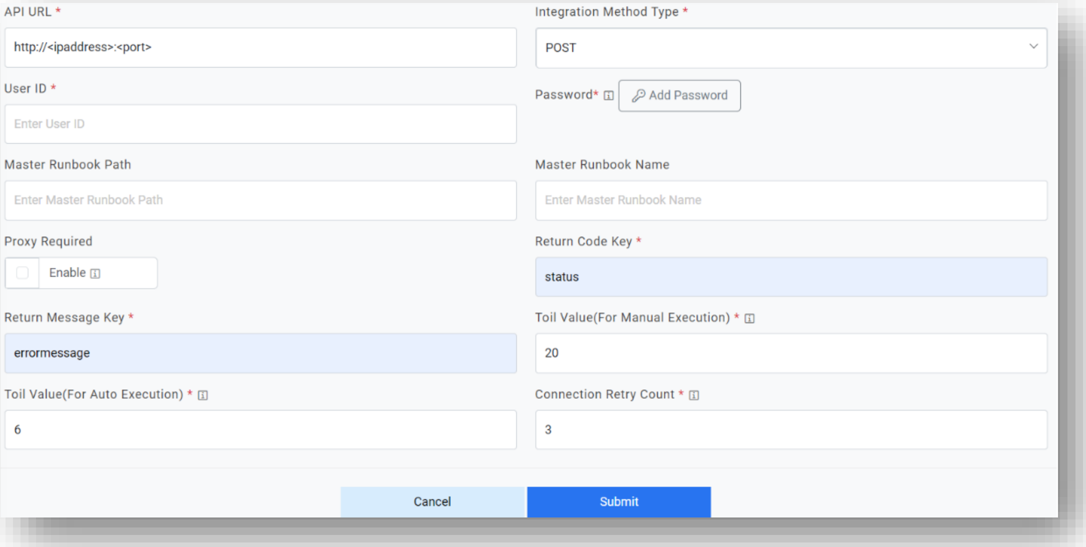
8. Type the URL in the **API URL**. field.
9. **Sample URL** - https://<ip>:<port>
10. Select the Integration Method Type as POST
11. Type the username and password in the **User ID** and **Password** field to get access to API web services.

API URL, User ID, and Password are dependent on the selected integration method

12. Specify the path to get the consolidated scripts for the execution of runbooks in the **Master Runbook Path** field. This will be provided by respective **Runbook Tool** teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

13. Select **Proxy Required**, if the environment needs access to content from servers outside a firewall.
14. Type the return code key in the **Return Code Key** field to identify the success or failure of runbook execution. E.g. status
15. Type the return message key in the **Return Message Key** field to display the success or failure of runbook execution. E.g. result.
16. Type the **Toil Value** (For Manual Execution) which implies the manual execution time of runbook under this tool (in minutes).
17. Type the **Toil Value** (For Auto Execution) which implies the auto execution time of runbook under this tool (in minutes), if available else it's a non-mandatory field.



| | | | |
|---|---|---|---|
| API URL * | http://<ipaddress>:<port> | Integration Method Type * | POST |
| User ID * | Enter User ID | Password* | Add Password |
| Master Runbook Path | Enter Master Runbook Path | Master Runbook Name | Enter Master Runbook Name |
| Proxy Required | □ Enable | Return Code Key * | status |
| Return Message Key * | errormessage | Toil Value(For Manual Execution) * | 20 |
| Toil Value(For Auto Execution) * | 6 | Connection Retry Count * | 3 |

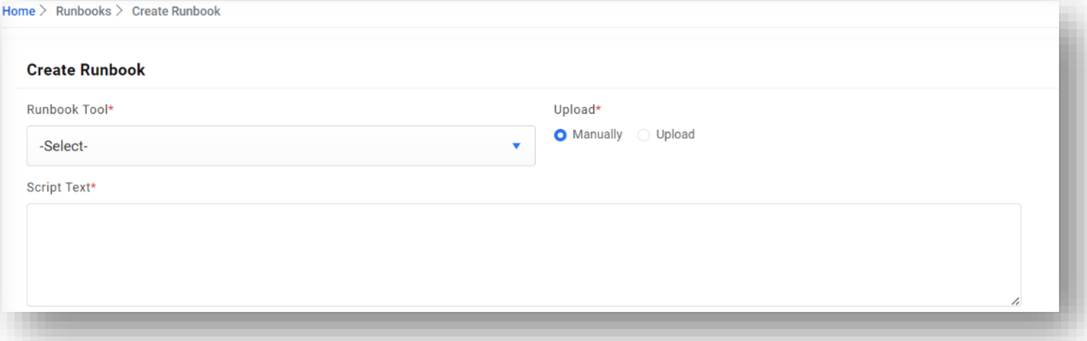
Figure 415 - Manage Runbook Tool (Cont.)

18. Click **Submit** / **Update** for adding a new tool or making changes to an existing tool. An appropriate success message will be displayed.

### 5.7.1 Integration with Bigfix Master Fixlet

To create Bigfix master runbook, perform the following steps:

1. On the left menu bar, click **Runbooks**, then click **Create Runbook**. The **Create Runbook** page appears.
2. Select **Runbook Tool**, the tool against which master runbook has to be created.
3. Either **Upload** or type **Script Text**, file has to be uploaded which are of extensions .ps1/.bat/.py/.sh.



Home > Runbooks > Create Runbook

Create Runbook

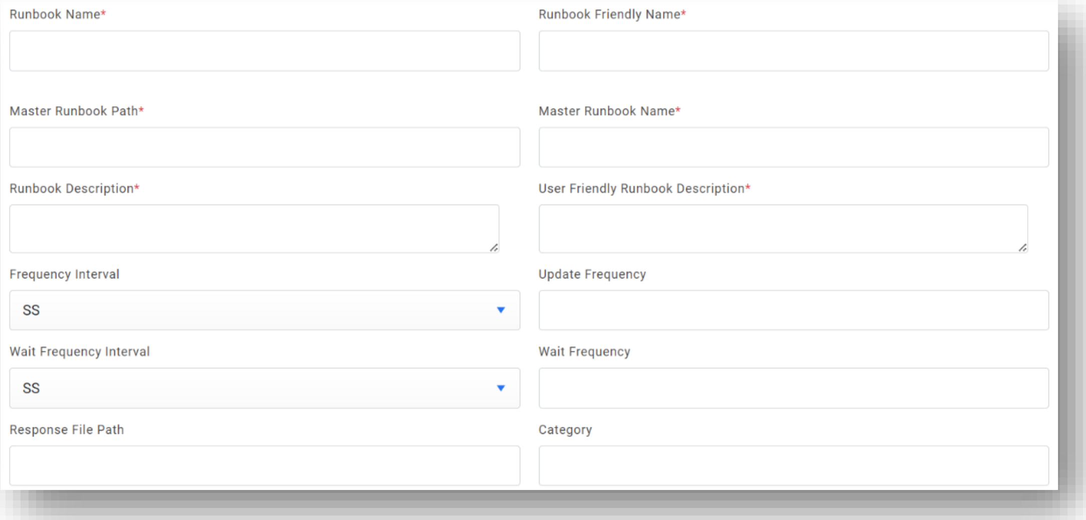
Runbook Tool* -Select-

Upload* ● Manually ○ Upload

Script Text*

Figure 416 - Create Runbook

4. Type the name of the runbook in **Runbook Name** field.
5. Add runbook path in the field **Master Runbook Path**. Although in case of BigFix, this can be given any value, since BigFix integration is independent of runbook path.
6. Type the value of master fixlet ID in the field **Master Runbook Name**.
7. Add the path of 'error_folder' in the field **Response File Path**. While creation of Bigfix Master Runbook, this field is mandatory.



Runbook Name*

Runbook Friendly Name*

Master Runbook Path*

Master Runbook Name*

Runbook Description*

User Friendly Runbook Description*

Frequency Interval SS

Update Frequency

Wait Frequency Interval SS

Wait Frequency

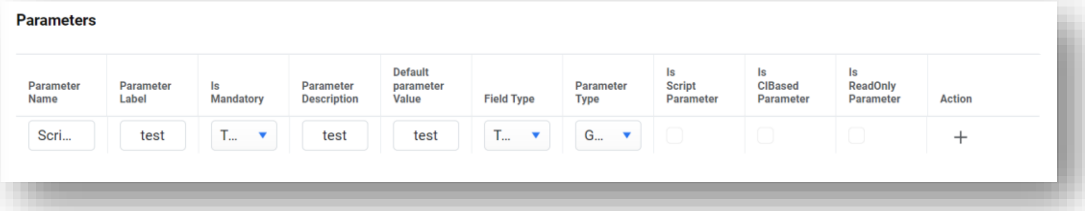
Response File Path

Category

Figure 417 - Create Runbook (Cont.)

8. Add the following Parameter Names in the parameter grid:
9. **ScriptPath** – The default parameter value consists of the shared path.
10. **ScriptType** – The default parameter value consists of the type of script uploaded.
11. **Hostname** – The default parameter value consists of the target server on which script is being executed.
12. **Fixletid** - The default parameter value consists of the value of the ID of child fixlet executed.
13. **Computername** – The default parameter consists of the value of the master server or the root server.

14. **TicketNumber** – The default parameter consists of the static value 'TicketNumber' and it is mapped with TicketNumber in Parameter Type.
15. **TenantID** – The default parameter consists of the static value 'TenantID' and it is mapped with TenantID in Parameter Type.
16. **Param1** – The default parameter consists of the parameter value user wants to add in. If user wants to add multiple parameters, those are also added in the similar manner like param1. Furthermore, it needs to be checked in for 'IsScript Parameter'.



Parameters

| Parameter Name | Parameter Label | Is Mandatory | Parameter Description | Default parameter Value | Field Type | Parameter Type | Is Script Parameter | Is CIBased Parameter | Is ReadOnly Parameter | Action |
|---|---|---|---|---|---|---|---|---|---|---|
| Scri... | test | T... ▾ | test | test | T... ▾ | G... ▾ | □ | □ | □ | + |

Figure 418 - Parameter grid in Create Runbook for ScriptType Powershell

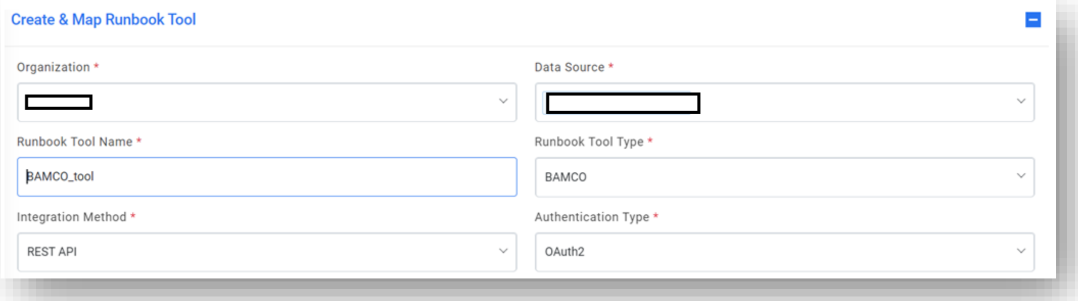
17. Select 'Save' button after adding all the details for the master runbook.

The master runbook created on 'Create Runbook' will be visible in Manage Runbooks. (On main menu, go to Runbooks and select manage runbooks.

## 5.8 Integration with BMCAO

To manage / onboard BMCAO as the RBA tool, perform the following steps:

1. On the main menu bar, click **Runbooks**, and then click **Manage Runbook Tool**. The **Manage Runbook Tool** appears.
2. Click **Add New** to add a new tool or click ✎ to edit an existing runbook automation tool.
3. Select organization for which you need to create runbook tool in the **Organization Name** field.
4. Type the runbook tool name in the **Runbook Tool Name** field.
5. Select **BMCAO** from the **Runbook Tool Type** drop-down
6. Select **REST** as the integration method for BMCAO for the **Integration Method** field.



Create & Map Runbook Tool

Organization *

Data Source *

Runbook Tool Name * BAMCO_tool

Runbook Tool Type * BAMCO

Integration Method * REST API

Authentication Type * OAuth2

Figure 419 - Manage Runbook Tool (Cont.)

7. Select one of the Authentication Type from BasicAuth, OAuth 2.0

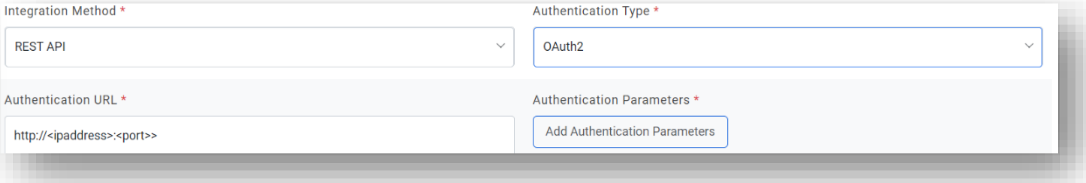
– Selection of from **BasicAuth** requires you to enter:

  - User Id

- Password

- Selection of from **OAuth 2.0** requires you to enter:
  - User Id
  - Password
  - Authentication URL
  - Client Secret

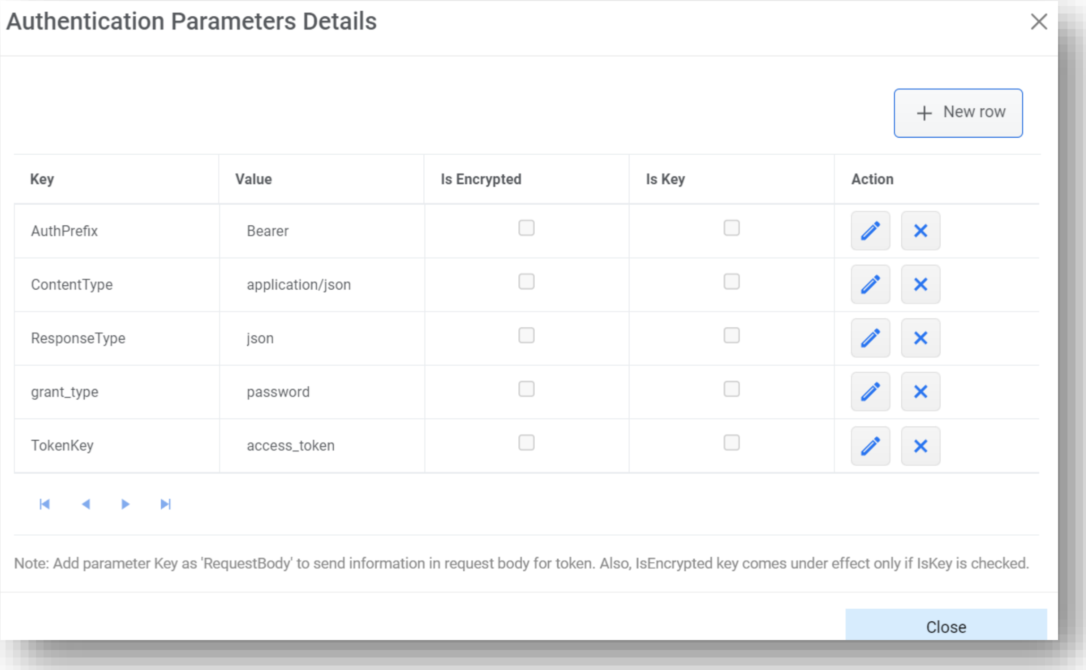
8. Type the URL in the **API URL**. field.
9. Sample URL – http://MyHost:MyPort



| | |
|---|---|
| Integration Method * | Authentication Type * |
| REST API | OAuth2 |
| Authentication URL * | Authentication Parameters * |
| http://<ipaddress>:<port>> | Add Authentication Parameters |

Figure 420 - Manage Runbook Tool (Cont.)

10. Click on **Edit Authentication Parameters** if authentication type is OAuth2 and provide below details:



Authentication Parameters Details

| Key | Value | Is Encrypted | Is Key | Action |
|---|---|---|---|---|
| AuthPrefix | Bearer | □ | □ | |
| ContentType | application/json | □ | □ | |
| ResponseType | json | □ | □ | |
| grant_type | password | □ | □ | |
| TokenKey | access_token | □ | □ | |

Note: Add parameter Key as 'RequestBody' to send information in request body for token. Also, IsEncrypted key comes under effect only if IsKey is checked.

Figure 421 - Manage Runbook Tool (Cont.)

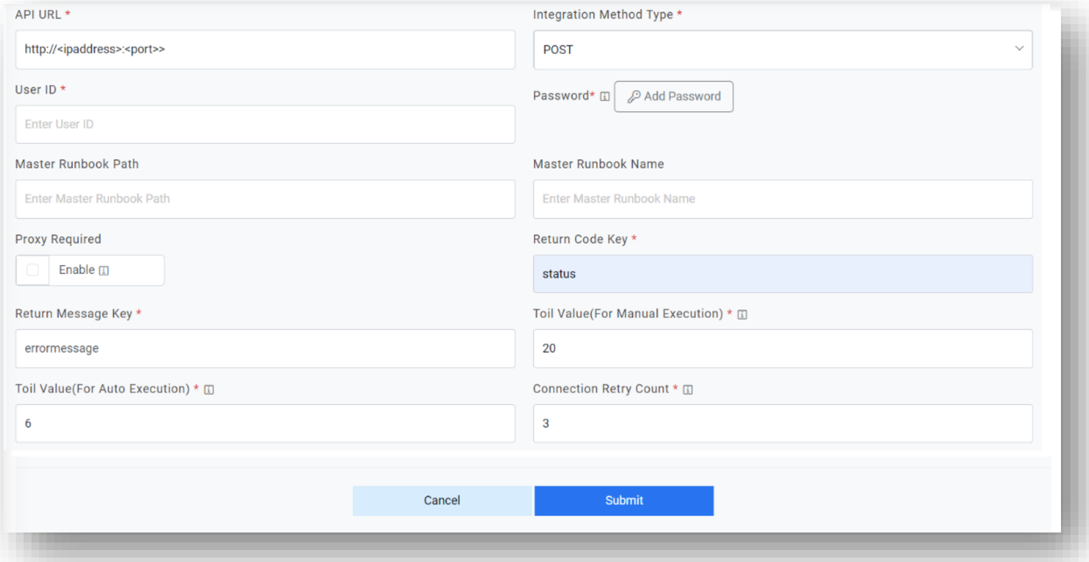
11. Select 'POST' in the **Integration Method Type** field.
12. Type the username and password in the **User ID** and **Password** field to get access to API web services.

API URL, User ID, and Password are dependent on the selected integration method

13. Specify the path to get the consolidated scripts for the execution of runbooks in the Master Runbook Path field. This will be provided by respective Runbook Tool teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

14. Select **Proxy Required**, if the environment needs access to content from servers outside a firewall.
15. Type the return code key in the **Return Code Key** field to identify the success or failure of runbook execution. E.g. status_codes
16. Type the return message key in the **Return Message Key** field to display the success or failure of runbook execution. E.g. iautomate_success
17. Type the Toil Value (For Manual Execution) which implies the manual execution time of runbook under this tool (in minutes).
18. Type the Toil Value (For Auto Execution) which implies the auto execution time of runbook under this tool (in minutes), if available else it's a non-mandatory field.



API URL * http://<ipaddress>:<port>>

Integration Method Type * POST

User ID * Enter User ID

Password* Add Password

Master Runbook Path: Enter Master Runbook Path

Master Runbook Name: Enter Master Runbook Name

Proxy Required: □ Enable

Return Code Key * status

Return Message Key * errormessage

Toil Value(For Manual Execution) * 20

Toil Value(For Auto Execution) * 6

Connection Retry Count * 3

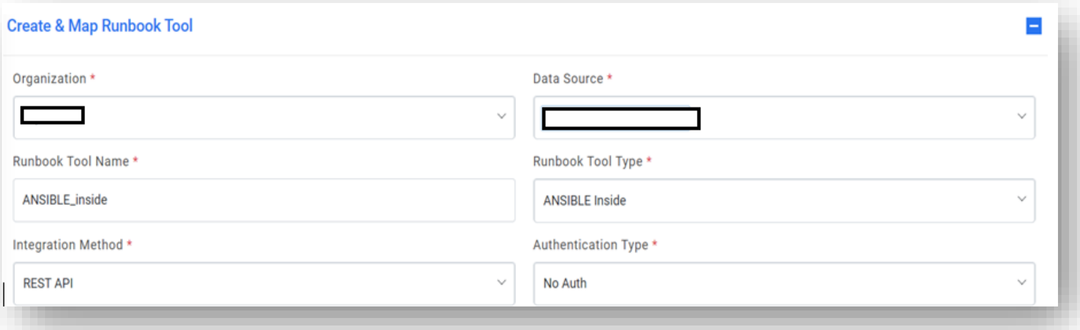
Figure 422 - Manage Runbook Tool (Cont.)

19. Click **Submit** / **Update** for adding a new tool or making changes to an existing tool. An appropriate success message will be displayed.

## 5.9 Integration with ANSIBLE Inside

To manage / onboard Ansible Inside as the RBA tool, perform the following steps:

1. On the left menu bar, click **Runbooks**, and then click **Manage Runbook Tool**. The **Manage Runbook Tool** appears.
2. Click **Add New** to add a new tool or click ✎ to edit an existing runbook automation tool.
3. Select Organization for which you need to create runbook tool in the **Organization** Name field.
4. Type the runbook tool name in the **Runbook Tool Name** field.
5. Select ANSIBLE Inside from the **Runbook Tool Type** drop-down.
6. Select REST API as the integration method for ANSIBLE Inside for the **Integration Method** field.



Create & Map Runbook Tool

Organization * [redacted]

Data Source * [redacted]

Runbook Tool Name * ANSIBLE_inside

Runbook Tool Type * ANSIBLE Inside

Integration Method * REST API

Authentication Type * No Auth

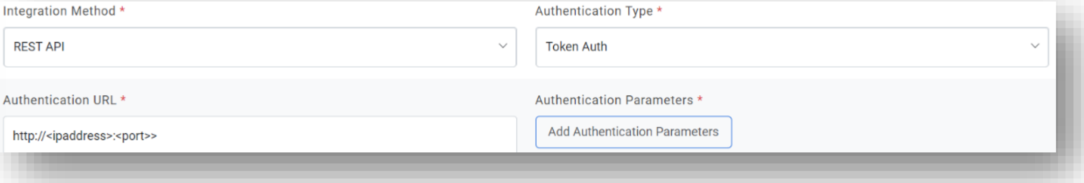
Figure 423 - Manage Runbook Tool (Cont.)

7. Select one of the **Authentication Type** from No Auth, Certificate Based Auth, or Token Auth.

– Selection of No Auth, Certificate Based Auth as **Authentication Type** will not require any user id or password.

– Selection of Token Auth as **Authentication Type** requires you to enter –

  - **Authentication URL**: URL of iAutomate API to generate token.



Integration Method * REST API

Authentication Type * Token Auth

Authentication URL * http://<ipaddress>:<port>>

Authentication Parameters * Add Authentication Parameters

Figure 424 - Manage Runbook Tool (Cont.)

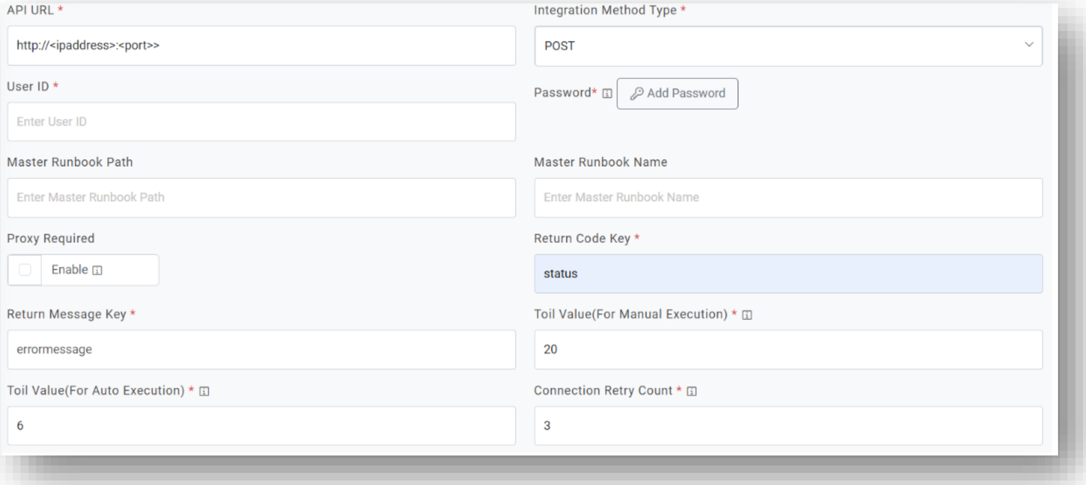
8. Provide the **API URL** of the tool.
9. Select 'POST' in the **Integration Method Type** field.

API URL is dependent on the selected integration method.

10. Specify the path to get the consolidated scripts for the execution of runbooks in the Master Runbook Path field. This will be provided by respective Runbook Tool teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

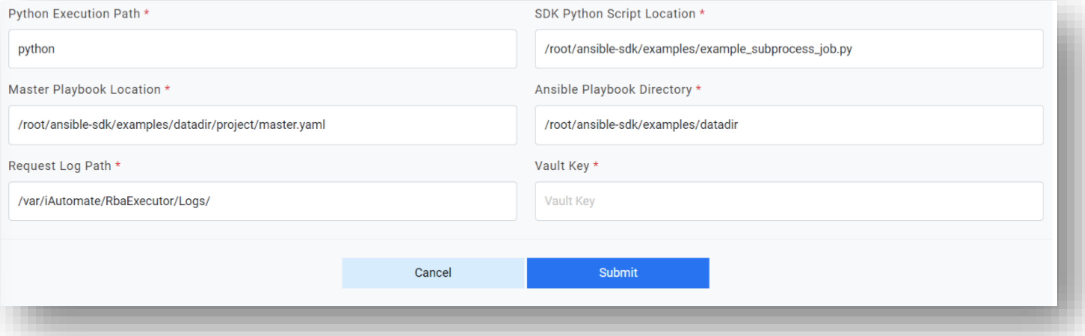
11. Select **Proxy Required**, if the environment needs access to content from servers outside a firewall.
12. Type the return code key in the **Return Code Key** field to identify the success or failure of runbook execution. E.g. status_codes
13. Type the return message key in the **Return Message Key** field to display the success or failure of runbook execution. E.g. iautomate_success
14. Type the **Toil Value** (For Manual Execution) which implies the manual execution time of runbook under this tool (in minutes).
15. Type the **Toil Value** (For Auto Execution) which implies the auto execution time of runbook under this tool (in minutes), if available else it's a non-mandatory field.



| | |
|---|---|
| API URL * | Integration Method Type * |
| http://<ipaddress>:<port>> | POST |
| User ID * | Password* Add Password |
| Enter User ID | |
| Master Runbook Path | Master Runbook Name |
| Enter Master Runbook Path | Enter Master Runbook Name |
| Proxy Required | Return Code Key * |
| □ Enable | status |
| Return Message Key * | Toil Value(For Manual Execution) * |
| errormessage | 20 |
| Toil Value(For Auto Execution) * | Connection Retry Count * |
| 6 | 3 |

Figure 425 - Manage Runbook Tool (Cont.)

16. Type the **Python Execution Path** which will remain constant as 'python'.
17. Type the **SDK Python Script Location** which is the Location of the core python script of Ansible-Inside that manages the execution of Ansible roles.
18. Type the **Master Playbook Location** which is the Location that will host the master yaml within the Ansible-Inside setup.
19. Type the **Ansible Playbook Directory** which is the Location that will host all the ansible roles to be executed within the Ansible -Inside setup.
20. Type the **Request Log Path** which is the location where all log files will be created.
21. Type the **Vault Key** which is used to securely access the tool.



| | |
|---|---|
| Python Execution Path * | SDK Python Script Location * |
| python | /root/ansible-sdk/examples/example_subprocess_job.py |
| Master Playbook Location * | Ansible Playbook Directory * |
| /root/ansible-sdk/examples/datadir/project/master.yaml | /root/ansible-sdk/examples/datadir |
| Request Log Path * | Vault Key * |
| /var/iAutomate/RbaExecutor/Logs/ | Vault Key |

Figure 426 - Manage Runbook Tool (Cont.)

22. Click **Submit / Update** for adding a new tool or making changes to an existing tool. An appropriate success message will be displayed.

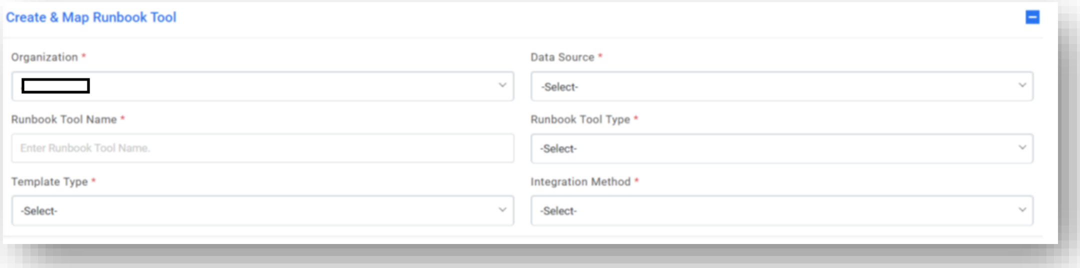
The fields SDK Python Script Location, Master Playbook Location, Ansible Playbook Directory and Vault Key are also given at runbook level for tool type ANSIBLE SDK. By default, for a runbook, these values will be populated from the corresponding tool, but if there's any case that these values differ at runbook level then user can define them accordingly at runbook level.

To get the logs of the tickets executed for the tool Ansible Inside, login with org admin and navigate to **Reports→ Ansible Inside Logs** page. Here all the tickets executed under Ansible Inside tool will be populated. Corresponding to each ticket, user can download the component log and the console logs. For detailed information, refer **HCL iAutomate 6.4.1 Configuration Guide.**

## 5.10 Integration with Jenkins

To manage / onboard Jenkins as the RBA tool, perform the following steps:

1. On the main menu bar, click Runbooks, and then click Manage Runbook Tool. The Manage Runbook Tool appears.
2. Click Add New to add a new tool or click to edit existing runbook automation tool.
3. Select an organization for which you need to create a runbook tool in the Organization Name field.
4. Type the runbook tool name in the Runbook Tool Name field.
5. Select Jenkins from the Runbook tool type drop-down.
6. Select REST as integration method for Jenkins for the integration method field.



Create & Map Runbook Tool

Organization *

Data Source * -Select-

Runbook Tool Name * Enter Runbook Tool Name.

Runbook Tool Type * -Select-

Template Type * -Select-

Integration Method * -Select-

Figure 427 - Manage Runbook Tool

7. Select one of the Authentication Type from BasicAuth,

– Selection of from BasicAuth requires you to enter.

- User Id
- Password

8. Type the URL in the API URL field, sample: http://<ippaddress>:<port>
9. Select the integration method type as POST.
10. Type the username and password in the User Id and Password field to get access to API web service.

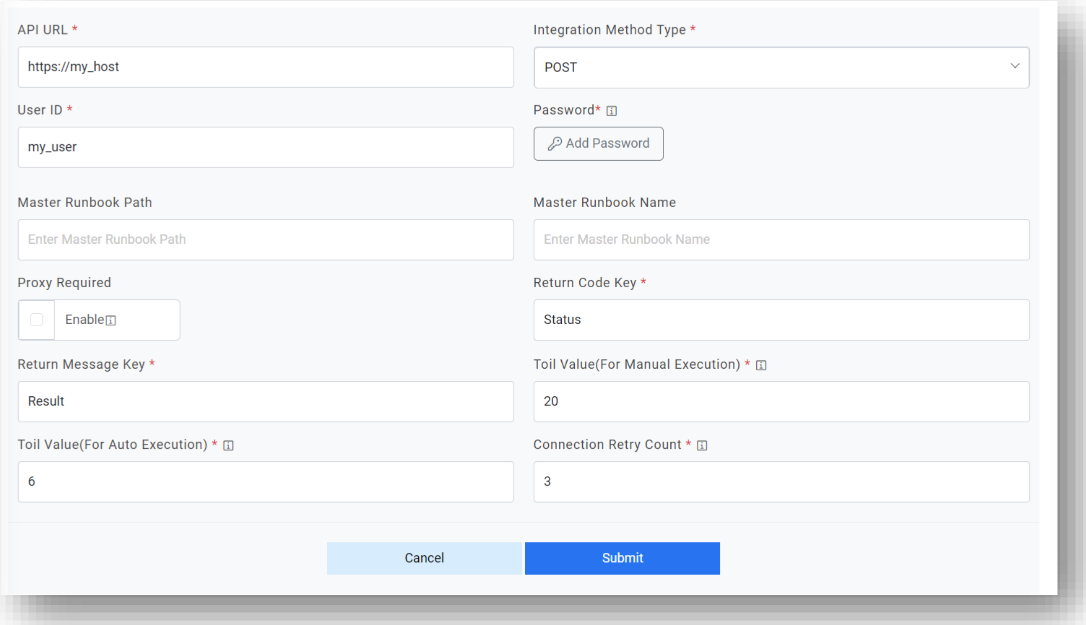
API URL, User ID, and Password are dependent on the selected integration method

11. Specify the path to get consolidate script for the execution of runbook in the master Runbook path field. This will be provided by respective runbook tool teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

12. Select Proxy required, if the environment needs access to contact from the server outside a firewall.
13. Type the return code in the return code Key field to identify the success or failure of runbook execution. E.g. status
14. Type the return message key in the return message key field to display the success or failure of runbook execution. E.g. result
15. Type the Toil Value (for manual execution) which implies the manual execution time of runbook under this tool (in minute)

16. Type the Toil Value (for auto execution) which implies the manual execution time of runbook under this tool (in minute) if available else it's a non- mandatory field.



API URL *: https://my_host

Integration Method Type *: POST

User ID *: my_user

Password*

Master Runbook Path: Enter Master Runbook Path

Master Runbook Name: Enter Master Runbook Name

Proxy Required: □ Enable

Return Code Key *: Status

Return Message Key *: Result

Toil Value(For Manual Execution) *: 20

Toil Value(For Auto Execution) *: 6

Connection Retry Count *: 3

Figure 428 - Manage Runbook Tool (Cont.)

17. Click Submit/Update for adding a new tool or making changes in an existing tool. An appropriate success message will be displayed.
18. Type the Connection Retry Count which implies the no. of time connection would be made with RBA tool server in case of connection failure.

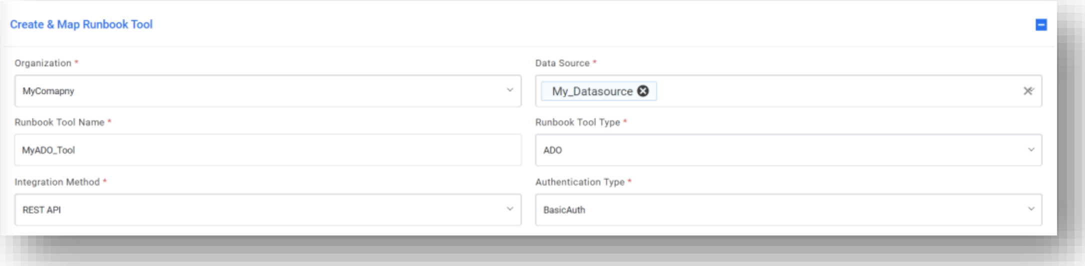
    Note: The URL of the Jenkins should be of below format:

    http://<Host_Server>:<Port>/job/<Collection>/job/<ProjectName>/

## 5.11 Integration with ADO

To manage / onboard ADO as the RBA tool, perform the following steps:

1. On the main menu bar, click Runbooks, and then click Manage Runbook Tool. The Manage Runbook Tool appears.
2. Click Add New to add a new tool or click to edit existing runbook automation tool.
3. Select an organization for which you need to create a runbook tool in the Organization Name field.
4. Type the runbook tool name in the Runbook Tool Name field.
5. Select ADO from the Runbook tool type drop-down.
6. Select REST API as integration method for ADO for the integration method field.



Create & Map Runbook Tool

Organization *: MyComapny

Data Source *: My_Datasource

Runbook Tool Name *: MyADO_Tool

Runbook Tool Type *: ADO

Integration Method *: REST API

Authentication Type *: BasicAuth

Figure 429 - Manage Runbook Tool

7. Select one of the Authentication Type from BasicAuth,

– Selection of from BasicAuth requires you to enter.

  - User Id
  - Password

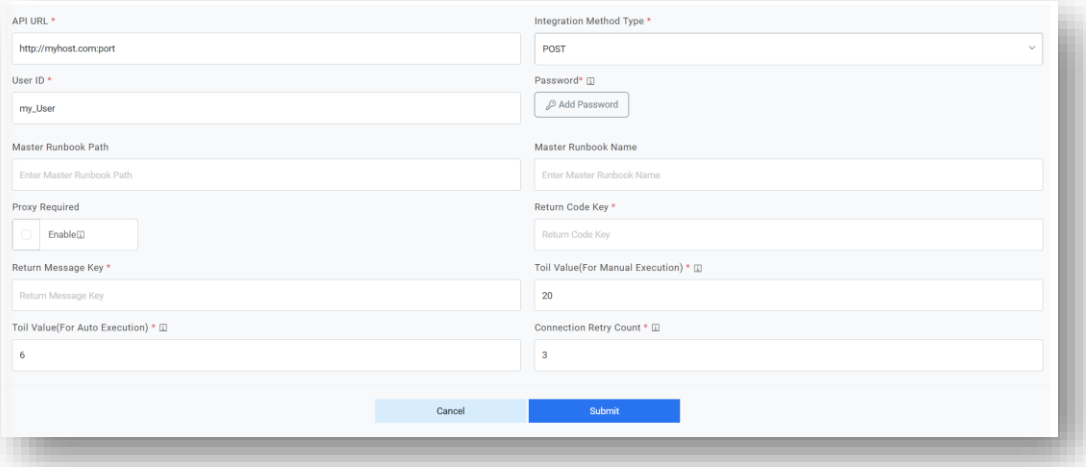
8. Type the URL in the API URL field, sample: http://<ippaddress>:<port>
9. Select the integration method type as POST.
10. Type the username and password in the User Id and Password field to get access to API web service.

API URL, User ID, and Password are dependent on the selected integration method

11. Specify the path to get consolidate script for the execution of runbook in the master Runbook path field. This will be provided by respective runbook tool teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

12. Select Proxy required, if the environment needs access to contact from the server outside a firewall.
13. Type the return code in the return code Key field to identify the success or failure of runbook execution. **E.g.** status
14. Type the return message key in the return message key field to display the success or failure of runbook execution. **E.g.** result
15. Type the Toil Value (for manual execution) which implies the manual execution time of runbook under this tool (in minute)
16. Type the Toil Value (for auto execution) which implies the manual execution time of runbook under this tool (in minute) if available else it's a non- mandatory field.
17. Type the Connection Retry Count which implies the no. of time connection would be made with RBA tool server in case of connection failure.



API URL *
http://myhost.com:port

Integration Method Type *
POST

User ID *
my_User

Password*
Add Password

Master Runbook Path
Enter Master Runbook Path

Master Runbook Name
Enter Master Runbook Name

Proxy Required
□ Enable

Return Code Key *
Return Code Key

Return Message Key *
Return Message Key

Toil Value(For Manual Execution) *
20

Toil Value(For Auto Execution) *
6

Connection Retry Count *
3

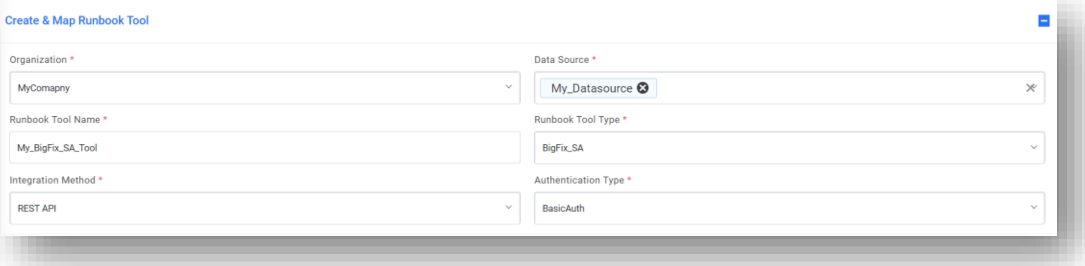
Figure 430 - Manage Runbook Tool (Cont.)

18. Click Submit/Update for adding a new tool or making changes in an existing tool. An appropriate success message will be displayed.

## 5.12 Integration with BigFix_SA

To manage / onboard BigFix_SA as the RBA tool, perform the following steps:

1. On the main menu bar, click Runbooks, and then click Manage Runbook Tool. The Manage Runbook Tool appears.
2. Click Add New to add a new tool or click to edit existing runbook automation tool.
3. Select an organization for which you need to create a runbook tool in the Organization Name field.
4. Type the runbook tool name in the Runbook Tool Name field.
5. Select BigFix_SA from the Runbook tool type drop-down.
6. Select REST API as integration method for BigFix_SA for the integration method field.



Create & Map Runbook Tool

Organization *
MyComapny

Data Source *
My_Datasource

Runbook Tool Name *
My_BigFix_SA_Tool

Runbook Tool Type *
BigFix_SA

Integration Method *
REST API

Authentication Type *
BasicAuth

Figure 431 - Manage Runbook Tool

7. Select one of the Authentication Type from BasicAuth,

– Selection of from BasicAuth requires you to enter.

  - User Id
  - Password

8. Type the URL in the API URL field, sample: http://<ippaddress>:<port>
9. Type the Platform API URL of Bigfix in the Platform API URL field, sample: https://<ippaddress>:<port>
10. Select the integration method type as POST.

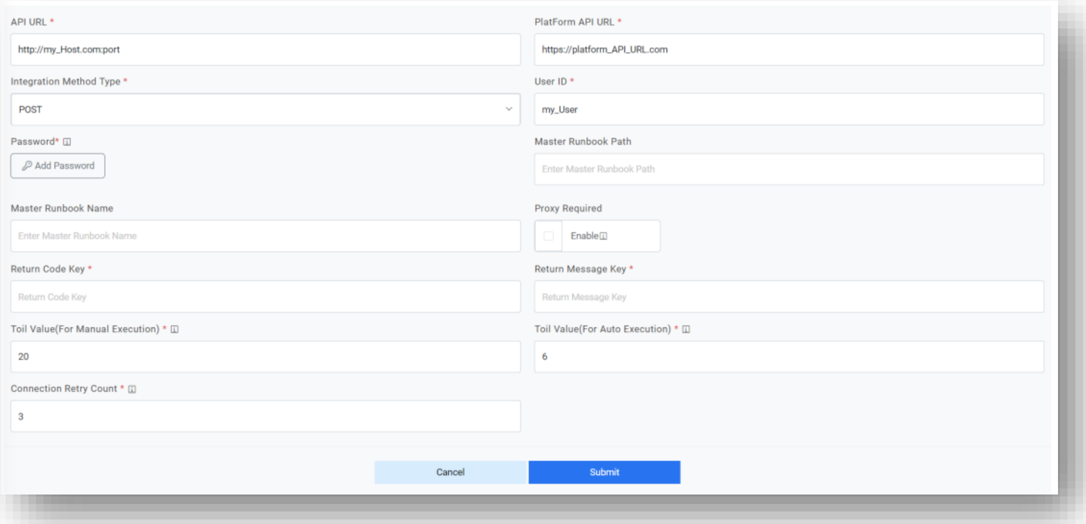
11. Type the username and password in the User Id and Password field to get access to API web service.

API URL, User ID, and Password are dependent on the selected integration method

12. Specify the path to get consolidate script for the execution of runbook in the master Runbook path field. This will be provided by respective runbook tool teams if they have a master runbook.

This is not a mandatory field. Users can change and run these scripts any time.

13. Select Proxy required, if the environment needs access to contact from the server outside a firewall.
14. Type the return code in the return code Key field to identify the success or failure of runbook execution. E.g. status
15. Type the return message key in the return message key field to display the success or failure of runbook execution. E.g. result
16. Type the Toil Value (for manual execution) which implies the manual execution time of runbook under this tool (in minute)
17. Type the Toil Value (for auto execution) which implies the manual execution time of runbook under this tool (in minute) if available else it's a non- mandatory field.
18. Type the Connection Retry Count which implies the no. of time connection would be made with RBA tool server in case of connection failure.



API URL *
http://my_Host.com:port

PlatForm API URL *
https://platform_API_URL.com

Integration Method Type *
POST

User ID *
my_User

Password* ◫

Master Runbook Path
Enter Master Runbook Path

Master Runbook Name
Enter Master Runbook Name

Proxy Required
□ Enable◫

Return Code Key *
Return Code Key

Return Message Key *
Return Message Key

Toil Value(For Manual Execution) * ◫
20

Toil Value(For Auto Execution) * ◫
6

Connection Retry Count * ◫
3

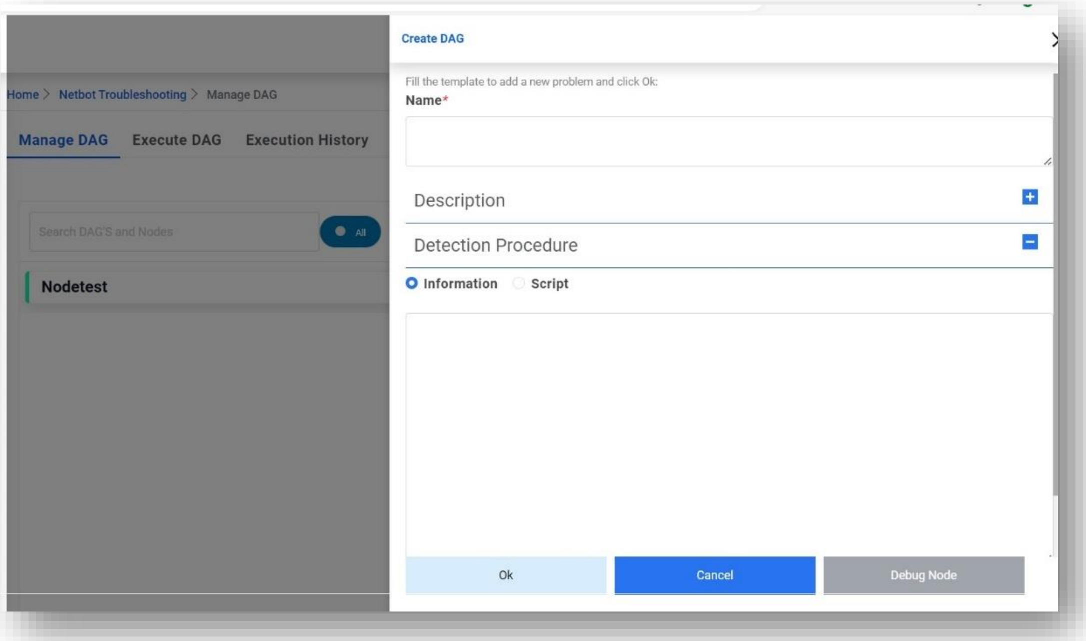
Figure 432 - Manage Runbook Tool (Cont.)

19. Click Submit/Update for adding a new tool or making changes in an existing tool. An appropriate success message will be displayed.

# 6 DAG

## 6.1 Creation of a Node

1. **Information Node**: Information node contains information and does not execute. Go to **Netbot Troubleshooting→Manage DAG**. Click on **Add**. Select the **Detection Procedure** as 'information' and fill in all the details. Click **Ok**.



Create DAG

Fill the template to add a new problem and click Ok:

Name*

Description

Detection Procedure

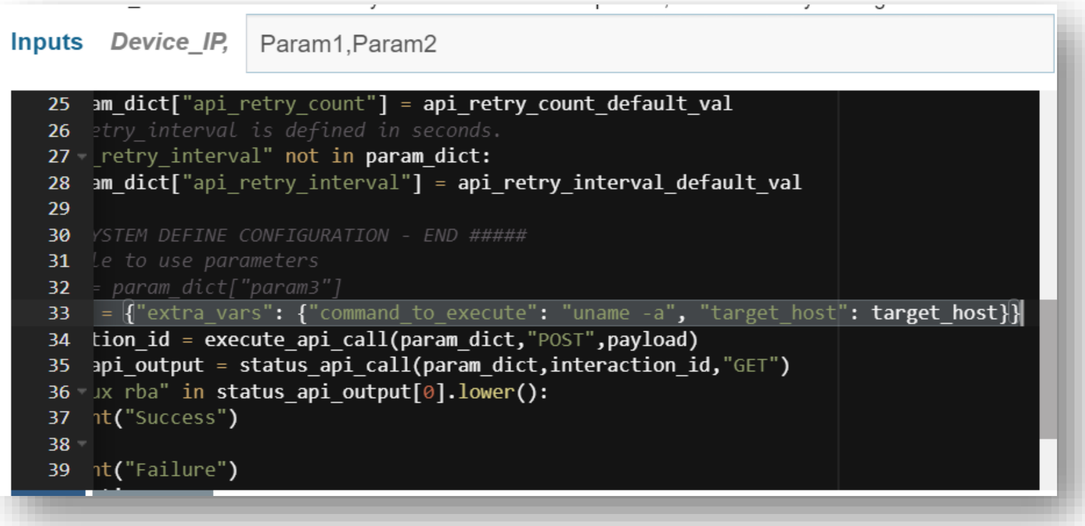
● Information ○ Script

Figure 433 - DAG - Create Information Node

2. **Script Node**: Script node contains python script and related parameters that execute by API. Go to **Netbot Troubleshooting→Manage DAG**. Click on **Add**. Select the **Detection Procedure** as 'Script' and fill in all the details. By default, a script is populated on the screen. If the command to be executed needs to be changed, then in the already defined script, search for "command_to_execute" and update its value to the command that you want to execute.

   Eg: payload = {"extra_vars": {"command_to_execute": "<Command to execute>", "target_host": target_host}}

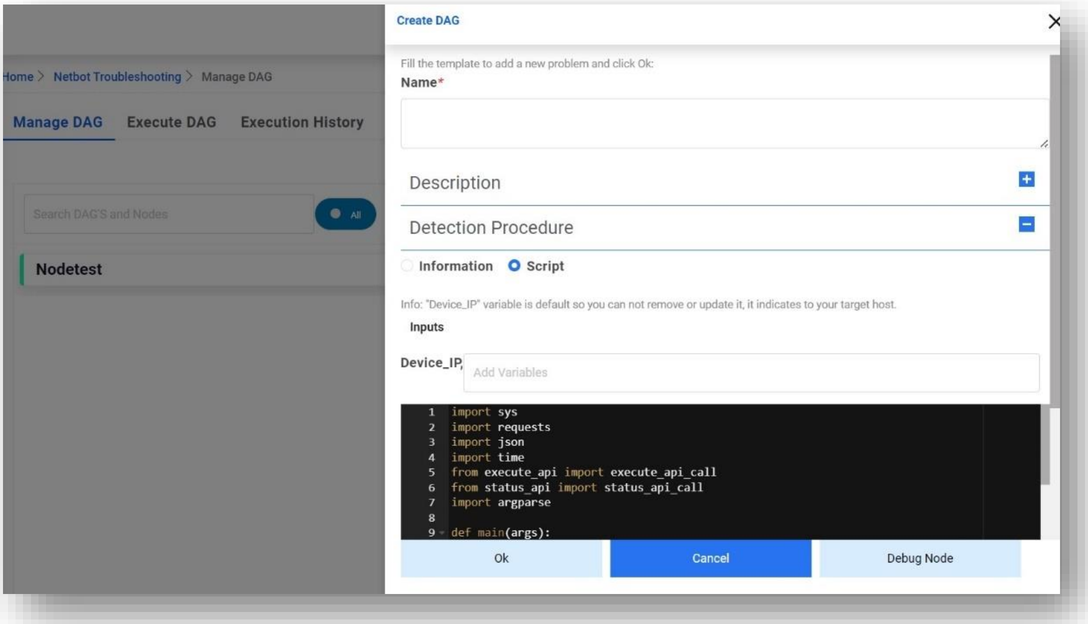
   If the command needs to have some parameters, then define those parameters (comma separated) under 'Inputs' field along with *Device_IP.*



Inputs Device_IP, Param1,Param2

```
am_dict["api_retry_count"] = api_retry_count_default_val
etry_interval is defined in seconds.
_retry_interval" not in param_dict:
am_dict["api_retry_interval"] = api_retry_interval_default_val

YSTEM DEFINE CONFIGURATION - END #####
le to use parameters
= param_dict["param3"]
 = {"extra_vars": {"command_to_execute": "uname -a", "target_host": target_host}}
tion_id = execute_api_call(param_dict,"POST",payload)
api_output = status_api_call(param_dict,interaction_id,"GET")
ux rba" in status_api_output[0].lower():
nt("Success")

nt("Failure")
```

Figure 434 - DAG - Define Parameters



Create DAG

Fill the template to add a new problem and click Ok:

Name*

Description

Detection Procedure

○ Information ● Script

Info: "Device_IP" variable is default so you can not remove or update it, it indicates to your target host.

Inputs

Device_IP, Add Variables

```
import sys
import requests
import json
import time
from execute_api import execute_api_call
from status_api import status_api_call
import argparse

def main(args):
```

Figure 435 - DAG - Create Script Node

## 6.2 Creation of a DAG

A DAG is a node don't have any parent and have one or child(s). If Dag mapped a child, parent will be treated as Dag and the DAG that mapped as child will be treated as Node. Dag are identified by in purple, and Node are in light green color.

- To create a DAG, create two or more nodes, click on a node, and link it to another one to identify the hierarchy of execution of nodes.

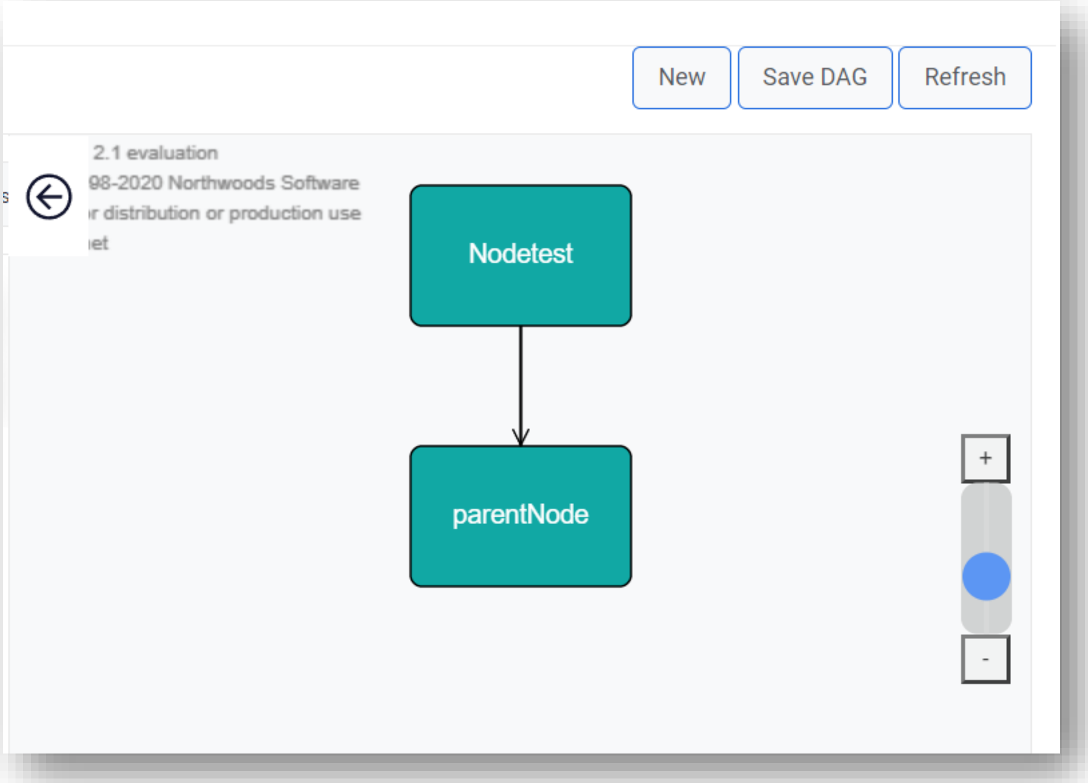


Figure 436 - Creation of a DAG

- You can also define the linking by clicking on the node, edit it and under 'Add Child Node' section, select the node that you want to link it with.
- After all node linking is defined, click on 'Save DAG'.

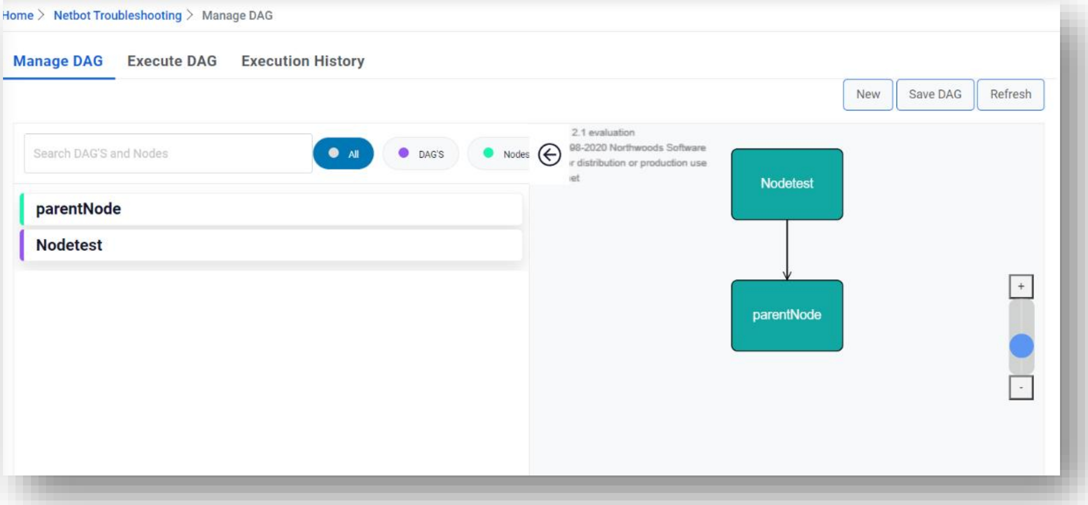


Figure 437 - Creation of a DAG (Cont.)

## 6.3 Execution of a DAG

To execute a Dag,

1. Go to Reports → Netbot Troubleshooting → Execute DAG.
2. Search for the DAG to be executed and click on **Execute** button for the node to be executed.

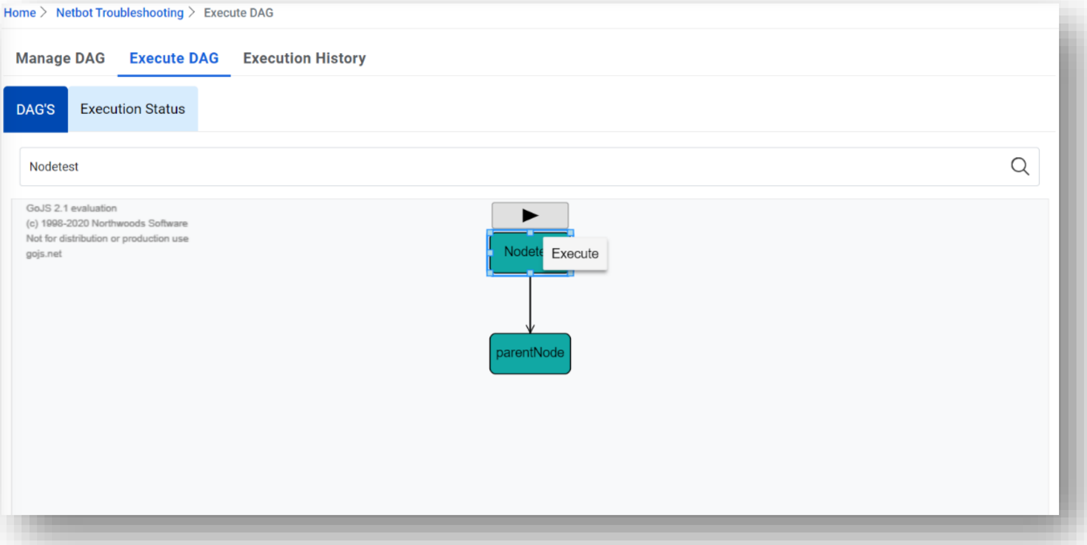
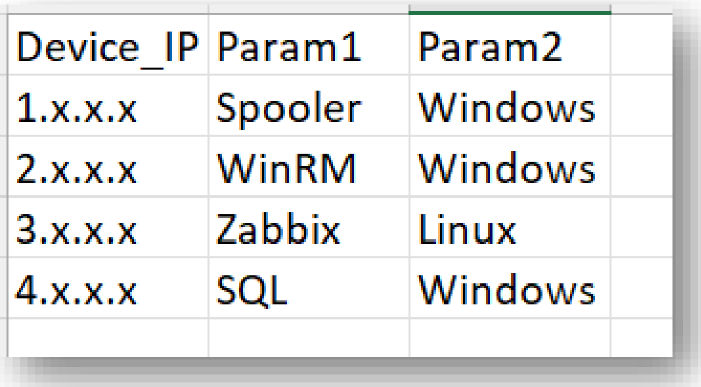
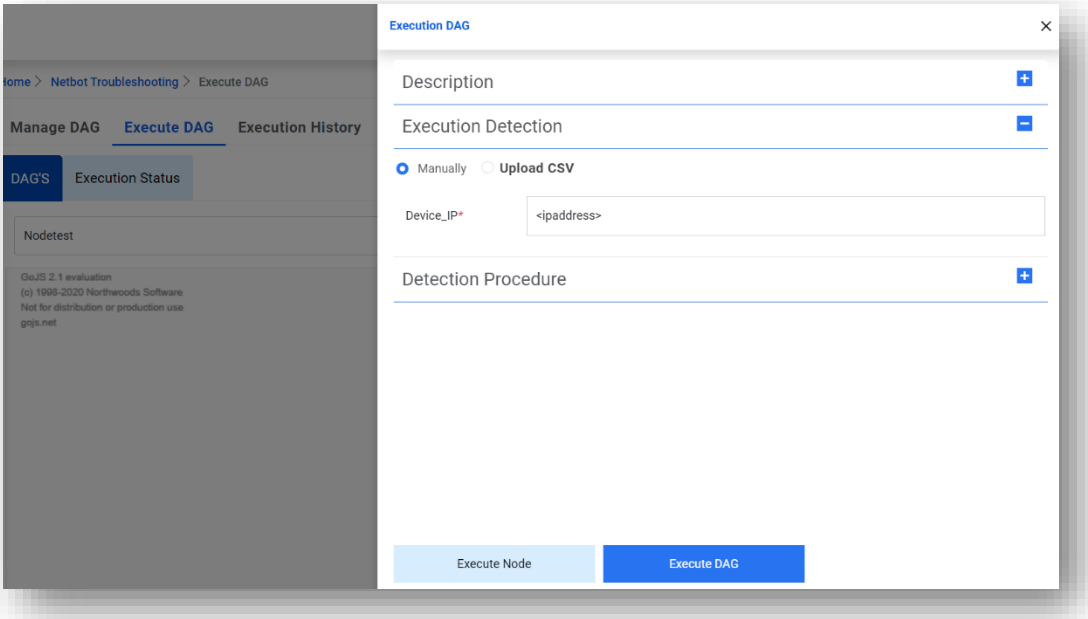


Figure 438 - Execution of a DAG

3. If the parameters were mentioned during the node creation, then it will ask the parameter value on execute button click. Provide those parameter values. There are two ways to pass the parameters:
   - Either pass the value of those parameters manually
   - Define the parameters in the CSV and upload it.
4. The benefit of this is that multiple values can be passed for a parameter. For that, download the template and define the value of parameters as below:



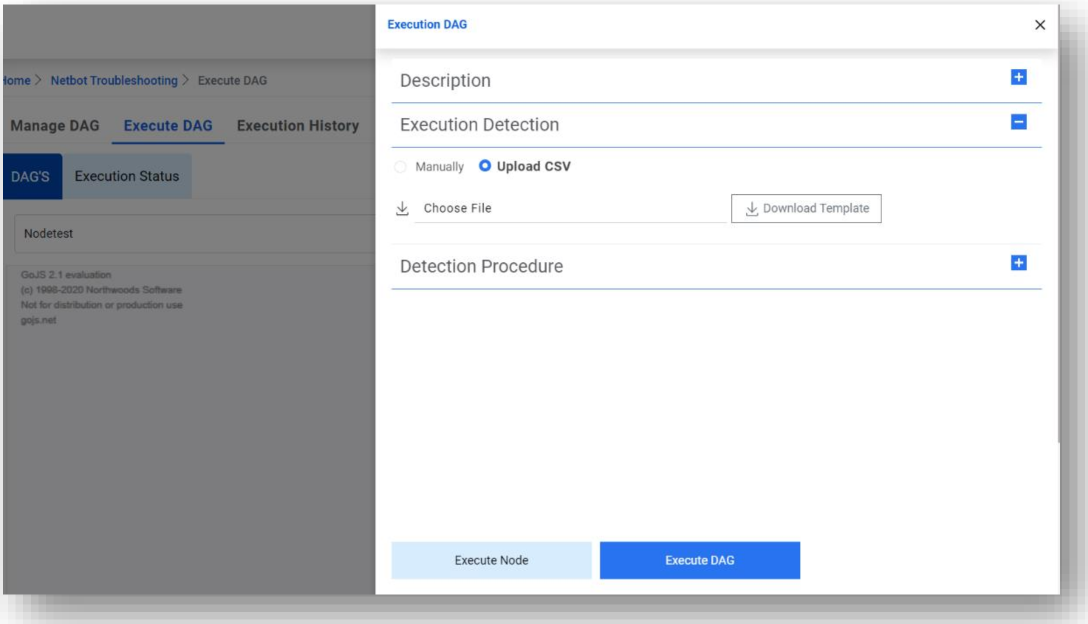
| Device_IP | Param1 | Param2 |
|---|---|---|
| 1.x.x.x | Spooler | Windows |
| 2.x.x.x | WinRM | Windows |
| 3.x.x.x | Zabbix | Linux |
| 4.x.x.x | SQL | Windows |



● Manually ○ Upload CSV

Device_IP* <ipaddress>

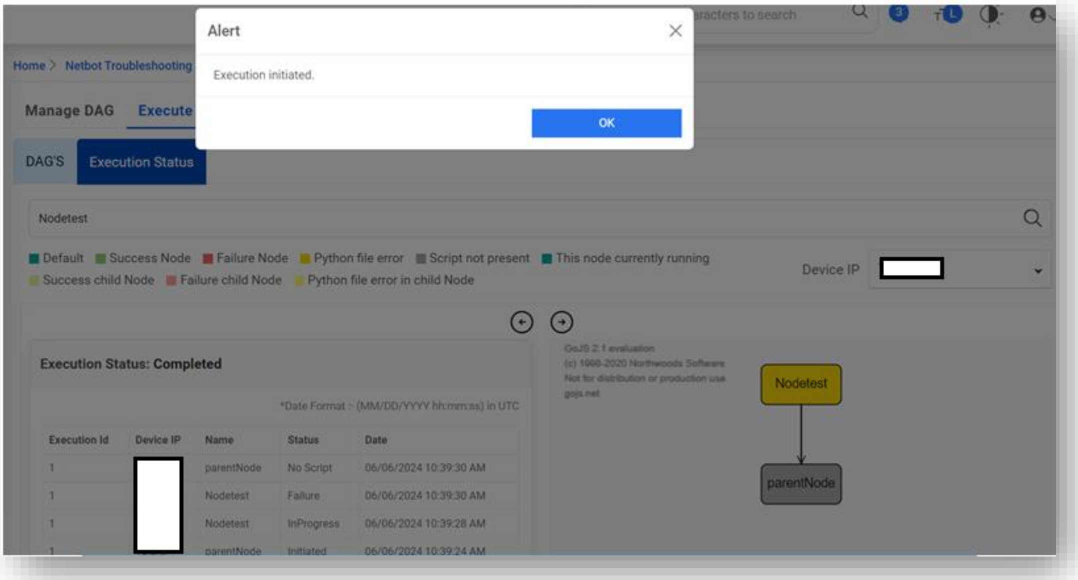
Figure 439 - Parameters Passed Manually



○ Manually ● Upload CSV

Figure 440 - Parameters Passed Using Upload CSV Option

5. Now if you want to only execute the selected Node then, click on '**Execute Node'**. But if you want to execute the selected node and all its child nodes then click on '**Execute DAG'**.
6. The moment execution is initiated, the user will be prompted "Execution Initiated" and is redirected to the **Execution Status** tab.


aracters to search
Home > Netbot Troubleshooting
Manage DAG
Execute
DAG'S
Execution Status
Nodetest
🟩 Default 🟩 Success Node 🟥 Failure Node 🟨 Python file error ▨ Script not present 🟩 This node currently running
🟩 Success child Node 🟥 Failure child Node 🟨 Python file error in child Node
Device IP
GoJS 2.1 evaluation
(c) 1998-2020 Northwoods Software
Not for distribution or production use
gojs.net
Nodetest
parentNode

Alert

Execution initiated.

OK

Device IP [redacted]

Execution Status: Completed

*Date Format :- (MM/DD/YYYY hh:mm:ss) in UTC

| Execution Id | Device IP | Name | Status | Date |
|---|---|---|---|---|
| 1 | [redacted] | parentNode | No Script | 06/06/2024 10:39:30 AM |
| 1 | [redacted] | Nodetest | Failure | 06/06/2024 10:39:30 AM |
| 1 | [redacted] | Nodetest | InProgress | 06/06/2024 10:39:28 AM |
| 1 | [redacted] | parentNode | Initiated | 06/06/2024 10:39:24 AM |

Figure 441 - Execution of a DAG (Cont.)

7. Entered Device IP should populate in the Device IP dropdown list and Dag/Node (diagram) should appear in right panel and related execution status should appear in left panel.

If multiple parameters were passed using upload CSV option, all the Device_IPs should appear in Device IP dropdown list with very first device IP in uploaded csv will appear in the Device IP dropdown list. User can select the different Device IP from Device IP dropdown list and see the status.

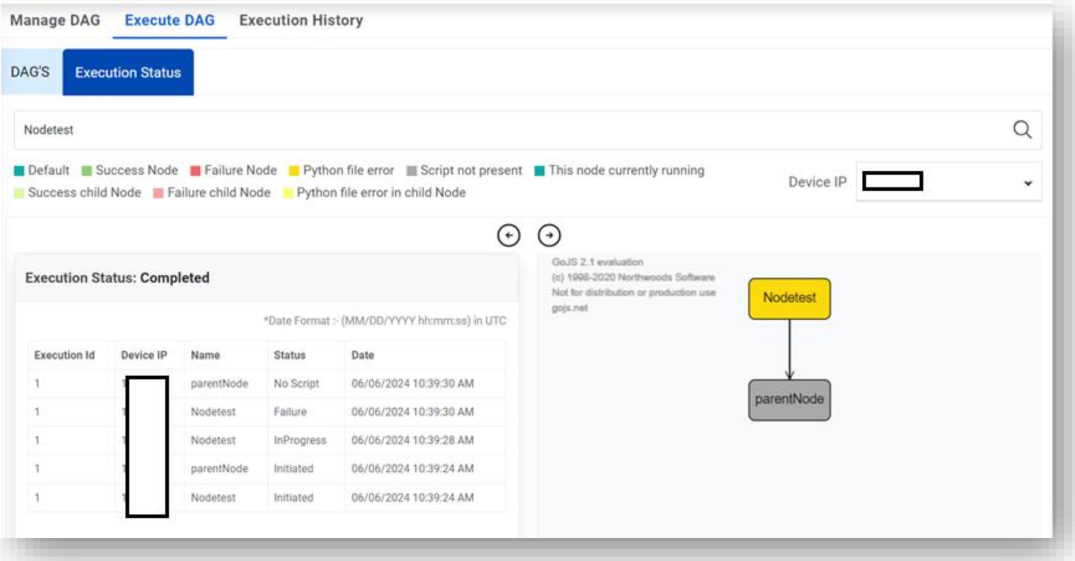
8. Over the period, there should be additional status log and node color should turn as per final execution of Node as screenshot below.

There is color code defined to represent different status that appears on after execution:

a.If node contains script, then actual color (green/red/yellow) will be displayed.

b.If the node contains information and doesn't have any child, then it will be grey.

c.If node contains information and has child, then color will be light version of child node (red/green/yellow) having script.



Nodetest

Device IP [redacted]

Execution Status: Completed

*Date Format :- (MM/DD/YYYY hh:mm:ss) in UTC

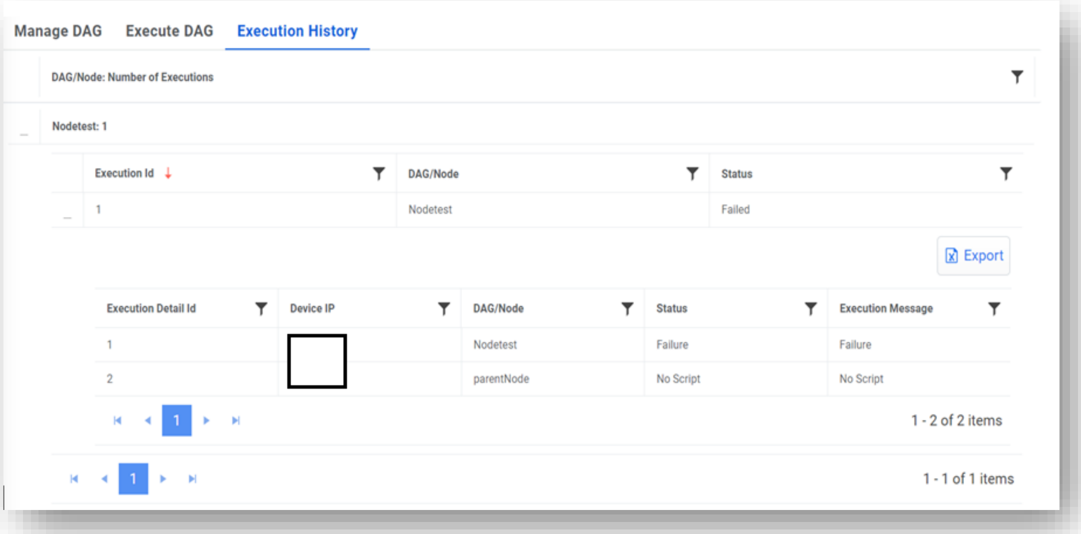
| Execution Id | Device IP | Name | Status | Date |
|---|---|---|---|---|
| 1 | [redacted] | parentNode | No Script | 06/06/2024 10:39:30 AM |
| 1 | [redacted] | Nodetest | Failure | 06/06/2024 10:39:30 AM |
| 1 | [redacted] | Nodetest | InProgress | 06/06/2024 10:39:28 AM |
| 1 | [redacted] | parentNode | Initiated | 06/06/2024 10:39:24 AM |
| 1 | [redacted] | Nodetest | Initiated | 06/06/2024 10:39:24 AM |

Figure 442 – DAG/Node Execution Status

To see the most recent execution status of any executed Dag/Node, user can navigate to Execute Status tab and search the Dag/Node in the search filter and can see the result for all the devices executed in most recent execution.

### 6.3.1 DAG Execution History

To see the historical execution status of a DAG/Node, you can filter a DAG/Node and see all the execution happened in the past by navigating to **Netbot Troubleshooting→ Execution History**



DAG/Node: Number of Executions

Nodetest: 1

| Execution Id | DAG/Node | Status |
|---|---|---|
| 1 | Nodetest | Failed |

| Execution Detail Id | Device IP | DAG/Node | Status | Execution Message |
|---|---|---|---|---|
| 1 | [redacted] | Nodetest | Failure | Failure |
| 2 | [redacted] | parentNode | No Script | No Script |

1 - 2 of 2 items

1 - 1 of 1 items

Figure 443 - DAG Execution History

# 7 Support

For any additional queries, please reach out to us at iAuto-Product-Supp@hcl.com.

# 8 Appendix

Table 71 - List of Abbreviations

| Abbreviation | Expansion |
|---|---|
| AD | Active Directory |
| AI | Artificial Intelligence |
| ITOPS | IT Operations |
| ITSMS | IT Service Management System |
| KEDB | Known Error Database |
| SNOW | ServiceNow |

HCLSoftware

hcltechsw.com