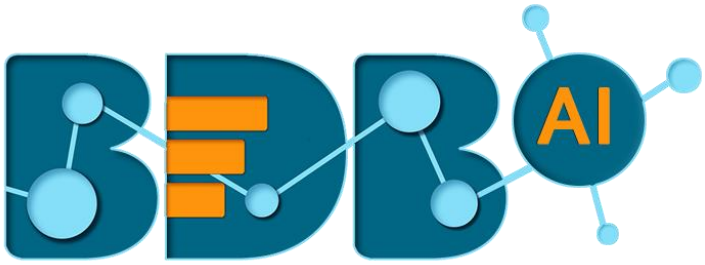


User Guide

Data Pipeline- 6.5

Contents

1. Introduction 6
   1.1. Introducing Data Pipeline 6
   1.2. Overview 6
   1.3. Target Audience 6
2. Supported Web Browsers 6
   2.1. Unsupported Browsers 6
3. Terminology 7
4. Logging In 9
5. Homepage Menu Panel 11
   5.1. List Pipeline 12
      5.1.1. Sharing a Pipeline 13
   5.2. Create a Pipeline 14
   5.3. List Components 15
   5.4. Delete Pods 17
   5.5. Scheduler 17
   5.6. Data Channel 18
   5.7. Settings 18
6. User Interface 18
   6.1. Creating a Pipeline 18
      6.1.1. Resource Allocation 21
   6.2. Adding Components to the Canvas 21
   6.3. Toggle Log Panel 25
   6.4. Search Component in Pipelines 26
   6.5. Global Variable 28
   6.6. Deleting a Pipeline 29
   6.7. Pipeline Monitoring 29
   6.8. Connecting Components 30
      6.8.1. Event Creation 31
      6.8.2. Updating an Event 32
      6.8.3. Connecting Event to a Component 33
      6.8.4. Shared Event 34
      6.8.5. Preview Data 35
      6.8.6. Flush Event 36
7. Component Pallet 37

7.1. Sorting and Grouping Components 37
7.2. Searching a Component 38
8. Component Properties 39
8.1. Basic Information Tab 40
8.2. Meta Information Tab 40
8.3. Configuration and Logs Tabs 42
9. Components 44
9.1. Reader 44
9.1.1. S3 Reader 44
9.1.2. HDFS Reader 48
9.1.3. Cassandra Reader 52
9.1.4. RDBMS Reader 53
9.1.5. ES Reader 56
9.1.6. SFTP Reader 58
9.1.7. GCS Reader 60
9.1.8. SFTP Stream Reader 63
9.1.9. MongoDB Reader 65
9.1.10. Notification Subscriber 67
9.1.11. Azure Reader 74
9.1.12. Azure Cosmos DB Reader 77
9.1.13. Azure Metadata Reader 79
9.1.14. Azure Blob Reader 81
9.2. Writer 85
9.2.1. S3 Writer 85
9.2.2. RDBMS Writer 87
9.2.3. HDFS Writer 89
9.2.4. Cassandra Writer 91
9.2.5. ES Writer 93
9.2.6. MongoDB Writer 94
9.2.7. Notification Publisher 97
9.2.8. Azure Writer 104
9.2.9. Azure Cosmos DB Writer 105
9.2.10. Eventhub Publisher 107
9.2.11. EventGrid Producer 108
9.3. Transformation 110

9.3.1. Split Component 110
9.3.2. Date Formatter Component 112
9.3.3. Join Component 115
9.3.4. SQL Component 117
9.3.5. Dataprep Script Runner Component 119
9.3.6. File Splitter 121
9.3.7. Python Script (Custom Python Script) 124
9.3.8. Rule Splitter 127
9.3.9. Stored Procedure Runner 130
9.3.10. Stream Component 132
9.3.11. Flatten JSON 134
9.3.12. Email Component 136
9.4. ML 138
9.4.1. R Model Runner 138
9.4.2. Python Model Runner 140
9.5. Ingestion 141
9.5.1. API Server Ingestion 142
9.5.2. Sqoop Executer 145
9.5.3. GCS Monitor 146
9.5.4. AWS SNS Monitor 150
9.5.5. Script Runner 155
9.5.6. MQTT Consumer 158
9.5.7. API Connector 159
9.5.8. SFTP Python Monitor 161
9.5.9. OPC UA 164
9.5.10. Eventhub Subscriber 165
9.5.11. Twitter Scrapper 167
9.5.12. API Ingestion 169
9.6. WebSocket 170
9.6.1. WebSocket Listener 170
9.6.2. WebSocket Producer 172
9.7. Scheduler 173
10. Settings 174
10.1. Kafka Configuration 175
10.2. Dataprep Scripts 176
10.3. Data Science Models 176

10.3.1. Using a Deployed Model in Data Pipeline 177
10.4. Logger 178
10.5. Default Configuration 178
11. Third-Party Data Ingestion using Data Pipeline 179
11.1. Data Ingestion from an MQTT Consumer 180
11.2. Configuring a Data Pipeline According to the Script 180
11.2.1. Making a POST request along with the data 182
12. Signing Out 183
12.1. Forgot Password Option 184
12.2. Force Login 185

1. Introduction

1.1. Introducing Data Pipeline

The Data Pipeline is an Enterprise data orchestration and transformation tool, that allows you to seamlessly design and deploy your DataOps or MLOps workflow. It can handle both Streaming and batch data seamlessly. The Data pipeline offers an extensive list of data processing components that help you automate the entire data workflow, Ingestion, transformations, and running AI/ML models.

In the Data Pipeline plugin, we treat data as events, Data Processing components can listen to events, as data hits those events, the process kick starts automatically, these processes then publish the output to another event. This allows data engineers to chain the process and build large data flows.

1.2. Overview

This guide covers:

- Features and menus provided on the Data Pipeline homepage
- Steps to create and manage Pipeline
- Create and manage various pipeline components

1.3. Target Audience

The document is targeted at the following audience:

- Data Engineers
- Data Scientists
- ML-Ops Engineers

2. Supported Web Browsers

The BDB Platform is a web browser-based application. The users can run the BDB Platform and its various plugins on the below-given versions of the browsers:

Google Chrome	Latest Version (recommended web browser)
Mozilla Firefox/ Firefox ESR	Latest Version
Microsoft Edge	Latest Version
Apple Safari	10

The supported browser versions are driven by the capabilities the UI employs and the dependencies it uses. UI features will be developed and tested against the supported browsers.

2.1. Unsupported Browsers

While the UI may run successfully in unsupported browsers, it is not actively tested against them. Additionally, the UI is designed as a desktop experience and is not currently supported in mobile browsers.

3. Terminology

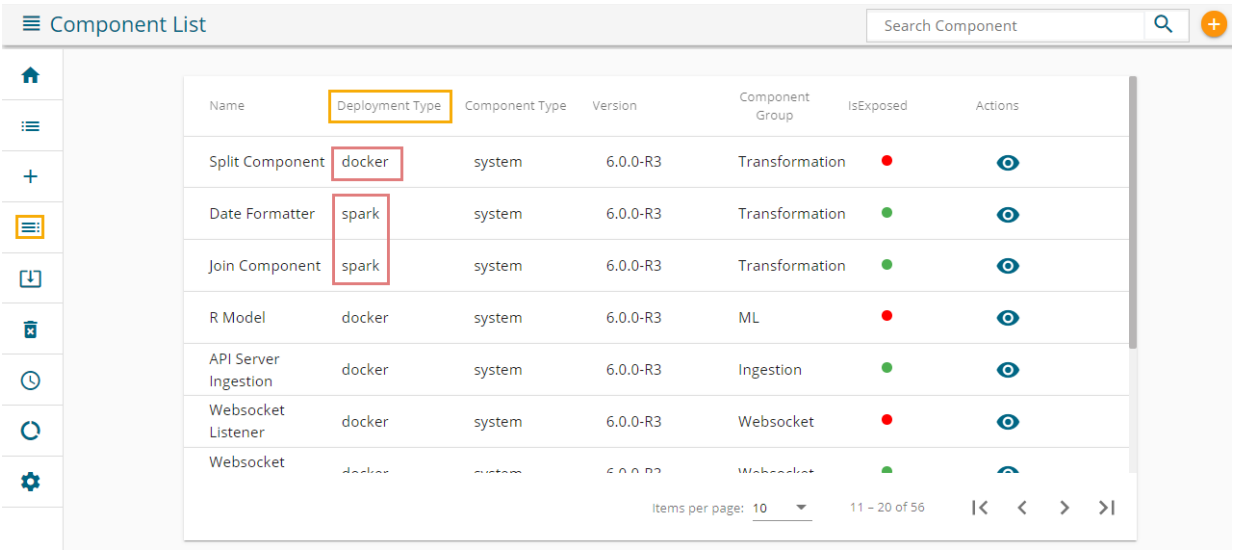
Component Deployment: Component Deployment means the Deployment on Kubernetes.

Components (Deployment Type): There are two types of component categories in context to deployment-type (Spark and Docker).

Spark: Drivers and Executers.

When these components get deployed, we get two pods one is for the component driver and another one is the executor. The number of executors can be increased according to the processing load or the throughput required by the user.

Docker: a single pod gets deployed in this case.



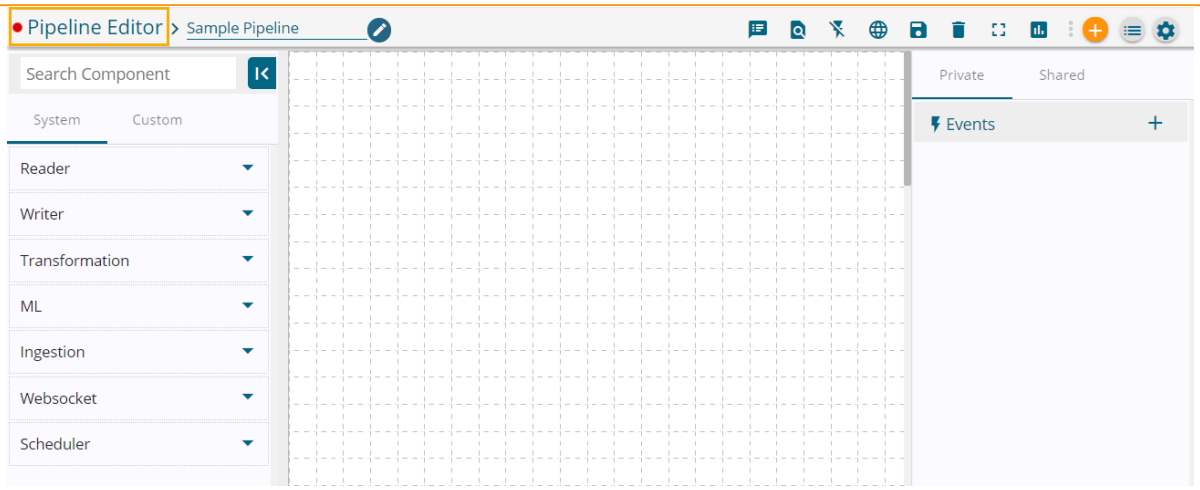
Base Components

These are the components that are responsible for the backend functionality of the pipeline.
Each of these components' Pod should be UP and Running for the smooth functioning of the pipeline module.
Base components are as given below:

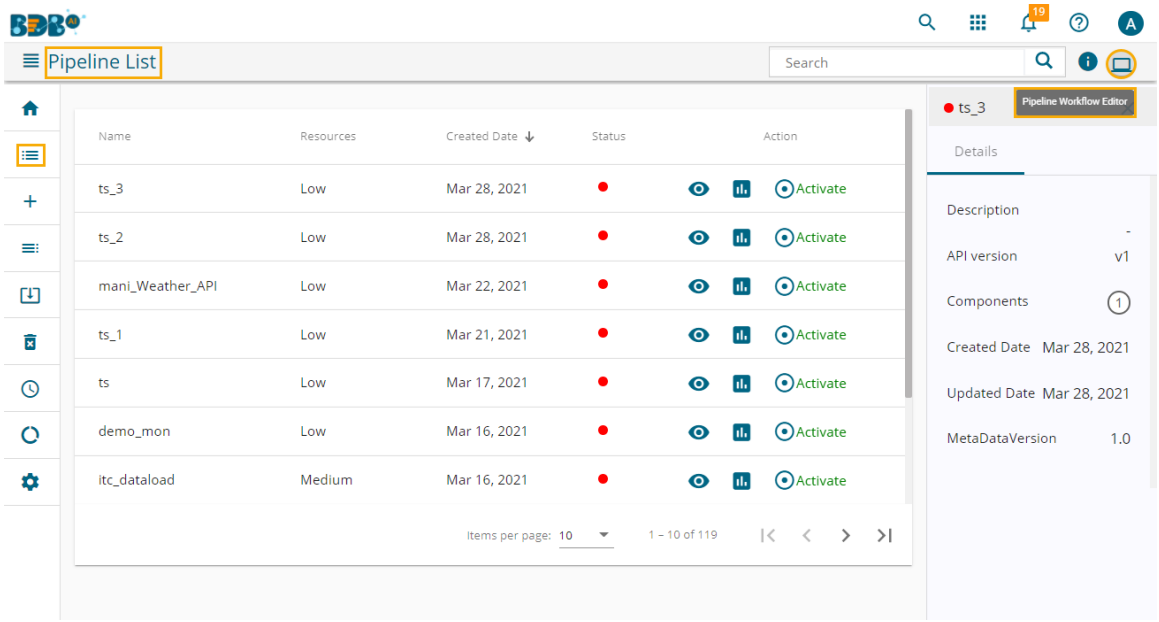
- event-trigger
- log manager
- pipeline-scheduler
- task invoker
- task manager
- webservice
- ws-producer
- pipeline-scheduler

Workflow Editor

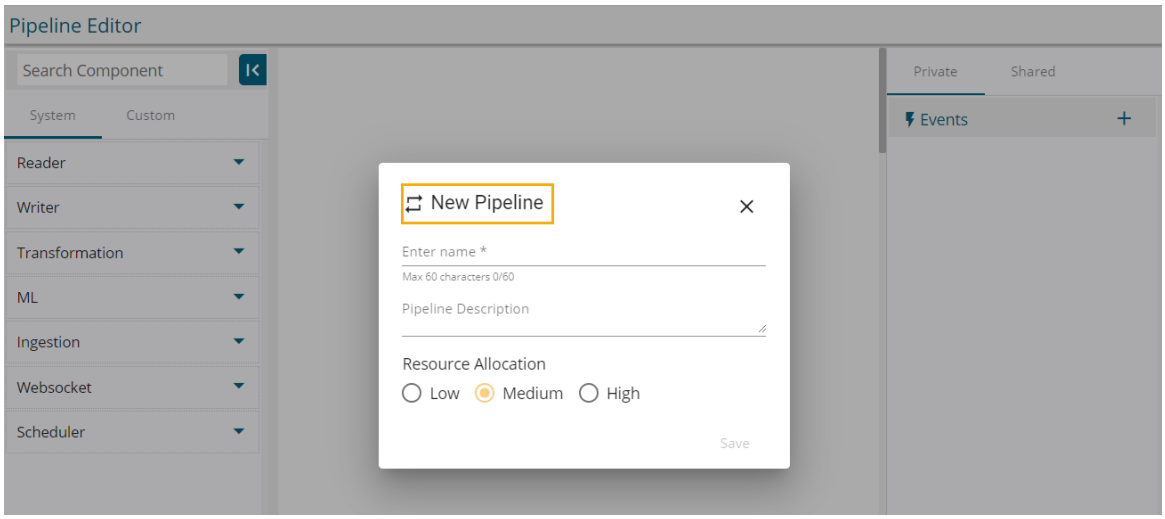
The canvas where we create/develop the pipeline is the workflow editor.



The user can open the Pipeline Editor page by clicking the '**Pipeline Workflow Editor**' icon provided on the top right side of the Pipeline List page.



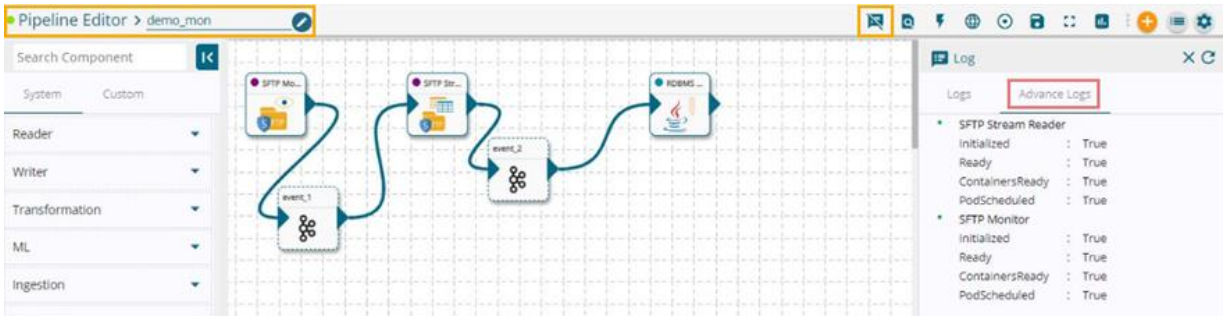
Note: While clicking the 'Pipeline Workflow Editor' icon the user gets the New Pipeline dialog box. The user can get the Pipeline Workflow Editor page only after creating a new pipeline.



Logs

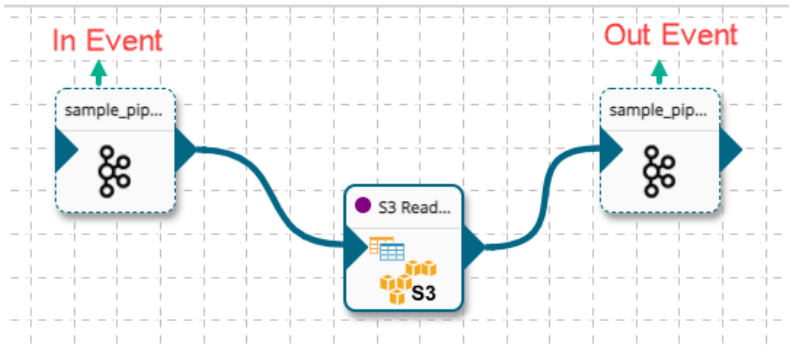
Advanced Logs: when someone refers to advanced logs, that person is asking to monitor whether the components on the workflow editor are deployed or not.

Note: This tab appears when the pipeline is active inside the log panel.



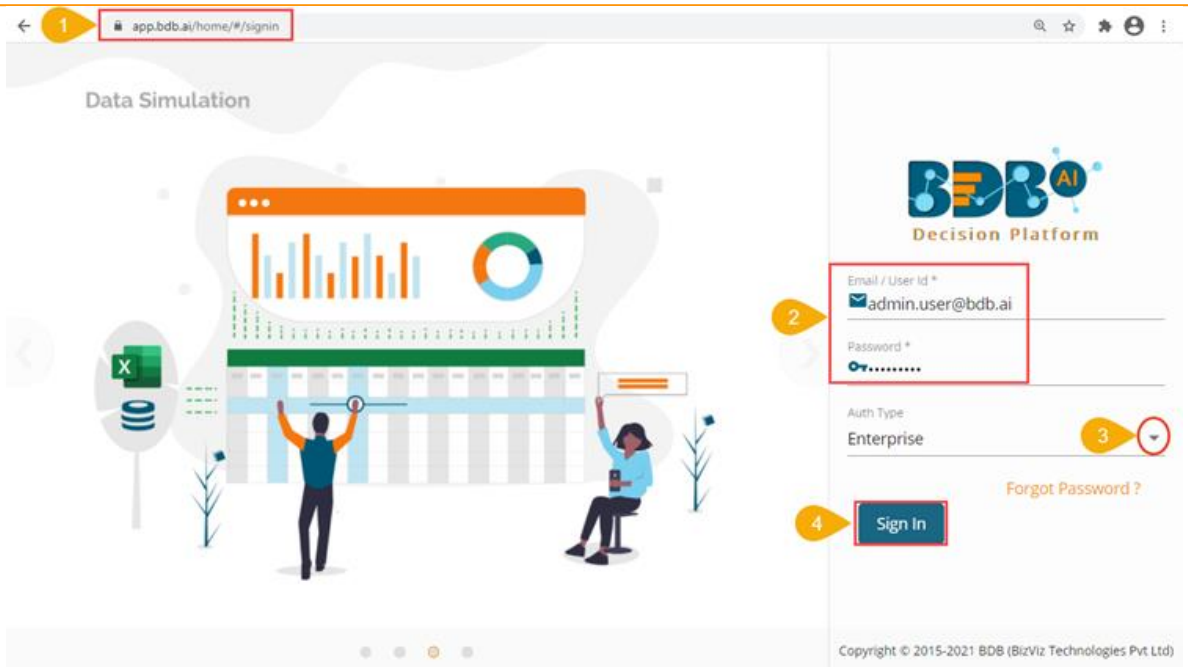
Event

An event, in general, is referred to as the Kafka topic that works as a communication channel between two components.
In-event: the previous event/ Kafka-topic. This gives input to the component.
Out-event: the event after the component is out event. This is where the output of the component is present.

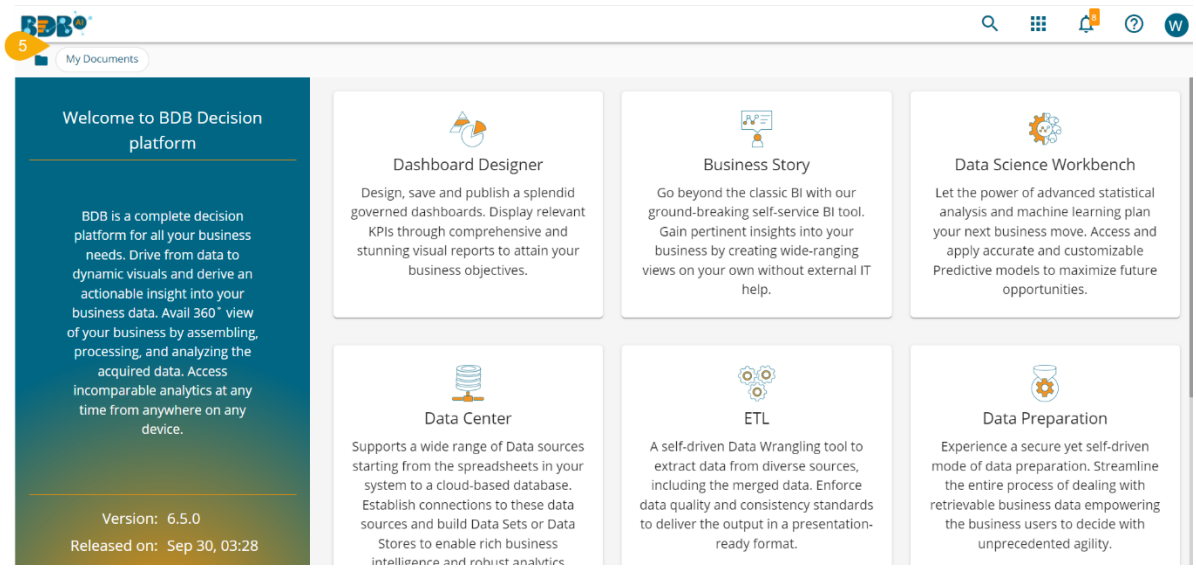


4. Logging In

i) Browse the URL link: https://app.bdb.ai to open the Sign In page.
ii) Enter user-specific credentials (Email ID with valid Password).
iii) Select an authentication type using the drop-down.
iv) Click the '**Sign In**' option.

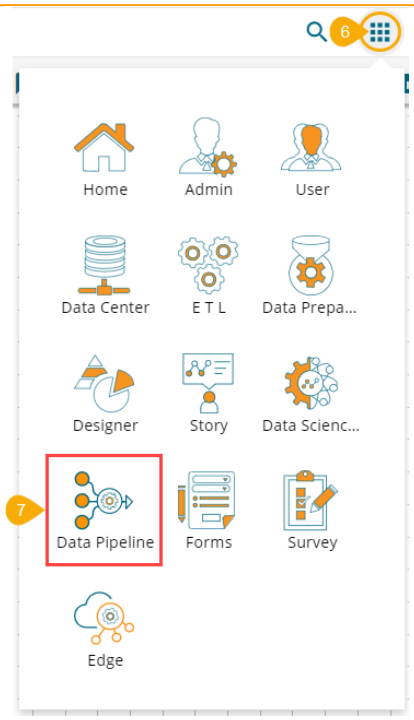


v) The Platform homepage opens.

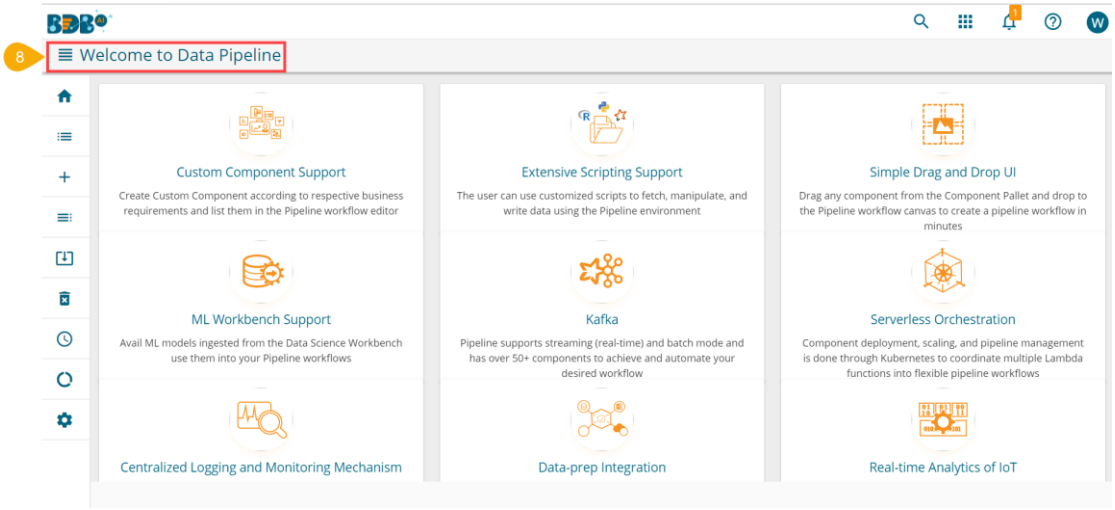


Note: Make sure that the user has permission to access the Data Pipeline plugin.

vi) Click the Apps menu.
vii) Select the Data Pipeline Plugin.

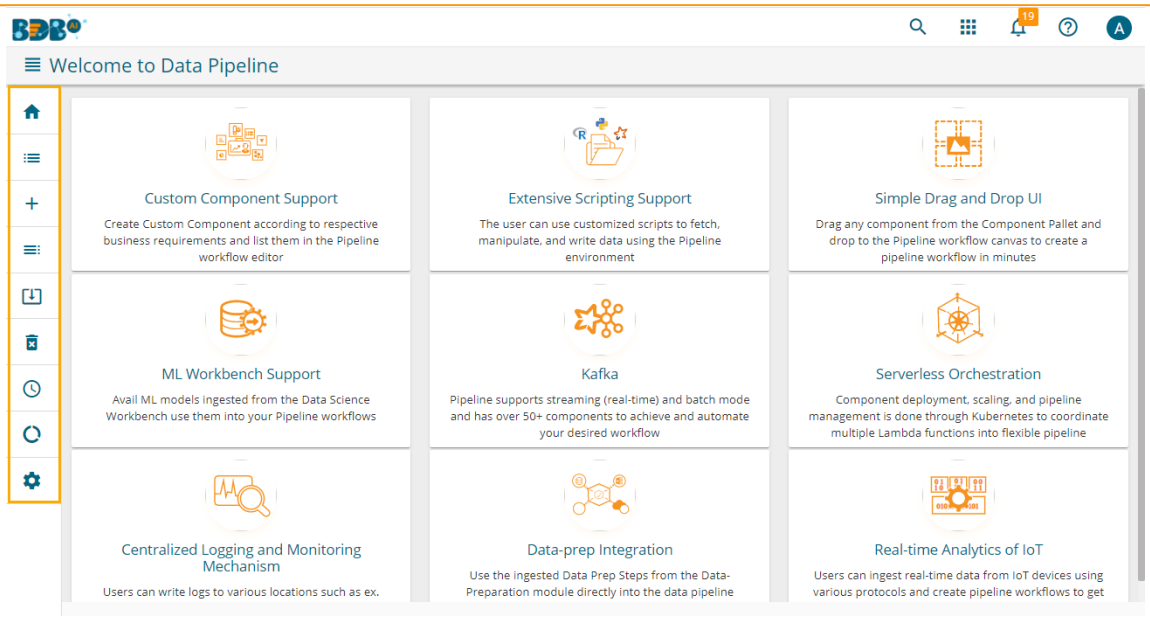


viii) The Data Pipeline Homepage opens.



5. Homepage Menu Panel

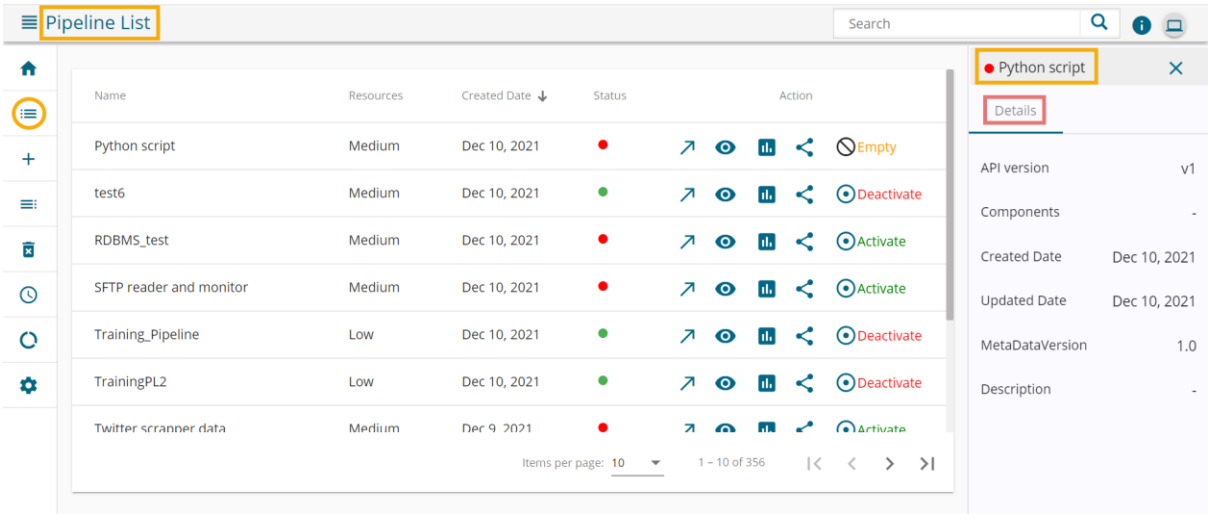
The Data Pipeline homepage opens with a menu panel on the left side of the screen. The user can expand the menu panel by a click on it as highlighted in the following image:



The user can access various options from the Homepage Menu Panel to work upon the data flow.

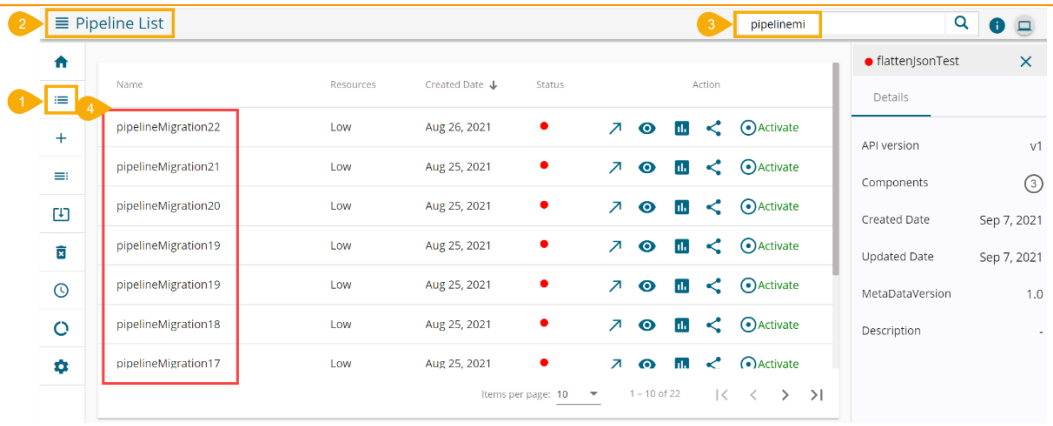
5.1. List Pipeline

The '**List Pipelines**' option opens the available Pipeline List for the logged-in user. All the saved pipelines by a user get listed on this page. The 'Details' tab on the right side of the page displays the basic details of the selected pipeline.



Note:

a. The user can open the Pipeline Editor for the selected pipeline from the list by clicking the '**View**' icon or the '**Pipeline Workflow Editor**' icon on the Pipeline List page.
b. The user can search for a specific pipeline by using the Search bar on the Pipeline List. By typing a common name all the existing pipelines having that word will list. E.g., By typing '**Sam**' all the existing pipelines with the word 'Sample' in it get listed in the following image:

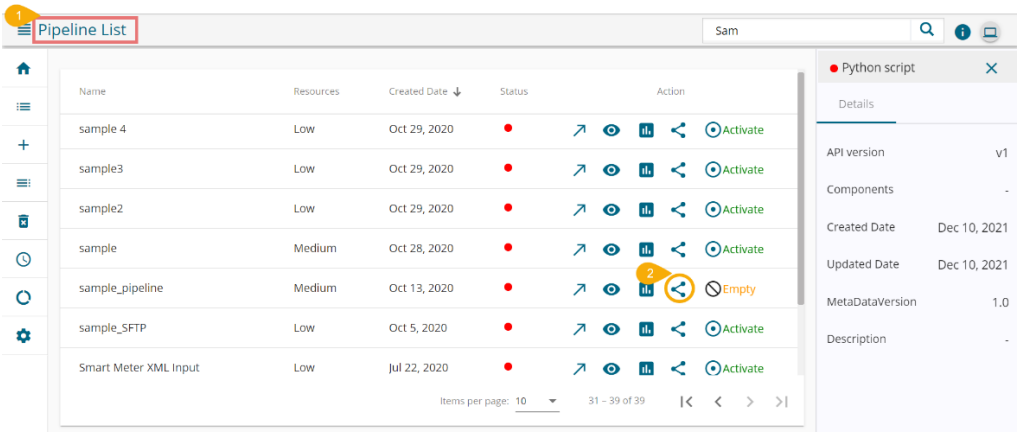


5.1.1. Sharing a Pipeline

The user can share pipeline with one or multiple users and user Groups using the '**Share Pipeline**' option.

i) Click the '**List Pipelines**' icon to open the List Pipelines.

ii) Select a Pipeline and click on the '**Share Pipeline**' icon.

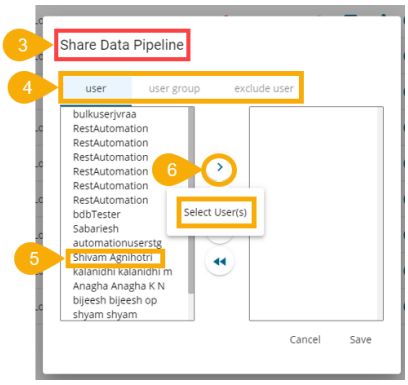


iii) The '**Share Pipeline**' window opens.

iv) Select an option out of the given choices: User and User Group or Exclude User (the '**Exclude User**' option can be chosen, if the document is already shared with a user/user group and you wish to exclude them from the privilege).

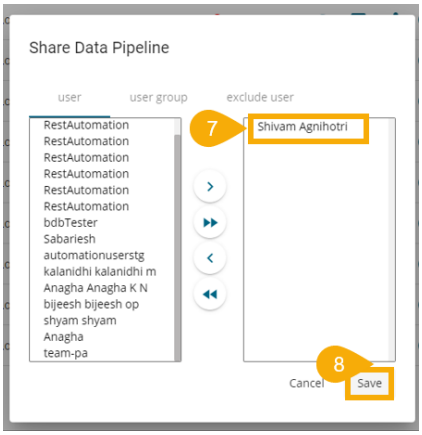
v) Select a user or user group using the displayed list of users/ user groups.

vi) Click the arrow to move the selected User(s)/ User Group(s).

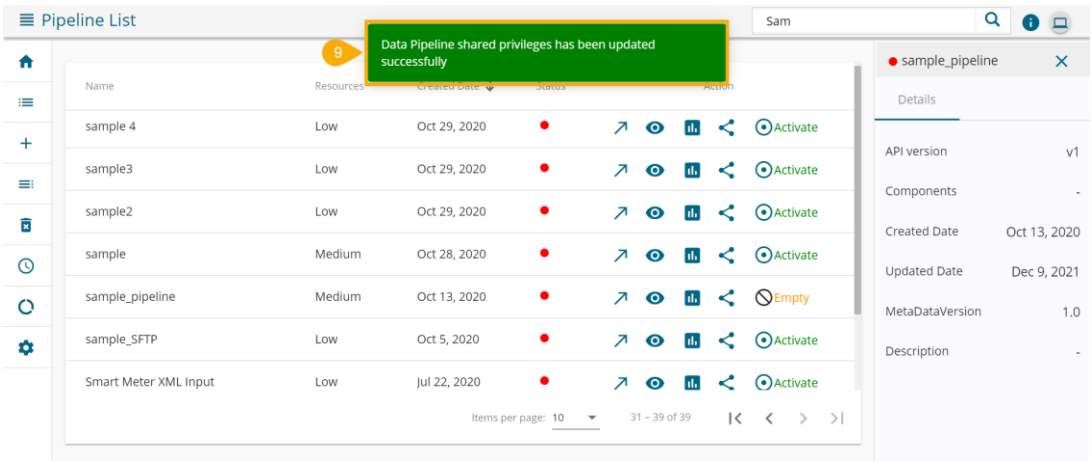


vii) The selected user(s)/user group(s) get moved to the box given on the right.

viii) Click the '**Save**' option.



ix) Privilege for the selected pipeline gets updated and the same gets communicated through a message.



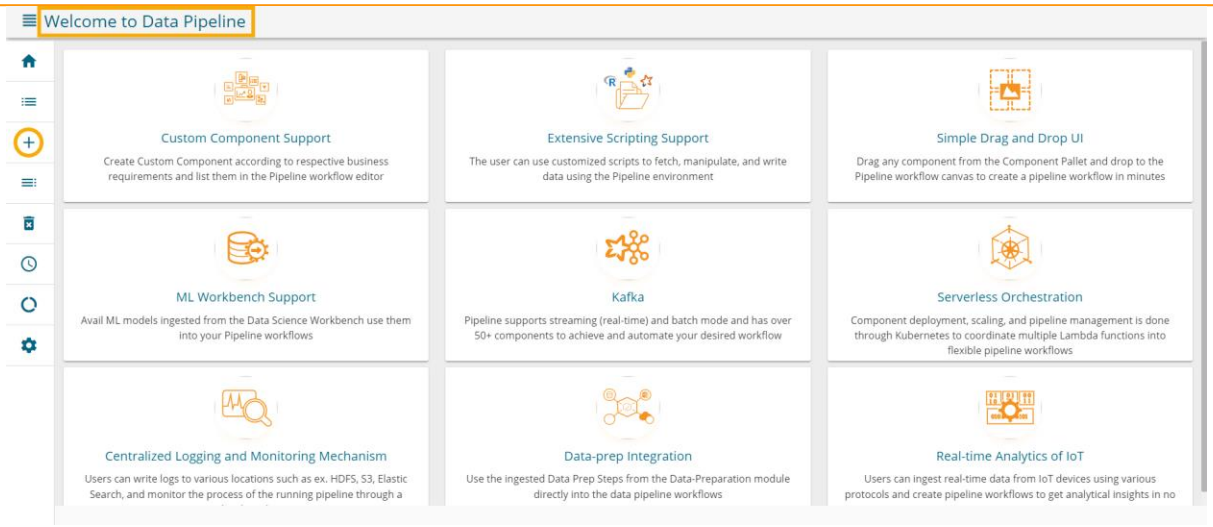
x) By completing the steps mentioned above, the concerned pipeline gets successfully shared with the selected user/user group or the selected users can also be excluded from their privileges for the concerned pipeline.

Note: A non-admin user can '**Edit/Modify'** and '**View'** a pipeline but cannot '**Share'** it or '**Delete'** a shared pipeline.

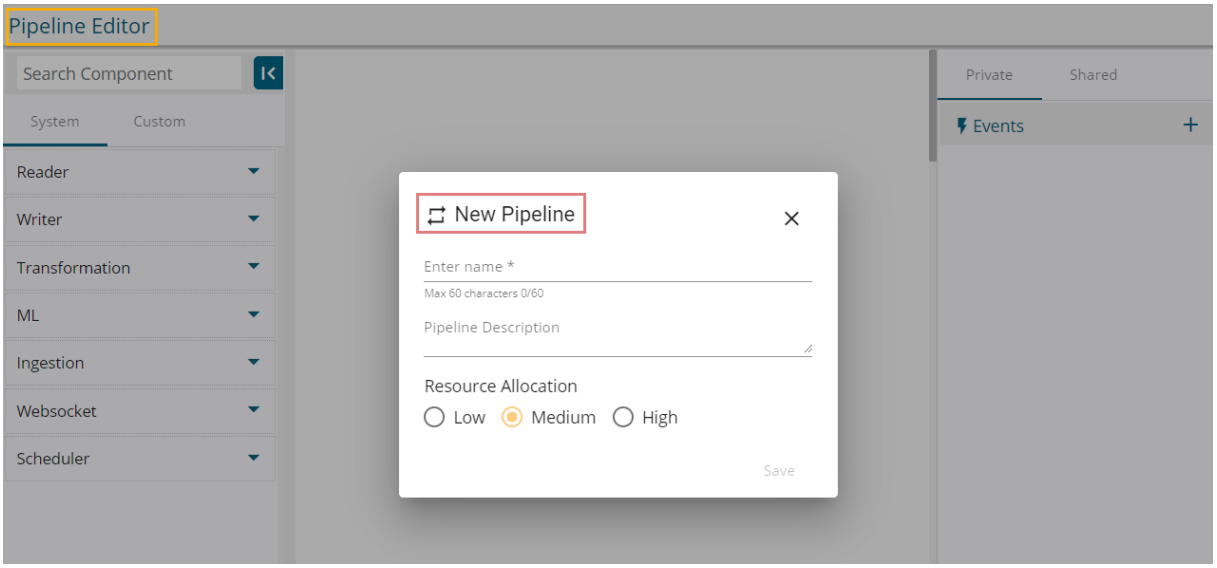
5.2. Create a Pipeline

The '**Create Pipeline**' option redirects the user to create a new pipeline and eventually to the Pipeline Editor (the user interface) to create, update, activate, and save the Pipeline Workflow.

Click the '**Create Pipeline**' icon from the menu panel.



A new dialog box opens redirecting the user to the Pipeline Editor.

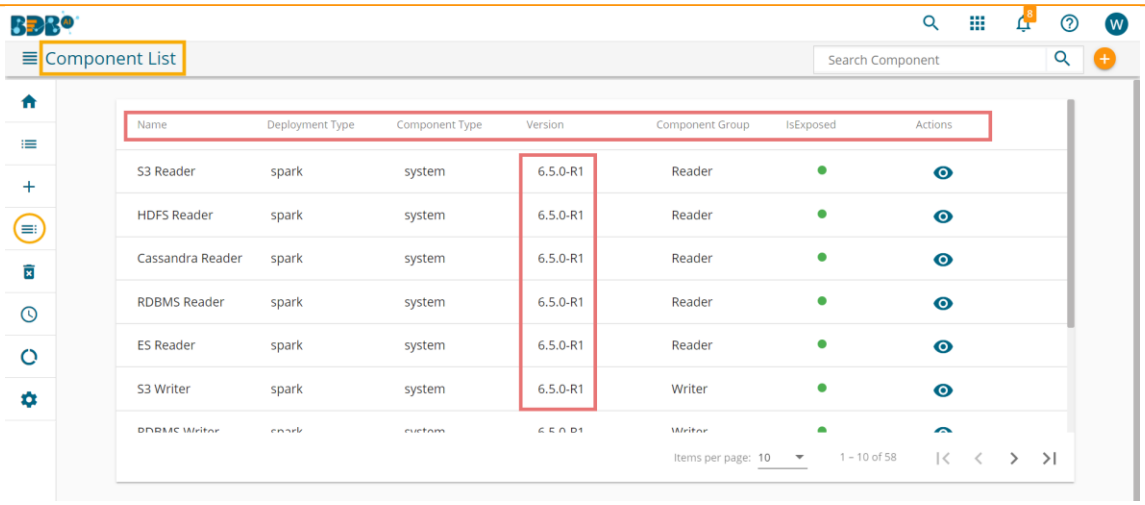


Note: The stepwise process to create a Pipeline workflow is described later in this document.

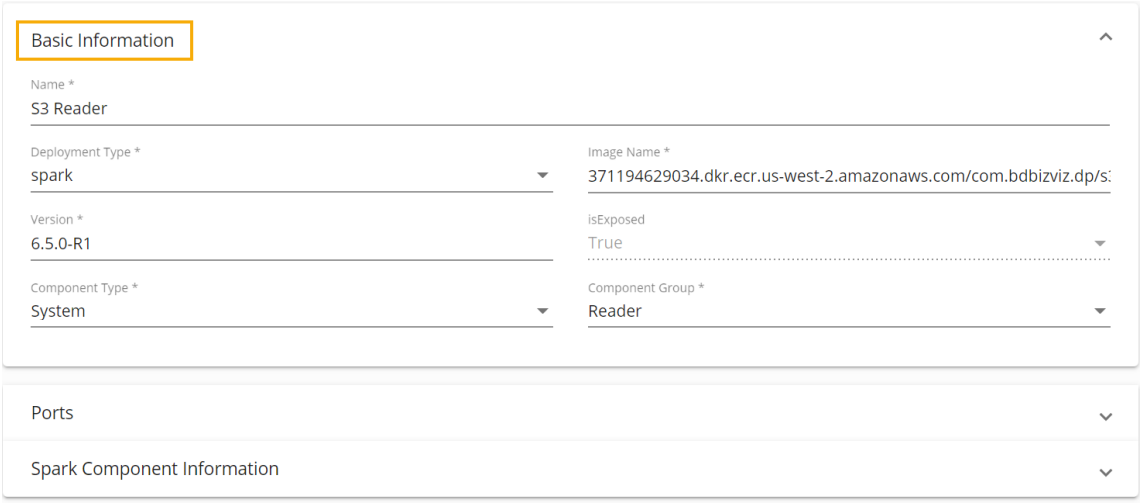
5.3. List Components

All the available component names are listed on this page. The user gets to know about the Deployment Type, Component Type, Version, Component Group about a component. The Expose status is indicated through the green color coding if the related component is exposed.

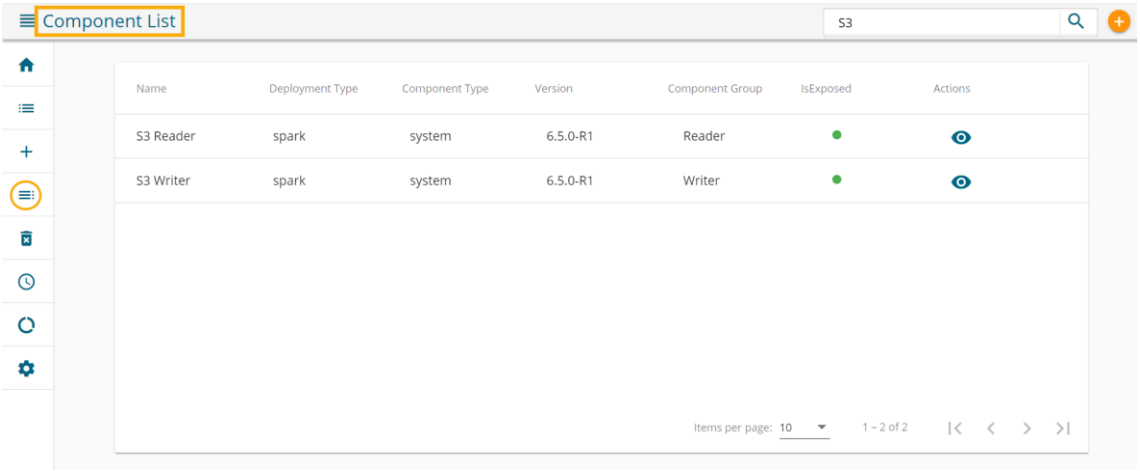
The user gets the Component version information from the List Component page. The version of the components get updated with each new release of the respected components.



The user can get the Basic Information about the selected component by clicking the '**Actions**' icon. The basic information can be edited from this window if needed. The user gets Ports and Spark Component Information if the selected Deployment Type option is '**Spark**'.



The user can search for the desired components using the Search bar. The Component List gets customized based on the searched components.



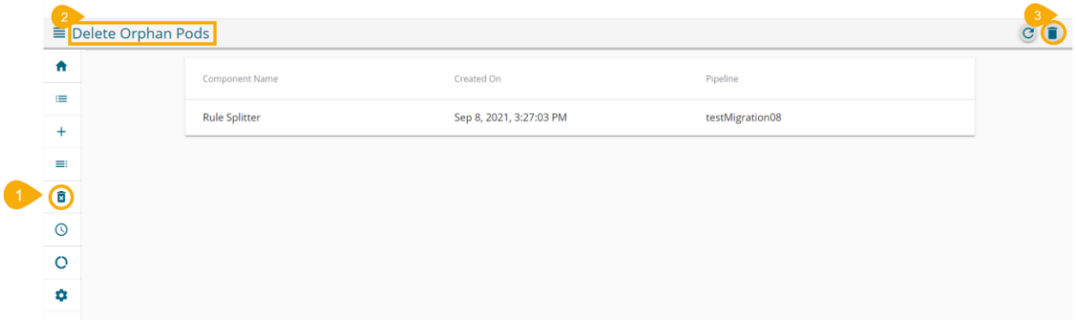
5.4. Delete Pods

The user can delete the orphan pods using this option.

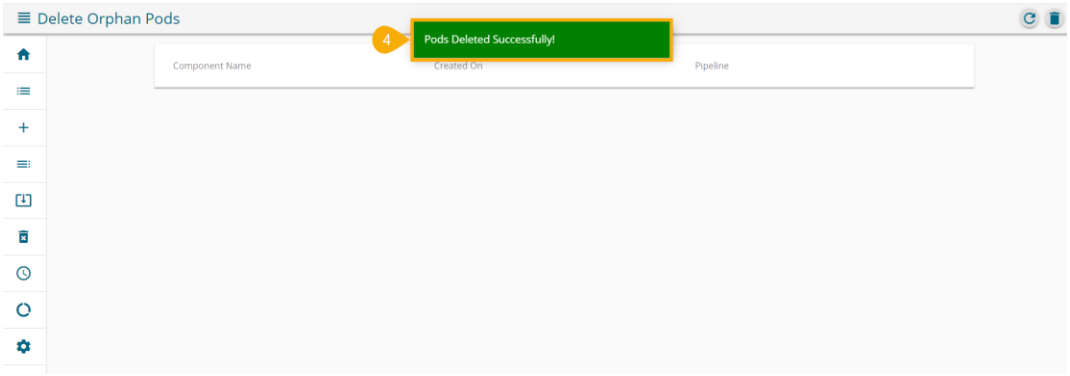
i) Click the '**Delete**' icon.

ii) The '**Delete Orphan Pods**' page opens.

iii) Click the '**Delete**' icon.

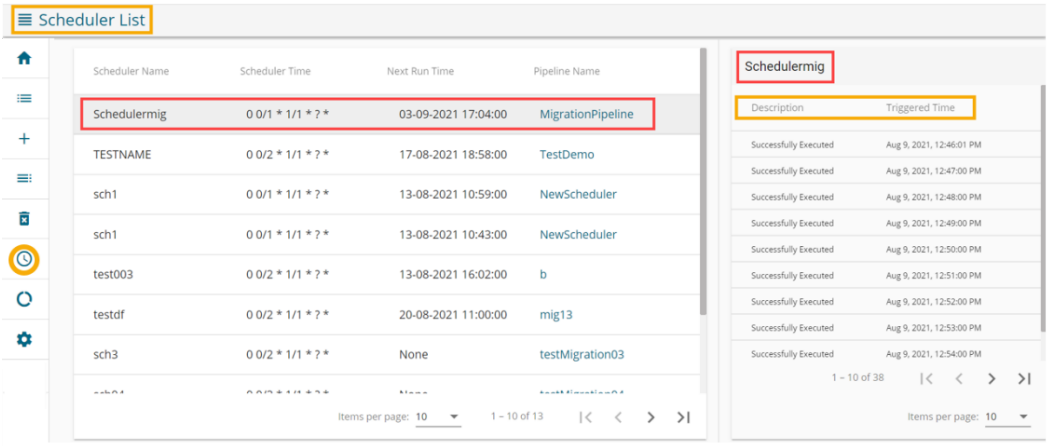


iv) A notification message appears, and the listed orphan pods get deleted.



5.5. Scheduler

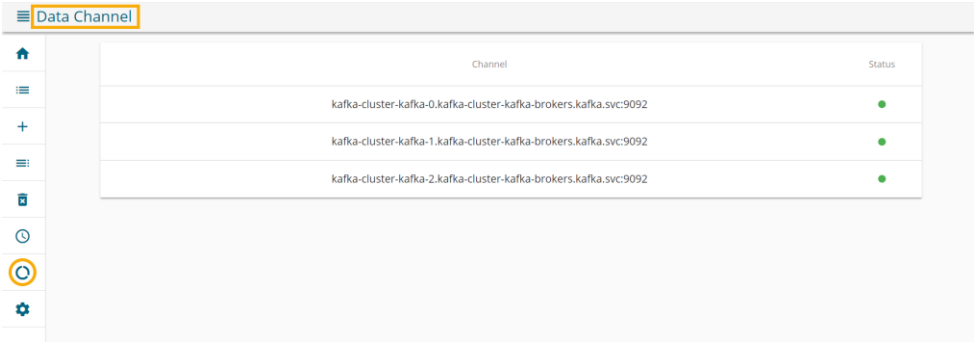
It opens the Scheduler List with the status and Triggered Time of the scheduled pipeline components mentioned in the right-side panel.



Note: The Scheduler Time column displays the time as Cron expressions.

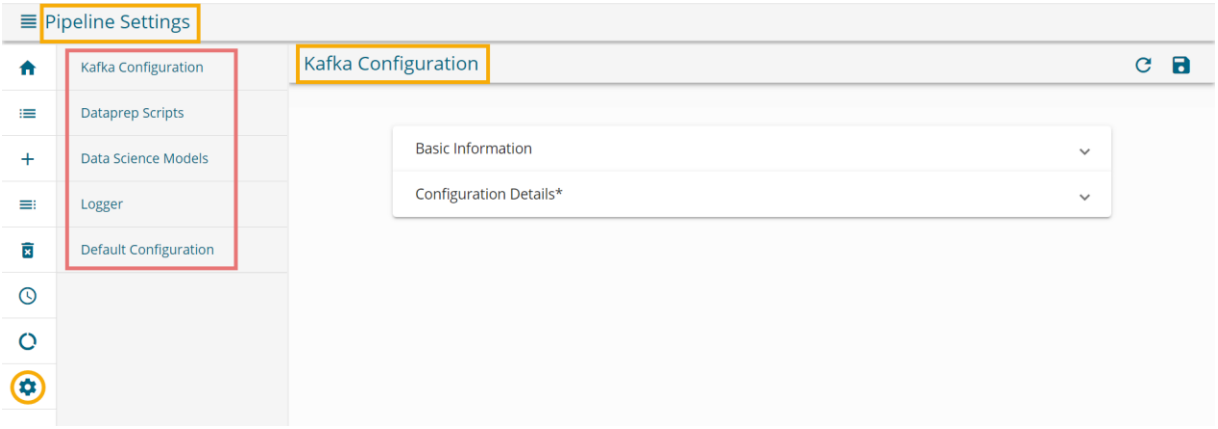
5.6. Data Channel

The Data Channel window displays the status of the Kafka Server.



5.7. Settings

The user gets settings information about various pipeline functionalities. The Settings page provides information about Kafka Configuration, Data Prep Scripts, Data Science Models, and Logger. The Default Configuration and Pod Status get displayed under the Settings option.



The details of each Settings option are described under the Settings section of this document.

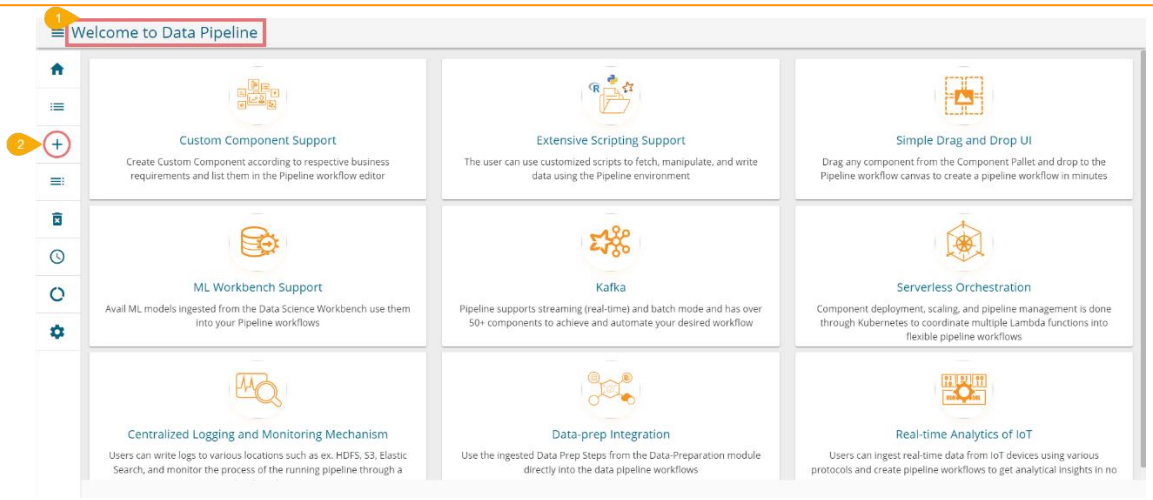
6. User Interface

The User Interface of the Data Pipeline provides a canvas for the user to build the data flow (Pipeline Workflow).
The Data Flow creation process can be divided into three parts as mentioned below:

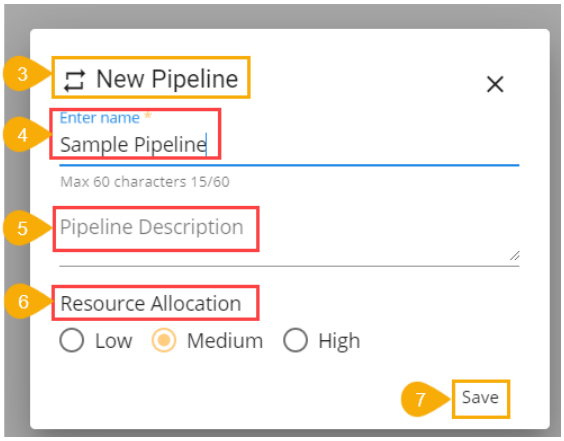
1. Creating a Pipeline
2. Adding Components to the Canvas
3. Connecting Components

6.1. Creating a Pipeline

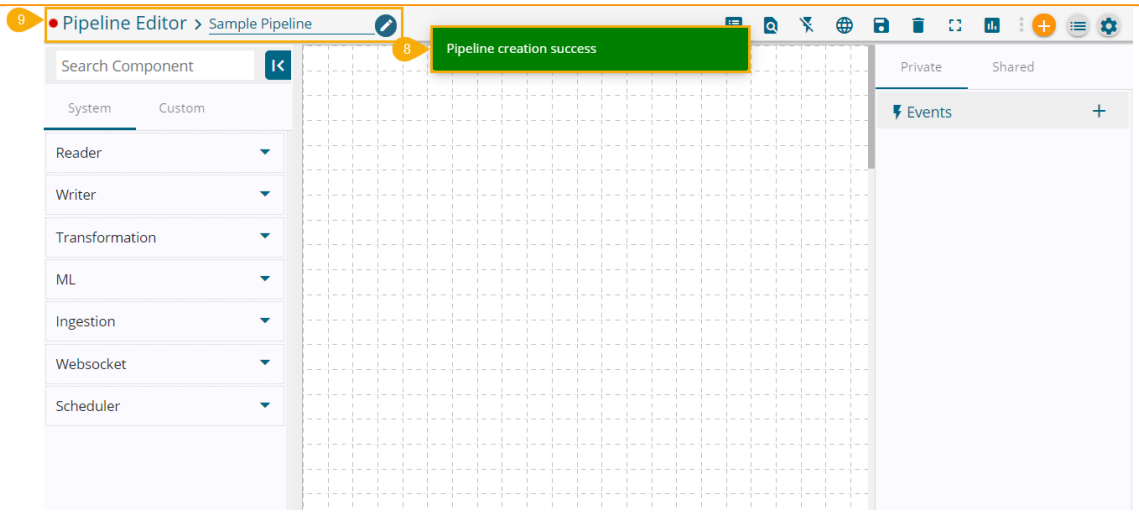
i) Navigate to the Data Pipeline landing page.
ii) Click the '**Create Pipeline**' option provided on the top right side of the Pipeline landing page.



iii) The '**New Pipeline**' window opens asking for the basic information.
iv) Enter a name for the new Pipeline.
v) Describe the Pipeline (Optional).
vi) Select a resource allocation option using the radio button- the given choices are:
   i. Low
   ii. Medium
   iii. High
vii) Click the '**Save**' option to create the pipeline. By clicking the 'Save' option, the user gets redirected to the pipeline workflow editor.



viii) A success message appears to confirm the creation of a new pipeline.
ix) The Pipeline Editor page opens for the newly created pipeline.

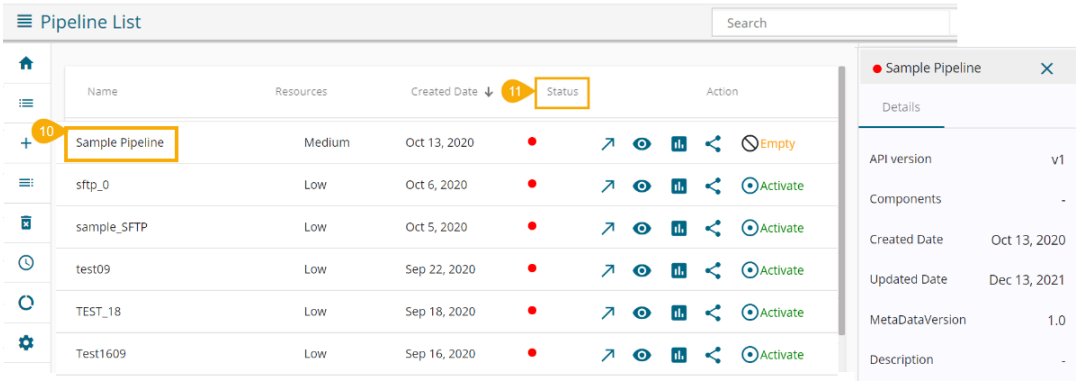


x) The newly created pipeline appears at the top of the Pipeline List.

xi) The status of the pipeline is indicated through color-coding under the 'Status' column.

a. Green-Activated Pipeline

b. Red- Deactivated Pipeline



The following icons get displayed on the Pipeline Editor page:

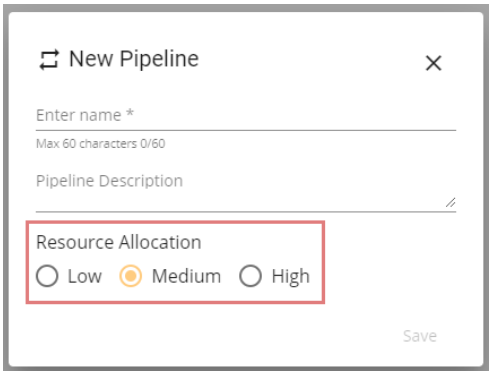
Icon	Name	Function
	Open/ Close the Components Pallet	Opens or Closes the Component Pallet
	Edit the Pipeline Name/resource allocation type	Allows to modify or change the given Pipeline name or change default resource configuration for pipeline components.
	Toggle Log Panel	Displays the UI Logs.
	Search Component in Pipelines	Searches for a specific component across all the existing pipelines.
&	Toggle Event Panel	Opens and closes the Event Panel to create and display created Events
	Global Variable	Opens the Global Variables window to add new variables.
	Activate/Deactivate Pipeline	Activates or deactivates the Pipeline. It appears once the user clicks the Update Pipeline icon
	Update Pipeline	Saves and updates the pipeline

	Delete Pipeline	Deletes the pipeline
	Full Screen	Displays the present page in full screen
	Pipeline Monitoring	Displays all the essential details about the selected pipeline to monitor it.
	Create a New Pipeline	Redirects the users to create a new pipeline
	List Pipeline	Redirects the users to the page listing all the existing pipelines
	Settings	Redirects the users to the Pipeline Settings page

6.1.1. Resource Allocation

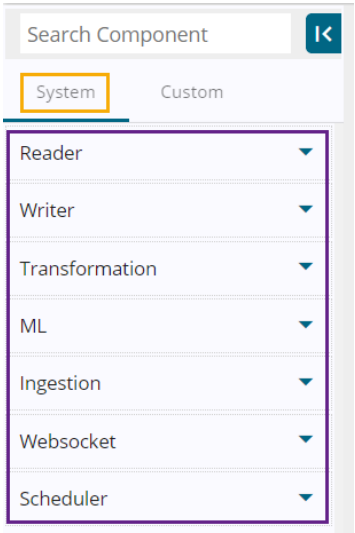
The following is used to deploy the pipeline with high, medium, or low-end configurations according to the velocity and volume of data that the pipeline must handle.

All the components saved in the pipeline are then allocated resources based on the selected Resource Allocation option depending on the component type (Spark and Docker).



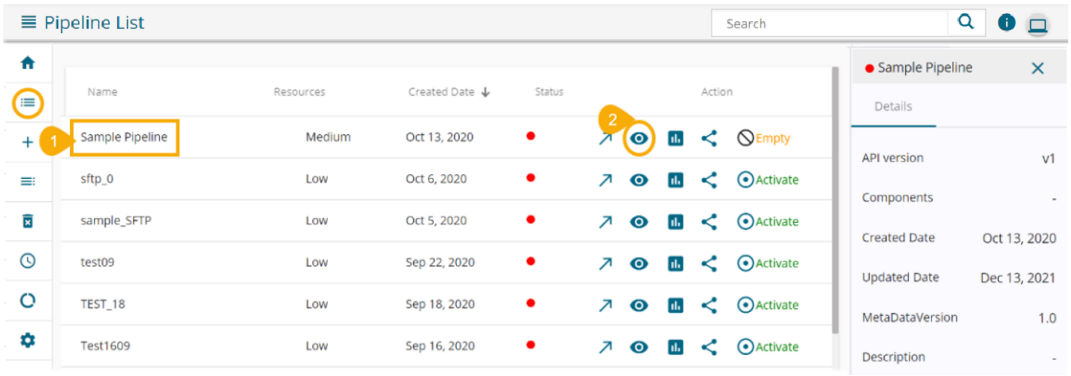
6.2. Adding Components to the Canvas

The Component Pallet is situated on the left side of the User Interface. It has a System and Custom Components tab listing the various components.



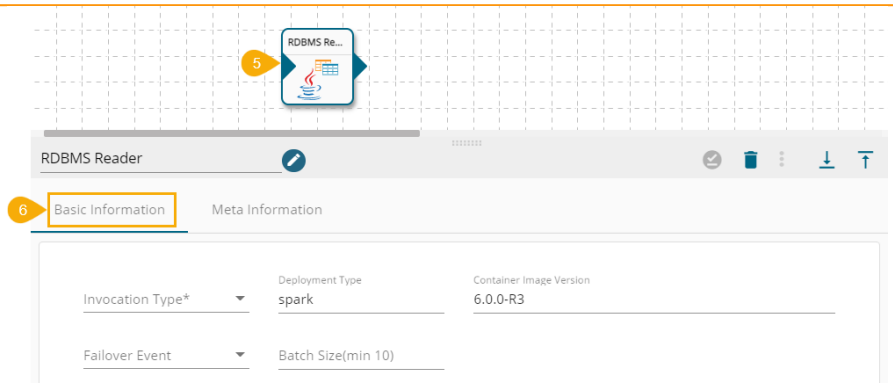
Once the Pipeline gets saved in the pipeline list, the user can add components to the canvas. The user can drag the required components to the canvas and configure it to create a Pipeline workflow or Data flow.

i) Navigate to the existing data pipeline from the Pipeline List page.
ii) Click the '**View**' icon for the pipeline.
(The user can perform this step-in continuation to the pipeline creation, but in case if the user has come out of the Pipeline Editor the above steps help him to access it again.)



iii) The Pipeline Editor opens for the selected pipeline.
iv) Drag and drop the new required components or make changes in the existing component's meta information or change the component configuration (E.g., the RDBMS Reader is dragged to the workspace in the below-given image).

v) Click on the component.
vi) The component-specific fields open asking the basic information about the dragged component:
a. Select an Invocation Type using the drop-down menu.
b. The Deployment Type appears based on the selected component.
c. The Container Image version displays the docker version.



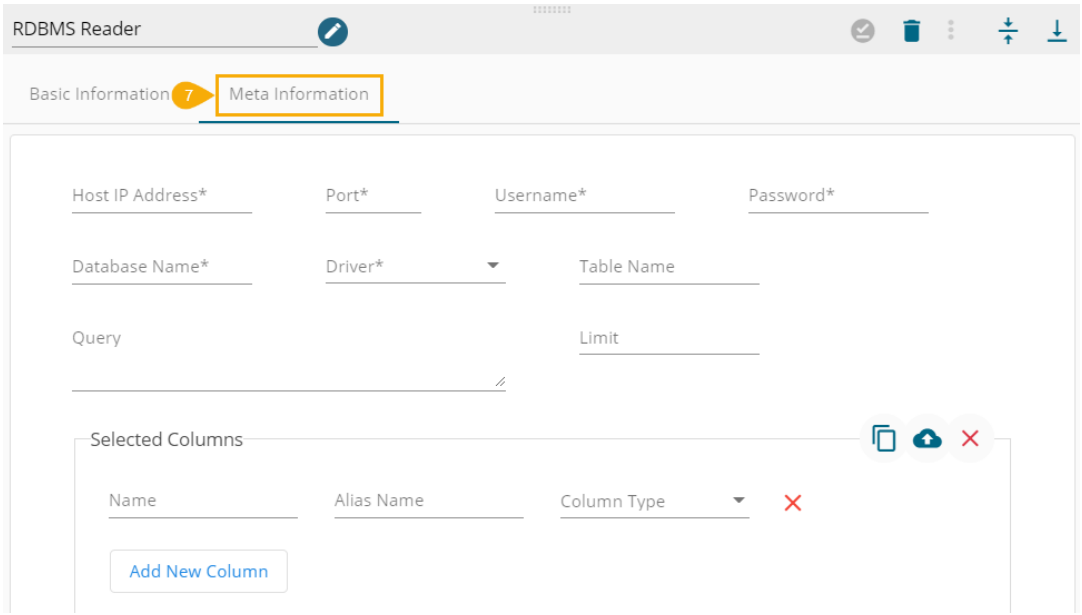
vii) Open the Meta Information tab and configure the required information for the dragged component.

i. Host IP Address
ii. Port number
iii. Username
iv. Password
v. Database Name
vi. Driver- Select from the drop-down menu
vii. Table Name
viii. Query
ix. Limit
x. Selected Columns: Fill the following details manually to select the required columns
   1. Name
   2. Alias Name
   3. Column Type

   Or

   Use 'Download Data' and 'Upload File' options to select the desired columns.

i. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
ii. Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.

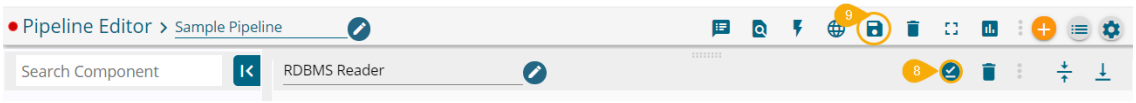


Note: The users can also upload the downloaded JSON file to another component using the 'Upload File'

icon.

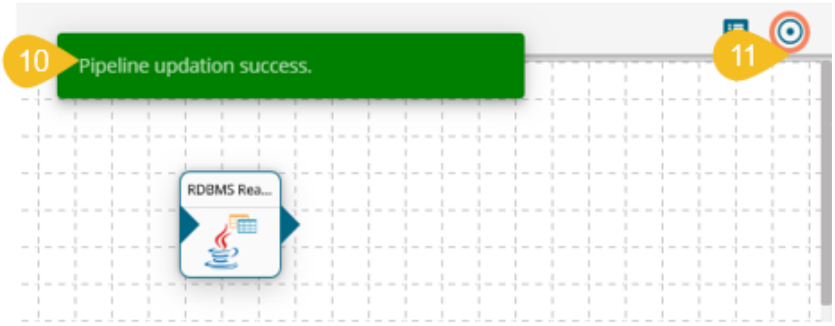
viii) Make sure to click the '**Save Component in Storage**' icon to update the component details and pipeline to reflect the recent changes in the pipeline. (The user can drag and configure other required components to create the Pipeline Workflow.

ix) Click the '**Update Pipeline**' icon to save the changes.



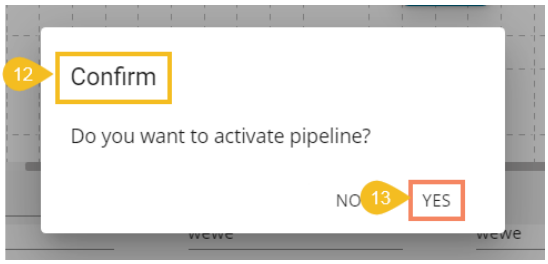
x) A success message appears to assure that the pipeline has been successfully updated.

xi) Click the '**Activate Pipeline**' icon to activate the pipeline (It appears only after the newly created pipeline gets successfully updated).

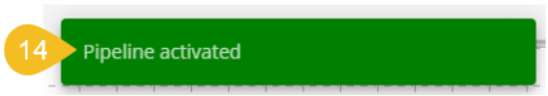


xii) A dialog window opens to confirm the action of pipeline activation.

xiii) Click the '**YES**' to activate the pipeline.

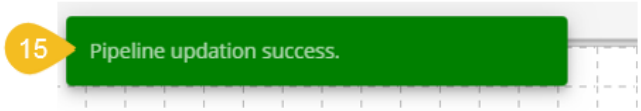


xiv) A success message appears confirming the activation of the pipeline.

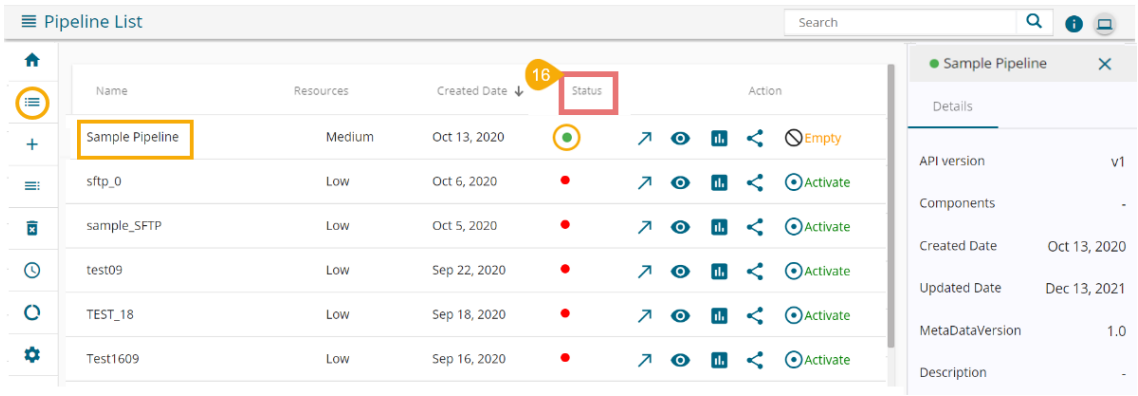


Note: The details of the selected pipeline from the Pipeline List appear on the right side of the page.

xv) Another success message appears to confirm that the pipeline has been updated.

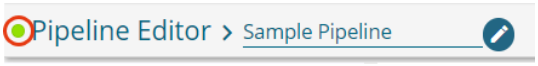


xvi) The Status for the pipeline gets changed on the Pipeline List page.



Note: Click the '**Delete**' icon from the Pipeline Editor page to delete the selected pipeline. The deleted Pipeline gets removed from the Pipeline list.

The status (activated/deactivated) of the pipeline is indicated through color-coding as highlighted in the below image. It turns green if the pipeline is active and remains red for the inactive pipeline.



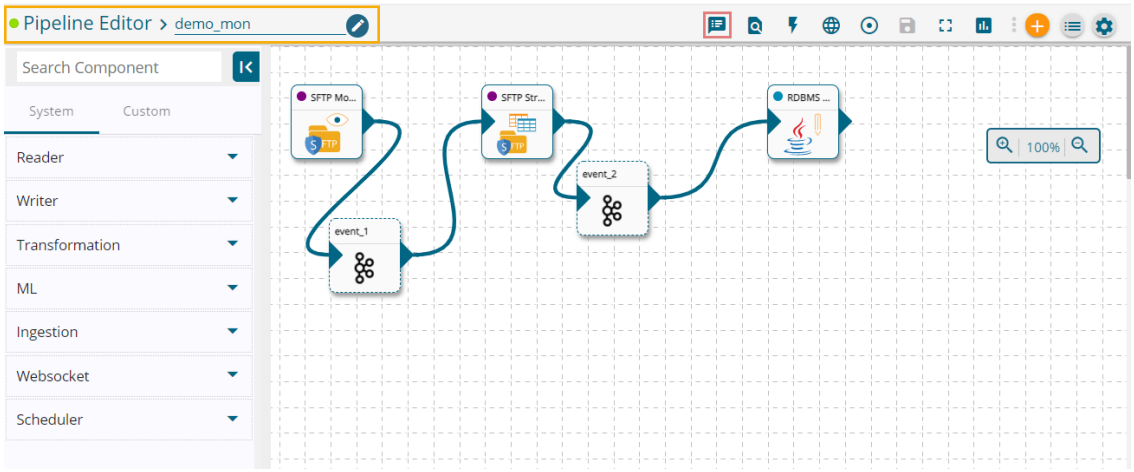
Please refer to the Component Pallet section to get detailed information on the Components.

6.3. Toggle Log Panel

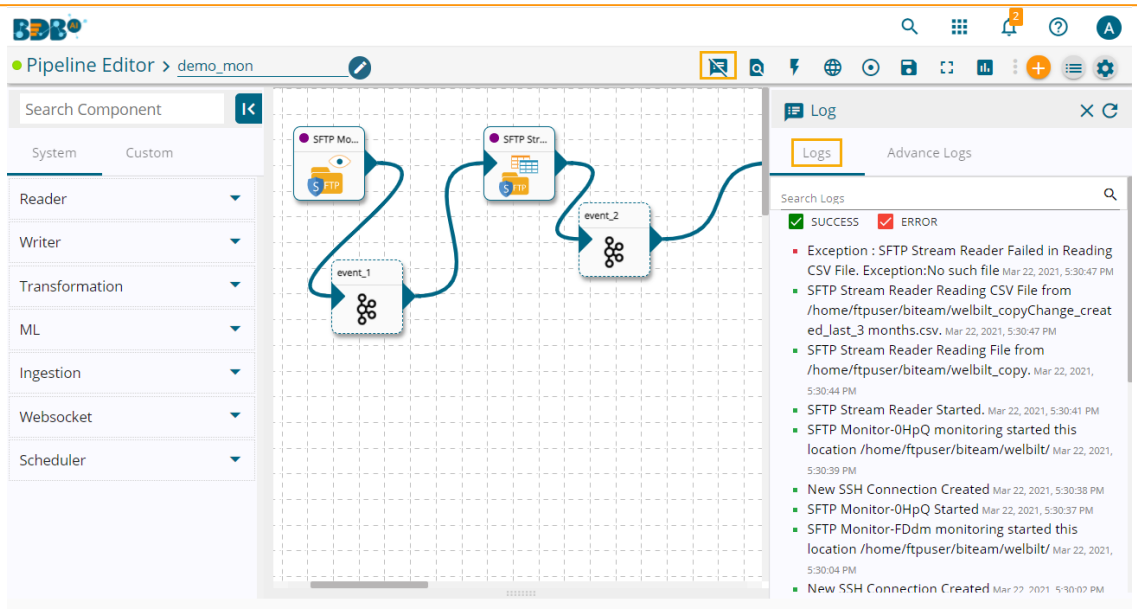
The Toggle Log Panel displays the Logs and Advanced Logs tabs for the Pipeline Workflows.

i) Navigate to the '**Pipeline Editor**' page.

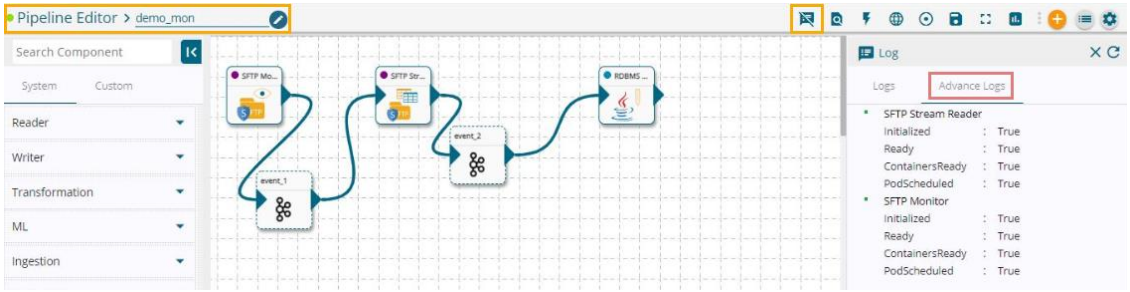
ii) Click the '**Toggle Log Panel**' icon on the header.



iii) A panel **Toggles** displaying the collective component logs of the pipeline under the '**Logs'** tab.



iv) Select the '**Advanced Logs'** tab to display the status of the component containers.

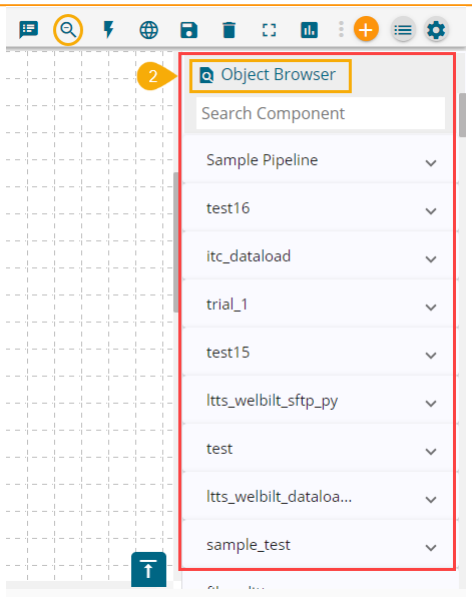


6.4. Search Component in Pipelines

This feature helps the user to search a specific component across all the existing pipelines. The user can drag the required components to the pipeline editor to create a new pipeline workflow.

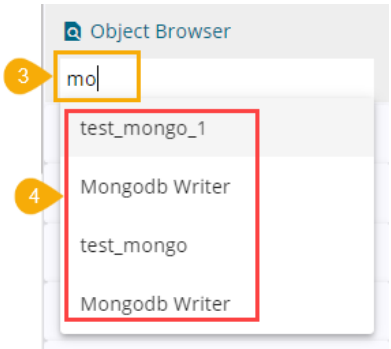
i) Click the '**Search Component in pipelines**' icon from the header panel of the Pipeline Editor.

ii) The Object Browser window opens displaying all the existing pipeline workflows.



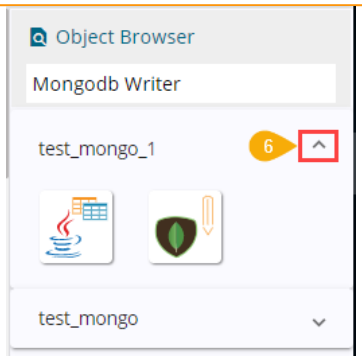
iii) The user can search a component using the '**Search Component**' space.

iv) The user gets prompt suggestions while searching for a component.

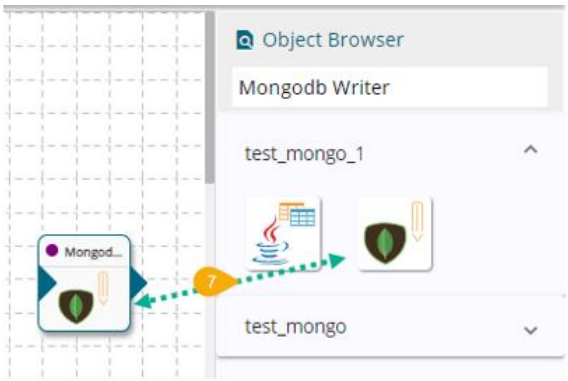


v) Once the component name is entered, the pipeline workflows containing the searched pipeline gets listed below.

vi) The user can click the collapse/expand button to expand the component panel for the selected pipeline.



vii) The user can drag a component to the Pipeline Editor canvass from the object browser window.

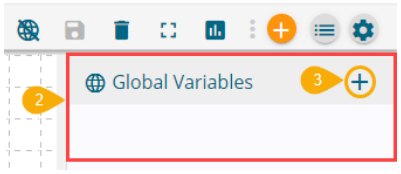


6.5. Global Variable

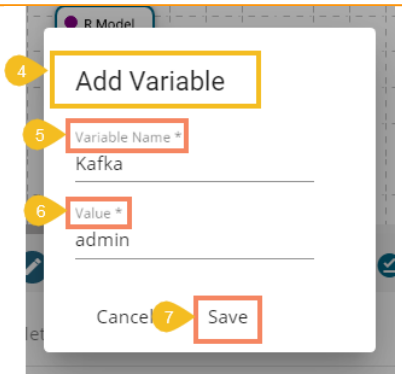
The user can create a Global Variable by following the below-given steps:

i) Click the Global Variable icon from the User Interface.

ii) Global Variables panel opens.
iii) Click the '**Add New Variable**' from the Global Variables panel.

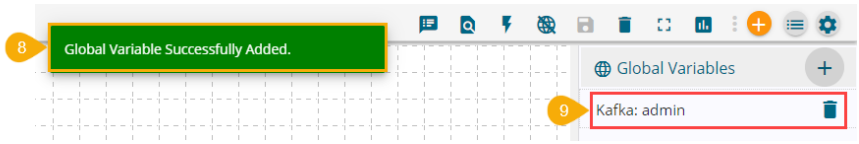


iv) The Add Variable window opens.
v) Insert Variable name.
vi) Provide Value for the Variable.
vii) Click the '**Save**' option.



viii) A notification message appears.

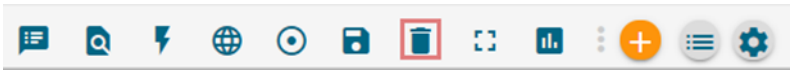
ix) The Global Variable gets created and added to the panel.



6.6. Deleting a Pipeline

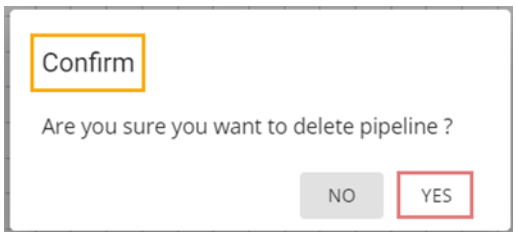
i) Navigate to the Pipeline Workflow Editor.

ii) Click the '**Delete Pipeline**' icon on the header panel.



iii) The '**Confirm**' dialog opens to confirm the deletion.

iv) Click the '**Yes**' option to confirm.



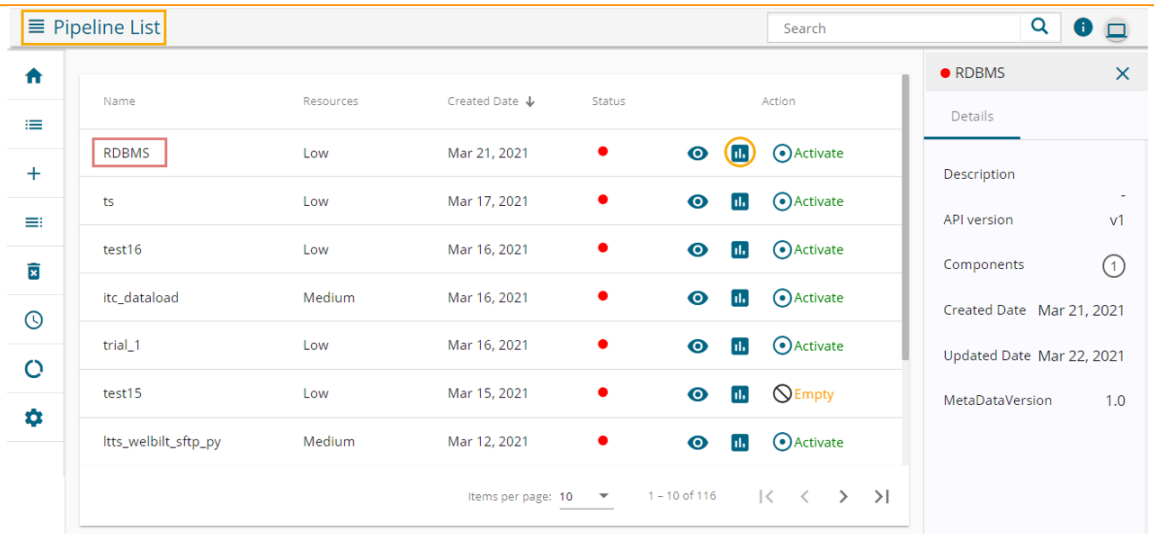
v) The selected pipeline gets deleted and gets removed from the List of Pipelines.

6.7. Pipeline Monitoring

The user can monitor a pipeline together with all the components associated with the same by using the '**Pipeline Monitoring**' icon. The user gets information about Pipeline components, Status, Types, Last Activated (Date and Time), Last Deactivated (Date and Time), Total Allocated and Consumed CPU %, Total allocated and consumed memory, Number of Records, and Component logs all displayed on the same page.

i) Navigate to the Pipeline List page.

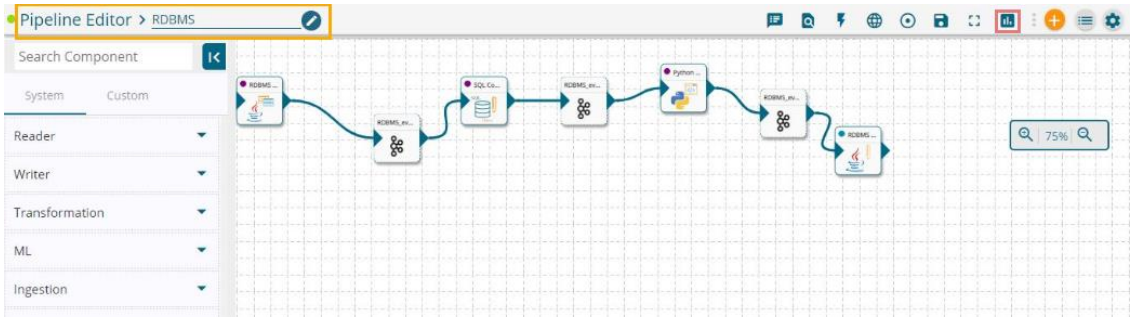
ii) Click the '**Monitor**' icon.



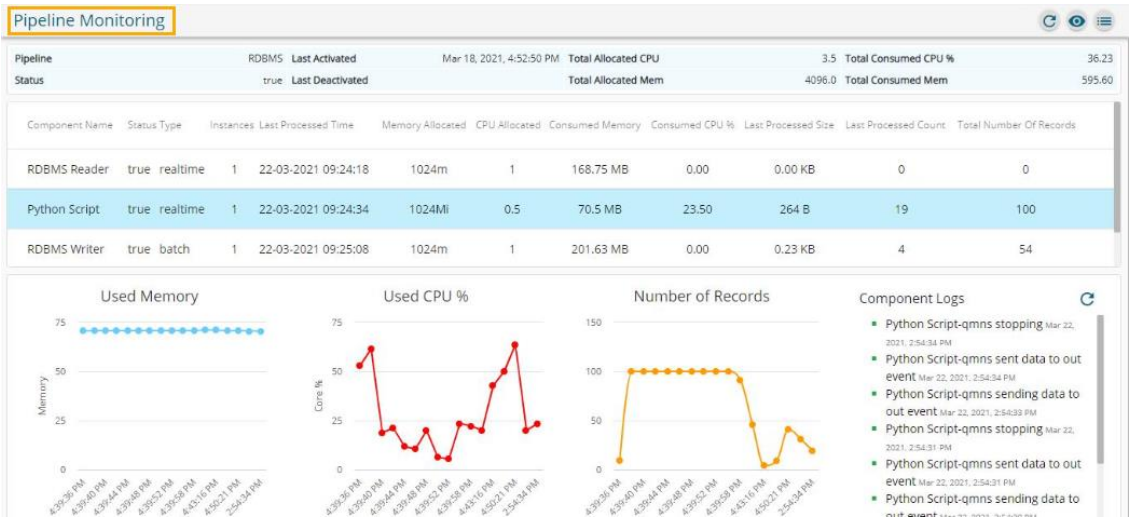
Or

Navigate to the Pipeline Workflow Editor page.

Click the '**Pipeline Monitoring**' icon on the Header panel.



iii) The '**Pipeline Monitoring**' page opens displaying details of the selected pipeline.



6.8. Connecting Components

Each component in the Data Pipeline works as an independent entity. The user needs to create an Event as a Kafka-based messaging channel to create a data flow. An Event helps the users to read data from various

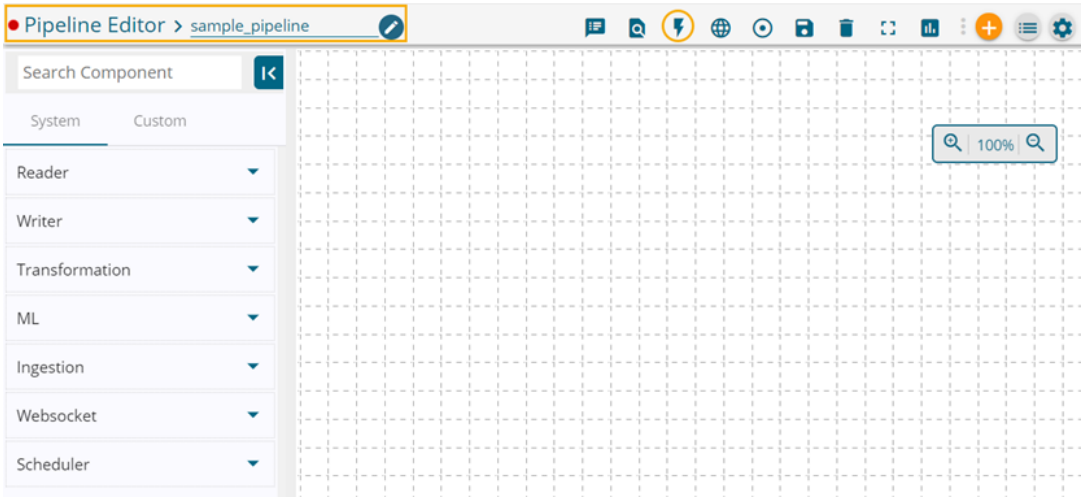
readers or use the data coming from one pipeline into another pipeline. The Kafka Event helps to deal with the scenarios where one pipeline is dependent on another pipeline's data.

There are two tabs in the Event Panel as described below:

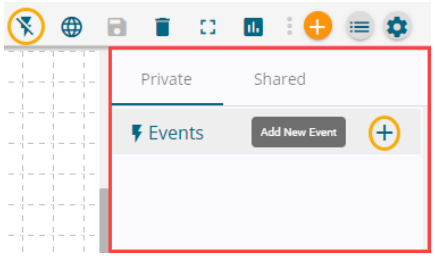
i) Private: This tab contains a list of all the created events by the user for the current pipeline.
ii) Shared: This tab contains a list of all the shared events from another pipeline.

6.8.1. Event Creation

i) Navigate to the Pipeline Editor page.
ii) Click the '**Toggle Event Panel**' icon from the header.



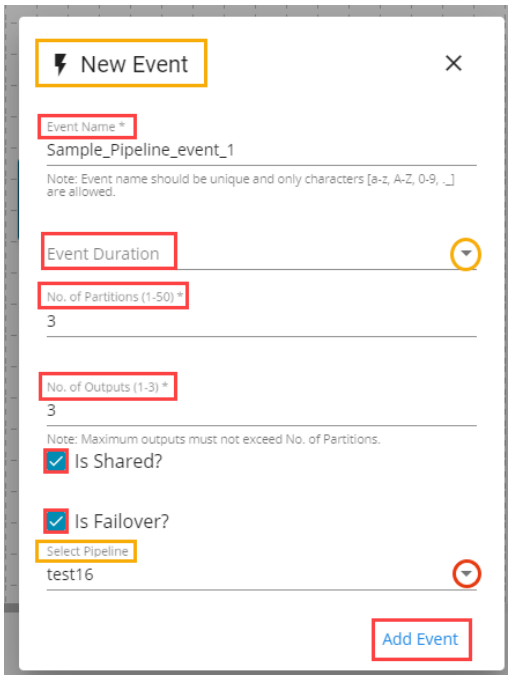
iii) The Events Panel appears, and the Toggle Event Panel icon gets changed as suggesting that the event panel is displayed.
iv) Click the '**Add New Event**' icon from the Event Panel.



v) The New Event window opens.
   a. Provide a name for the new Event.
   b. Select Event Duration: Select an option from the below-given options.
      i. Short (4 hours)
      ii. Medium (8 hours)
      iii. Long (48 hours)

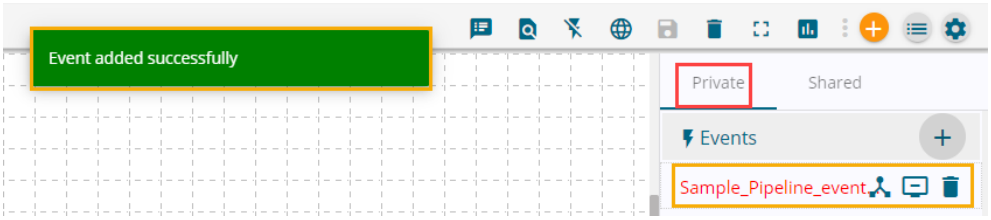
      Note: The event data gets erased after 7 days if no duration option is selected from the available options.
   c. Provide No. of Partitions (1-10).
   d. No. of Outputs: The user can define the number of outputs using this field.
   e. Checkmark the '**Is Shared**' option to make the created Event available as a Shared Event.
   f. Checkmark the '**Is Failover'** option to enable the Failover Event.
   g. Select a Pipeline using the drop-down menu if you have selected the '**Is Shared**?' option.

h. Click the '**Add Event**' option.



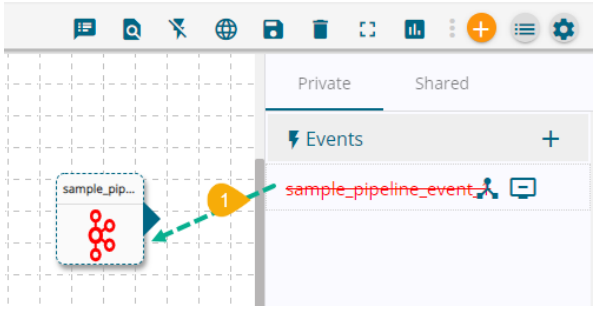
vi) A notification message appears.

vii) The newly created Event gets added to the 'Private' tab in the Events panel.



6.8.2. Updating an Event

i) Drag the created Event component to the Pipeline Editor canvas.

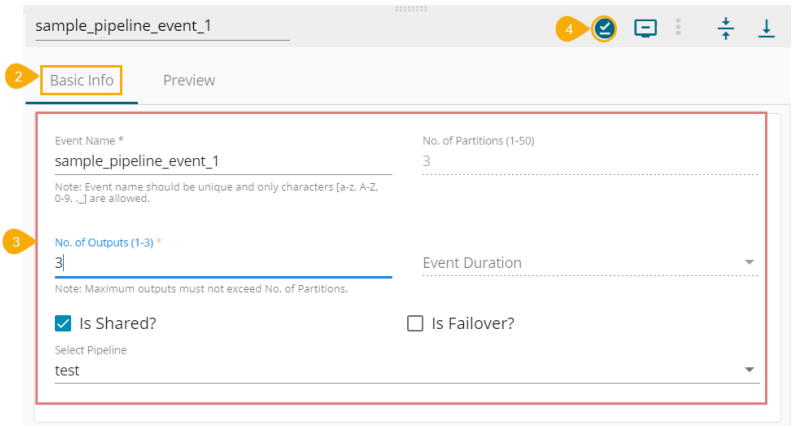


ii) Click the dragged Event component to open the '**Basic Info**' configuration fields.

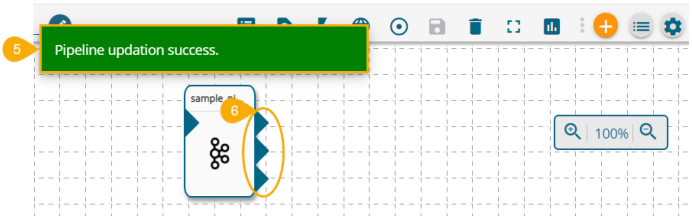
iii) The user can edit the following information (except No. of Partition and Event Duration):

a. Event Name: Modify the event name.
b. No. of Outputs: Set the number of outputs (the maximum allowed number of outputs are 3)
c. 'Is Shared?': Enable this option to share the created event.
d. 'Is Failover?': Enable this option to create a failover event.
e. Select Pipeline: Select one pipeline or multiple pipelines using the drop-down list.

iv) Click the '**Save Event**' icon to save the changes.

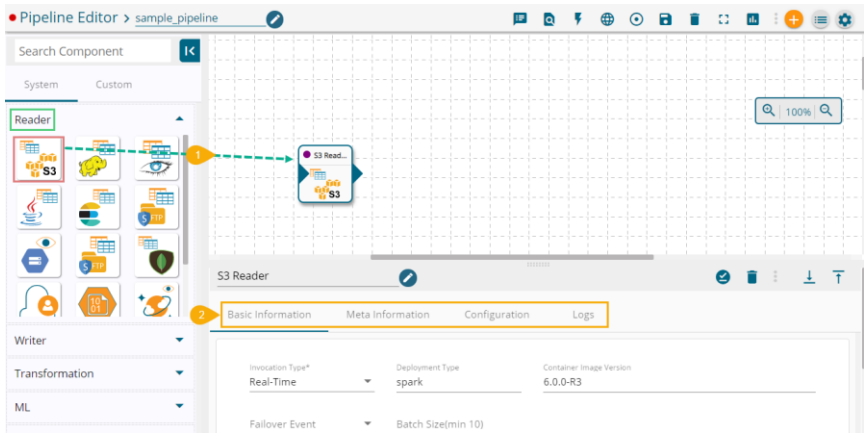


v) The user gets a notification message stating that the pipeline update is a success.
vi) The targeted Event component gets updated.

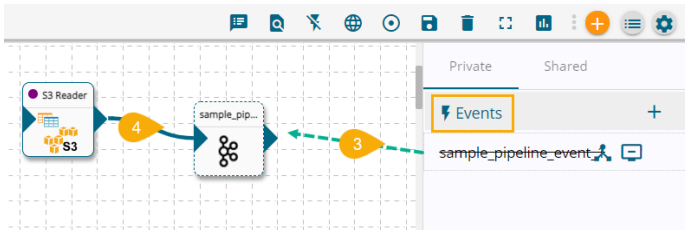


6.8.3. Connecting Event to a Component

i) Drag a (reader) component to the canvas.
ii) Configure the parameters of the dragged reader.



iii) Drag the Event from the Events Panel.
iv) Connect the dragged reader component as the input connection to the dragged Event.

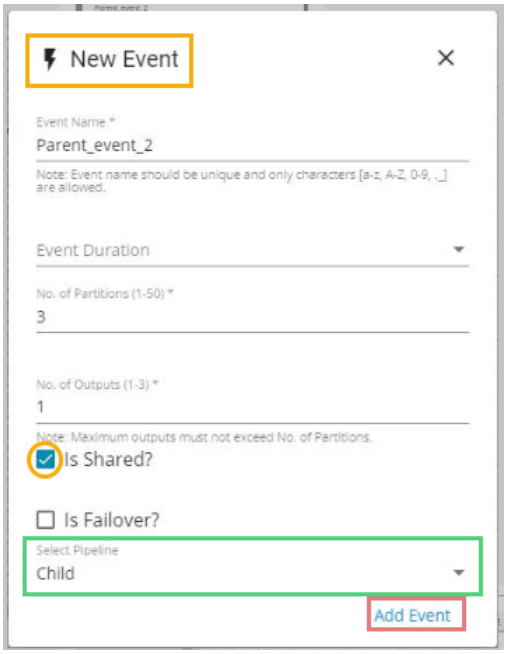


v) Drag a Writer component to the Editor canvass from the Component Pallet.
vi) Configure the writer component.
vii) Connect the Writer component as an output connection to the dragged Event to complete the data flow.
viii) Click the '**Update Pipeline**' icon to save the pipeline workflow.
ix) Click the '**Activate Pipeline**' icon to activate the Pipeline.

Note: The user gets notification messages on the successful update and activation of the Data Pipeline.

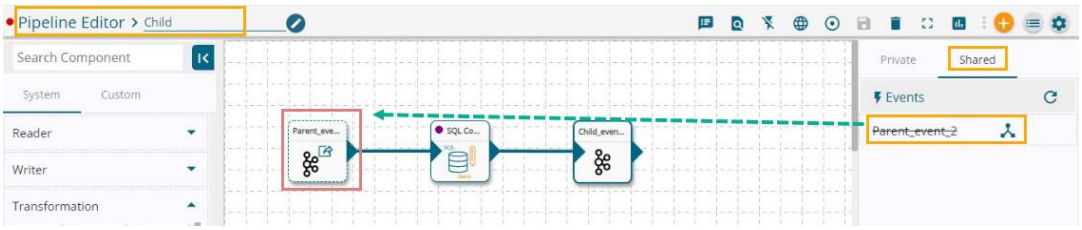
6.8.4. Shared Event

i) Enable the '**Is Shared?**' option while creating an Event.



ii) Open the pipeline where the same event is shared.
iii) The shared event gets added to the '**Shared**' section of the Event.

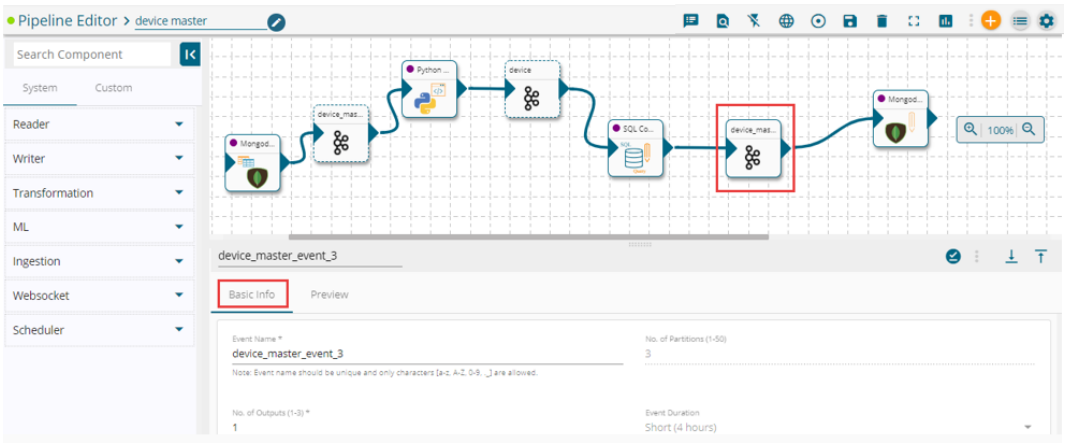
iv) The user can use the Shared Event in the new pipeline with a writer as displayed in the following image:



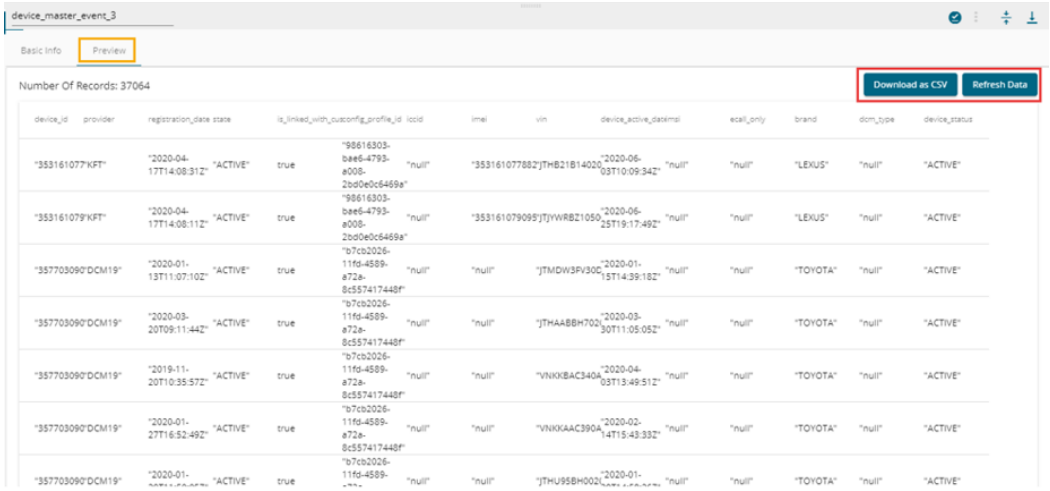
6.8.5. Preview Data

The user can preview the fetched data from the configuration panel of the Event component in the pipeline workflow.

i) Navigate to a pipeline workflow (The pipeline should be activated).
ii) Click on a connected Event component.
iii) The Basic Information tab of the Event Component opens with the default configuration which is given by the user on Event creation.



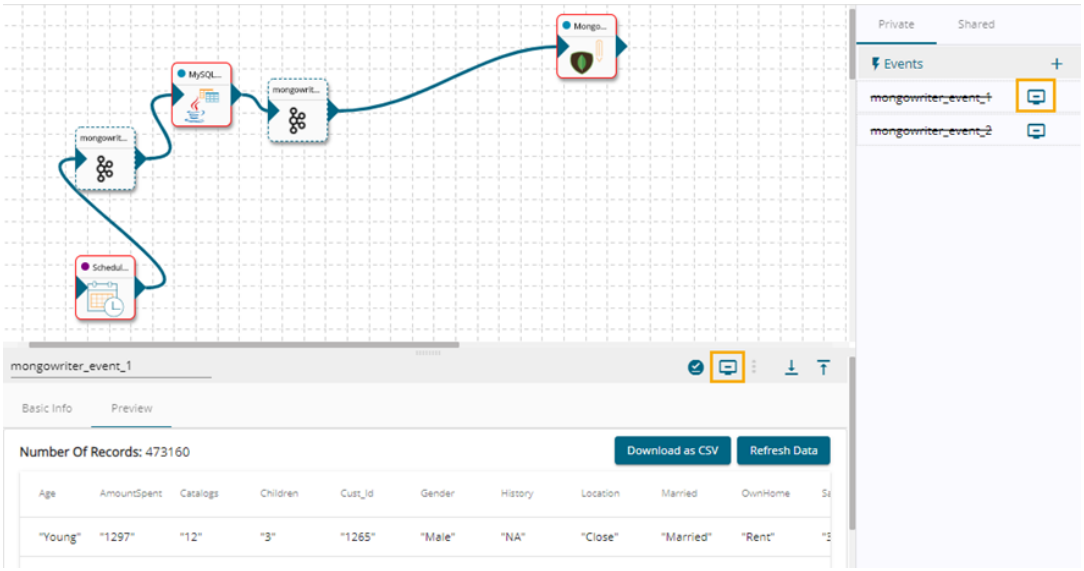
iv) Click the '**Preview**' tab from the Event configuration panel.
v) The Preview Data tab appears displaying the fetched data with the count of the Number of Records.
vi) Click the '**Download as CSV**' option to download the data in a CSV file.
vii) Click the '**Refresh Data**' option to refresh the fetched data.



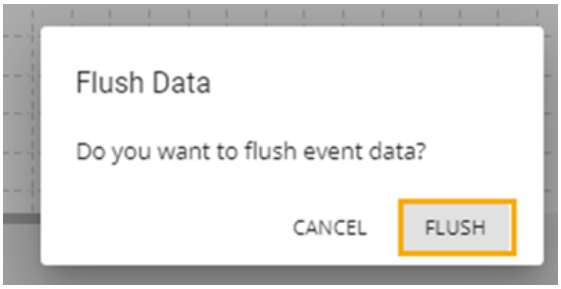
6.8.6. Flush Event

This feature helps the user to clear the fetched data from the Kafka Event by using the Flush Event.

i) Click on an Event component.
ii) Click on the Preview option from the configuration panel.
iii) Click the '**Flush Event**' icon.

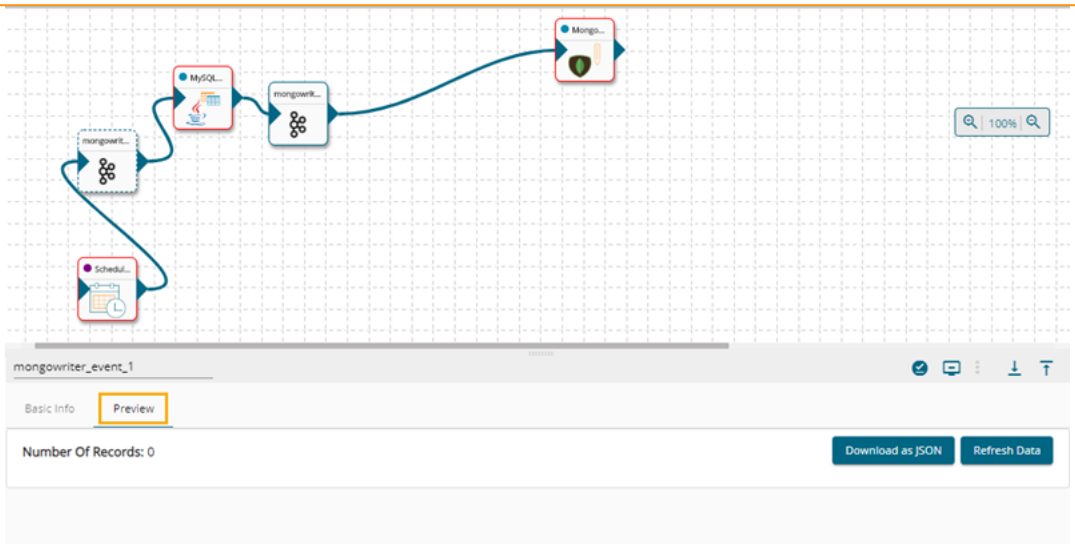


iv) The Flush Data dialog window opens.
v) Click the '**FLUSH**' option.



vi) A notification message appears to inform that the data has been flushed.

vii) Open the Preview tab, there will be no fetched data to preview.



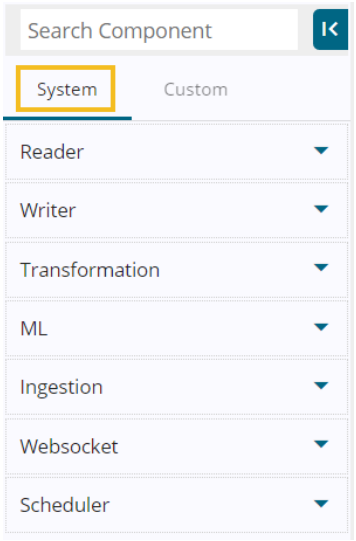
7. Component Pallet

The user interface of the Data Pipeline has a component pallet on the left side of the page. The Component Pallet lists System and Custom components of the Data Pipeline.

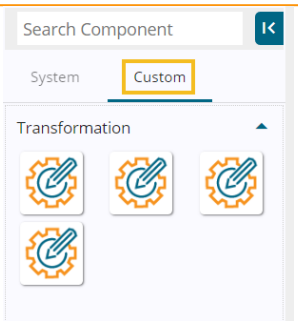
7.1. Sorting and Grouping Components

The Component Pallet displays a list of all the pipeline components. The Components are majorly divided into two groups.

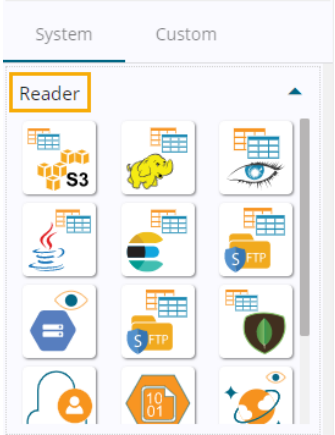
i) System-The pre-designed pipeline components are listed under the '**System**' tab.



ii) Custom- The Custom tab lists all the customized pipeline components created by the user.



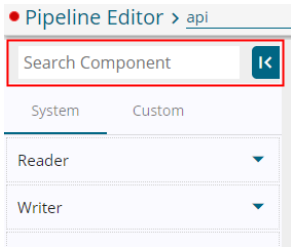
The Component gets grouped based on the Component type. E.g., All the reader components are provided under the 'Reader' menu tab of the Component Pallet.



7.2. Searching a Component

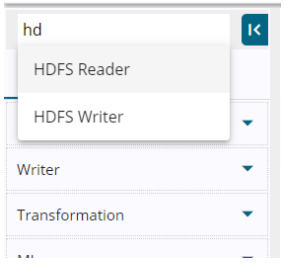
A search bar is provided to search across the 50+ components. It helps to find a specific component by inserting the name in the Search Bar.

i) Navigate to the Search bar provided in the Component Pallet.



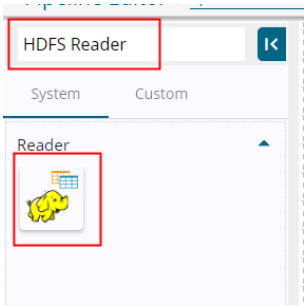
ii) Type in the given search bar.

iii) The user gets prompt suggestions for the searched components. E.g., While searching 'hd', it lists HDFS Reader and HDFS Writer components.



iv) Select an option from the prompted choices.

v) The searched component appears under the 'System' tab. (E.g., HDFS Reader component as displayed in the below image).



8. Component Properties

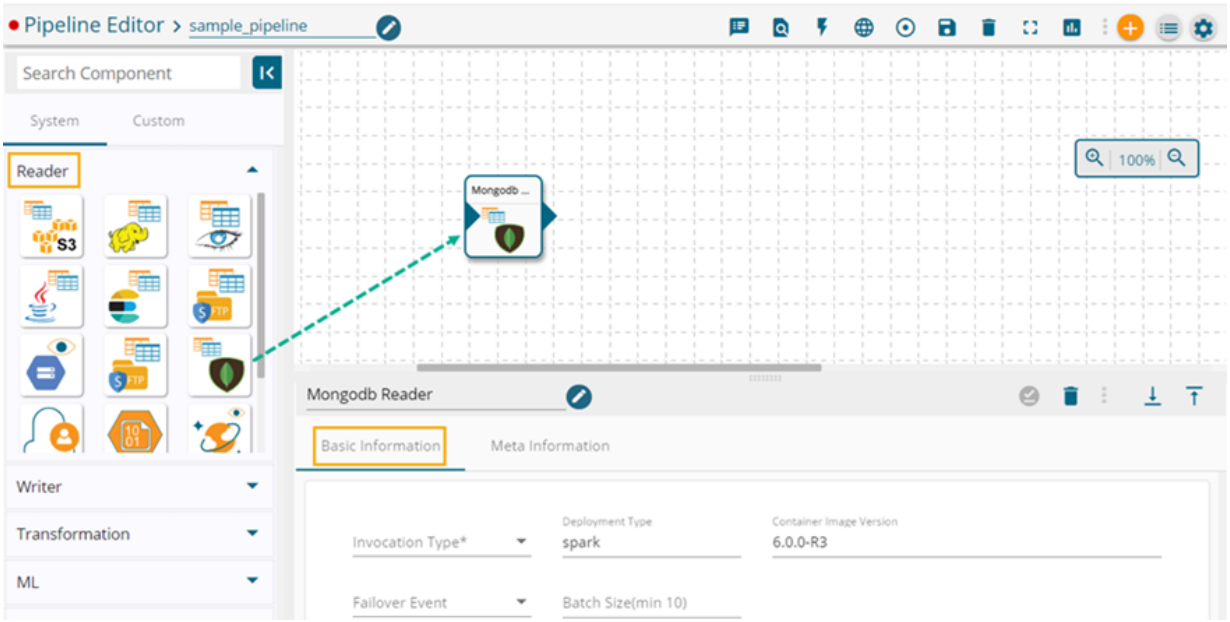
Each component has properties to configure. The user can use a specific component in the Pipeline Workflow only after configuring the required parameters for that component.

This section aims at describing all the component parameters as per the given groups under the Component Pallet.

The following four main tabs appear for each Pipeline component:

1. Basic Information
2. Meta Information
3. Configuration
4. Logs

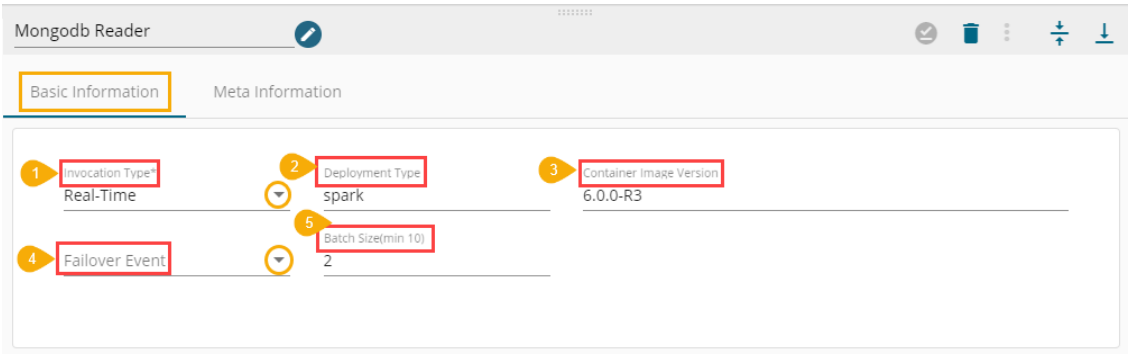
Click on a dragged pipeline component to open the component properties tabs below.



8.1. Basic Information Tab

It is the default properties tab to open for the dragged Pipeline Component while configuring the component (E.g., Mongo DB Reader as displayed in the below image).

- Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.
- Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
- Container Image Version: It displays the image version for the docker container. This field comes pre-defined.
- Failover Event: Select a failover Event from the drop-down menu.
- Batch Size: Provide a maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

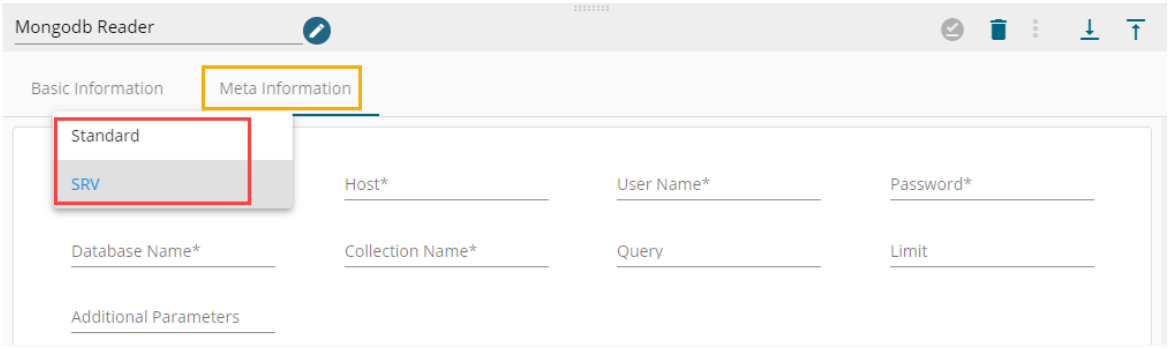


8.2. Meta Information Tab

Open the '**Meta Information**' tab and fill in all the connection-specific details of Mogo DB Reader. The Meta

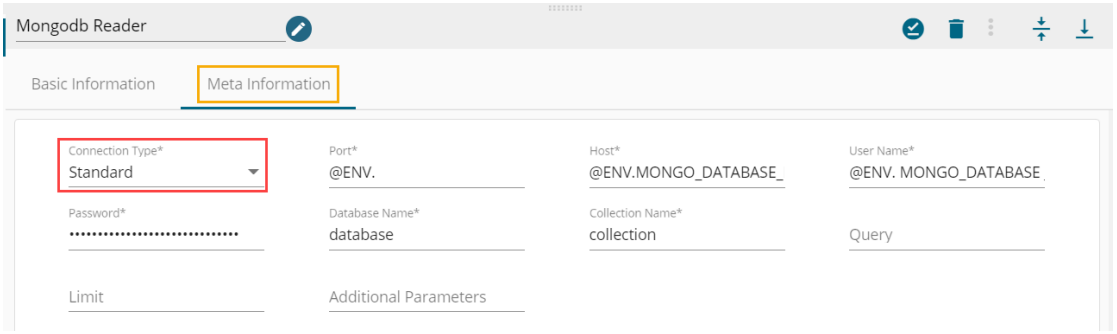
Information details may vary for each component.

The user can configure the Mongo DB Reader via two connection types as explained below:

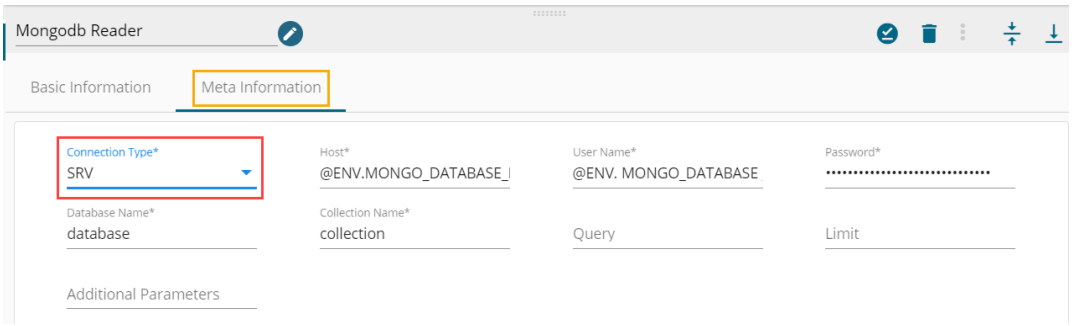


- **Meta Information** tab Configuration with '**Standard**' as the '**Connection Type**'
  - Connection Type: Select the '**Standard**' option as the connection type.
  - Port (*): Port number.
  - Host IP Address (*): IP address of the host.
  - Username (*): Provide username.

- Password (*): Provide a valid password to access the Mongo DB.
- Database Name (*): Provide the name of the database from where you wish to read data.
- Collection Name (*): Provide the name of the collection.
- Query: Insert an SQL query (it takes a query containing a Join statement as well).
- Limit: Set a limit for the number of records.
- Additional Parameters: Provide the details of the additional parameters.



- **Meta Information** tab Configuration with '**SRV**' as the '**Connection Type'**
  - Host IP Address (*): Hadoop IP address of the host.
  - Username (*): Provide username.
  - Password (*): Provide a valid password to access the Mongo DB.
  - Database Name (*): Provide the name of the database from where you wish to read data.
  - Collection Name (*): Provide the name of the collection.
  - Query: Insert an SQL query (it takes a query containing a Join statement as well).
  - Limit: Set a limit for the number of records.
  - Additional Parameters: Provide the details of the additional parameters.



- **Selected Columns**: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

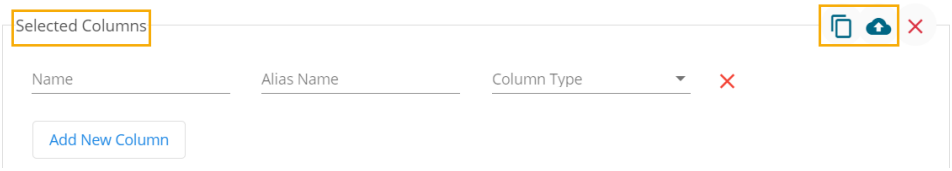
  The supported column types are String, Long, Integer, Float, Double, Date, Datetime.

  or

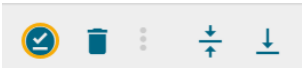
  Use '**Download Data**' and '**Upload File**' options to select the desired columns.

  1. Upload File: The user can upload the existing system files (CSV, JSON) using the '**Upload File**' icon (file size must be less than 2 MB).

2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.



- After doing all the Meta Information configurations click the '**Save Component in Storage**' icon to save the component.



- A notification message appears to inform about the component configuration success.

- Click the '**Update Pipeline**' icon.

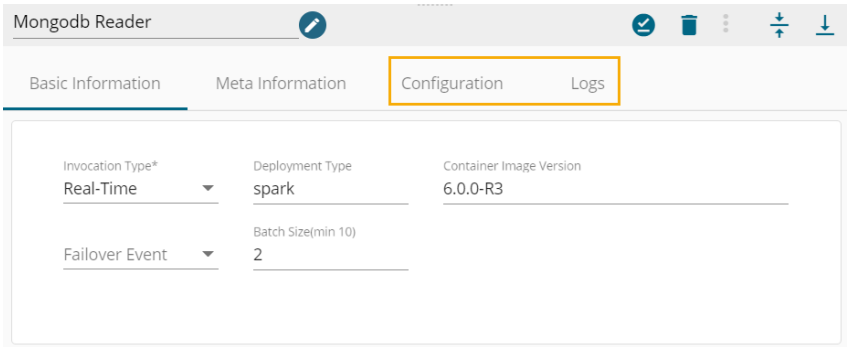
- A notification message appears to inform that the pipeline is successfully updated.

Note: The user needs to click on the '**Save Component in Storage**' icon for each component of a Pipeline Workflow and the '**Update Pipeline**' icon to save the configured component and create pipeline workflow.

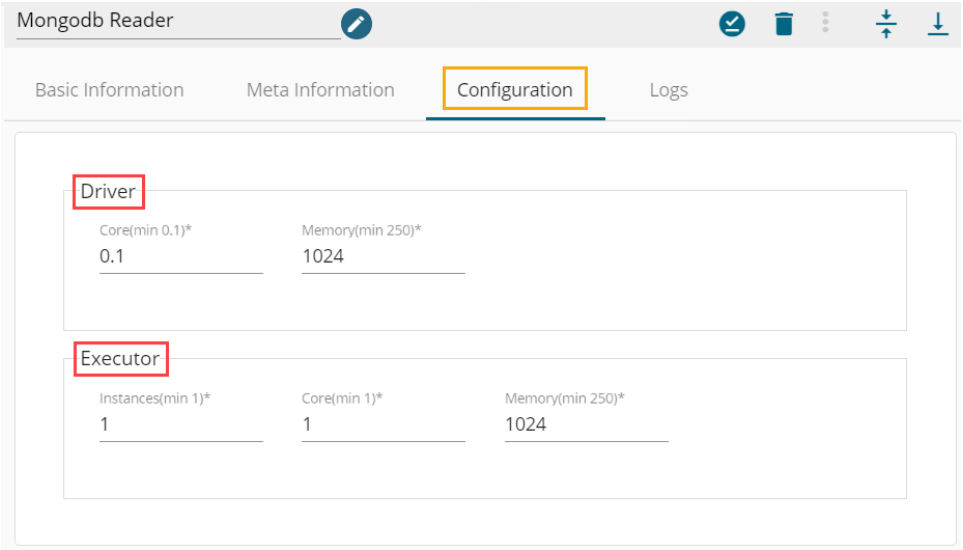
8.3. Configuration and Logs Tabs

The Configuration tab appears after saving the component with Basic and Meta information and updating the Pipeline it is part of.

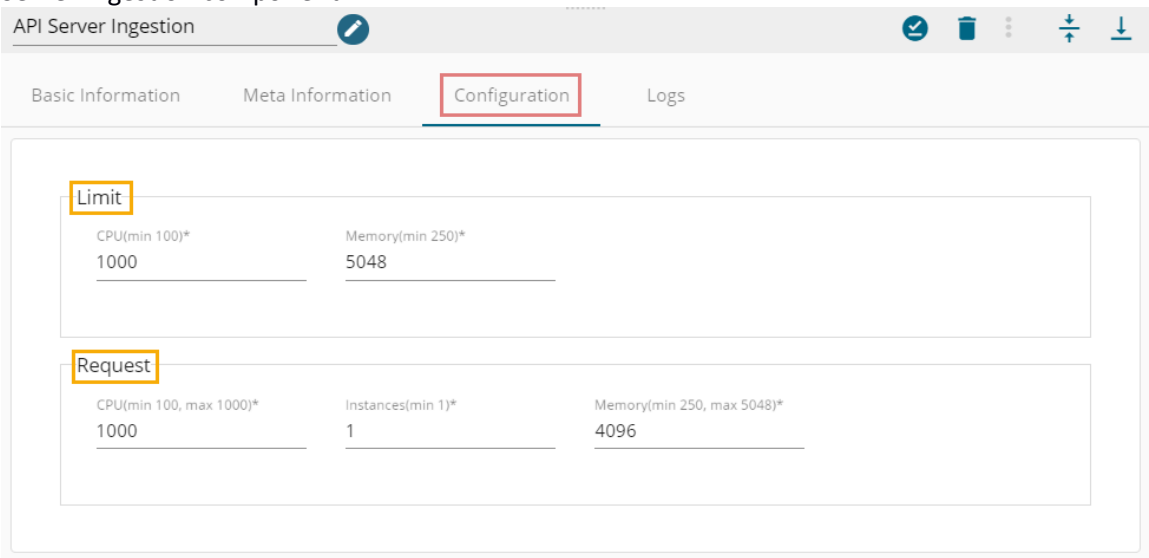
- Open a component from the Pipeline Workflow which has been updated.
- The concerned component gets the '**Configuration**' and '**Logs**' properties tabs.



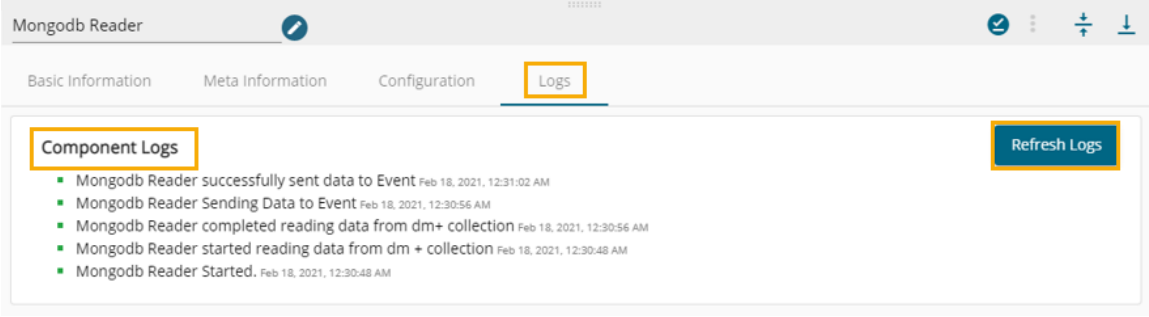
- Click on the '**Configuration**' tab to open it.
- Since the selected component (MongoDB Reader) is a Spark component the Configuration tab displays Driver and Executor information.



- If the concerned component has Deployment Type as Docker, then the Configuration tab will display the details of the Limit and Request. E.g., The below image shows the '**Configuration'** tab for the API Server Ingestion component.



- Click on the '**Logs**' tab to open the Component Logs for a specific pipeline component (E.g., Component Logs for the MongoDB Reader are displayed in the following image).



Note:

a. The **Basic Information** and **Meta Information** tabs need to be configured by the user before saving the concerned component while the **Configuration** and **Logs** tabs appear after saving the component.

b. The user can modify the values for the Spark or Docker components (if needed) using the '**Configuration**' tab.

c. The user needs to click the '**Save Component in Storage**' icon after modifying the '**Configuration**' tab.

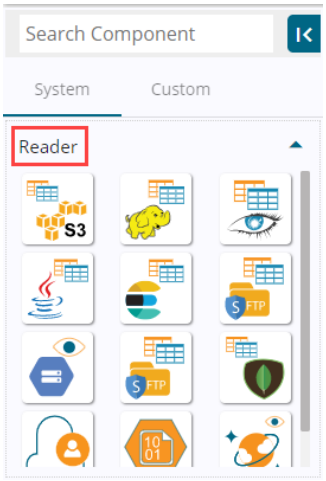
9. Components

The Pipeline Editor contains a Component Pallet containing System and Custom components to create Pipeline Workflow.

9.1. Reader

The data Reader component helps to read data from different sources and ingest that data into the Pipeline for further processing.

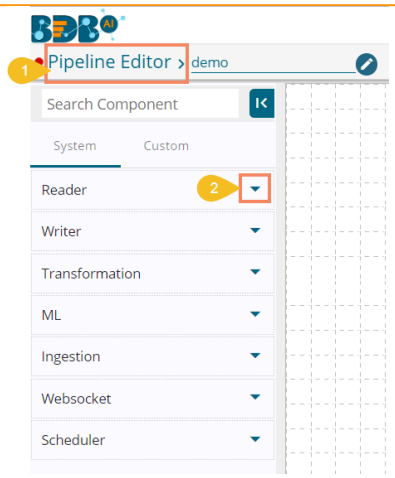
The Data Pipeline contains the following Data Readers under the Component Pallet:



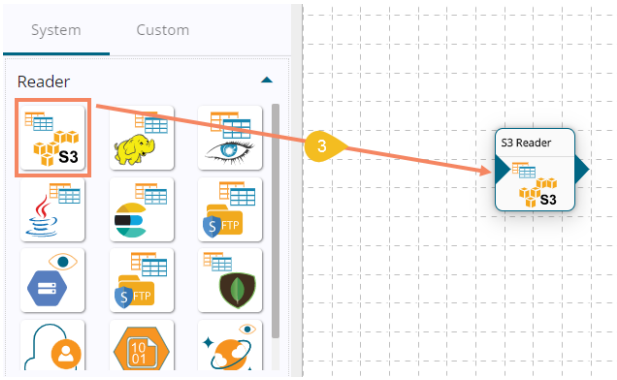
9.1.1. S3 Reader

i) Navigate to the Data Pipeline Editor.

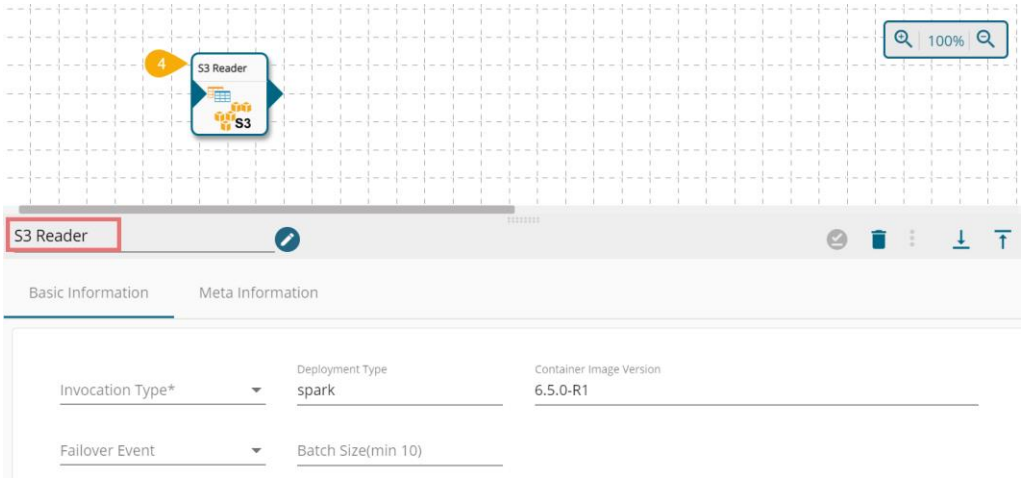
ii) Expand the Reader section provided under the Component Pallet.



iii) Drag and drop the S3 Reader component to the Workflow Editor.



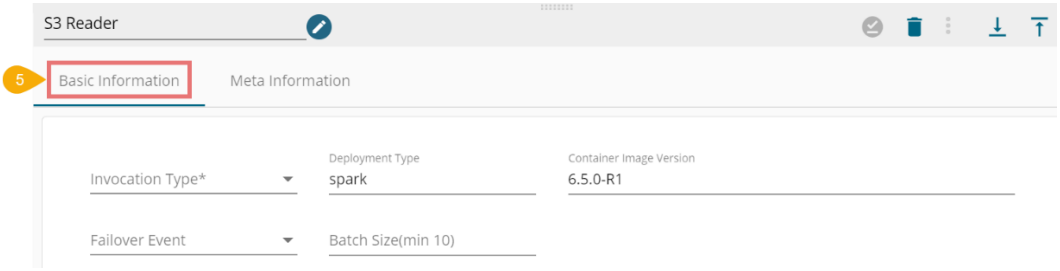
iv) Click on the dragged S3 Reader to get the component properties tabs.



v) **Basic Information**: It is the default tab to open for the component while configuring it.

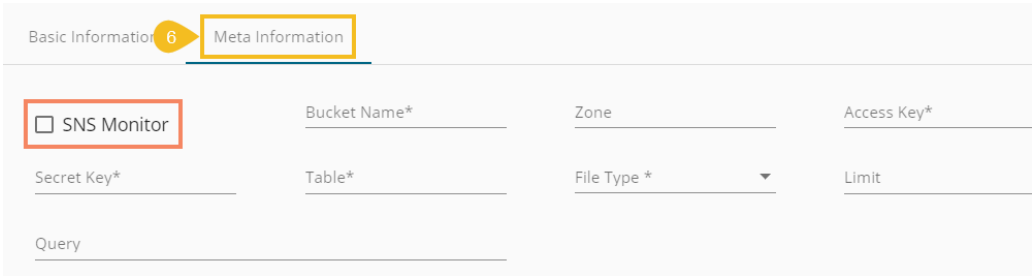
a. Invocation Type: Select an invocation mode out of 'Real-Time' or 'Batch' using the drop-down menu.
b. Deployment Type: It displays the deployment type for the reader component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

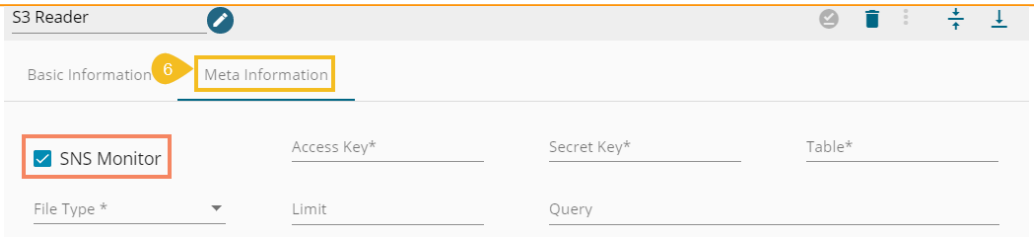


vi) Open the '**Meta Information**' tab and fill in all the connection-specific details for the S3 Reader.

- Configuration fields when SNS Monitor is disabled:
  i) Bucket Name (*): Folder Name
  ii) Zone (*): S3 Zone location
  iii) Access Key (*): Access key shared by AWS to login
  iv) Secret Key (*): Secret key shared by AWS to login
  v) Table (*): Mention the Table or object name which is to be read
  vi) File Type (*): Select a file type from the drop-down menu (CSV, JSON, PARQUET, AVRO are the supported file types)
  vii) Limit: Set a limit for the number of records.
  viii) Query: Insert an SQL query (it supports query containing a join statement as well)



- Configuration fields when SNS Monitor is enabled:
  1. Access Key (*): Access key shared by AWS to login
  2. Secret Key (*): Secret key shared by AWS to login
  3. Table (*): Mention the Table or object name which has to be read
  4. File Type (*): Select a file type from the drop-down menu (CSV, JSON, PARQUET, AVRO are the supported file types)
  5. Limit: Set limit for the number of records
  6. Query: Insert an SQL query (it supports query containing a join statement as well)



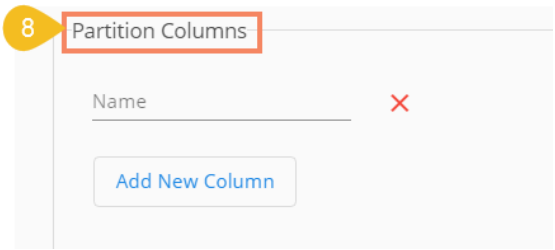
vii) Selected Columns: There is also a section for the selected columns in the Meta Information tab if the user can select some specific columns from the table to read data instead of selecting a complete table so this can be achieved by using the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

or

Use '**Download Data**' and '**Upload File**' options to select the desired columns.

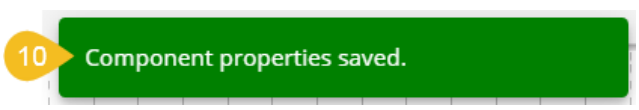
1. Upload File: The user can upload the existing system files (CSV, JSON) using the '**Upload File**' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.

viii) **Partition Columns**: Provide a unique Key column name to partition data in Spark.



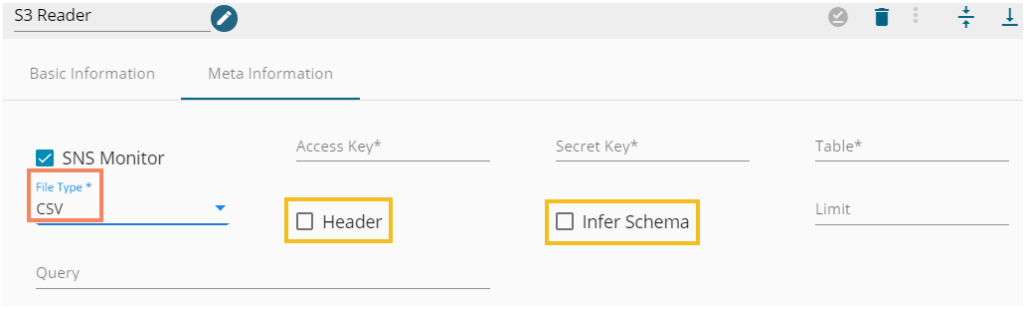
ix) Click the '**Save Component in Storage**' icon after doing all the configurations to save the reader component.

x) A notification message appears to inform about the component configuration success.

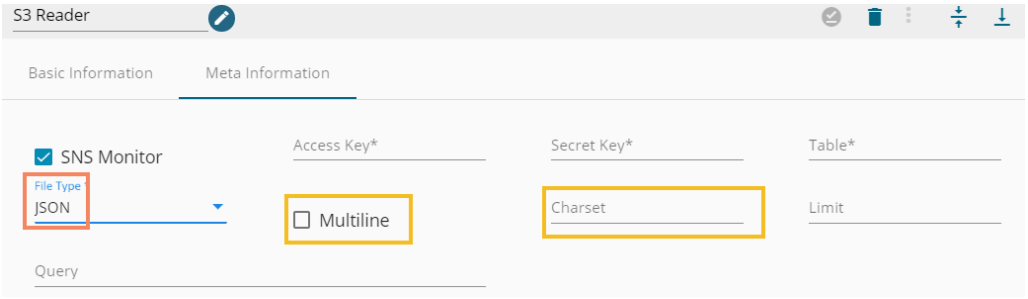


Note:

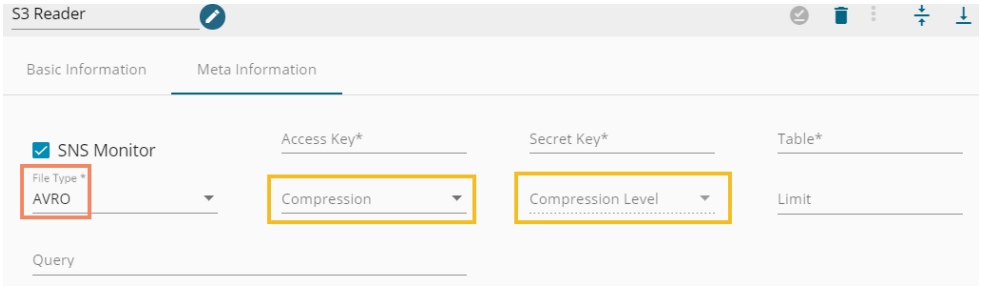
a. (*) the symbol indicates that the field is mandatory.
b. Either table or query must be specified for the data readers except for SFTP Reader.
c. Selected Columns- There should not be a data type mismatch in the Column Type for all the Reader components.
d. The Meta Information fields may vary based on the selected File Type. All the possibilities are mentioned below:
   i. CSV: '**Header**' and '**Infer Schema**' fields get displayed with CSV as the selected File Type.



   ii. JSON: '**Multiline**' and '**Charset**' fields get displayed with JSON as the selected File Type.

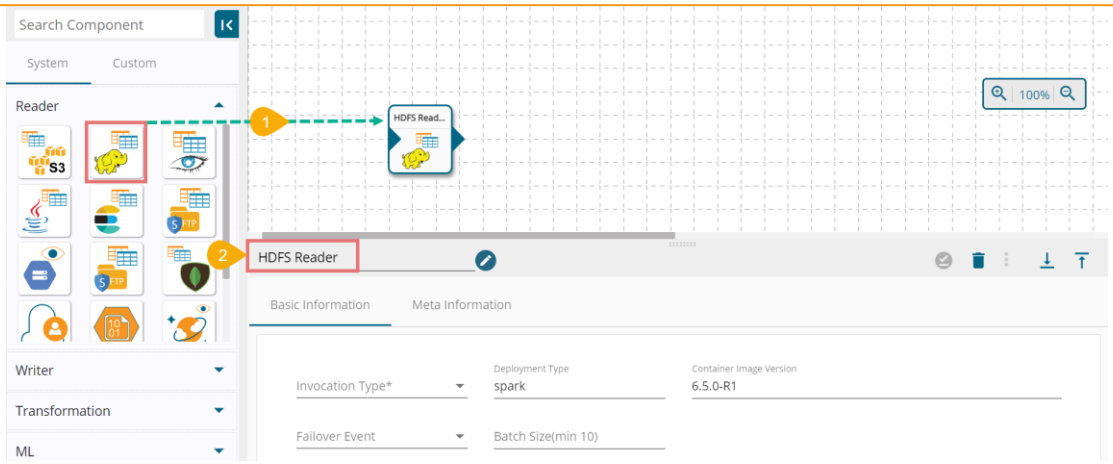


   iii. PARQUET: No extra field gets displayed with PARQUET as the selected File Type.
   iv. AVRO: This File Type provides two drop-down menus.
      1. Compression: Select an option out of the '**Deflate**' and '**Snappy**' options.
      2. Compression Level: This field appears for the Deflate compression option. It provides 0 to 9 levels via a drop-down menu.



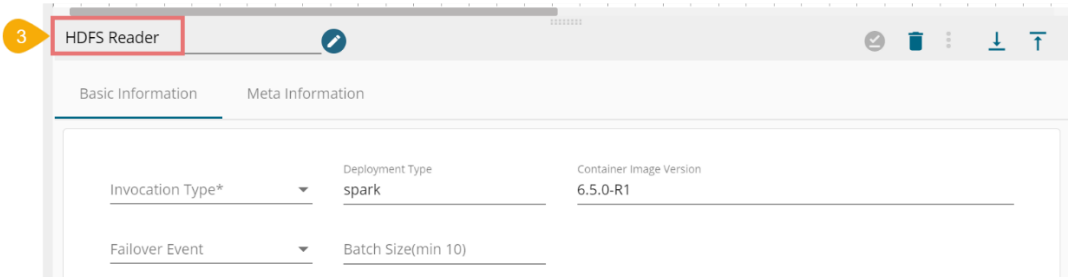
9.1.2. HDFS Reader

i) Drag & Drop the HDFS Reader component on the Workflow Editor.
ii) Click on the dragged reader component to open the component properties tabs below:



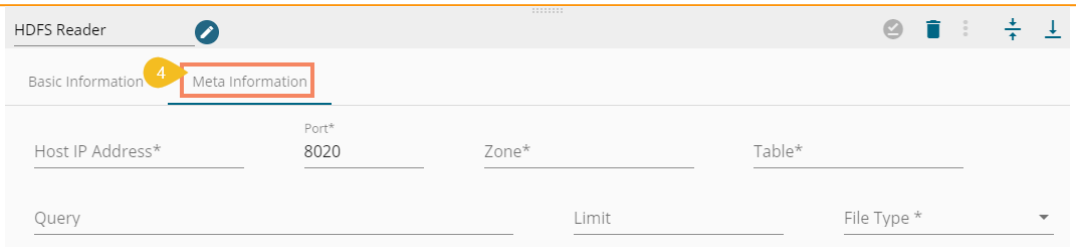
iii) Basic Information: It is the default tab to open for the component while configuring it.

a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the HDFS Reader.

1. Host IP Address (*): Hadoop IP address of the host
2. Port (*): Port number
3. Zone (*): HDFS Zone location
4. Table (*): Table or object name which must be read
5. Query: Insert an SQL query (it takes query containing a Join statement as well)
6. Limit: Set a limit for the number of records.
7. File Type: Select a file type from the drop-down menu (CSV, JSON, PARQUET, AVRO are the supported file types).



i. CSV

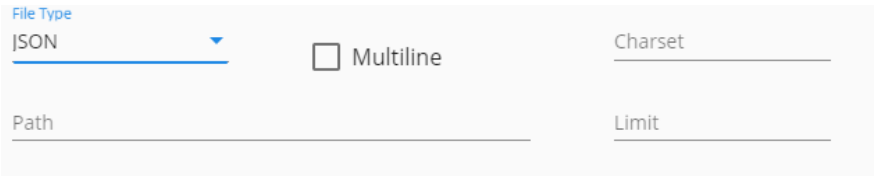
Configure the following fields when the selected file type is CSV:

a. Header- Enable or Disable the Header option to confirm whether the stored data has a header or not
b. Infer Schema- Enable or disable the Infer Schema option to confirm whether the schema is provided or not
c. Path- Provide a specific path for the file
d. Limit- Set limit for the number of rows

ii. JSON

Configure the following fields when the selected file type is JSON:

a. Multiline-Enable this option, if the stored JSON has line breaks in-between
b. Charset-Character set encoding, E.g., UTF-8
c. Path-provide a specific path for the file
d. Limit-set limit for the number of rows



iii. PARQUET

Configure the following fields when the selected file type is PARQUET:

a. Path-provide a specific path for the file
b. Limit- set limit for the number of rows

iv. AVRO

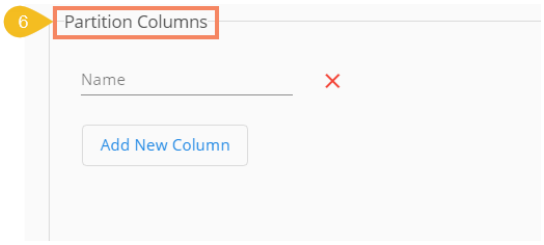
Configure the following fields when the selected file type is AVRO:

a. Compression: Select a compression option from the drop-down menu out of the '**Deflate**' and '**Snappy**' options.
b. Compression Level- If the selected Compression option is '**Deflate**' then select the compression level from 0-9 (where 0 and 9 are included).
c. Path-provide the specific path for the file.
d. Limit- set limit for the number of rows.

v) **Selected Columns**: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the **'Selected Columns'** section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.
or
Use '**Download Data**' and '**Upload File**' options to select the desired columns.

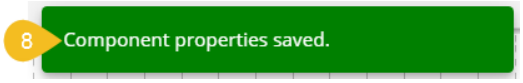
1. Upload File: The user can upload the existing system files (CSV, JSON) using the '**Upload File**' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.

vi) Partition Column- Provide a unique Key column name to partition data in Spark.



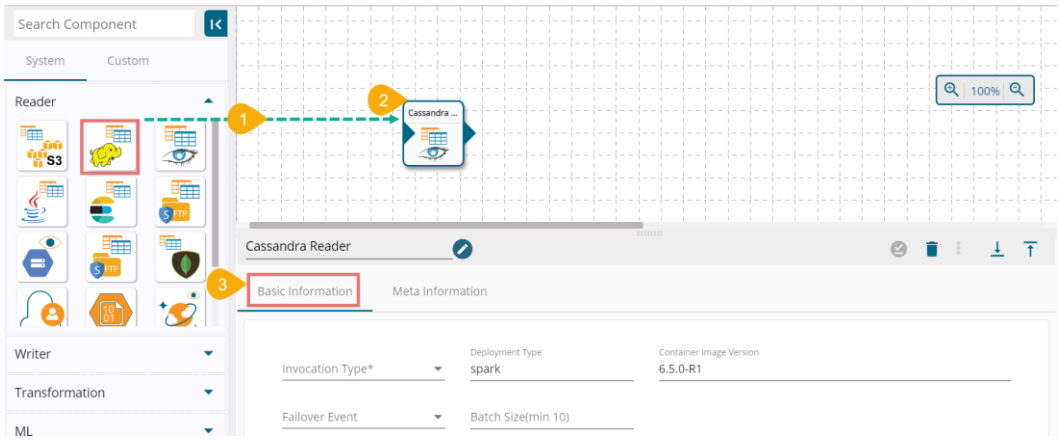
vii) After doing all the configurations click the '**Save Component in Storage**' icon to save the component.

viii) A notification message appears to inform about the component configuration success.

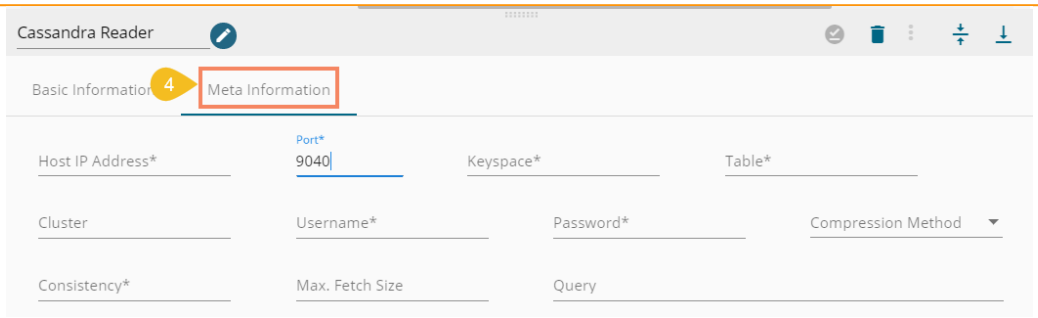


9.1.3. Cassandra Reader

i) Drag & Drop the Cassandra Reader component on the Workflow Editor.
ii) Click on the dragged reader component to open the component properties tabs below.
iii) Basic Information: It is the default tab to open for the Cassandra Reader while configuring the component.
   a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select 'Real-Time' or 'Batch' from the drop-down menu.
   b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   d. Failover Event: Select a failover Event from the drop-down menu.
   e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the Cassandra Reader.
   1. Host IP Address (*): IP Address
   2. Port (*): Host server port number
   3. Keyspace (*): Key for the table
   4. Table (*): Table Name to read data
   5. Cluster: Name of the cluster
   6. Username (*): Provide a username for login
   7. Password (*): provide a valid password for login
   8. Compression Method: Select a compression method from the drop-down
   9. Consistency: the minimum number of Cassandra nodes that must acknowledge the operation
   10. Max. Fetch Size: Max number of records per batch
   11. Query: Insert an SQL query (it takes query containing a Join statement as well)



v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the '**Selected Columns**' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

or

Use '**Download Data**' and '**Upload File**' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the '**Upload File**' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.

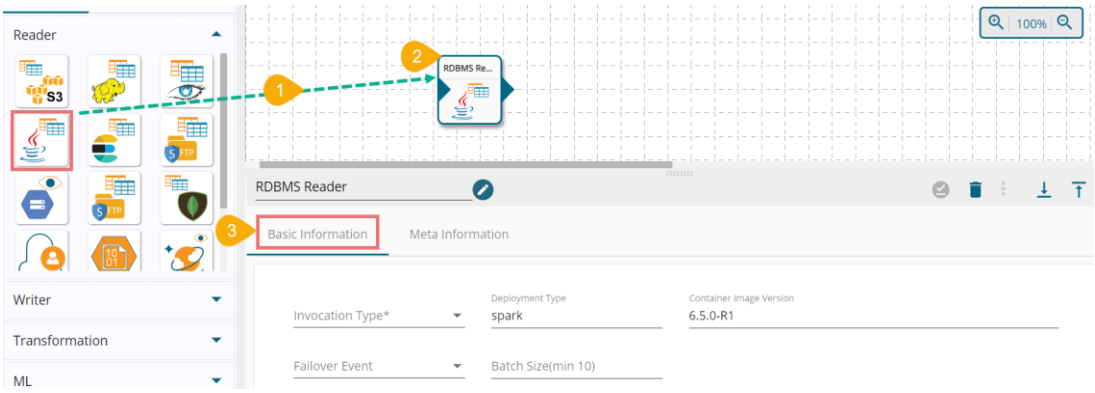
vi) After doing all the configurations click the '**Save Component in Storage**' icon provided in the component configuration panel to save the component.

vii) A notification message appears to inform about the component configuration success.

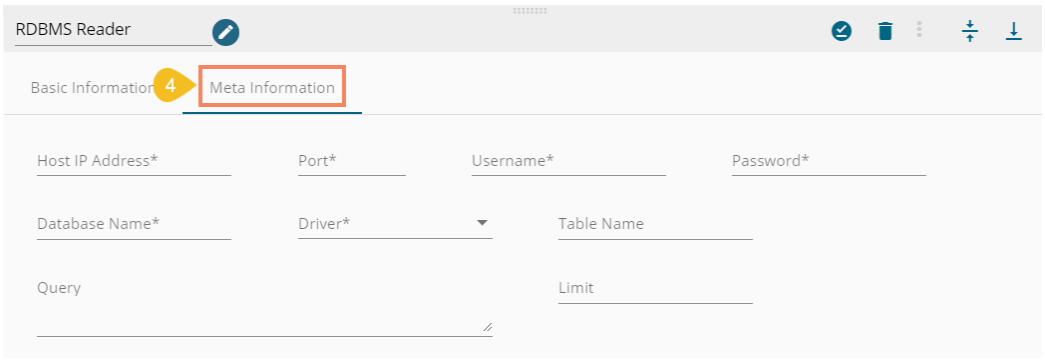
9.1.4. RDBMS Reader

i) Drag & Drop the RDBC Reader component on the Workflow Editor.
ii) Click on the dragged reader component to open the component properties tabs below.
iii) Basic Information: It is the default tab to open for the RDBMS Reader while configuring the component.
   a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select 'Real-Time' or 'Batch' from the drop-down menu.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the RDBMS Reader.
1. Host IP Address (*): IP Address
2. Port (*): Host server port number
3. Username (*): Username for login
4. Password (*): Password for login
5. Database Name (*): Provide the Database name
6. Driver (*): Select Database type (MYSQL, MSSQL, Oracle, Postgres)
7. Table Name: Provide the table name to read data
8. Query: Insert an SQL query (it takes query containing a Join statement as well)
9. Limit: Set limit for the number of records



v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the '**Selected Columns**' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

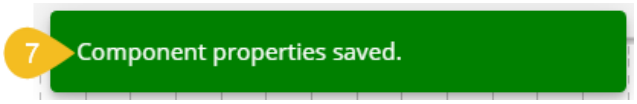
or

Use '**Download Data**' and '**Upload File**' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the '**Upload File**' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.

vi) After doing all the configurations click the '**Save Component in Storage**' icon to save the component.

vii) A notification message appears to inform about the component configuration success.



9.1.4.1. Payload

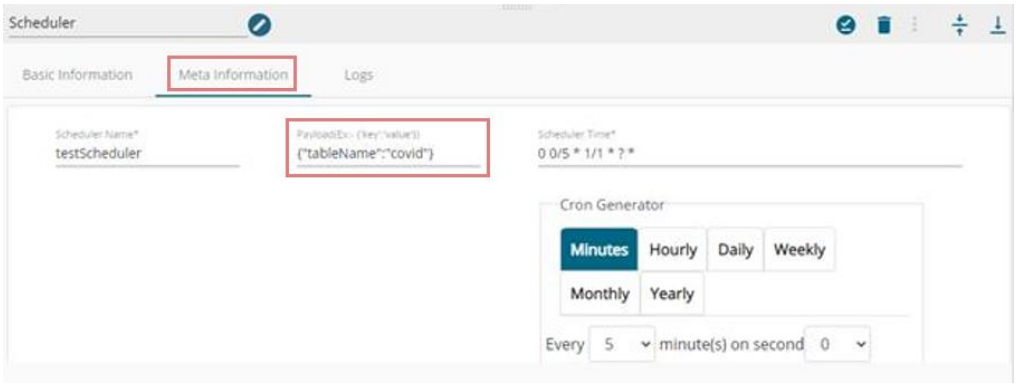
The RDBMS and Azure readers can consume the data from the previous events and inject it in specific fields of metadata and execute using the payload feature.

Syntax to use payload feature: @data.columnName

i) Navigate to an RDBMS reader component as a part of a pipeline.
ii) Open the '**Meta Information**' tab for the RDBMS reader.
iii) Fill the Table Name with the **'@data**.' Syntax.

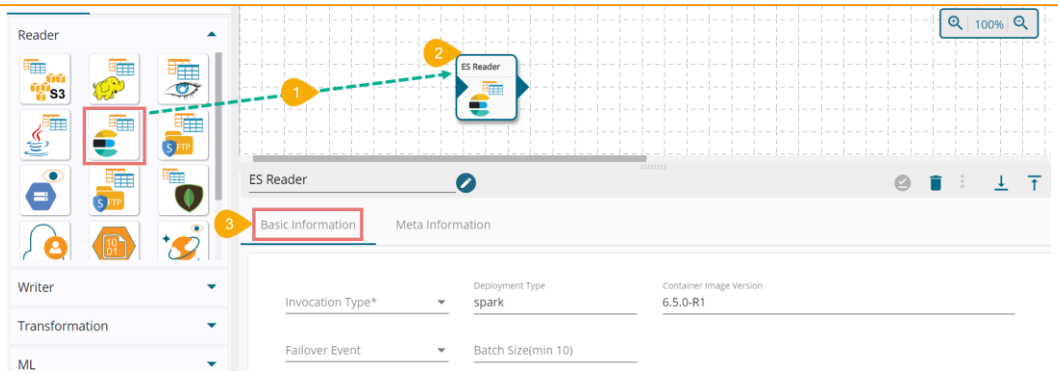
iv) Open the Preview Data for the RDBMS reader to see the fetched data.

Note: The user can send payload using the scheduler by giving syntax of the 'key: value pair. In the context of the above example, we can give the following information in the payload field of the scheduler.
{"table_name": "covid"}



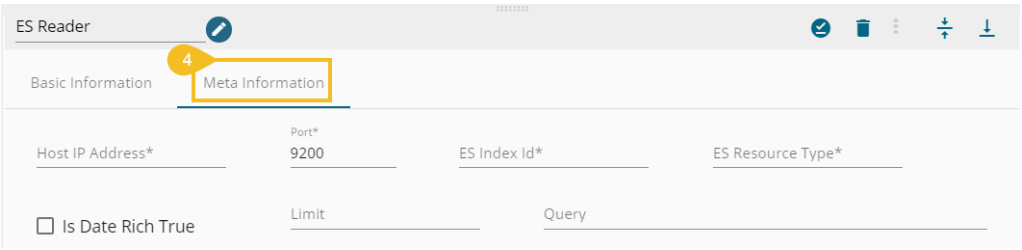
9.1.5. ES Reader

i) Drag & drop the ES Reader component on the Workflow Editor.
ii) Click on the dragged reader component to open the component properties tabs below.
iii) Basic Information: It is the default tab to open for the ES Reader while configuring the component.
   a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.
   b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   d. Failover Event: Select a failover Event from the dropdown menu.
   e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the 'Meta Information' tab and fill in all the connection-specific details of ES Reader.

1. Host IP Address (*): Database Host Address.
2. Port (*): Database Host port.
3. ES Index Id (*): Id of the elastic search index.
4. ES Resource Type: Type of the elastic search index (can be given the same as Id).
5. Is Data-Rich True: Required while reading data from indexes having dates.
6. Limit: Set a limit for the number of records.
7. Query: Provide a valid query to fetch data from the dataset.



v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

or

Use 'Download Data' and 'Upload File' options to select the desired columns.

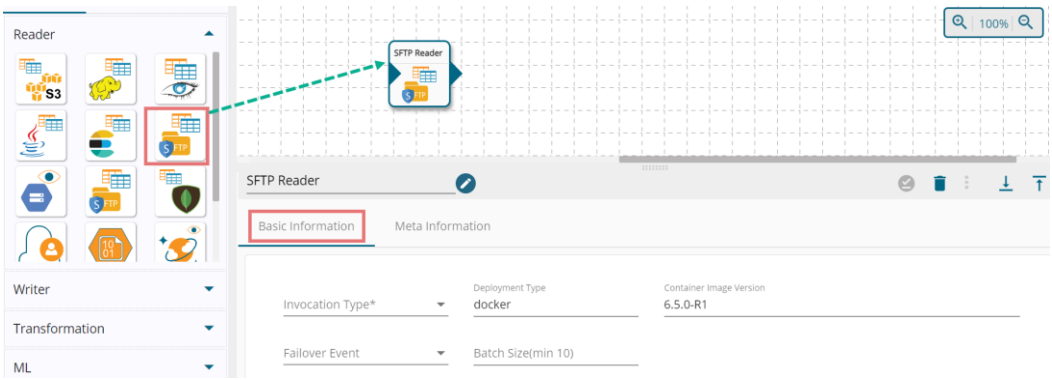
1. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.

vi) Click the '**Save Component in Storage**' icon to save the component properties.

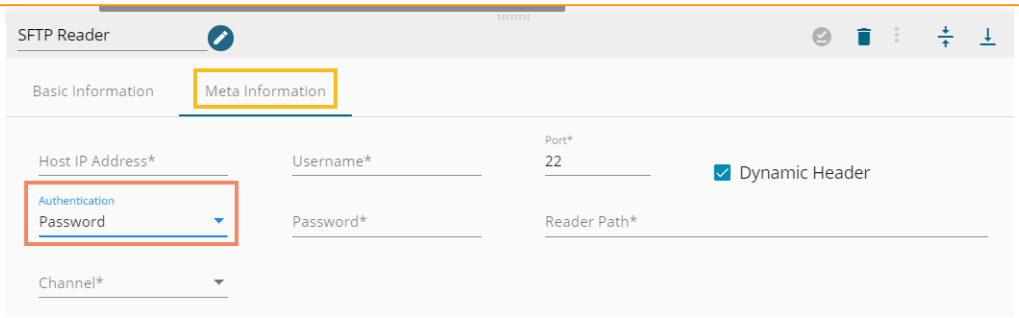
vii) A notification message appears to inform about the component configuration success.

9.1.6. SFTP Reader

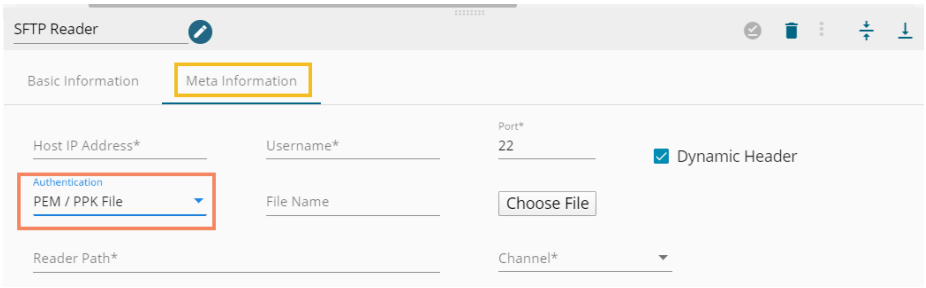
i) Drag & Drop SFTP Reader component on the Workflow Editor.
ii) Click on the dragged reader component to open the component properties tabs below.
iii) Basic Information: It is the default tab to open for the SFTP Reader while configuring the component.
   a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select 'Real-Time' or 'Batch' from the drop-down menu.
   b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   d. Failover Event: Select a failover Event from the drop-down menu.
   e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the SFTP Reader.
   1. Host IP Address (*): IP address of SFTP location.
   2. Username (*): Username of SFTP location.
   3. Port (*): Port of SFTP server (default:22).
   4. Dynamic Header: Enable this option by putting a checkmark in the box.
   5. Authentication (*): Select an authentication option for login to the SFTP server using the drop-down menu. The field gets displayed based on the Authentication option.
      a. Password: configure the following fields that appear after selecting the 'Password' option as authentication.
         i. Password: Provide a valid password in the 'Password' fields to log in.



b. PEM/PPK (file): Select PEM/PPK file database host port for login to SFTP server
   i. File Name: Provide a file name
   ii. Choose File: select a PEM/PPK file using this option



6. Reader Path (*) – complete path (absolute path) for the folder (from where the new files need to be read)
7. Channel (*) – the supported channel is SFTP.

v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

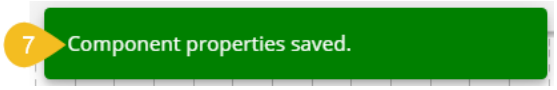
or

Use '**Download Data**' and '**Upload File**' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.

vi) After doing all the configurations click the '**Save Component in Storage'** icon to save the component.

vii) A notification message appears to inform about the component configuration success.

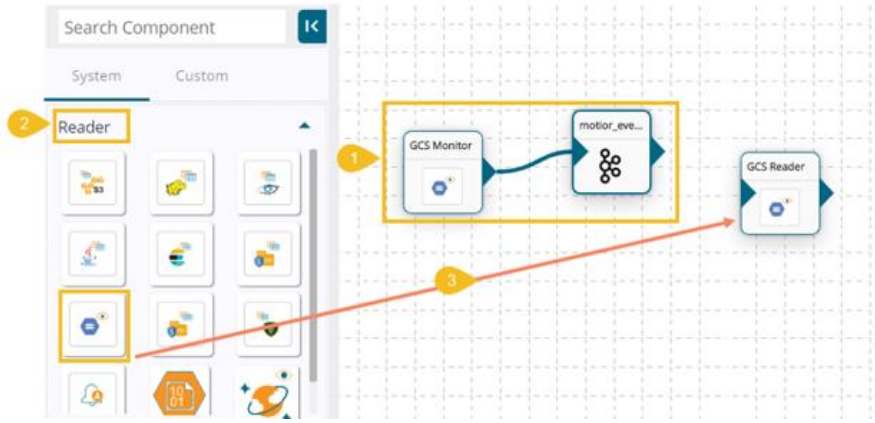


9.1.7. GCS Reader

GCS Reader pulls data from the GCS Monitor, so the first step is to implement GCS Monitor.

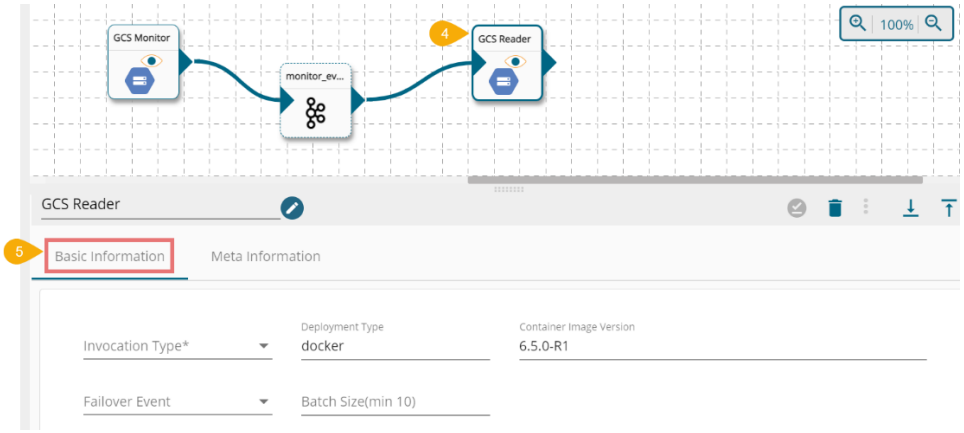
Note: The users can refer to the GCS Monitor section of this document for the details.

i) Navigate to the Pipeline Workflow Editor page for an existing pipeline workflow with GCS Monitor and Event component.
ii) Open the Reader section of the Component Pallet.
iii) Drag the GCS Reader to the Workflow Editor.



iv) Click on the dragged GCS Reader component to get component properties tabs below.
v) Basic Information: It is the default tab to open for the GCS Reader while configuring the component.
   a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select 'Real-Time' or 'Batch' from the drop-down menu.
   b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   d. Failover Event: Select a failover Event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



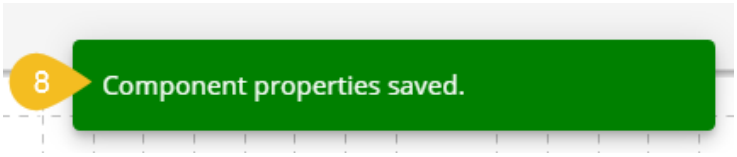
vi) Provide the following Meta Information for the GCS Reader component:

a. Bucket Name: Destination bucket name (Copy Bucket Name of the GCS Monitor)
b. Directory Path: Destination copy folder path (Copy Directory Path of the GCS Monitor)
c. Choose File: Upload the same Service account keys using the JSON file
d. File Name: It displays the file name of the uploaded file

Note: Provide the same destination configuration information that was used in the GCS Monitor.

vii) After doing all the configurations click the '**Save Component in Storage**' icon to save the component.

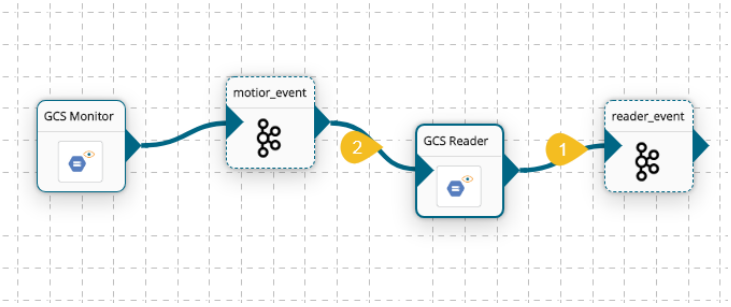
viii) A message appears to confirm the component properties update.



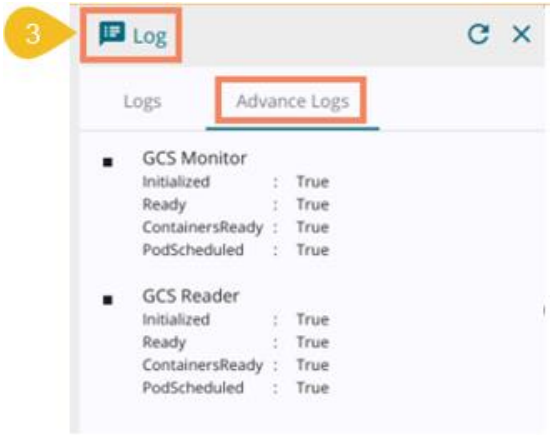
9.1.7.1. GCS Reader in a Workflow

i) Create a new Event to store all the read data and connect it with the dragged GCS Reader component.
ii) Connect the GCS Reader to the existing GCS Monitor Pipeline to create the

below-given workflow.

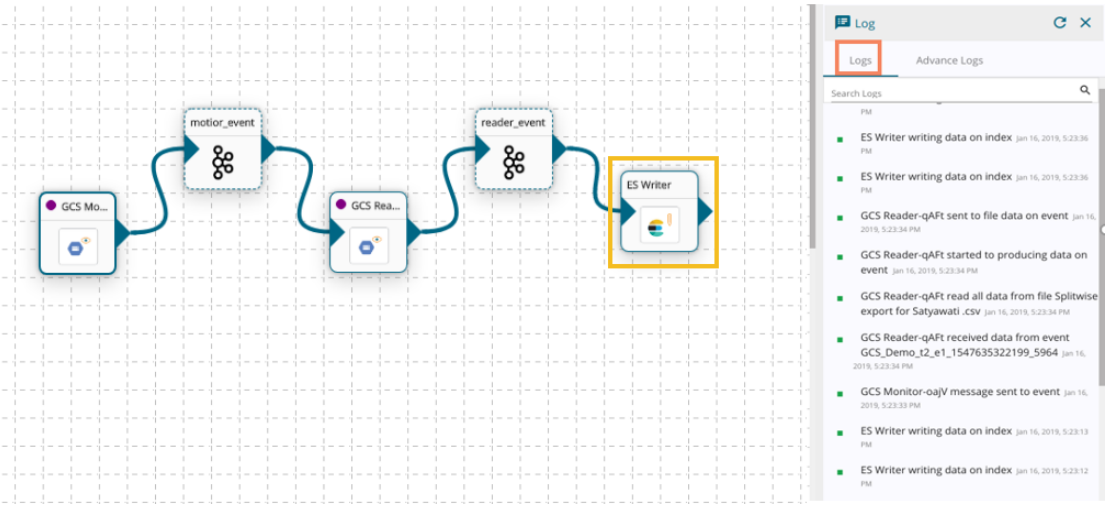


iii) Open the Advanced Log Tab to check the component status.



iv) Upload a CSV or JSON file on the monitoring location and check the Logs section displaying the ongoing operations.

v) Connect a Writer component to store the processed data (E.g., the ES Writer component is used in this pipeline workflow).



Note: GCS Reader component gets a message from an Event component that was initially sent by the GCS Monitor to the Event, so the users must configure the GCS Monitor component before using the GCS Reader component.

9.1.8. SFTP Stream Reader

i) Drag & Drop the SFTP Stream Reader component on the Workflow Editor.

ii) Click on the dragged reader component to open the component properties tabs below.

iii) Basic Information: It is the default tab to open for the SFTP Stream Reader while configuring the component.

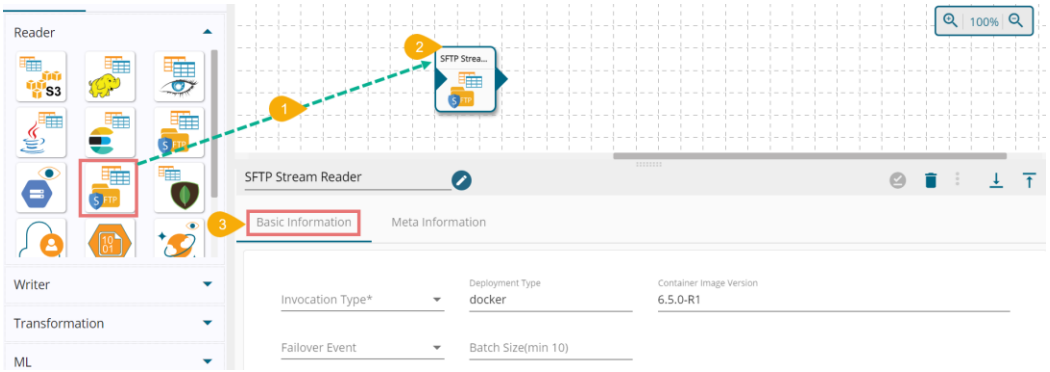
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select 'Real-Time' or 'Batch' from the drop-down menu.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

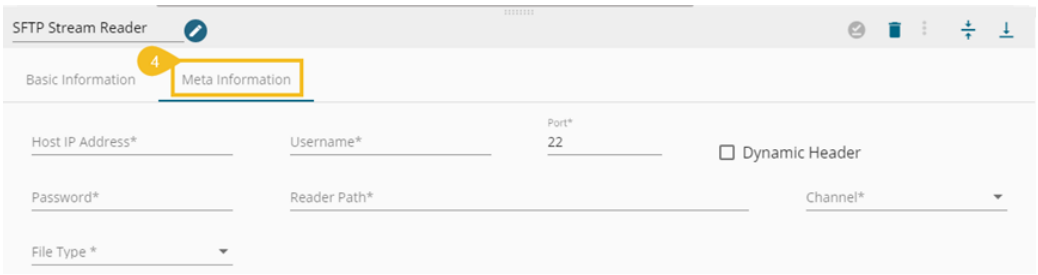
d. Failover Event: Select a failover Event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and provide the connection-specific information.

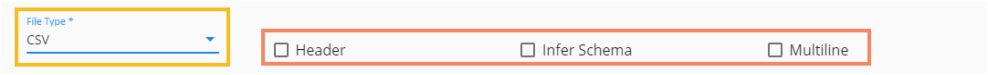
1. Host IP Address (*): IP address of SFTP location.
2. Username (*): Username of SFTP location.
3. Port (*): Port of SFTP server (default:22).
4. Dynamic Header: Enable this option by putting a checkmark in the box.
5. Password (*): Provide a valid password in the 'Password' field to log in.
6. Reader Path (*): Complete path (absolute path) for the folder (from where the new files need to be read)
7. Channel (*): The supported channel is SFTP.
8. File Type (*): Select a file from the drop-down menu.



Note: The user needs to configure some extra fields based on the selected File Type as displayed below.

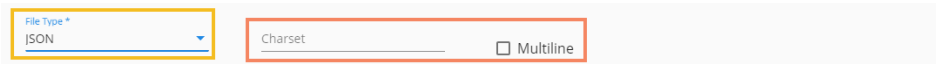
a. The following fields appear if the selected file type is CSV

1. Header: Put a checkmark in the box to get Column names/ headers from the CSV file.
2. Infer Schema: Put a checkmark in the box to get inferring datatypes.
3. Multiline: Enable this option to get a column with multiple lines in a single cell of the data. E.g., the address column can have multiple lines in a single cell.



b. The following fields appear if the selected file type is JSON

1. Charset-Character set encoding, E.g., UTF-8
2. Multiline: Enable this option to get a column with multiple lines in a single cell of the data.

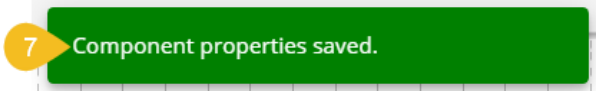


v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the '**Selected Columns**' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.
or
Use '**Download Data**' and '**Upload File**' options to select the desired columns.

3. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
4. Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.

vi) After doing all the configurations click the '**Save Component in Storage**' icon to save the component.

vii) A notification message appears to inform about the component configuration success.



9.1.9. MongoDB Reader

i) Drag & Drop the Mongo DB Reader on the Workflow Editor.

ii) Click on the dragged reader component to open the component properties tabs below.

iii) Basic Information: It is the default tab to open for the Mongo DB reader while configuring the component.

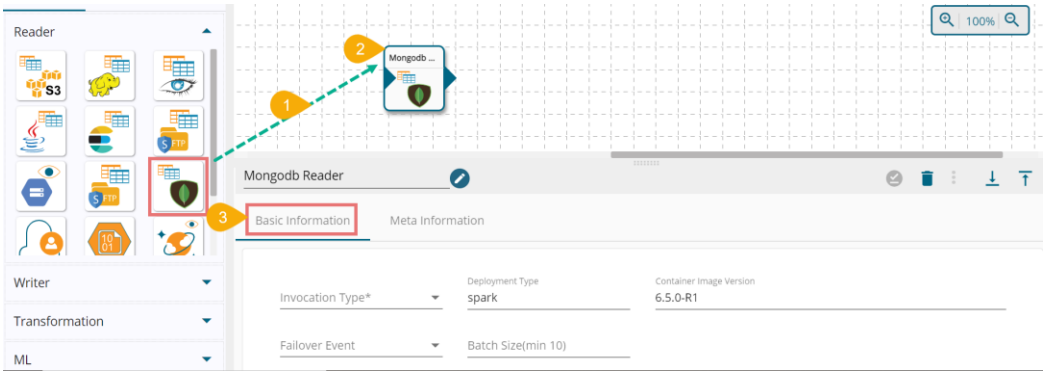
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select 'Real-Time' or 'Batch' from the drop-down menu.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

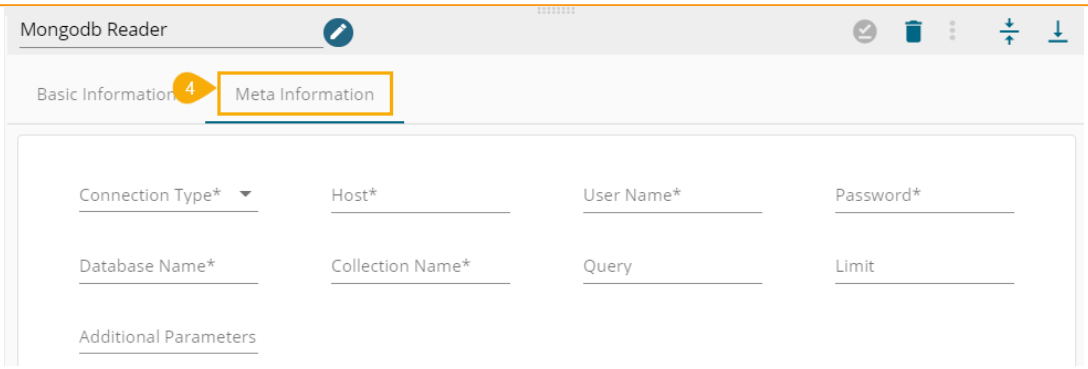
d. Failover Event: Select a failover Event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



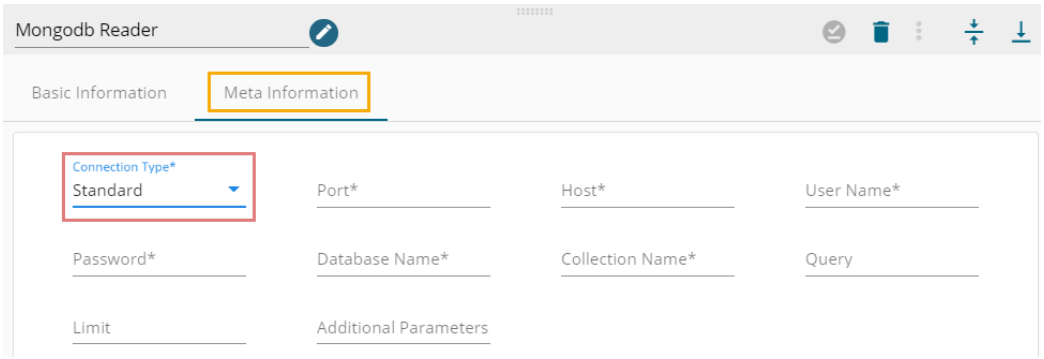
iv) Open the '**Meta Information**' tab and fill in all the connection-specific details of Mogo DB.

1. Connection Type: Select either of the connection type choices out of 'Standard' and 'SRV'.
2. Host IP Address (*): Hadoop IP address of the host.
3. Port (*): Port number (It appears only with the '**Standard**' Connection Type).
4. Username (*): Provide username.
5. Password (*): Provide a valid password to access the Mongo DB.
6. Database Name (*): Provide the name of the database from where you wish to read data.
7. Collection Name (*): Provide the name of the collection.
8. Query: Insert an SQL query (it takes a query containing a Join statement as well).
9. Limit: Set a limit for the number of records.
10. Additional Parameters: Provide the additional parameters.

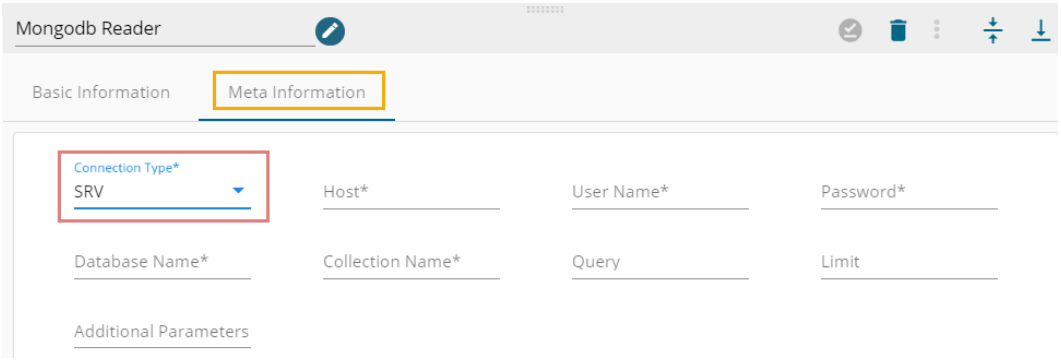


The Meta Information fields vary based on the selected '**Connection Type**' option.

i. **Meta Information** Tab with **Standard** as **Connection Type**



ii. **Meta Information** Tab with **SRV** as **Connection Type**



v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

or

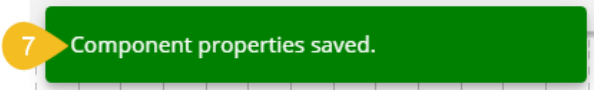
Use '**Download Data**' and '**Upload File**' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the '**Upload File**' icon (file size must be less than 2 MB).

2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.

vi) After doing all the configurations click the '**Save Component in Storage**' icon to save the component.

vii) A notification message appears to inform about the component configuration success.



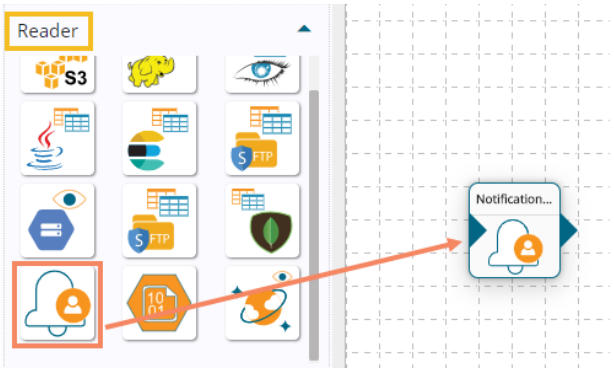
9.1.10. Notification Subscriber

Notification Subscriber Component as the name suggests is meant to subscribe to the messages published as a notification. This can be very helpful in acting in scenarios where the immediate reaction is required in some conditions.

Example: A pipeline is monitoring an IoT sensor that sends the temperature of a machine. If the temperature reaches a certain threshold; the process engineer is to be notified immediately to maintain the same to stabilize the machine. Here, the user can subscribe to the same SNS and further use it in a pipeline for sending this directly to the respective dashboard.

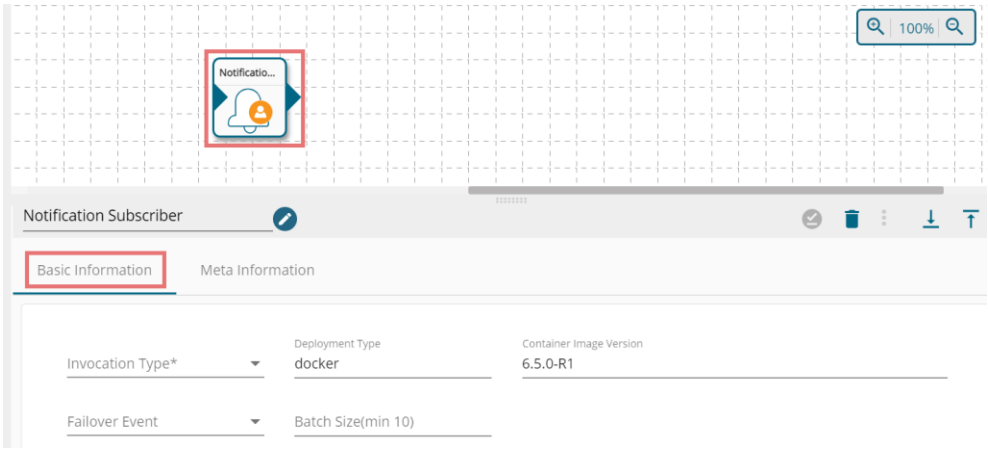
The steps to configure the Notification Subscriber component is described in this section.

i) Navigate to the Pipeline Editor.
ii) Drag the Notification Subscriber from the Reader section provided under the Components Pallet and drop it to the Pipeline Workspace.

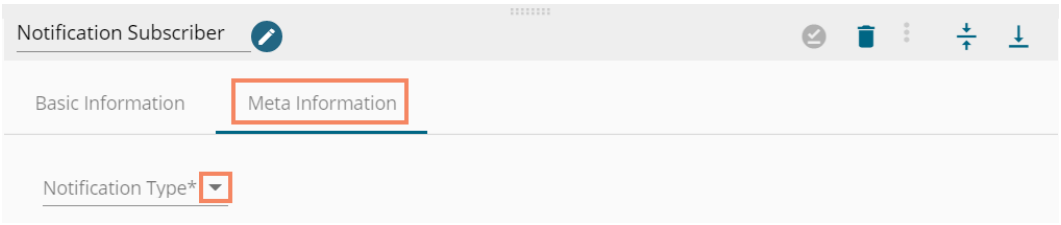


iii) Click the dragged Notification Subscriber component to get the Configuration fields:
iv) The Basic Information tab opens by default.

a. Select the invocation type (Real-Time/ Batch).
b. Deployment Type: It comes preselected based on the component.
c. Container Image Version: It comes preselected based on the component.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



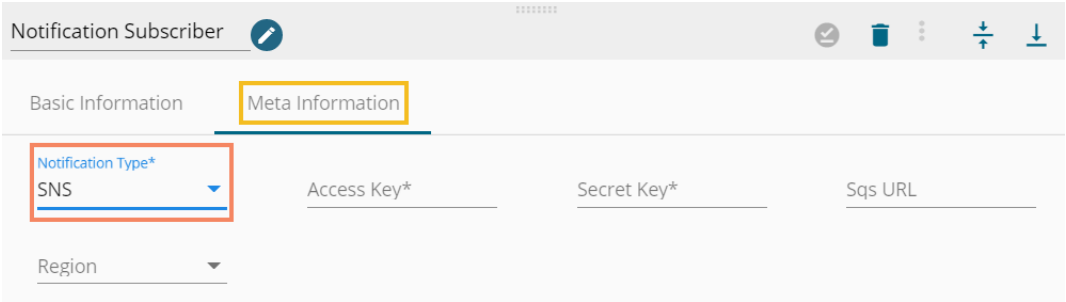
v) Open the '**Meta Information**' tab and provide the connection-specific fields.
a. Notification Type: Select a notification type using the drop-down menu. The provided options are:
   i. SNS
   ii. Google Pub/Sub
b. Based on the selected Notification Type the remaining configuration fields get displayed.
c. The user needs to complete the configuration process for the selected notification type and click the '**Save Component in Storage**' icon to complete the configuration process for the Notification Subscriber.



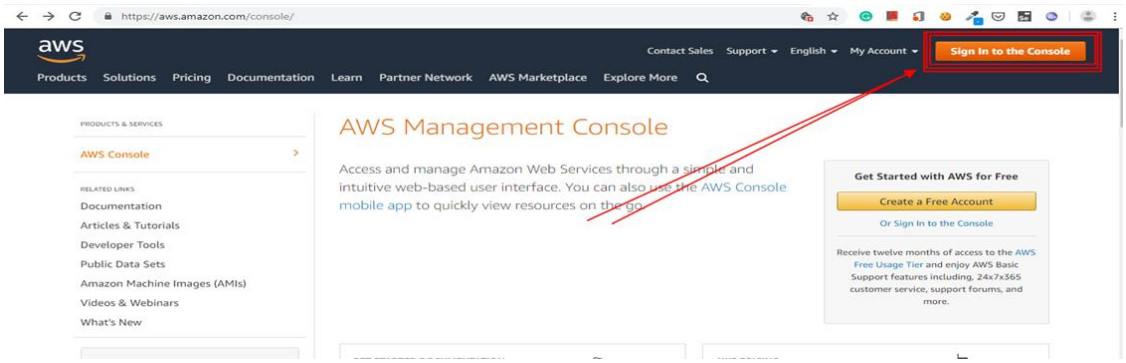
9.1.10.1. SNS as Notification Type

i) Navigate to the Meta Information tab provided for the Notification Subscriber.
ii) Select the 'SNS' as the Notification Type option using the drop-down menu.
iii) The following configuration fields appear:
   a. Access Key: Access keys consist of two parts: an access key ID (for example, AKIAIOSFODNN7EXAMPLE) can be retrieved from AWS key management.
   b. Secret Key: A secret access key (for example, wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY) can also be retrieved from the AWS key management.

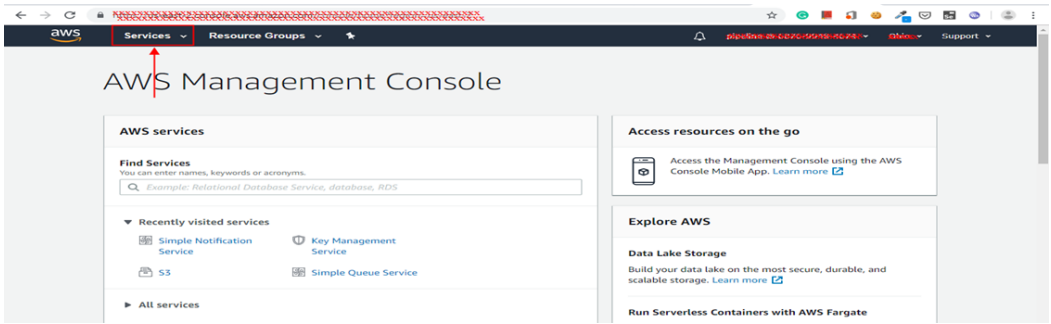
c. SQS URL: The field gets filled after the topic creation. Navigate to Topics and then click the desired topic.
d. Region: Select an option from the drop-down menu where the topic has been created.



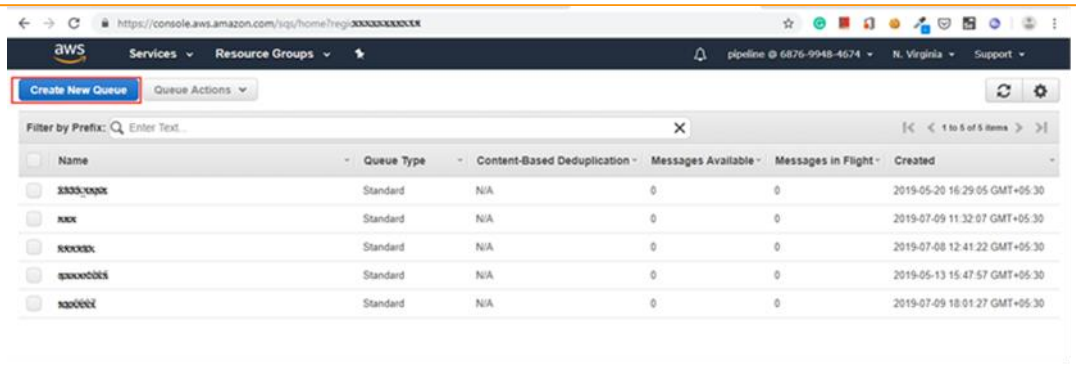
iv) The user needs to follow some more steps to create and subscribe to a topic. The below-given description explains those steps:
a. Navigate to the URL: https://aws.amazon.com/console/
b. Click the 'Sign In to the Console' option.



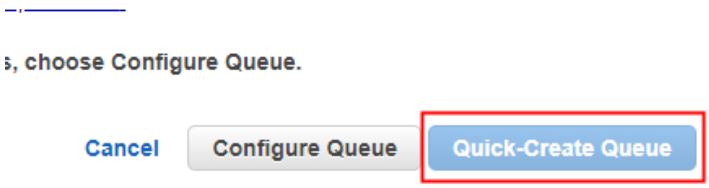
c. Sign in with your credentials.
d. The AWS Management Console opens.
e. Click the 'Services' option from the header drop-down.
f. Search SQS service using the 'Find Services' space.



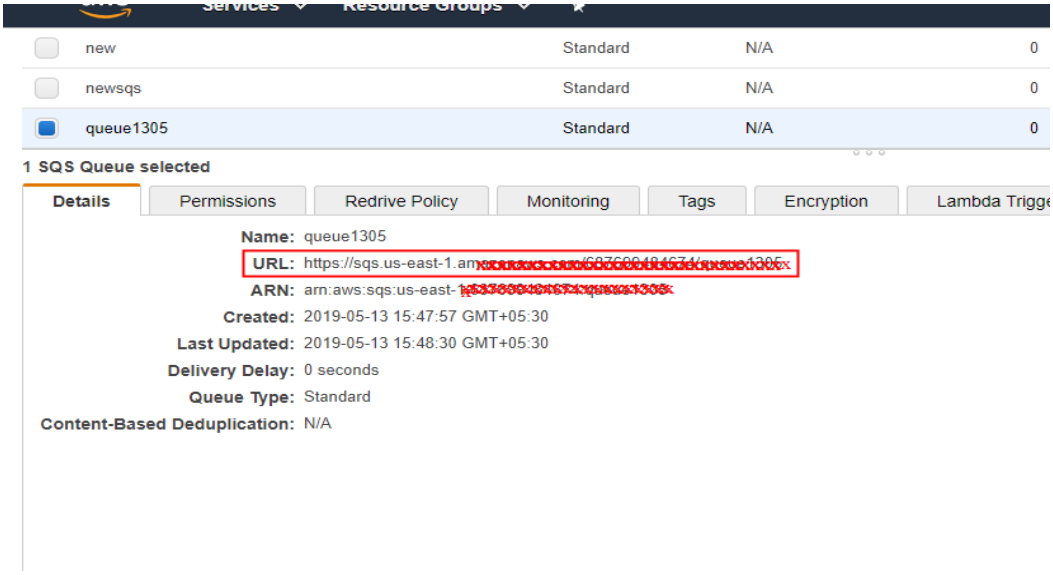
g. The Amazon SNS page opens.
h. Click the 'Create New Queue' option.



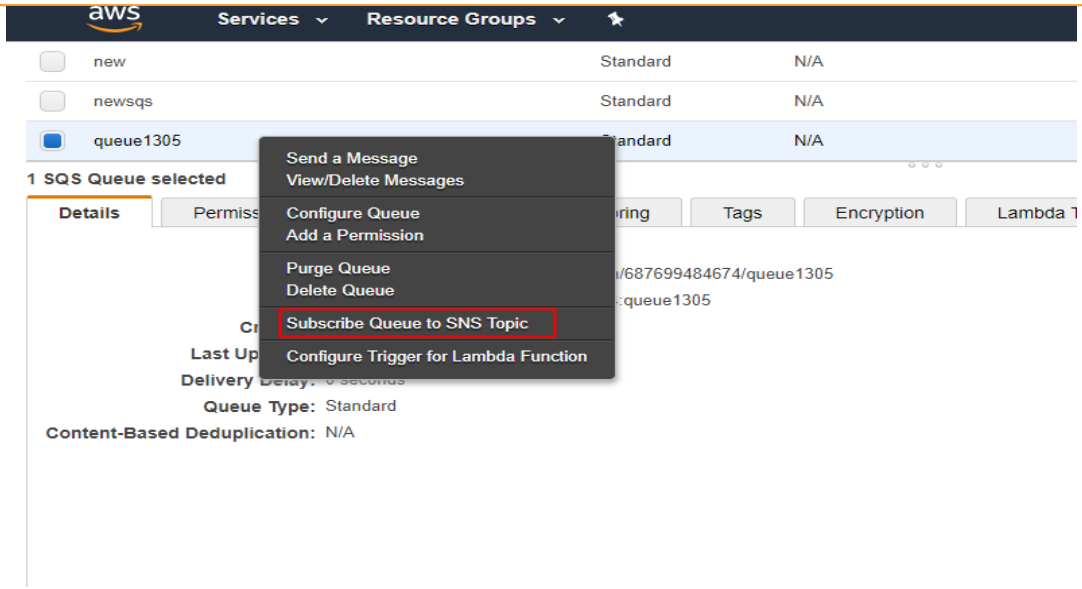
i. Click the 'Quick-Create Queue' option after giving SQS name.



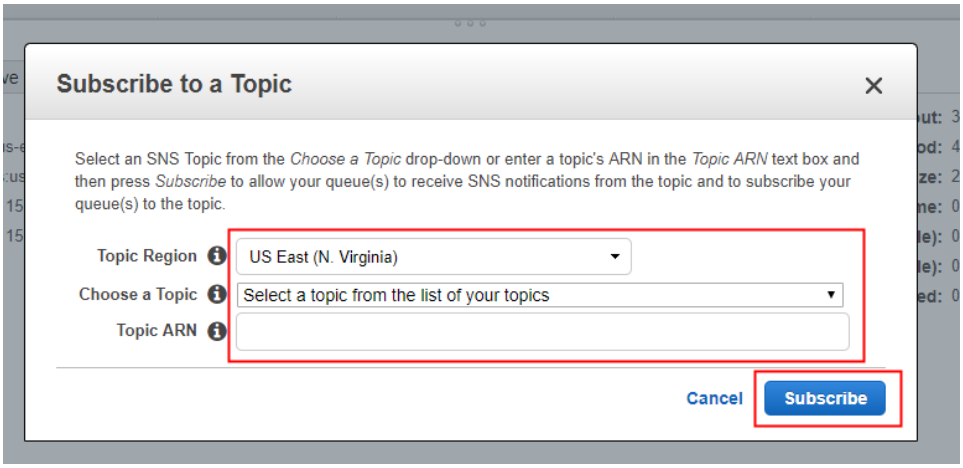
j. Click on the created SQS to see the details.
k. Copy the SQS URL from there.



l. Use right-click on the created SQS.
m. Select the 'Subscribe Queue to SNS Topic' to subscribe to the desired SNS Topic.



n. The 'Subscribe to a Topic' window opens.
o. Provide the following information:
   i. Topic Region
   ii. Choose a Topic
   iii. Topic ARN
p. Click the 'Subscribe' option.

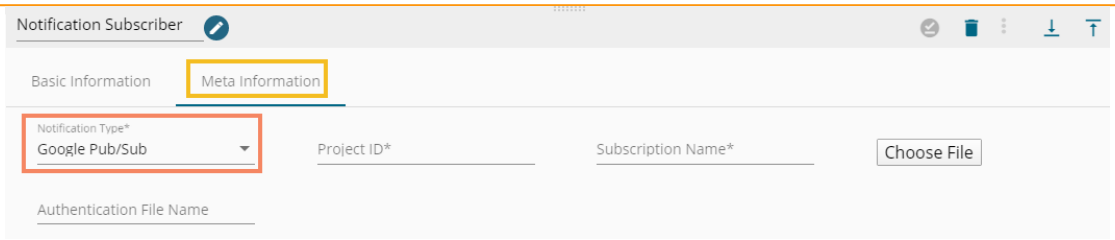


Note:

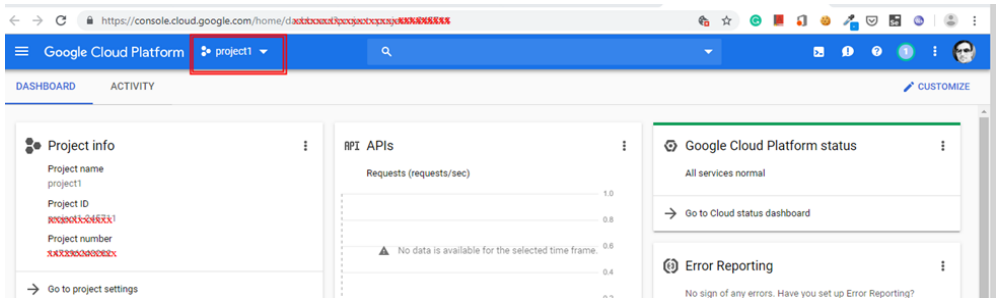
a. Choose the Topic from where your messages are being published.
b. SNS Region: The same can be seen in the SNS ARN URL. The user needs to use the same.

9.1.10.2. Google Pub/Sub as the selected Notification Type

i) Navigate to the Meta Information tab provided for the Notification Subscriber.
ii) Select the 'Google Pub/Sub' as the Notification Type option using the drop-down menu.

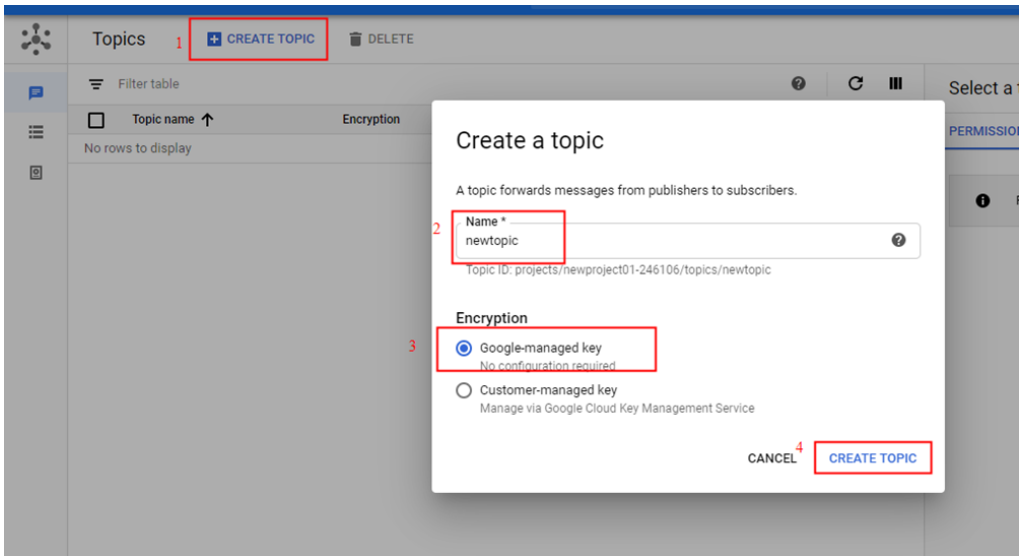


iii) The user needs to follow some more steps to get the Pub/Subtopic ID
iv) Navigate to the https://console.cloud.google.com/
v) Log in with your credentials.
vi) The Landing page opens.
vii) Click on the marked area.
viii) Create a Project or use the desired one.



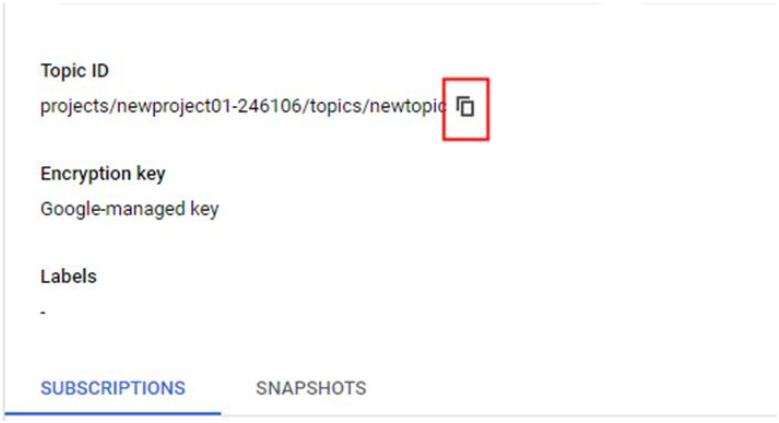
ix) Search for the Pub/Sub-option in the Search bar.

x) Create a Topic using the following steps:
   a. Click the 'CREATE TOPIC' option.
   b. Provide a name for the new topic
   c. Select the 'Google-managed key' as the Encryption option.
   d. Click the 'CREATE TOPIC' option.

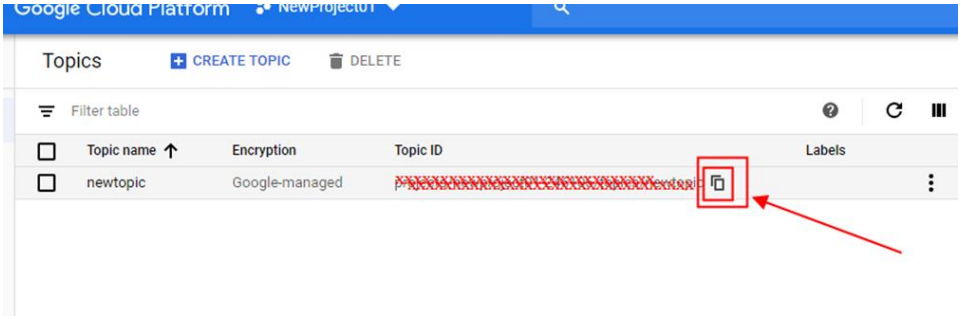


xi) The user gets either of the following screens:

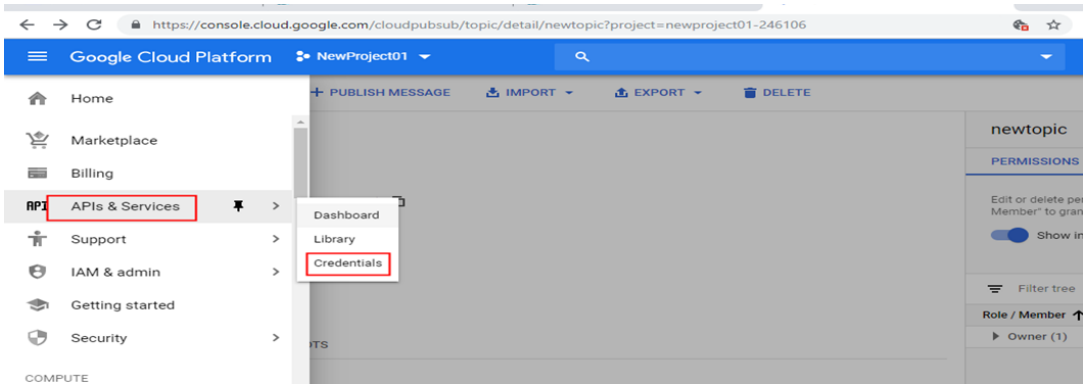
a. While using an existing Topic ID



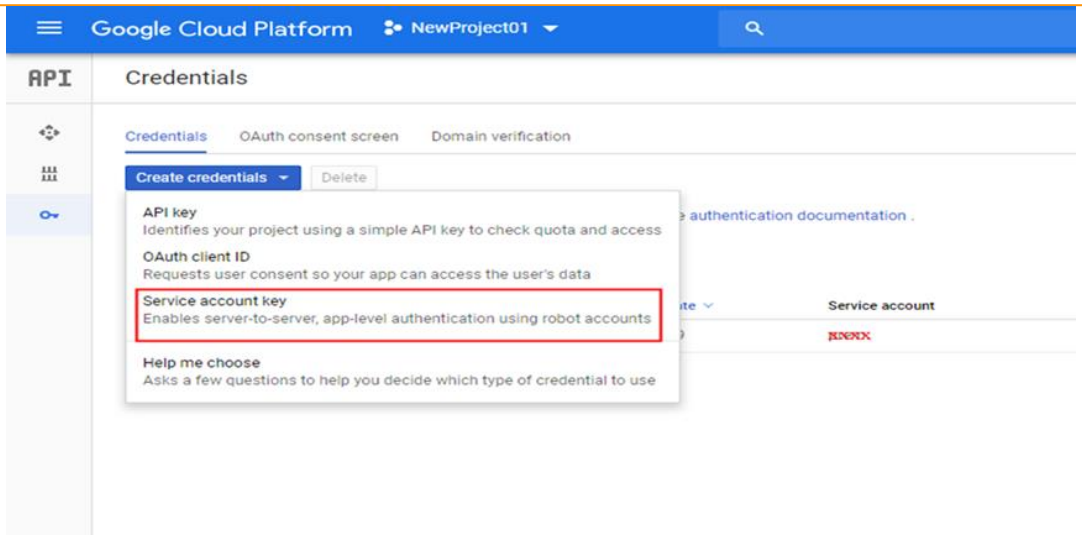
Alternatively, while using the newly created Topic ID



b. Create Auth-JSON for the current Project.

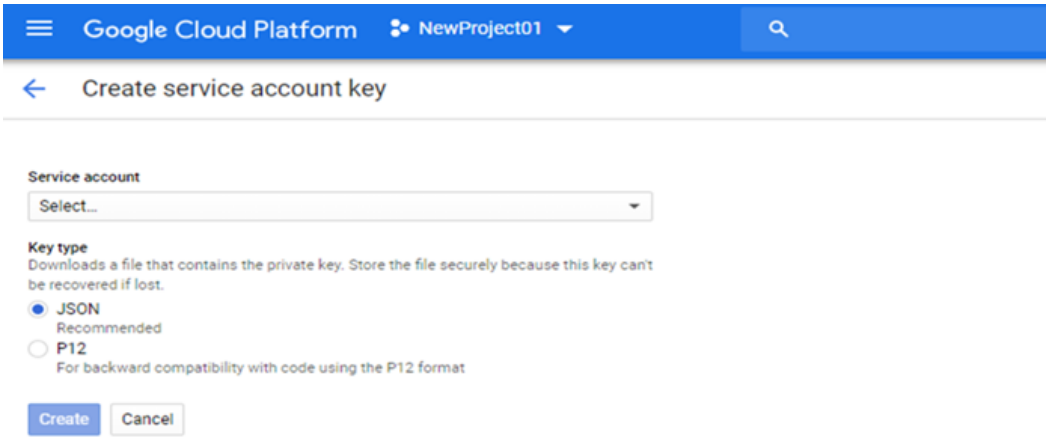


c. Create a Service account key by following the given steps:
   i. Click the '**Service account key**' option from the Credentials tab.



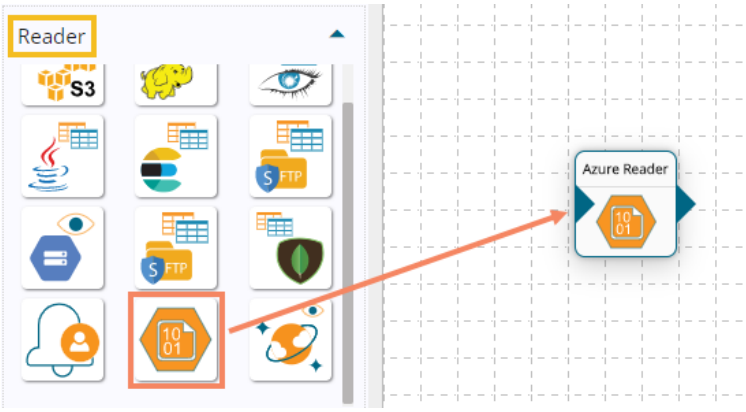
ii. Select a Service account using the drop-down menu.
iii. Download the JSON file using the radio button.
iv. Click the 'Create' option.

Use the copied Topic ID or the service account as metadata input.



9.1.11. Azure Reader

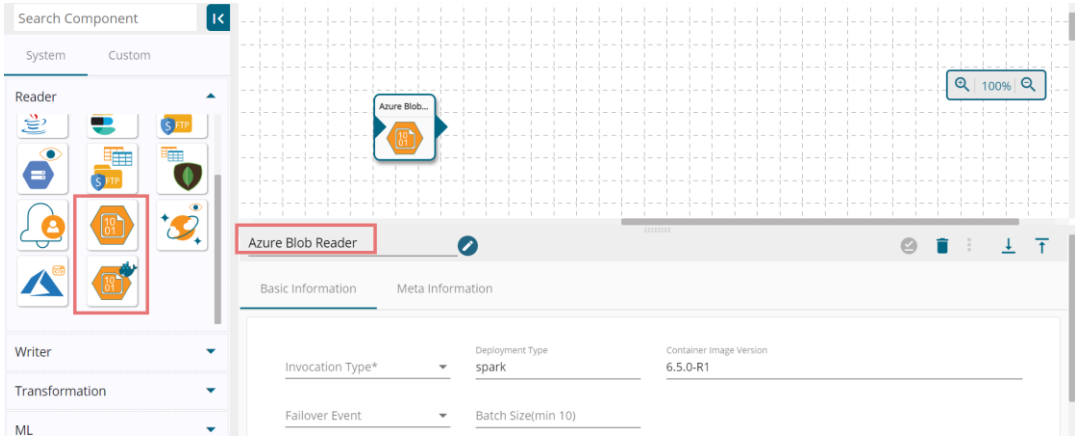
i) Drag and drop the Azure Reader component to the Workflow Editor.



ii) Click on the dragged Azure Reader to get the component properties tabs.

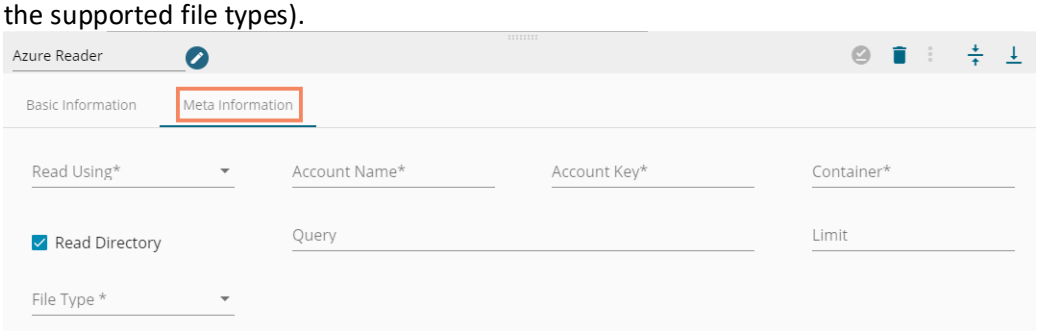
iii) **Basic Information**: It is the default tab to open for the component while configuring the component.

a. Invocation Type: Select an invocation mode out of '**Real-Time**' or '**Batch**' using the drop-down menu.
b. Deployment Type: It displays the deployment type for the reader component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the Azure Reader.

1. Read Using (*): Select an option from the drop-down menu to connect to data. The supported options are '**Connection String**' and '**Secret Key**'.
2. Account Name (*): Name of the Azure account.
3. Account Key (*): Provide Account Key to login.
4. Container (*): Container details.
5. Read Directory: Enable this option by putting a checkmark in the box to be read from the Directory.
6. Blob Name: Provide a blob name. The Blob field appears if the '**Read Directory**' option is not enabled.
7. Query: Insert relevant query.
8. Limit: Set a limit for the number of records.
9. File Type (*): Select a file type from the drop-down menu (CSV, JSON, PARQUET, AVRO are the supported file types).

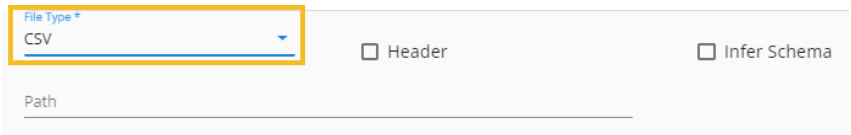


The configuration fields may vary based on the selection of the file type.

i. CSV

Configure the following fields when the selected file type is CSV:

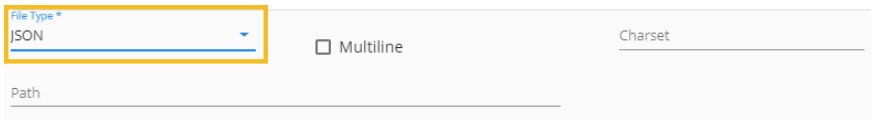
a. Header- Enable or Disable the Header option to confirm whether the stored data has a header or not.
b. Infer Schema- Enable or disable the Infer Schema option to confirm whether the schema is provided or not.
c. Path- Provide a specific path for the file.



ii. JSON

Configure the following fields when the selected file type is JSON:

a. Multiline-Enable this option, if the stored JSON has line breaks in-between
b. Charset-Character set encoding, E.g., UTF-8
c. Path-provide a specific path for the file.



iii. PARQUET

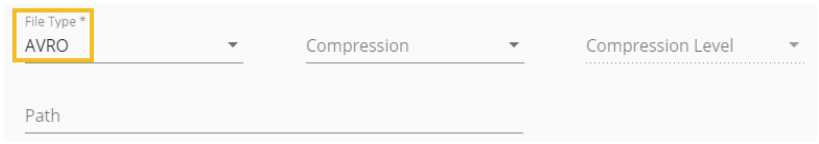
Configure the following fields when the selected file type is PARQUET:

a. Path-provide a specific path for the file.

iv. AVRO

Configure the following fields when the selected file type is AVRO:

a. Compression: Select a compression option from the drop-down menu out of the '**Deflate**' and '**Snappy**' options.
b. Compression Level- If the selected Compression option is 'Deflate' then select the compression level from 0-9 (where 0 and 9 are included).
c. Path-provide the specific path for the file.
d. Limit- set limit for the number of rows.

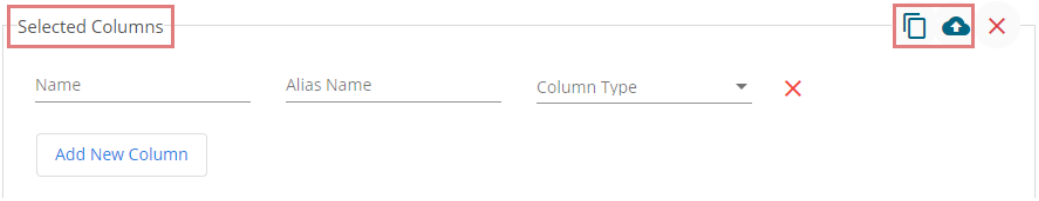


v) Selected Columns: There is also a section for the selected columns in the Meta Information tab if the user can select some specific columns from the table to read data instead of selecting a complete table so this can be achieved by using the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

Or

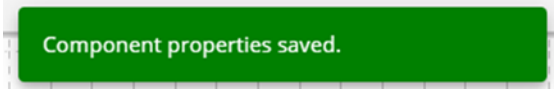
Use 'Download Data' and 'Upload File' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.



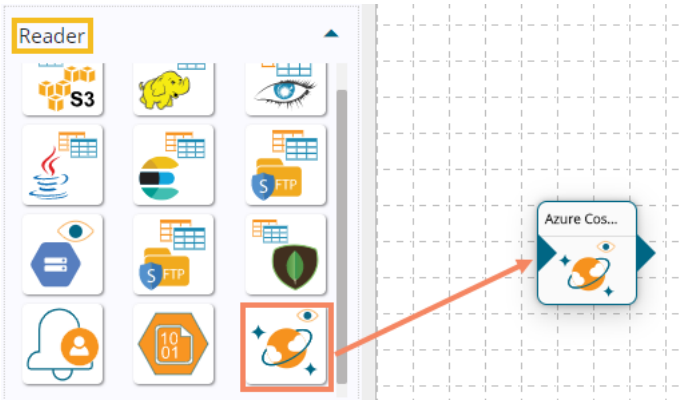
vi) Click the '**Save Component in Storage**' icon after doing all the configurations to save the reader component.

vii) A notification message appears to inform you that the component properties have been saved.



9.1.12. Azure Cosmos DB Reader

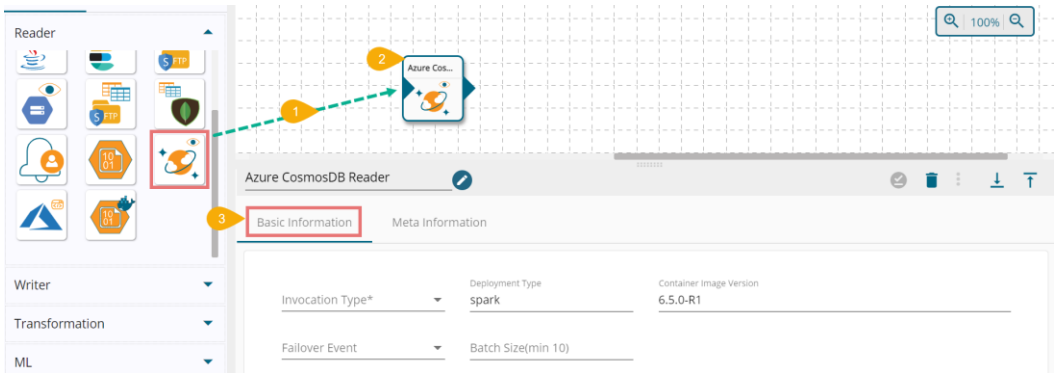
i) Drag and drop the Azure Cosmos DB Reader component to the Workflow Editor.



ii) Click on the dragged Azure Cosmos DB Reader to get the component properties tabs.

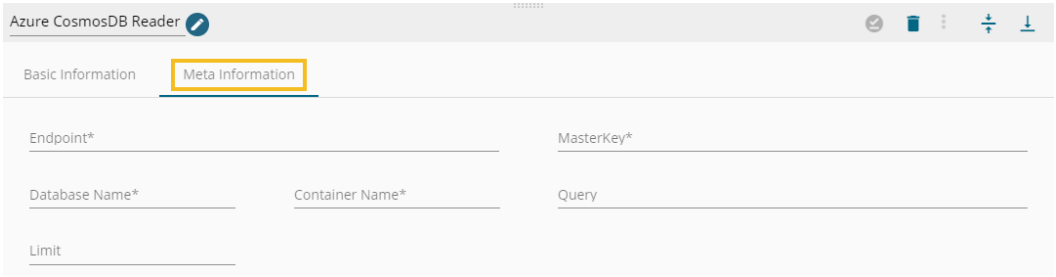
iii) **Basic Information**: It is the default tab to open for the component while configuring the component.

a. Invocation Type: Select an invocation mode out of '**Real-Time**' or '**Batch**' using the drop-down menu.
b. Deployment Type: It displays the deployment type for the reader component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the Azure Cosmos DB Reader.

1. Endpoint (*): Provide endpoint information.
2. Master Key (*): Provide the master key.
3. Database Name (*): Provide the name of the database.
4. Container Name (*): Container details.
5. Query: Insert relevant query.
6. Limit: Set a limit for the number of records.

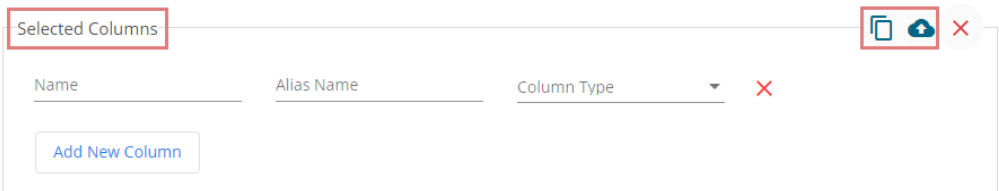


v) Selected Columns: There is also a section for the selected columns in the Meta Information tab if the user can select some specific columns from the table to read data instead of selecting a complete table so this can be achieved by using the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

Or

Use 'Download Data' and 'Upload File' options to select the desired columns.

3. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
4. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.

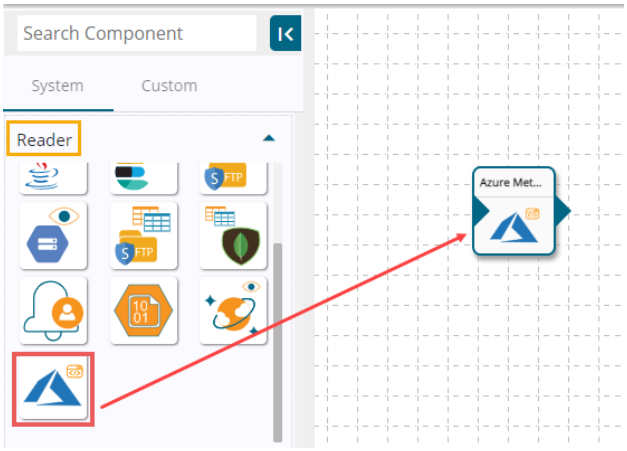


vi) Click the '**Save Component in Storage**' icon after doing all the configurations to save the reader component.

vii) A notification message appears to inform you that the component properties have been saved.

9.1.13. Azure Metadata Reader

i) Drag and drop the Azure Metadata Reader component to the Workflow Editor.



ii) Click on the dragged Azure Metadata Reader to get the component properties tabs.

iii) **Basic Information**: It is the default tab to open for the component while configuring the component.

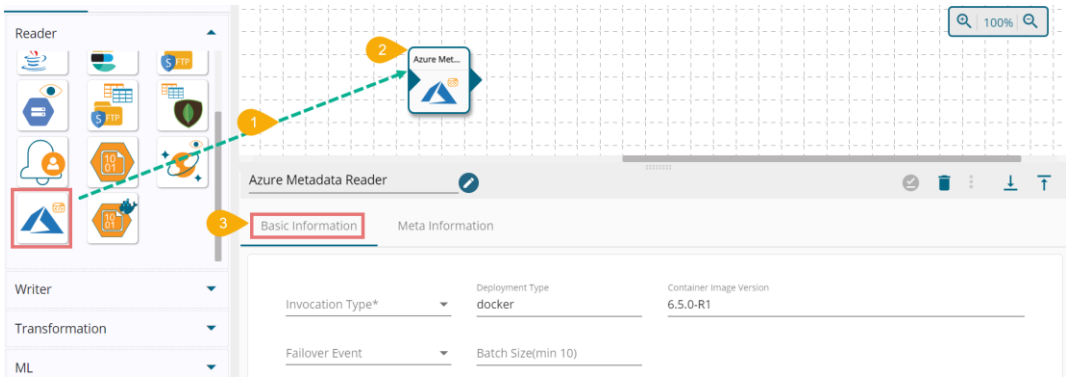
a. Invocation Type: Select an invocation mode out of '**Real-Time**' or '**Batch**' using the drop-down menu.

b. Deployment Type: It displays the deployment type for the reader component. This field comes pre-selected.

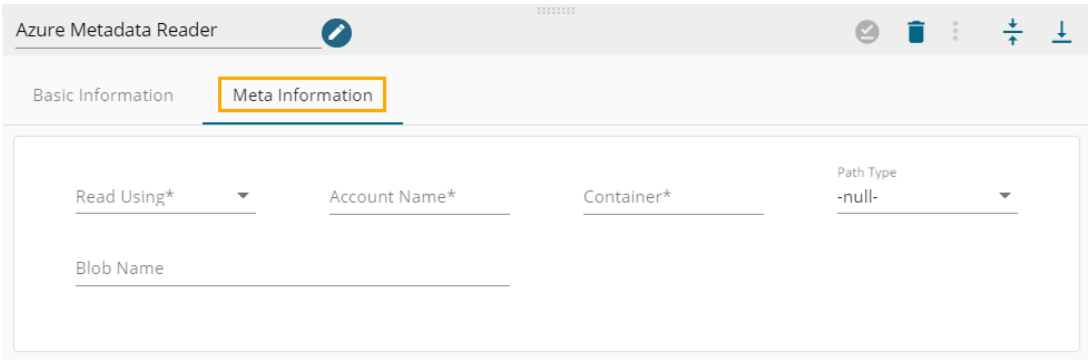
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover Event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



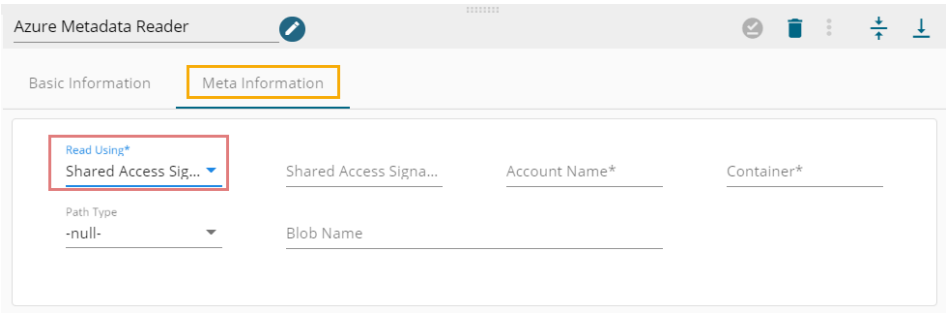
iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the Azure Metadata Reader.



Based on the selected option using the field '**Read Using'** the displayed fields may change. Available option (*): Select either of the options out of '**Shared Access Signature'** or '**Secret Key'**

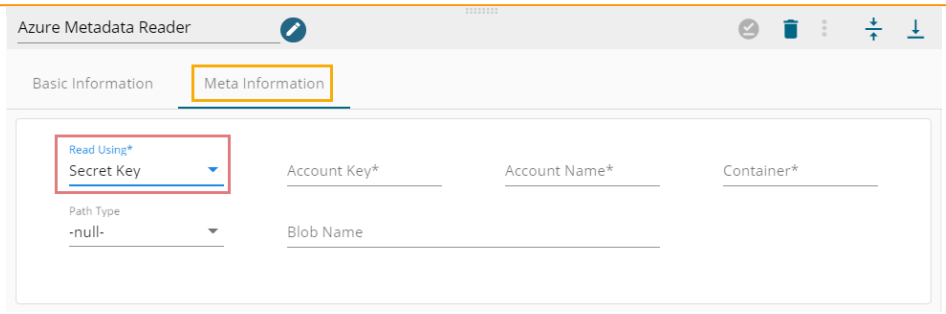
The Meta Information tab when the selected Read Using option is '**Shared Access Signature'**:

1. Read Using: the selected option is '**Shared Access Signature**'
2. Shared Access Signature: the user needs to provide the shared access signature.
3. Account Name (*): Provide the name of the account.
4. Container (*): Container details.
5. Path Tye: Select a Path type from the given choices (Null, Directory Path, Blob Name)
6. Blob Name: Provide a blob name.



The Meta Information tab when the selected Read Using option is '**Secret Key':**

1. Read Using: the selected option is '**Secret Key**'.
2. Account Key (*): Provide account key.
3. Account Name (*): Provide Account Name.
4. Container (*): Container details.
5. Path Type: Select a Path type from the given choices (Null, Directory Path, Blob Name)
6. Blob Name: Provide a blob name.

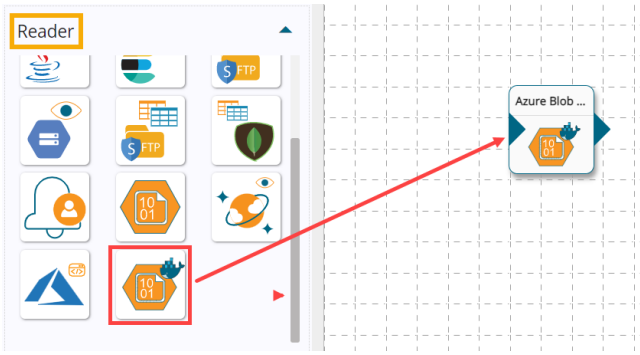


v) Click the '**Save Component in Storage**' icon after doing all the configurations to save the reader component.

vi) A notification message appears to inform you that the component properties have been saved.

9.1.14. Azure Blob Reader

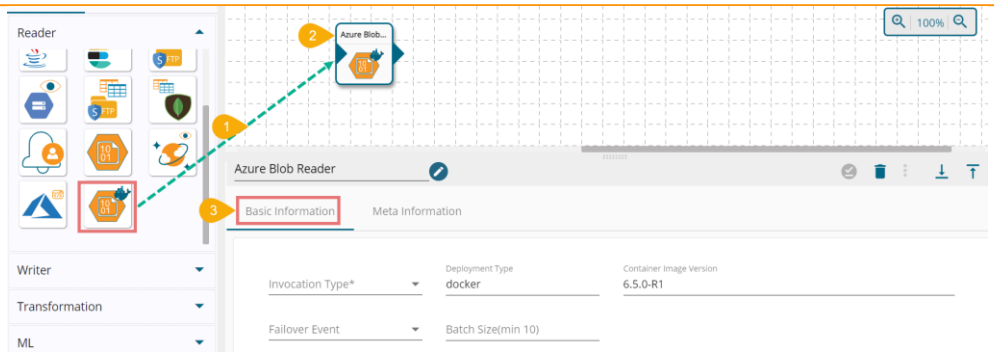
i) Drag and drop the Azure Blob Reader component to the Workflow Editor.



ii) Click on the dragged Azure Blob Reader to get the component properties tabs.

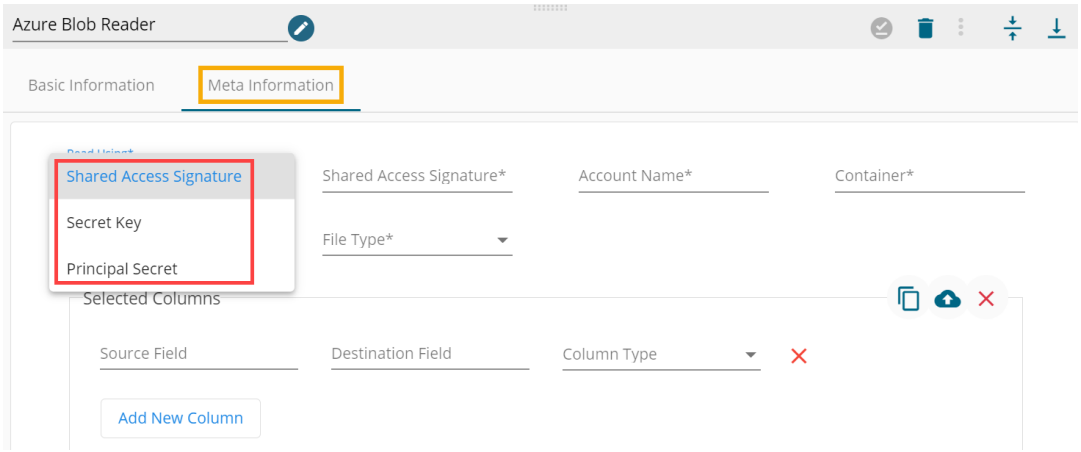
iii) **Basic Information**: It is the default tab to open for the component while configuring the component.

a. Invocation Type: Select an invocation mode out of '**Real-Time**' or '**Batch**' using the drop-down menu.
b. Deployment Type: It displays the deployment type for the reader component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



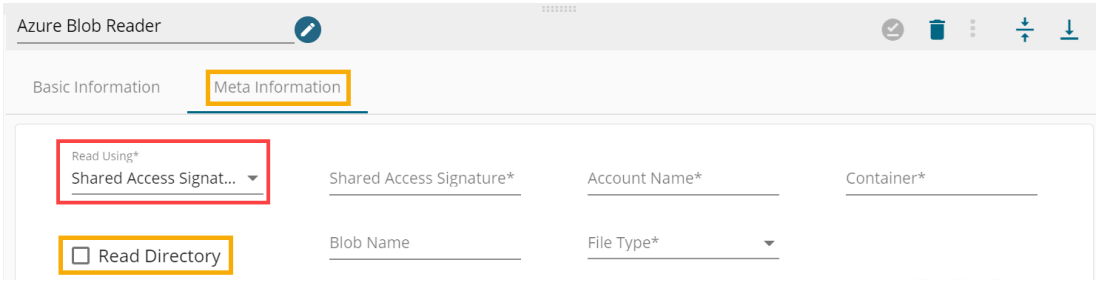
iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the Azure Blob Reader.

Based on the selected option using the field '**Read Using'** the displayed fields may change. Available option (*): Select either of the options out of '**Shared Access Signature'**, **'Secret Key' or 'Principal Secret'**

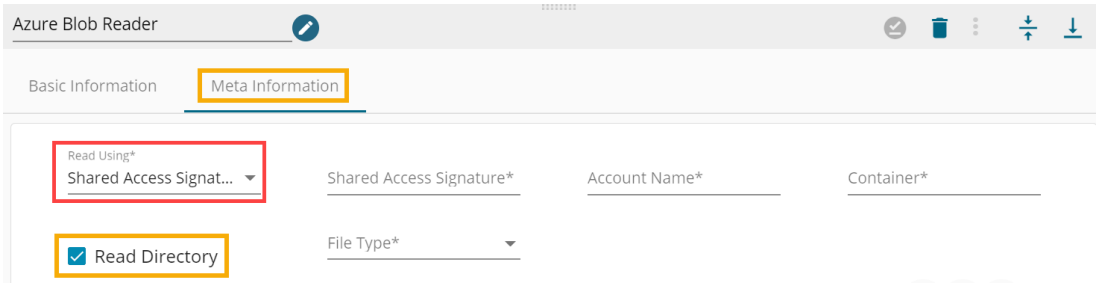


The Meta Information tab when the selected Read Using option is '**Shared Access Signature'**:

1. Read Using: the selected option is '**Shared Access Signature**'
2. Shared Access Signature: the user needs to provide the shared access signature.
3. Account Name (*): Provide the name of the Azure Blob account.
4. Container (*): Container details.
5. Read Directory: Select this option using the check box to read the blob name from the directory.
6. Blob Name: Provide a blob name. (It appears if the '**Read Directory**' option is not check marked).
7. File Type: Select a file type from the drop-down menu. (The supported file types are CSV, JSON, PARQUET, AVRO)

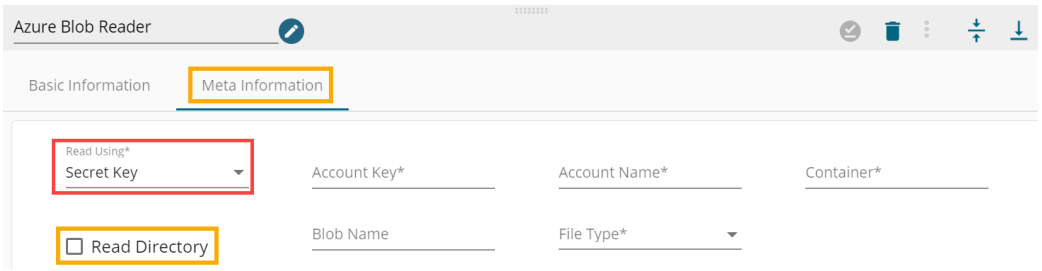


Note: If the '**Read Directory**' option is marked/enabled the '**Blob Name**' field will not appear.

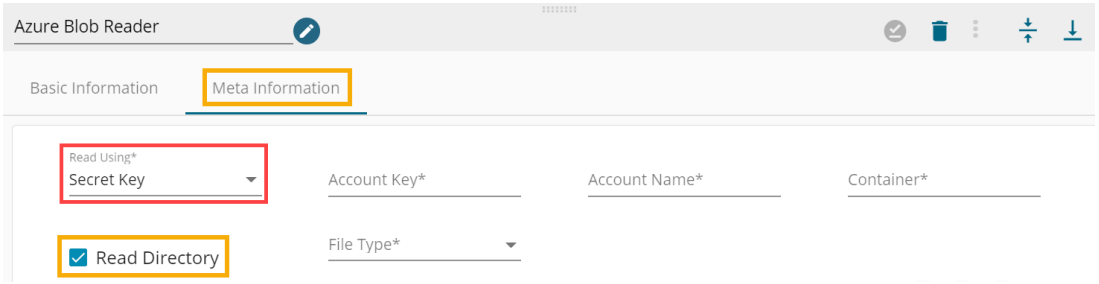


The Meta Information tab when the selected Read Using option is '**Secret Key**':

1. Read Using: the selected option is '**Secret Key**'.
2. Account Key (*): Provide azure blob account key.
3. Account Name (*): Provide the name of the Azure Blob account.
4. Container (*): Container details.
5. Read Directory: Select this option using the check box to read the blob name from the directory.
6. Blob Name: Provide a blob name.
7. File Type: Select a file type from the drop-down menu. (The supported file types are CSV, JSON, PARQUET, AVRO)



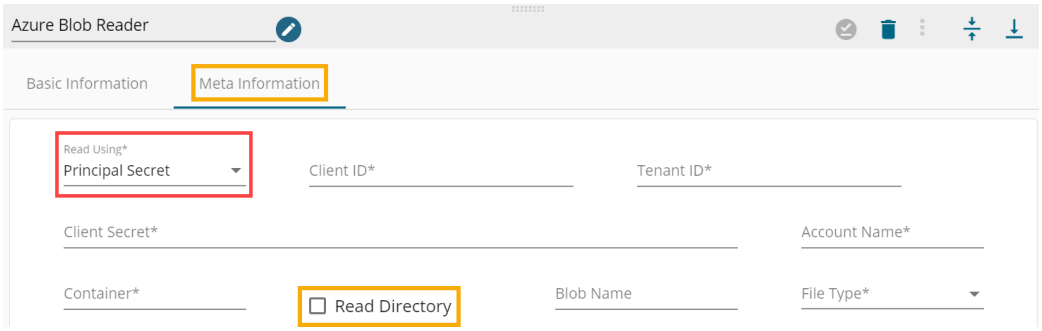
Note: If the '**Read Directory**' option is marked/enabled the '**Blob Name**' field will not appear.



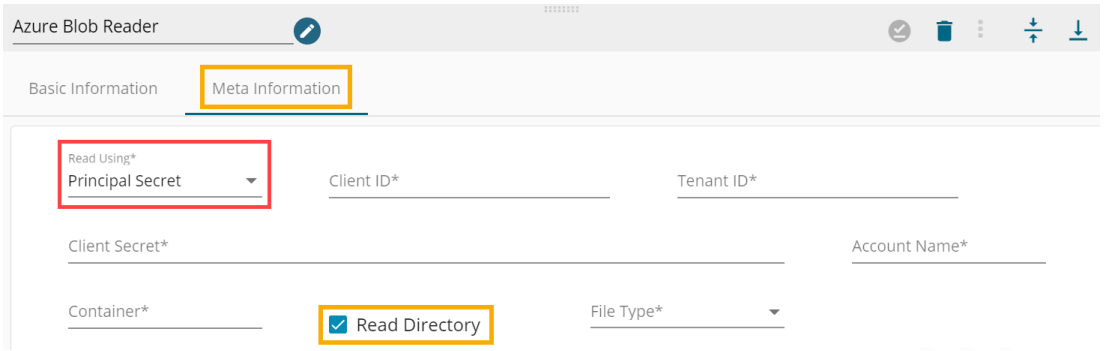
The Meta Information tab when the selected Read Using option is '**Principal Secret**':

1. Read Using: the selected option is '**Principal Secret**'.
2. Client ID (*): Provide client ID.
3. Tenant ID (*): Provide Tenant ID.
4. Client Secret (*): Provide client secret key
5. Account Name (*): Provide the name of the Azure Blob account.
6. Container (*): Container details.
7. Reader Directory: Enable or mark the box if wish to read the blob name from the directory.

8. Blob Name: Provide a blob name. (This field appears if the 'Read Directory' option is not marked/enabled.
9. File Type: Select a file type from the drop-down menu. (The supported file types are CSV, JSON, PARQUET, AVRO)



Note: If the '**Read Directory**' option is marked/enabled the '**Blob Name**' field will not appear.



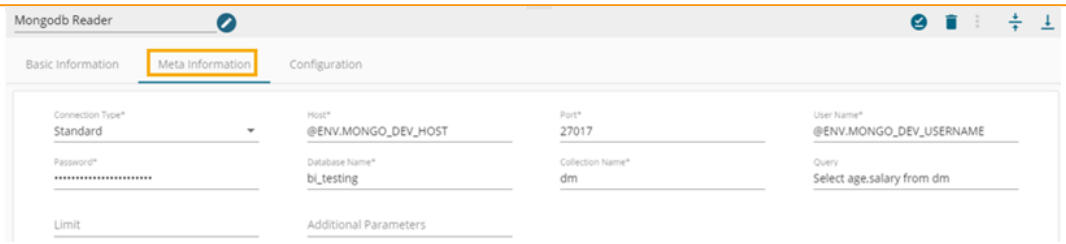
v) Click the '**Save Component in Storage**' icon after doing all the configurations to save the reader component.

vi) A notification message appears to inform you that the component properties have been saved.

Note:

a. The (*) symbol indicates that the field is mandatory.
b. Either table or query must be specified for the data readers except for SFTP Reader.
c. Selected Columns- There should not be a data type mismatch in the Column Type for all the Reader components.
d. The user can pass credentials as Environment Variable in some reader components instead of the actual credential values.

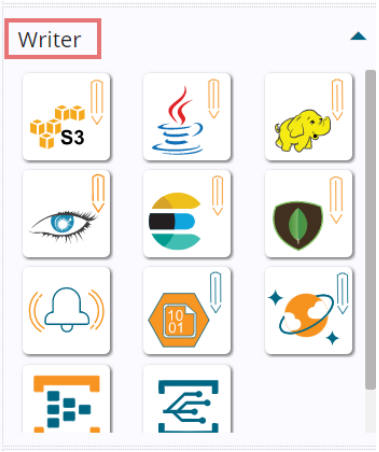
The supported readers are RDBMS, Mongo DB, Azure Reader, Azure Metadata Reader.
E.g., The following image displays the Environment Variable used to configure the 'Meta Information' tab for the Mongodb Reader.



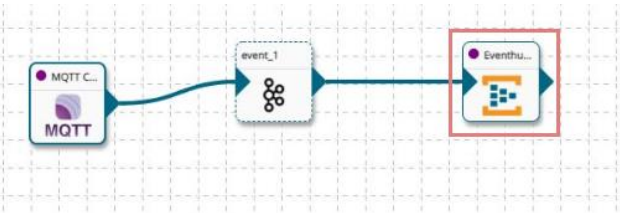
9.2. Writer

The Data Pipeline provides Writer components to write input data to the required place and create a pipeline workflow.

i) Navigate to the Pipeline Workflow Editor.
ii) Expand the Writer section using the Components Pallet.
iii) The following Writer components are available under the System tab of the Component Pallet.



Writers are generally used with the combination of a Reader and an Event in the Pipeline Workflow. As the Eventhub Publisher component is shown in the below image:

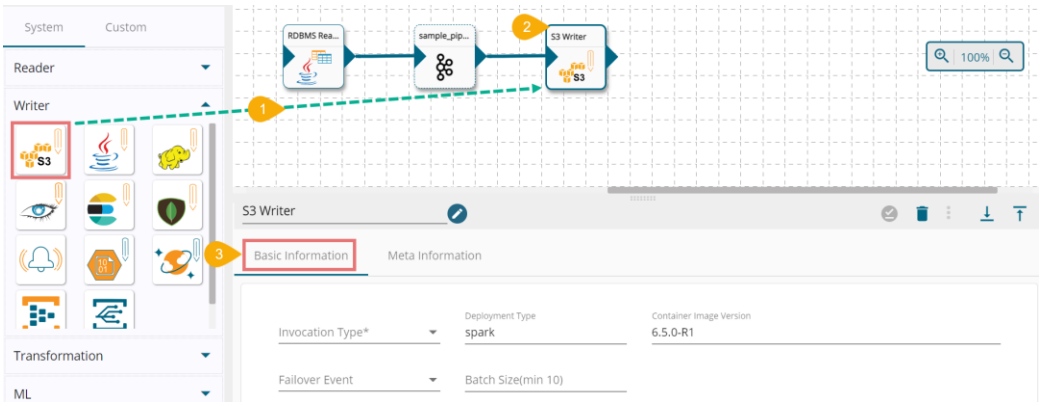


9.2.1. S3 Writer

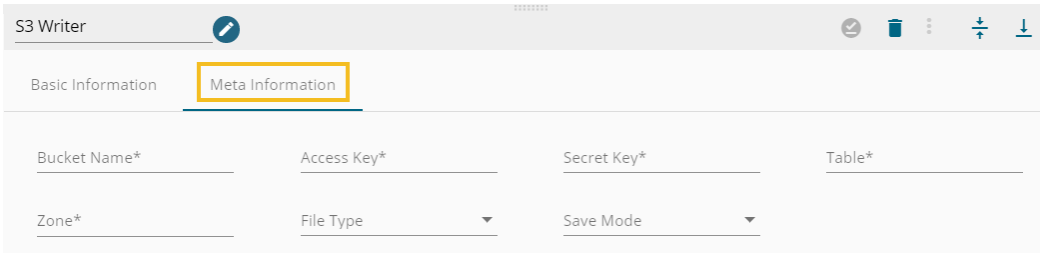
The S3 writer component writes data to the AWS S3 location.

i) Drag & Drop the '**S3 Writer**' component to the Workflow Editor and connect to a pipeline workflow containing a reader and an Event component to get the data.
ii) Click the dragged Writer component to get the Properties tabs below.
iii) Basic Information: It is the default tab to open for the S3 writer while configuring the component.
    1) Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.

2) Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
4) Failover Event: Select a failover Event from the drop-down menu.
5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

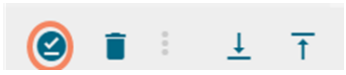


iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the S3 Writer.
a. Bucket Name (*): Provide Bucket name
b. Access Key (*): Access key shared by AWS to login
c. Secret Key (*): Secret key shared by AWS to login
d. Table (*): Mention the Table or object name which is to be read
e. Zone (*): Provide the zone information
f. File Type: Select a file type from the drop-down menu (CSV, JSON, PARQUET, AVRO are the supported file types)
g. Save Mode: supported save mode is '**Append**' at present.



v) Selected Columns- Select the columns that you wish to store in the S3 location. Provide Column Name, Alias Name, Column Type for the selected column(s).
(Add multiple columns to the Selected Columns section using the '**Add New Column**' option.)

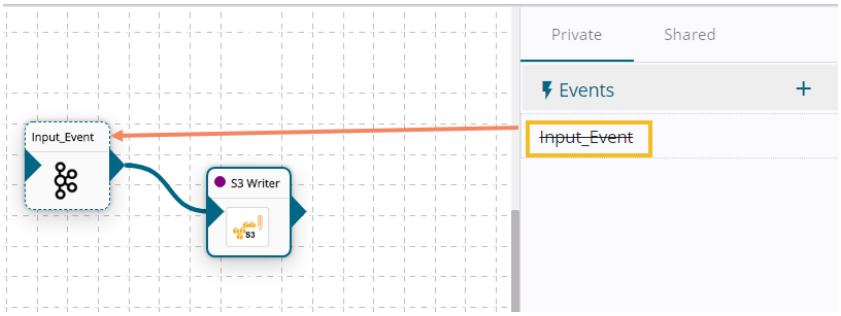
vi) Save the component properties.



vii) A message appears to assure that the component properties have been saved.

Note:

a. The S3 Writer requires data input from an Event and writes the data to the S3 location.
   i) Create one event and drag it to the workspace.
   ii) Connect the created input Event to the dragged S3 writer component.



b. The data in the input event can come from any Ingestion, Readers, or shared events.

9.2.2. RDBMS Writer

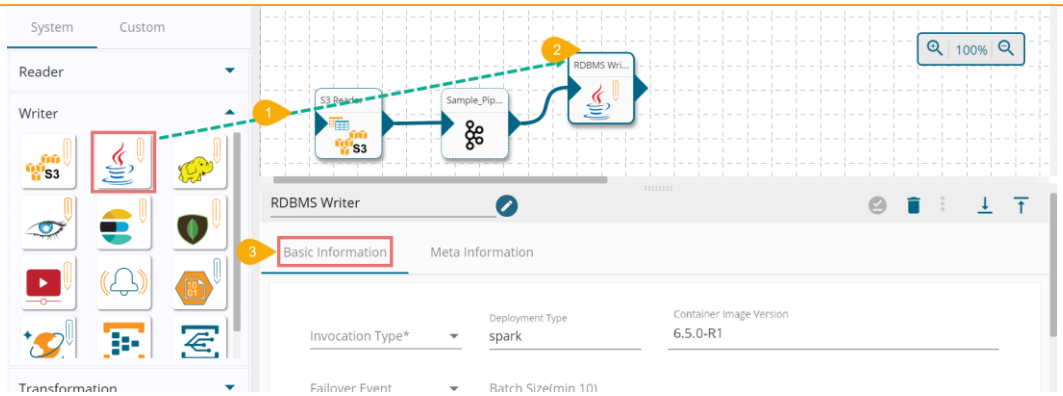
The RDBMS writer helps to write data to the supported relational databases.

i) Drag & drop the RDBMS Writer component to the Workflow Editor and connect with a pipeline workflow containing a reader and an event component to get the data.

ii) Click the RDBMS writer to open component properties tabs below.

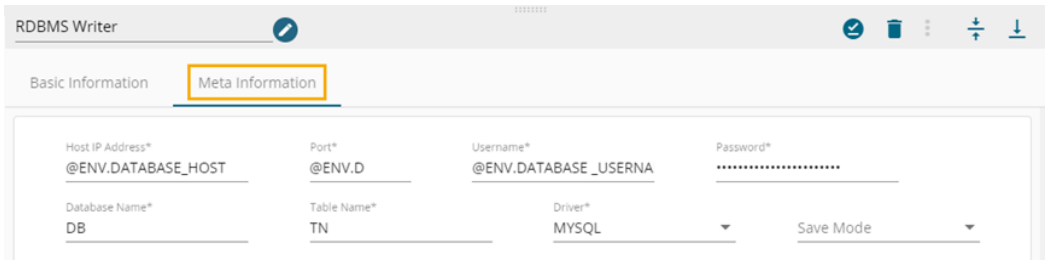
iii) Basic Information: It is the default tab to open for the RDBMS writer while configuring the component.

1) Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select either the 'Real-Time' or 'Batch' option from the drop-down menu.
2) Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
4) Failover Event: Select a failover Event from the drop-down menu.
5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the RDBMS Writer.

a. Host IP Address (*)
b. Port (*)
c. Username (*)
d. Password (*)
e. Database Name (*)
f. Table Name (*)
g. Driver (*) (MySQL/MSSQL/Oracle/Postgres)
h. Save Mode (Append/Overwrite/Upsert)

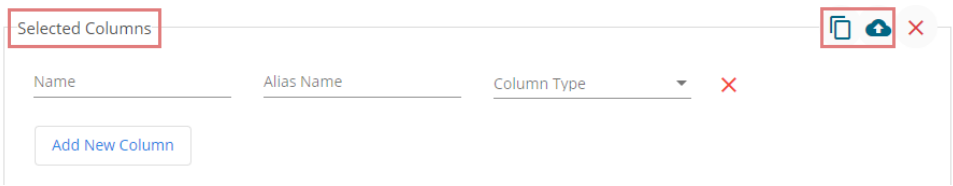


v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the '**Selected Columns**' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

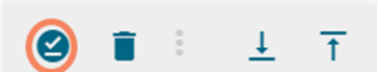
or

Use the '**Download Data**' and '**Upload File**' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the '**Upload File**' icon (file size must be less than 2 MB).
2. Download Data (Schema): The user can download the schema structure in JSON format by using the '**Download Data**' icon.



vi) Save the configured properties for the RDBMS writer.



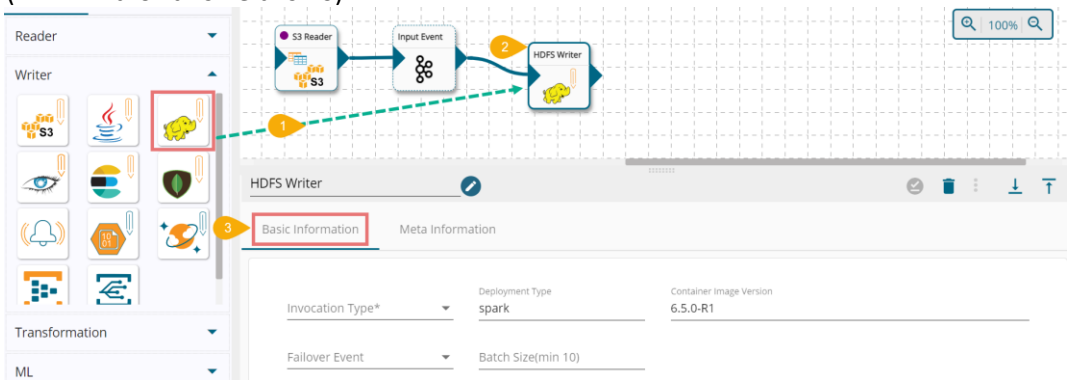
vii) A message appears to notify that the component properties got saved.

Note: The data input event can take data from Ingestion, Readers, or shared Event.

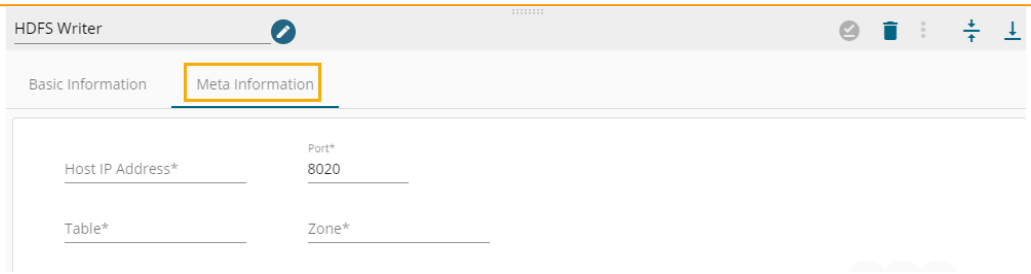
9.2.3. HDFS Writer

The HDFS writer writes the data to the HDFS location.

i) Drag the HDFS Writer component to the Workflow Editor and connect in a proper pipeline workflow to get the data.
ii) Click the HDFS Writer to open the component properties tabs below.
iii) Basic Information: It is the default tab to open for the HDFS Writer while configuring the component.
   1) Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select either 'Real-Time' or 'Batch' from the drop-down menu.
   2) Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   4) Failover Event: Select a failover Event from the drop-down menu.
   5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



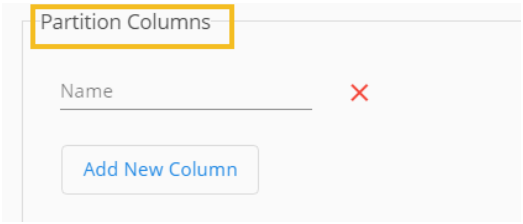
iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the HDFS Writer.
   1. Host IP Address (*): Hadoop IP address of the host
   2. Port (*): Port number
   3. Table (*): Table or object name which must be read
   4. Zone (*): HDFS Zone location



5. Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.
or
Use 'Download Data' and 'Upload File' options to select the desired columns.

- Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
- Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.

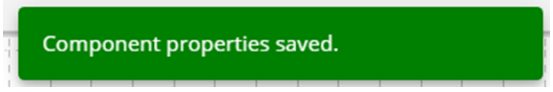
6. Partition Column- Provide a unique Key column name to partition data in Spark.



7. File Format (*): Select a file type from the drop-down menu (CSV, JSON, PARQUET, AVRO are the supported file types)
8. Save Mode (*): It displays the supported save mode. 'Append' is the supported save mode is at present.

v) After doing all the configurations click the '**Save Component in Storage**' icon provided in the Writer configuration panel to save the component.

vi) A message appears to notify about the component properties.

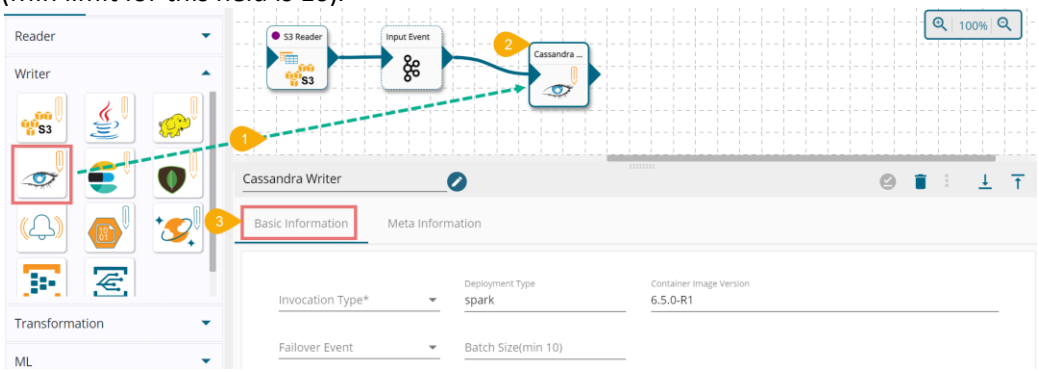


Note: The Input Event can take data from ingestion, Readers, or Shared Event.

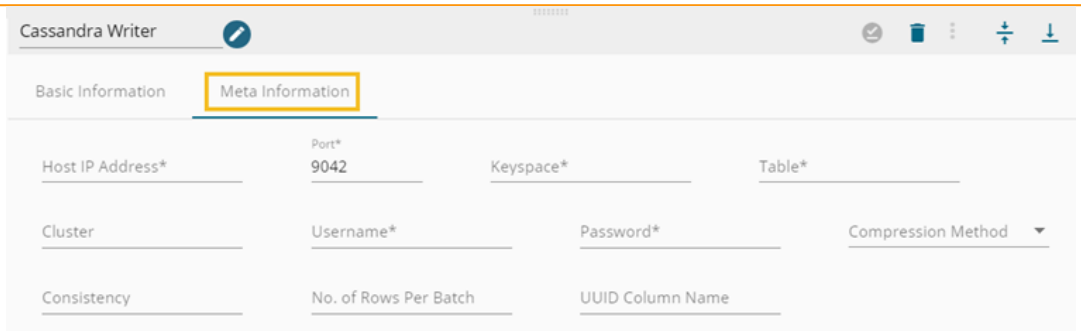
9.2.4. Cassandra Writer

The Cassandra Writer writes data to the Cassandra Database.

i) Drag & drop the Cassandra Writer component to the Workflow Editor and connect to a pipeline workflow to get the data.
ii) Click the dragged Cassandra Writer component to open the component properties tabs below.
iii) Basic Information: It is the default tab to open for the Cassandra Writer while configuring the component.
   1) Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select 'Real-Time' or 'Batch' from the drop-down menu.
   2) Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   4) Failover Event: Select a failover Event from the drop-down menu.
   5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



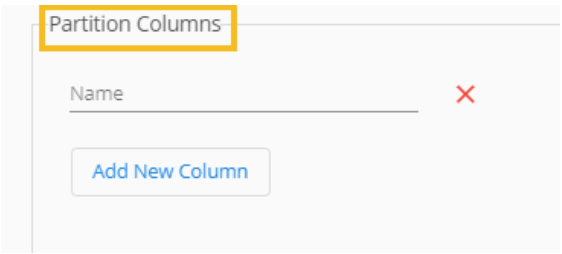
iv) Open the '**Meta Information**' tab and provide the connection-specific details.
   a. Host IP address (*): Provide hostname-based to connect the data connector.
   b. Port number (*): The server port number gets displayed by default.
   c. Keyspace (*): Provide a keyspace.
   d. Table (*): Provide a table name where you wish to save the data.
   e. Cluster: Provide a number to signify the number of clusters.
   f. Username (*): Provide the Username of the selected connection.
   g. Password (*): Provide a password for the database.
   h. Compression Method: Select a method from the drop-down menu (the supported method is SNAPPY)
   i. Consistency: Provide a number to define consistency.
   j. No. of Rows per Batch: Provide the number of rows to be fetched batch-wise.
   k. UUID Column Name- Enter the UUID column name (Optional)



v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.
or
Use 'Download Data' and 'Upload File' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.

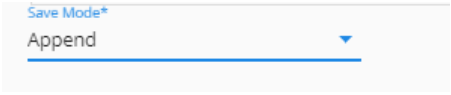
vi) **Partition Columns**: Provide the name of the partition column. Use the 'Add New Column' option to add a new partition column.



Note:

i. Users can add multiple Selected Columns and Partition Columns by clicking the '**Add New Column**' option.
ii. UUID Column Name, Selected Columns, and Partition Columns are optional.

vii) Save Mode- Select the mode to save the data from the drop-down menu (The supported save mode is the '**Append'**).



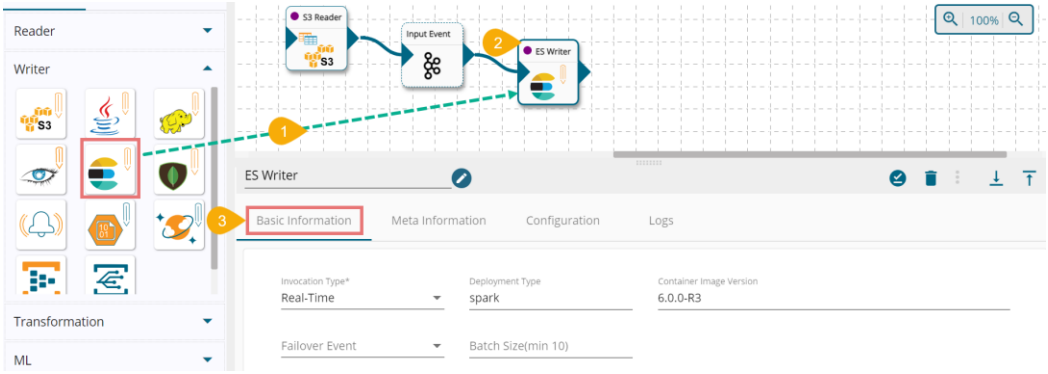
viii) Click the '**Save Component in Storage**' icon for the Cassandra Writer component.
ix) A message appears to notify the same.

Note: The input event can read data from Ingestion, Readers, and Shared Event.

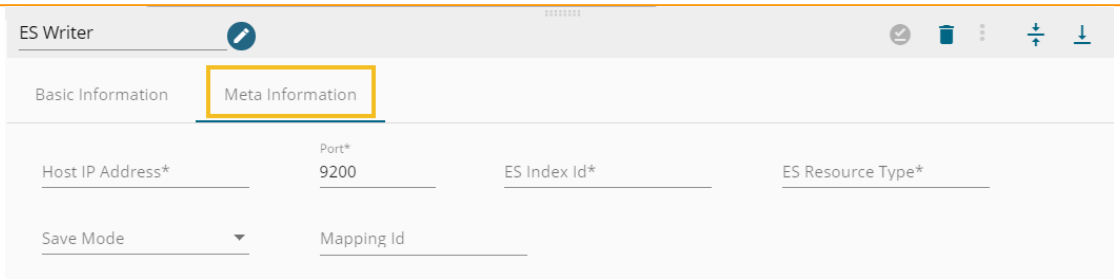
9.2.5. ES Writer

The ES writer writes the data to the Elastic Search engine.

i) Drag & drop the ES Writer component to the Workflow Editor and connect to a pipeline workflow to get the data.
ii) Click the dragged ES Writer component to open the component properties tabs below.
iii) Basic Information: It is the default tab to open for the ES Writer while configuring the component.
   1) Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select either '**Real-Time**' or '**Batch**' from the drop-down menu.
   2) Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   4) Failover Event: Select a failover Event from the drop-down menu.
   5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the ES Writer.
   a. Host IP Address
   b. Port number
   c. ES Index Id
   d. ES Resource Type,
   e. Save Mode: It displays the supported save modes. It supports only the '**Append**' option.
   f. Mapping Id- Provide mapping id (Optional)



v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the '**Selected Columns**' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.
or
Use '**Download Data**' and '**Upload File**' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.

vi) Click the '**Save Component in Storage**' icon for the ES Writer component.
vii) A message appears to notify the same.

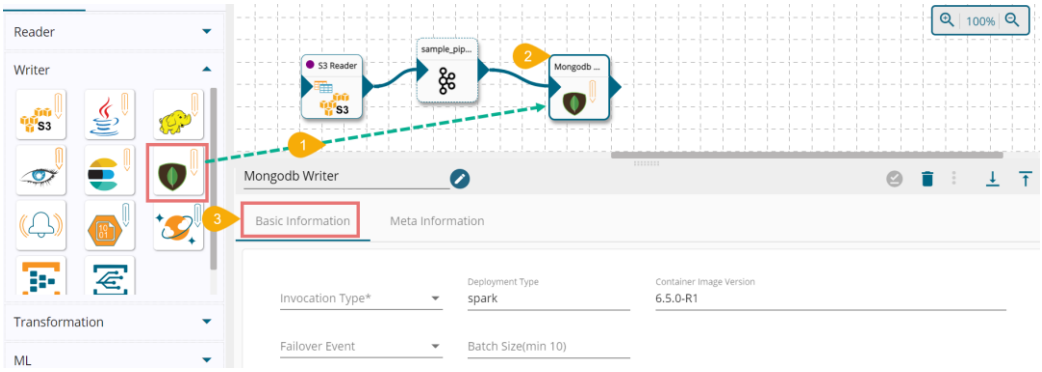
Note: The input event can read data from Ingestion, Readers, and Shared Event.

9.2.6. MongoDB Writer

The MongoDB writer writes the data to the database.

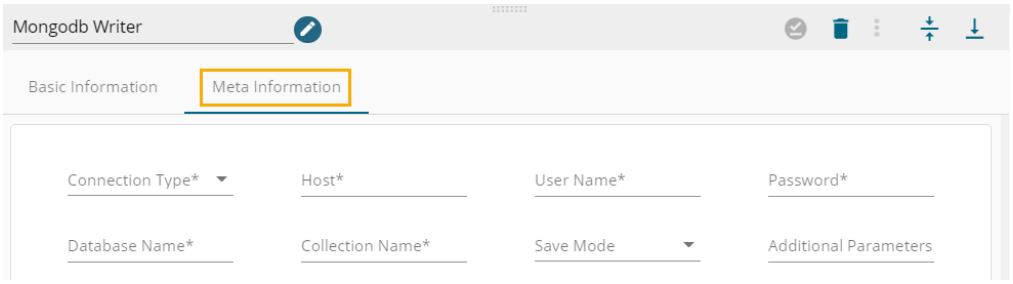
i) Drag & drop the MongoDB Writer component to the Workflow Editor and connect it to a pipeline workflow to get the data.
ii) Click the dragged MongoDB Writer component to open the component properties tabs below.
iii) Basic Information: It is the default tab to open for the MongoDB Writer while configuring the component.
   1) Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select 'Real-Time' or 'Batch' from the drop-down menu.
   2) Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   4) Failover Event: Select a failover Event from the drop-down menu.

5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



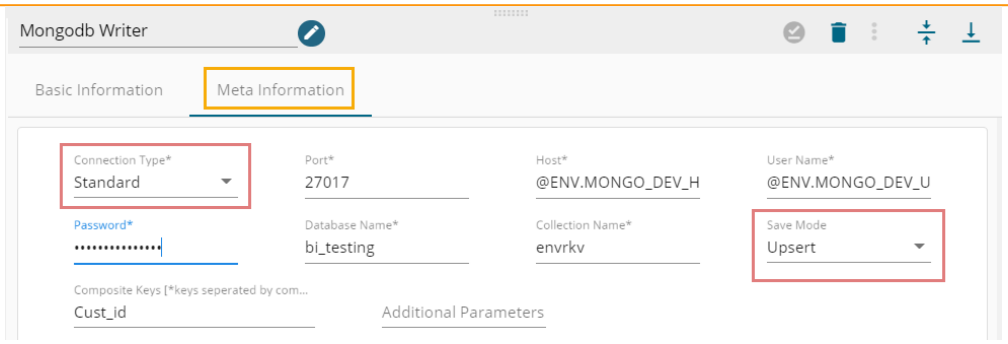
iv) Open the '**Meta Information**' tab and configure all the connection-specific details for the MongoDB Writer.

1. Connection Type: Select either of the options out of 'Standard' and 'SRV' connection types.
2. Port number (*): Provide the Port number (It appears only with the 'Standard' connection type).
3. Host IP Address (*): IP address of the host.
4. Username (*): Provide username.
5. Password (*): Provide a valid password to access the Mongo DB.
6. Database Name (*): Provide the name of the database where you wish to write data.
7. Collection Name (*): Provide the name of the collection.
8. Save Mode: Select an option from the drop-down menu (the supported options are Upsert, Ignore and Append).
9. Composite Keys (*): This field appears only when the selected save mode is '**Upsert**'. The user can enter multiple composite keys separated by commas.
10. Additional Parameters: Provide details of the additional parameters.

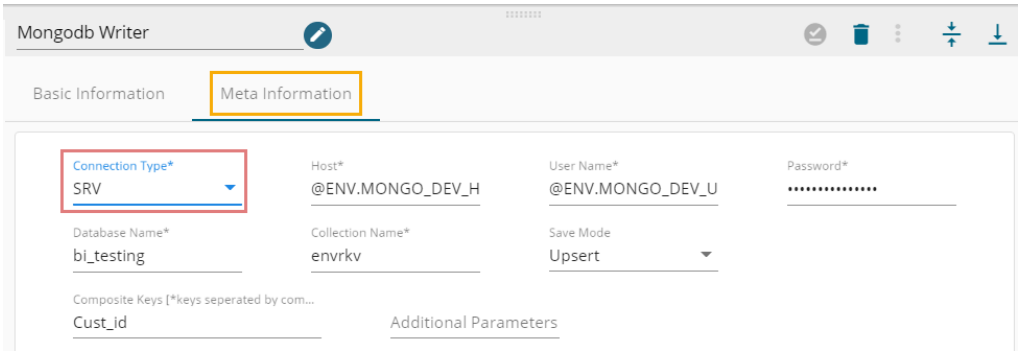


The Meta Information fields vary based on the selected '**Connection Type**' option.

i. **Meta Information** Tab with **Standard** as the **Connection Type**

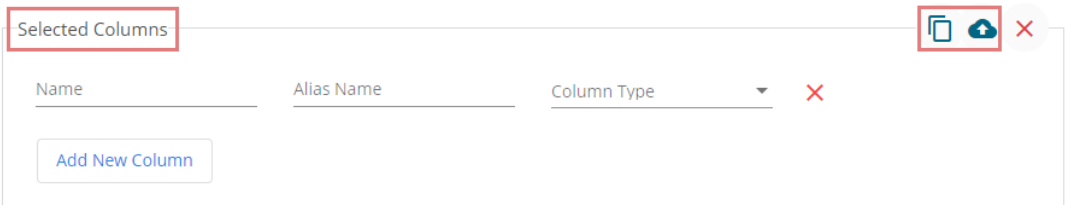


ii. **Meta Information** Tab with **SRV** as the **Connection Type**



v) Selected Columns: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.
or
Use '**Download Data**' and '**Upload File**' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the '**Upload File**' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.



vi) Click the '**Save Component in Storage**' icon for the MongoDB Writer component.
vii) A message appears to notify the same.

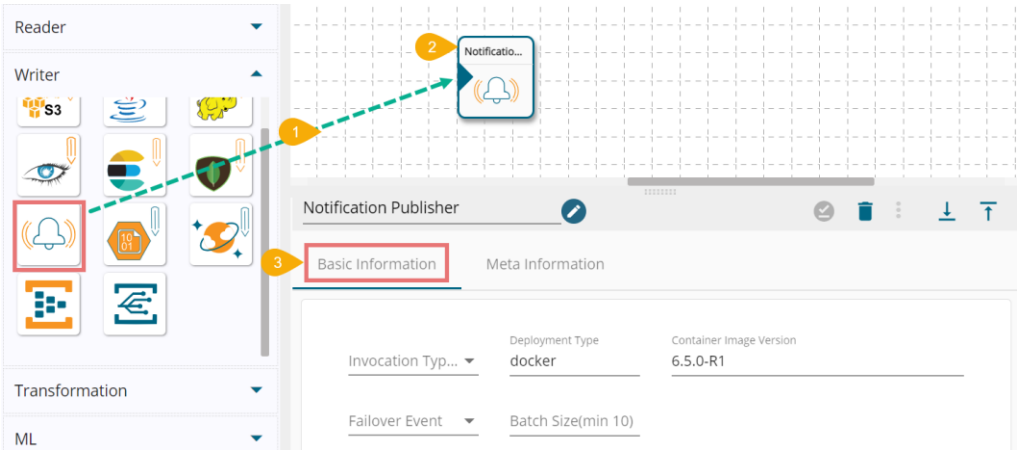
Note: The input event can read data from Ingestion, Readers, and Shared Event.

9.2.7. Notification Publisher

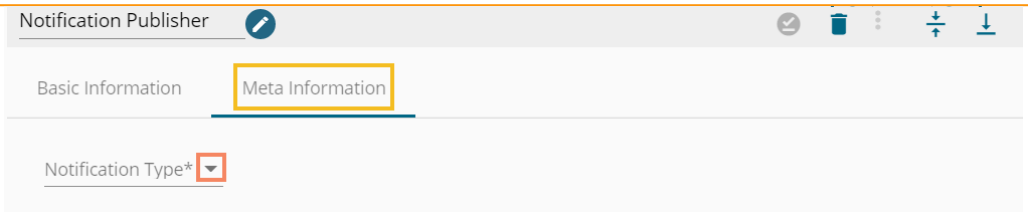
Notification Publisher Component as the name suggests is meant to publish the messages coming to Kafka as a notification. It is very helpful in acting scenarios where an immediate reaction is required in some conditions.

Example: A pipeline is monitoring an IoT sensor that sends the temperature of a machine. If the temperature reaches a certain threshold; the process engineer is to be notified immediately to maintain the same to stabilize the machine. Here, the Notification Publisher component is useful.

i) Drag and drop the Notification Publisher component to the Workflow Editor. (The user needs to connect the Notification component to a pipeline workflow to get the data).
ii) Click the 'Notification Publisher' component to open the component properties below.
iii) Basic Information: It is the default tab to open for the Notification Publisher while configuring the component.
   1) Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select 'Real-Time' or 'Batch' from the drop-down menu.
   2) Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   4) Failover Event: Select a failover Event from the drop-down menu.
   5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the Notification Writer.
   a. Type: Select a notification type using the drop-down menu. The provided options are:
      i. SNS
      ii. Google Pub/Sub
   b. Based on the selected Notification Type the remaining configuration fields get displayed.
   c. The user needs to complete the configuration process for the selected notification type and click the 'Save' icon to complete the configuration process for the Notification Publisher.



9.2.7.1. SNS as Notification Type

i) Navigate to the Meta Information tab provided for the Notification Publisher.

ii) Select the 'SNS' as the Notification Type option using the drop-down menu.

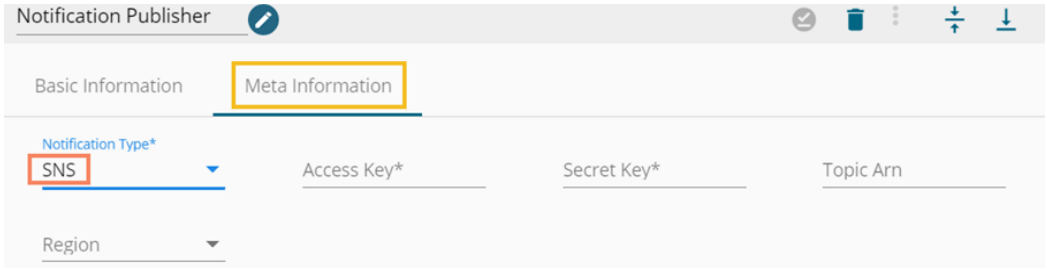
iii) The following configuration fields appear:

a. Access Key: Access keys consist of two parts: an access key ID (for example, AKIAIOSFODNN7EXAMPLE) can be retrieved from AWS key management.

b. Secret Key: A secret access key (for example, wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY) can also be retrieved from the AWS key management.

c. Topic Arn: The field gets filled after the topic creation. Navigate to Topics and then click the desired topic.

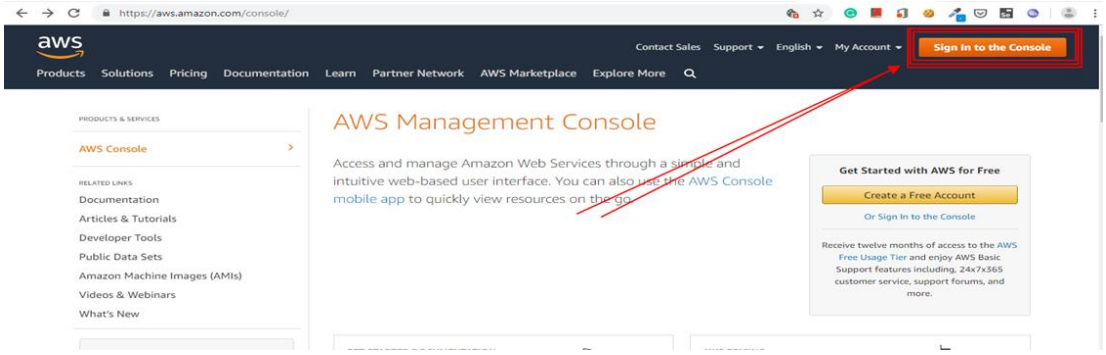
d. Region: Select an option from the drop-down menu where the topic has been created.



iv) The user needs to follow some more steps to create and subscribe to a topic. The below-given description explains those steps:

a. Navigate to the URL: https://aws.amazon.com/console/

b. Click the '**Sign In to the Console**' option.

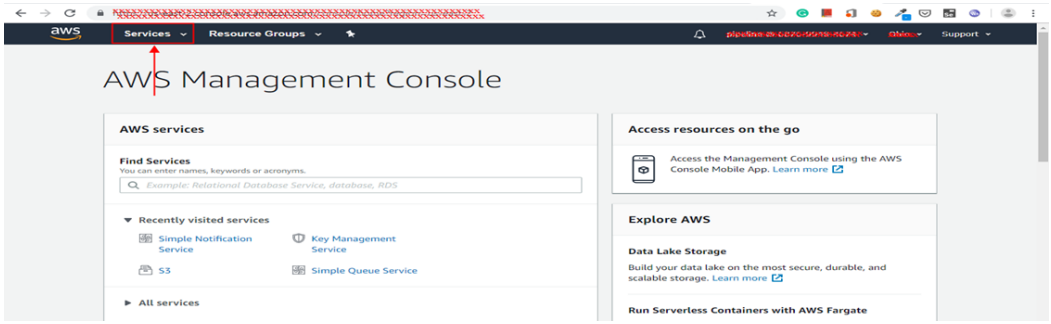


c. Sign in with your credentials.

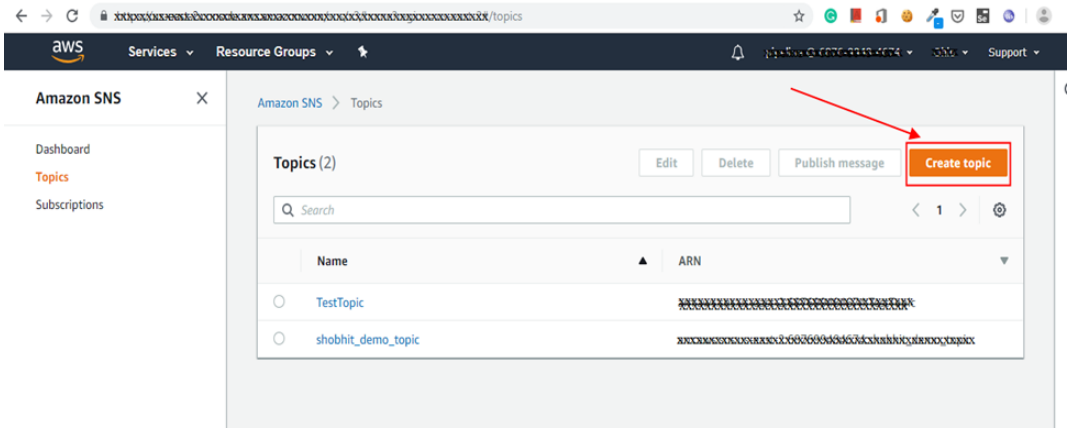
d. The AWS Management Console opens.

e. Click the '**Services**' option from the header drop-down.

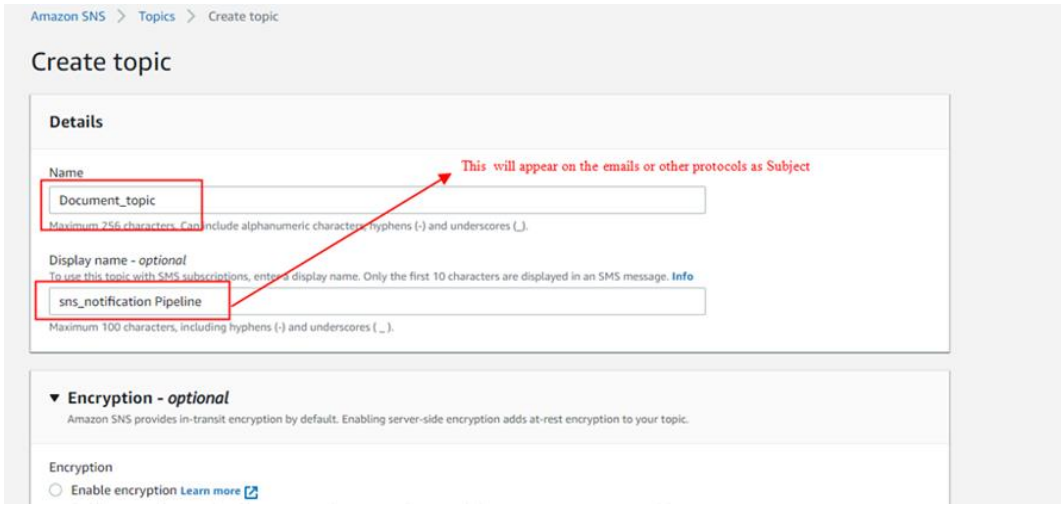
f. Search SNS service using the '**Find Services**' space.



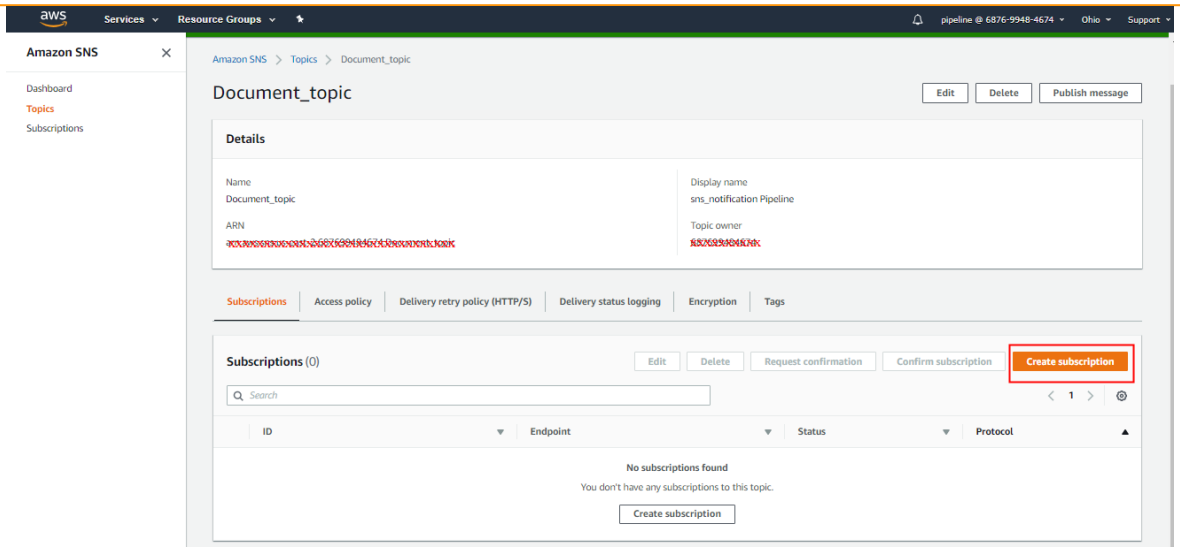
g. The Amazon SNS page opens.
h. Click the '**Create topic**' option.



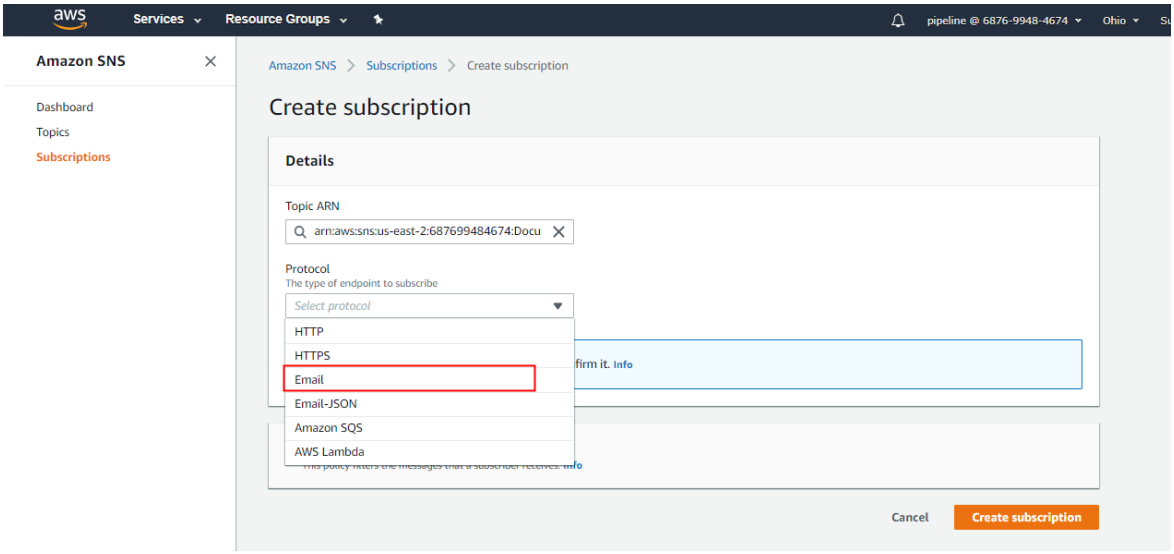
i. The Create topic page opens.
j. Provide essential information for the topic.



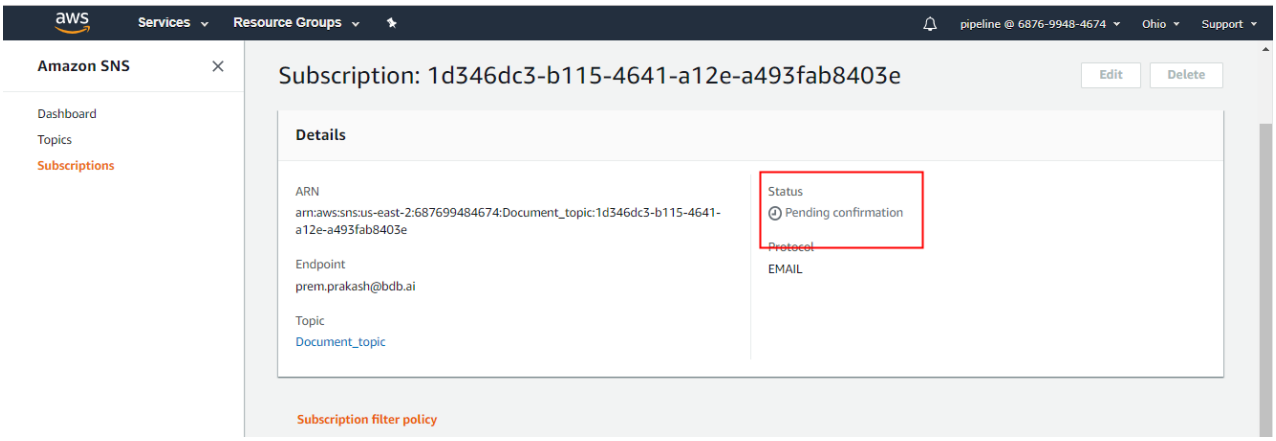
k. Click the '**Create Subscription**' option.



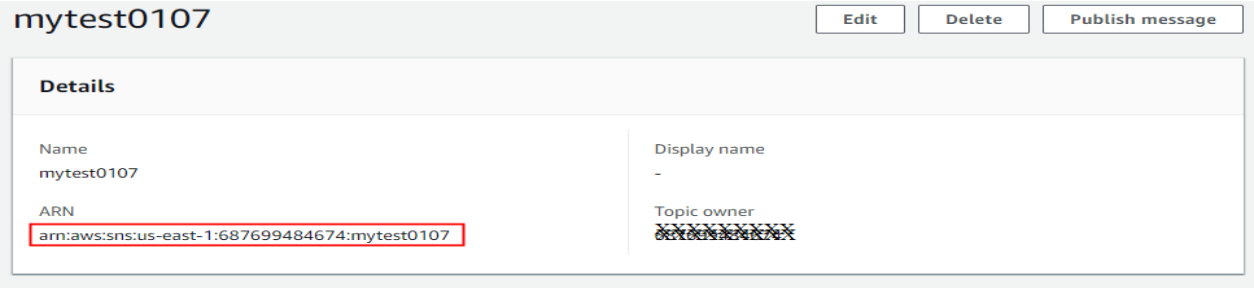
l. Search or provide the topic ARN.
m. Select a protocol from the available list.
n. Click the '**Create Subscription**' option.



NOTE: The user may get a '**Pending Confirmation**' notification of subscriptions as shown in the below image. In such cases navigate to the email id and confirm the subscription request.



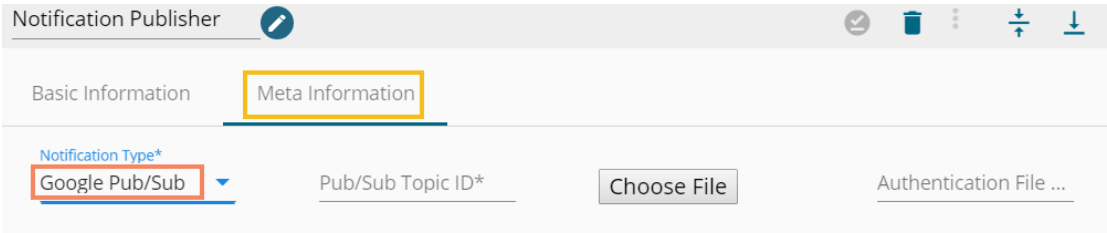
o. Copy the ARN of the Topic.



9.2.7.2. Google Pub/Sub as the selected Notification Type

i) Navigate to the Meta Information tab provided for the Notification Publisher.

ii) Select the '**Google Pub/Sub**' as the Notification Type option using the drop-down menu.



iii) The user needs to follow some more steps to get the Pub/Subtopic ID

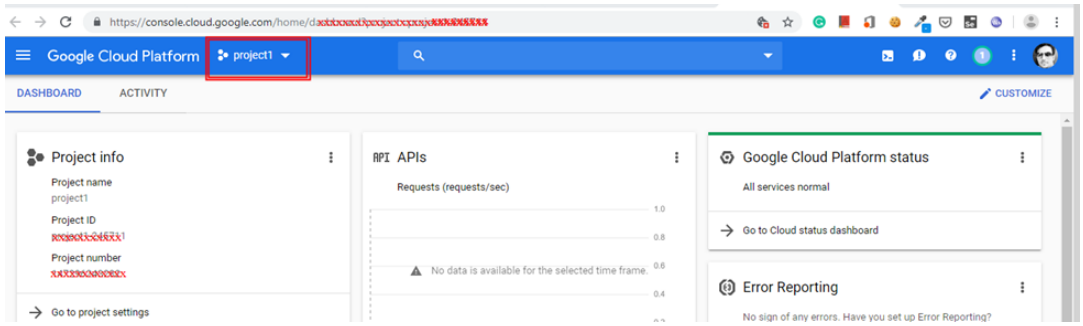
iv) Navigate to the https://console.cloud.google.com/

v) Log in with your credentials.

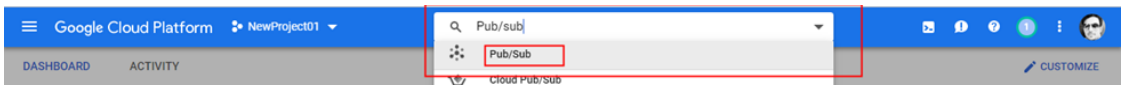
vi) The Landing page opens.

vii) Click on the marked area.

viii) Create a Project or use the desired one.



ix) Search for the Pub/Sub-option in the Search bar.



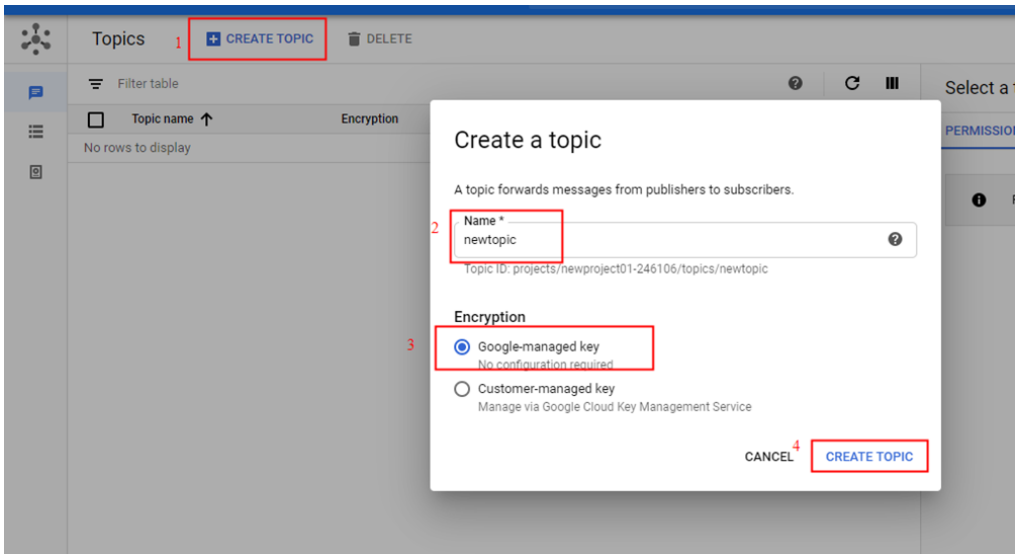
x) Create a Topic using the following steps:

a. Click the '**CREATE TOPIC**' option.

b. Provide a name for the new topic

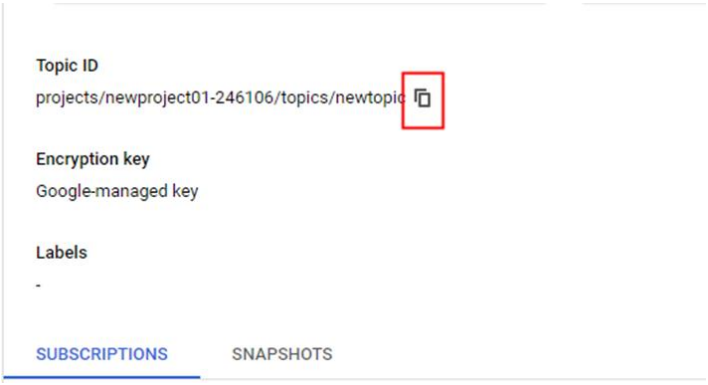
c. Select the '**Google-managed key**' as the Encryption option.

d. Click the '**CREATE TOPIC**' option.

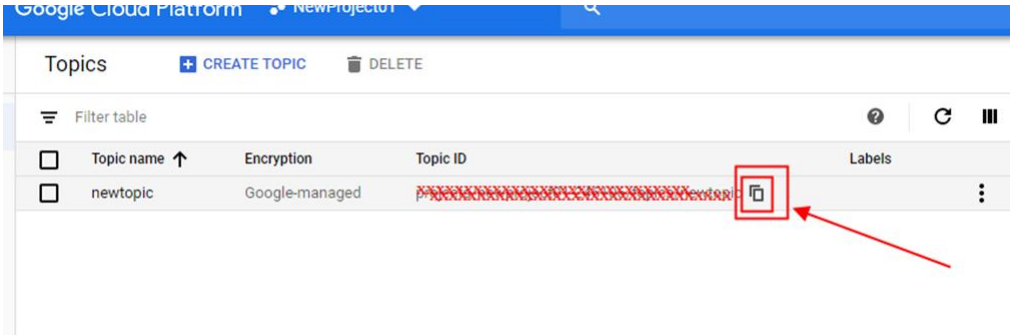


xi) The user gets either of the following screens:

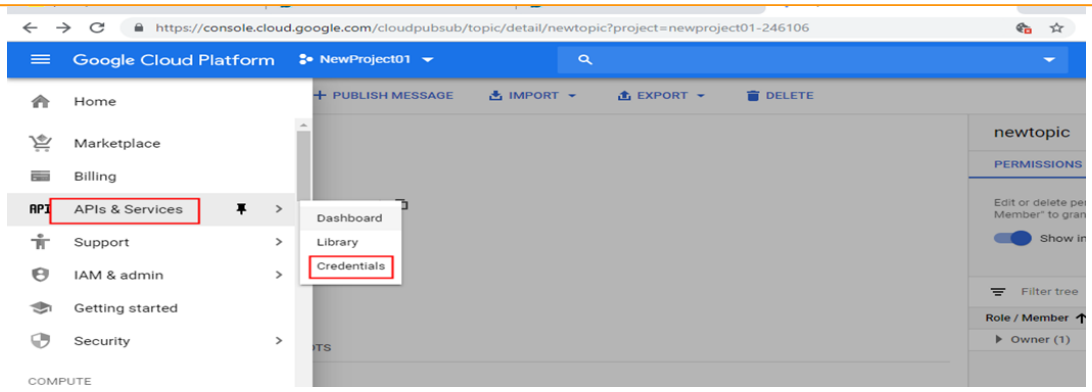
a. While using an existing Topic ID



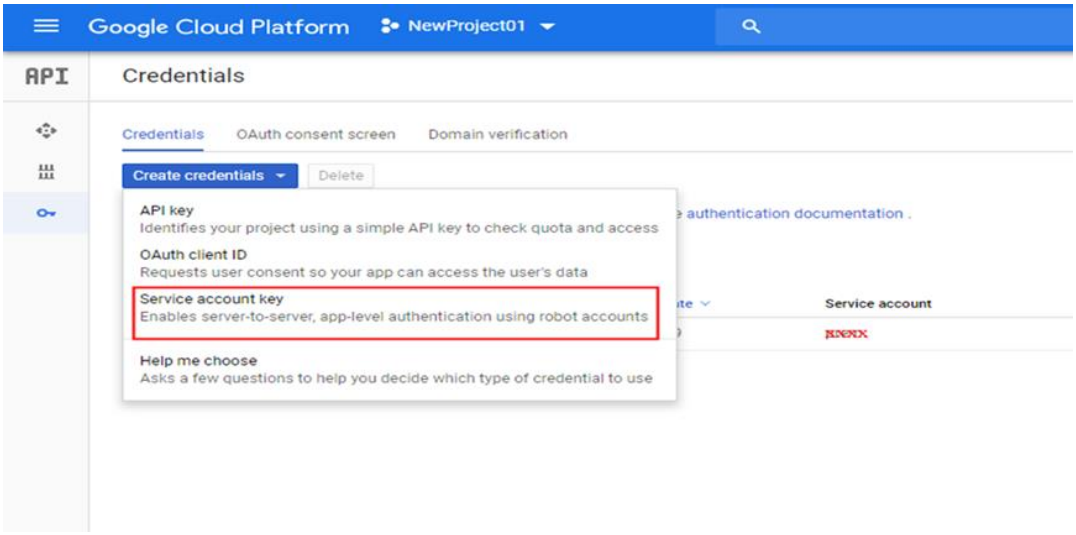
Alternatively, while using the newly created Topic ID



b. Create Auth-JSON for the current Project.

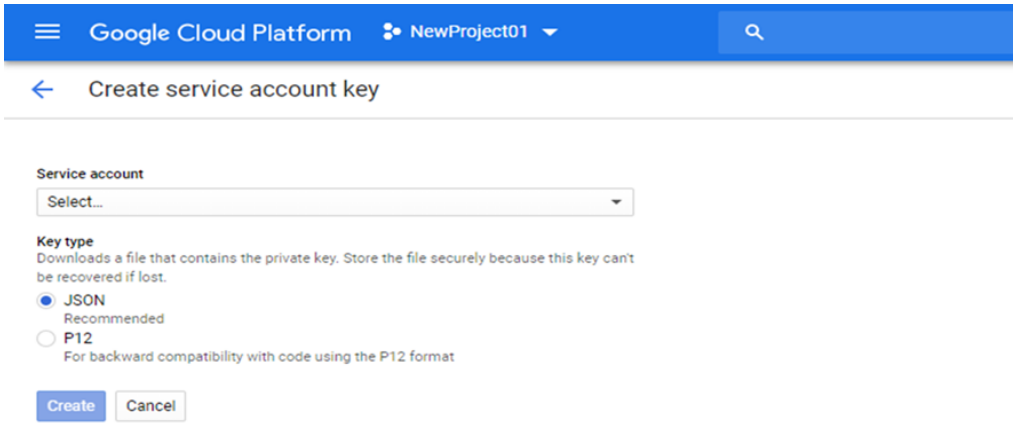


c. Create a Service account key by following the given steps:
   i. Click the '**Service account key**' option from the Credentials tab.



   ii. Select a Service account using the drop-down menu.
   iii. Download the JSON file using the radio button.
   iv. Click the '**Create**' option.

Use the copied Topic ID or the service account as metadata input.



.

v) Click the '**Save Component in Storage**' icon to save the component Properties for the Notification Publisher.

vi) A message appears to notify the same.

Note: The input event can read data from Ingestion, Readers, and Shared Event.

9.2.8. Azure Writer

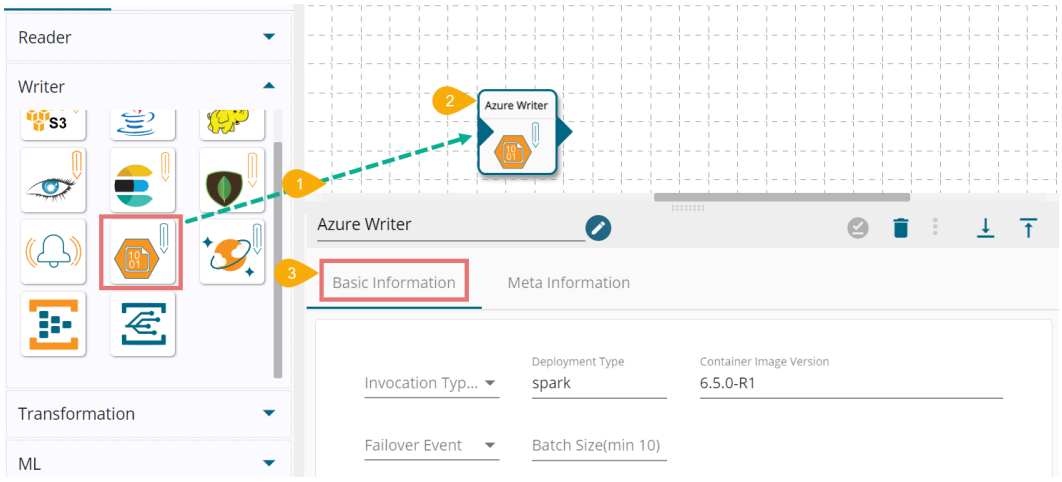
The Azure Writer writes the data to the Azure blob storage.

i) Drag and drop the Azure Writer component to the Workflow Editor (Connect the Azure Writer component to a pipeline workflow to get the data).

ii) Click on the dragged Azure Writer to get the component properties tabs.

iii) **Basic Information**: It is the default tab to open for the component while configuring the component.

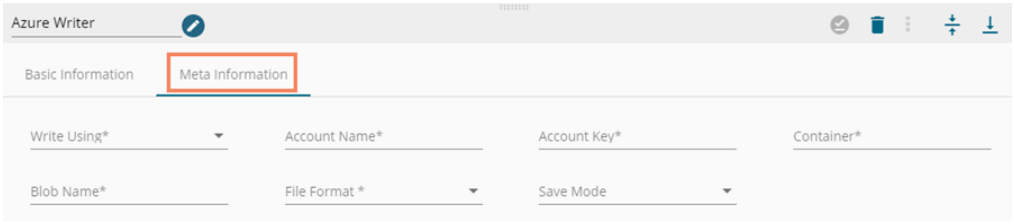
1) Invocation Type: Select an invocation mode out of '**Real-Time**' or '**Batch**' using the drop-down menu.
2) Deployment Type: It displays the deployment type for the reader component. This field comes pre-selected.
3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
4) Failover Event: Select a failover Event from the drop-down menu.
5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the Azure Writer.

1. Write Using (*): Select an option from the drop-down menu to connect to data. The supported options are '**Connection String**' and '**Secret Key**'.
2. Account Name (*): Name of the Azure account.
3. Account Key (*): Provide Account Key to login.
4. Container (*): Container details.

5. Blob Name (*): Provide a blob name. The Blob field appears if the '**Read Directory**' option is not enabled.
6. File Format (*): Select a file format option from the drop-down menu (CSV, JSON, PARQUET, AVRO are the supported file types).
7. Save Mode: Choose a mode to save data out of Append or Overwrite.



v) Selected Columns: There is also a section for the selected columns in the Meta Information tab if the user can select some specific columns from the table to read data instead of selecting a complete table so this can be achieved by using the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.
Or
Use 'Download Data' and 'Upload File' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.

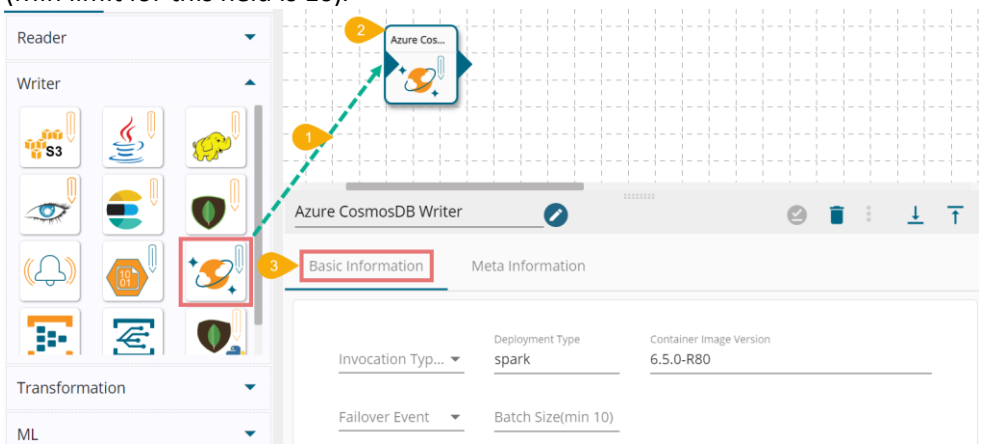
vi) Click the '**Save Component in Storage**' icon after doing all the configurations to save the writer component.
vii) A notification message appears to inform you that the component properties have been saved.

9.2.9. Azure Cosmos DB Writer

The Azure Cosmos DB Writer writes the data to the Cosmos DB.

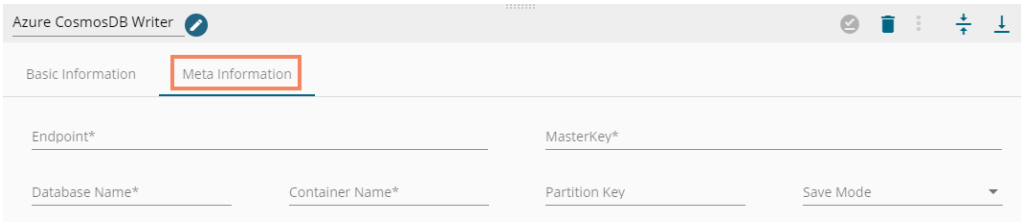
i) Drag and drop the Azure Cosmos DB Writer component to the Workflow Editor (connect the Azure Cosmos DB writer component to a pipeline workflow to get the data).
ii) Click on the dragged Azure Cosmos DB Writer to get the component properties tabs.
iii) **Basic Information**: It is the default tab to open for the component while configuring the component.

1) Invocation Type: Select an invocation mode out of '**Real-Time**' or '**Batch**' using the drop-down menu.
2) Deployment Type: It displays the deployment type for the reader component. This field comes pre-selected.
3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
4) Failover Event: Select a failover Event from the drop-down menu.
5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' tab and fill in all the connection-specific details for the Azure Cosmos DB Writer.

1. Endpoint (*):
2. Master Key (*): Provide the master key.
3. Database Name (*): Provide Account Key to login.
4. Container Name (*): Container details.
5. Partition Key: Provide the partition key.
6. Save Mode: Select a Save mode using the drop-down menu.

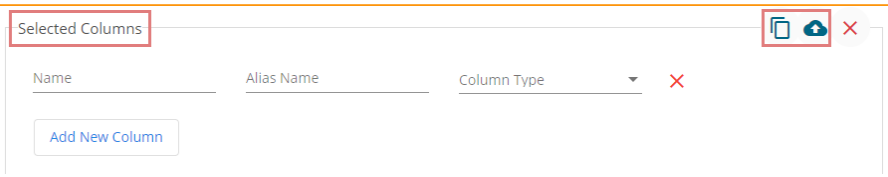


v) **Selected Columns**: There is also a section for the selected columns in the Meta Information tab if the user can select some specific columns from the table to read data instead of selecting a complete table so this can be achieved by using the '**Selected Columns**' section. Select the Source Field using the drop-down menu. Provide the name of the Destination Field. Select a Column Type from the drop-down menu.

Or

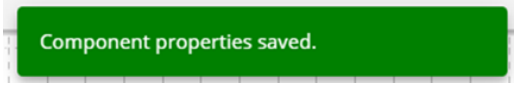
Use the 'Download Data' and 'Upload File' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.



vi) Click the '**Save Component in Storage**' icon after doing all the configurations to save the writer component.

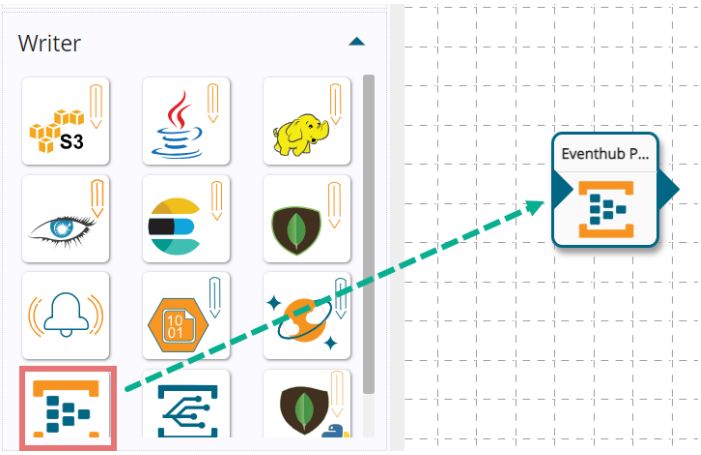
vii) A notification message appears to inform you that the component properties have been saved.



9.2.10. Eventhub Publisher

This Component consumes the data from the previous event and publishes the data on a given EventHub name. This data can be consumed by other applications. EventHub Subscriber in the data pipeline comes under the Writer component group.

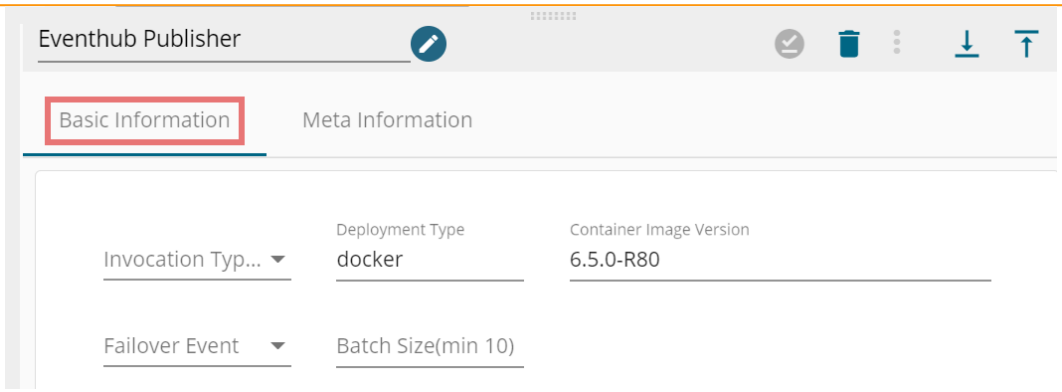
i) Drag and drop the Eventhub Publisher component to the Workflow Editor (connect the Eventhub Publisher to a pipeline workflow to get the data).



ii) Click on the dragged Eventhub Publisher component to get the component properties tabs.

iii) Configure the **Basic Information tab**.

1) Invocation Type: Select an invocation mode out of '**Real-Time**' or '**Batch**' using the drop-down menu.
2) Deployment Type: It displays the deployment type for the reader component. This field comes pre-selected.
3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
4) Failover Event: Select a failover Event from the drop-down menu.
5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



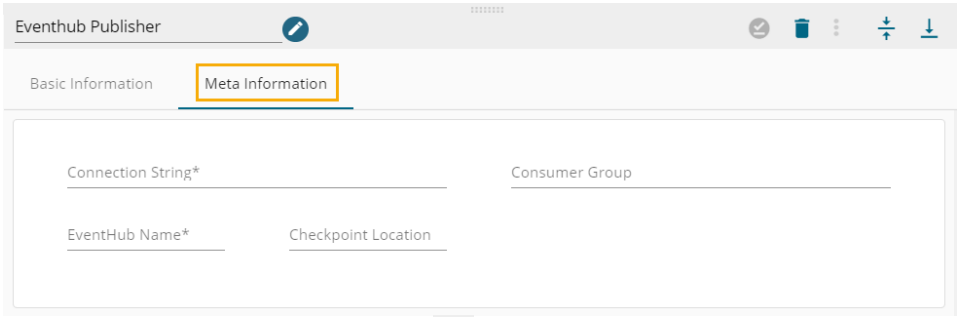
iv) Configure the **Meta Information tab** by providing the required fields.

a. **Connection String**: There is a connection string to connect to the event hub service (this can be found in Azure platform access manages].

b. **Consumer Group**: A consumer group is a view (state, position, or offset) of an entire event hub.

c. **Eventhub name**: Name of the subscribed event hub.

d. **Checkpoint location:** it is an HDFS location URL ("hdfs://<ip>/user/filename/Test"). It can be used when a component fails, and you want to read from the exact event where you left.



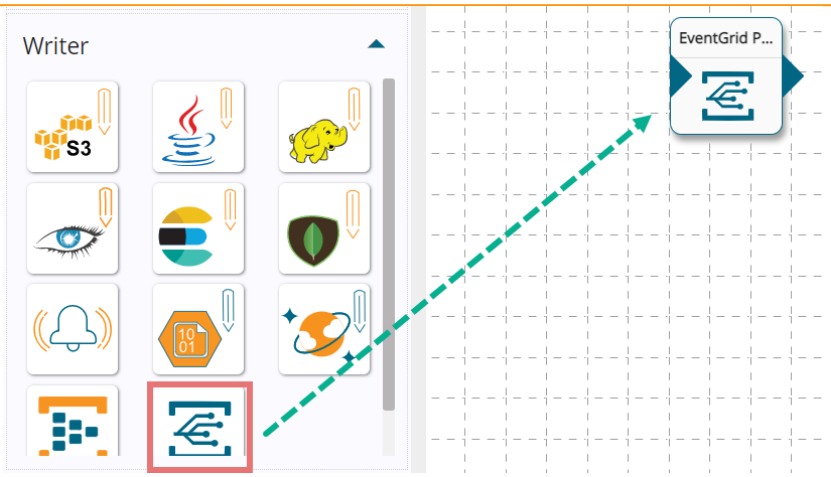
v) Click the '**Save Component in Storage**' icon after doing all the configurations to save the writer component.

vi) A notification message appears to inform you that the component properties have been saved.

9.2.11. EventGrid Producer

The Azure Event Grid provides the capability for building event-based applications.

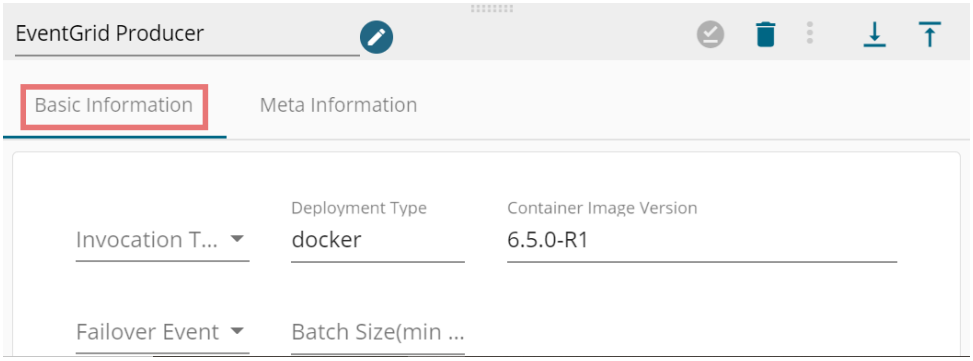
i) Drag and drop the EventGrid Producer component to the Workflow Editor (connect the EventGrid Producer to a pipeline workflow to get the data).



ii) Click on the dragged EventGrid Producer component to get the component properties tabs.

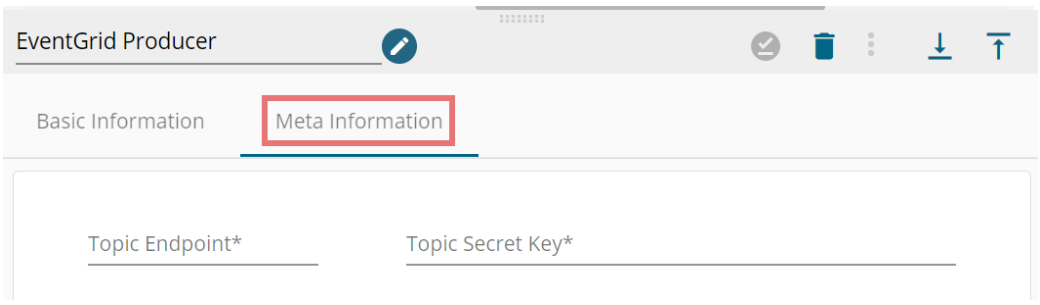
iii) Configure the **Basic Information tab**.

1) Invocation Type: Select an invocation mode out of '**Real-Time**' or '**Batch**' using the drop-down menu.
2) Deployment Type: It displays the deployment type for the reader component. This field comes pre-selected.
3) Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
4) Failover Event: Select a failover Event from the drop-down menu.
5) Batch Size: Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Configure the **Meta Information tab** by providing the required fields.

a. **Topic Endpoint**
b. **Topic Secret Key**

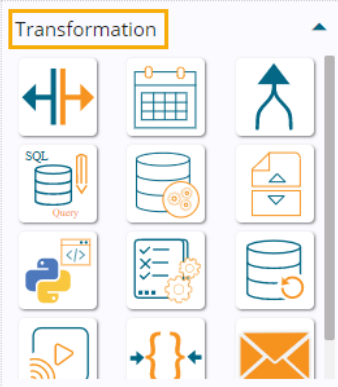


v) Click the '**Save Component in Storage**' icon after doing all the configurations to save the writer component.

vi) A notification message appears to inform you that the component properties have been saved.

9.3. Transformation

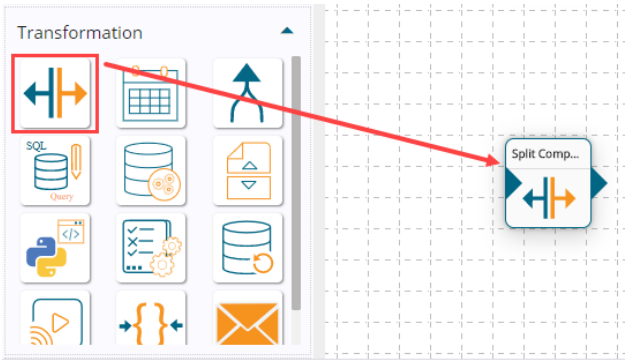
Transformation components allow users to transform the data. BDB Data Pipeline provides the following Transformation components to transform a variety of data.



9.3.1. Split Component

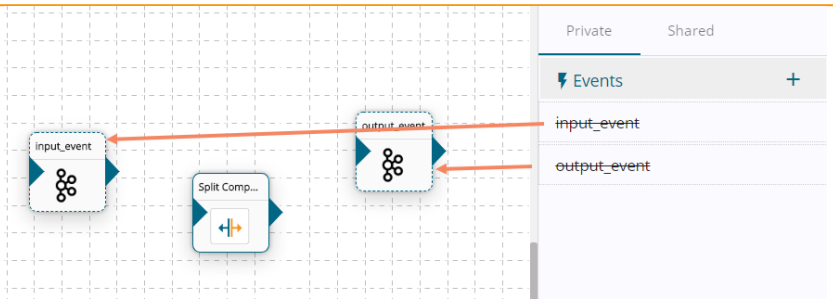
The Split component helps users to split the selected column(s) from the input data set based on the given regular expression.

i) Drag & drop the Split Component to the Workflow Editor.

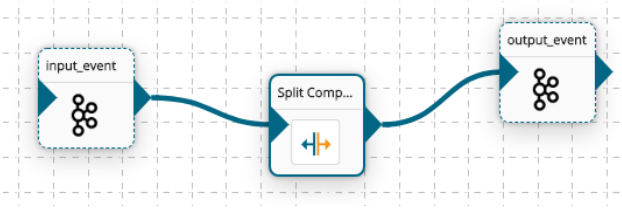


ii) The transformation component requires an input event (to get the data) and sends the data to an output event.

iii) Create two Events and drag them to the Workspace.



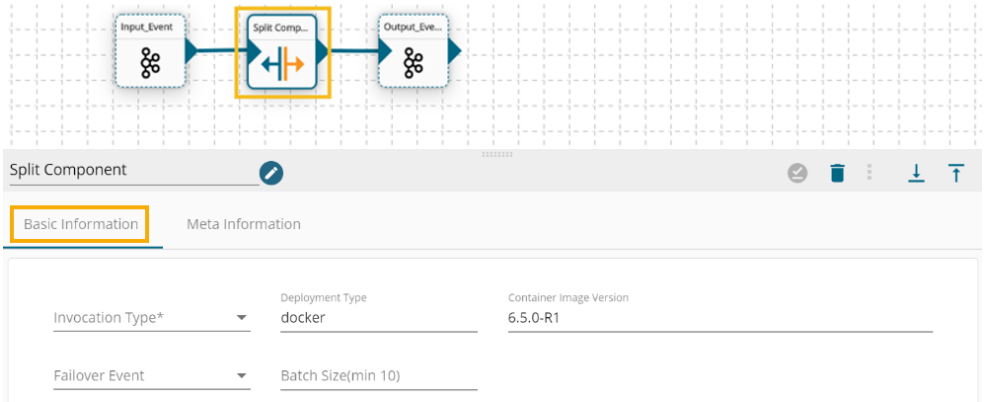
iv) Connect the input event and the output event (The data in the input event can come from any Ingestion, Readers, or shared events).



v) Click the dragged Split component to get the component properties tabs below.

vi) **Basic Information**: It is the default tab to open for the Split Component while configuring the component.

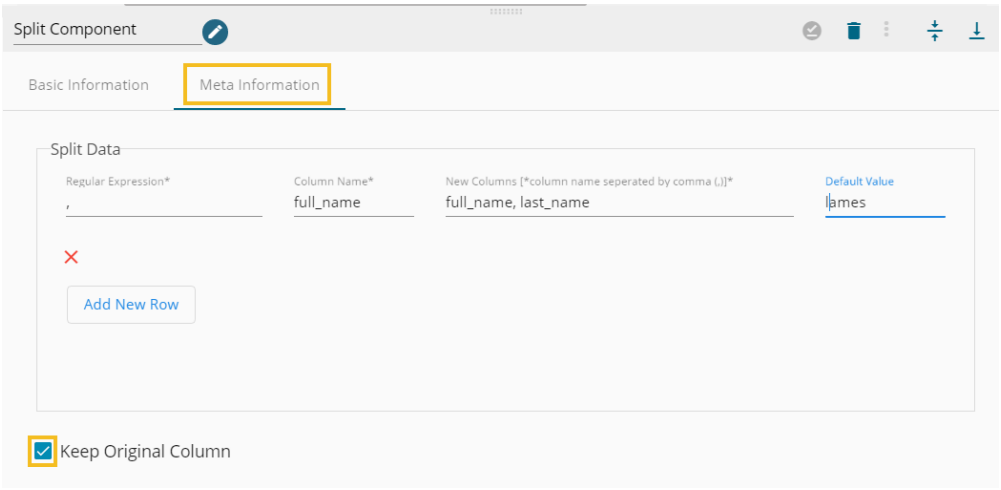
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



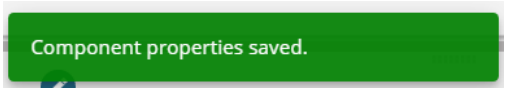
vii) Open the '**Meta Information' tab** and fill in all the connection-specific details for the Notification Writer.

1) Fill the Regular Expression by which the data gets split (E.g., Comma (,) is used as the Regular Expression in the below-given image).
2) Column Name- Provide the input column name.
3) New Columns- Provide Output columns name separated by a comma. These output columns get created after transformation.
4) Provide Default Value in the given field.

5) To perform multiple Split transformation actions, click the '**Add New Row**' option, and follow the above steps.
6) Enable the '**Keep Original Column**' option to keep the original column.



viii) Click the '**Save Component in Storage**' icon to save the configured properties for the Split component.
ix) A message appears to notify the successful update of the component.

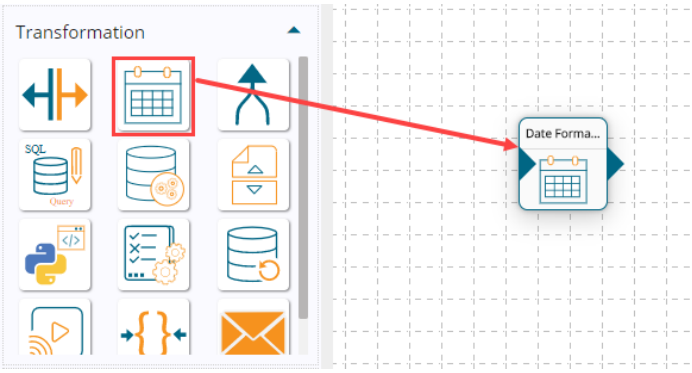


x) The Split component is ready to read the data coming from the input event, transform the data, and give the output data with the newly created column(s).

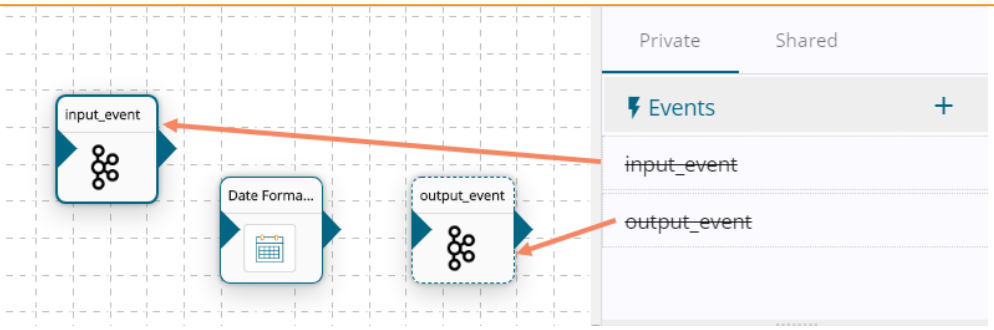
9.3.2. Date Formatter Component

Users can alter the Date format by using this transform component.

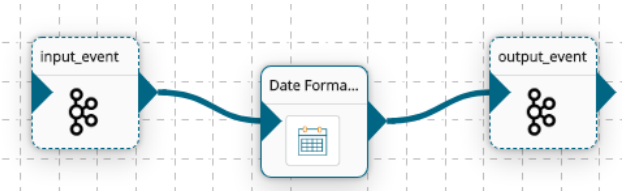
i) Drag & drop the Replace Text component to the Workflow Editor.



ii) The transformation component requires an input event (to get the data) and sends the data to an output event.
iii) Create two Events and drag them to the Workspace.

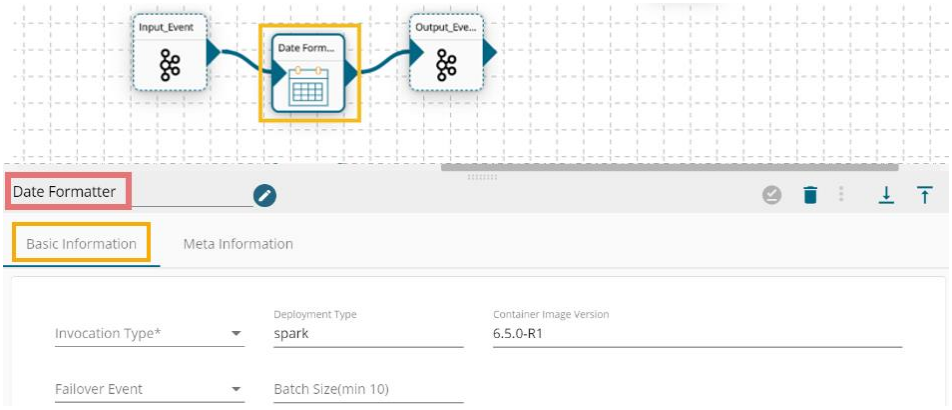


iv) Connect the input event and the output event (The data in the input event can come from any Ingestion, Reader, or shared events).



v) **Basic Information**: It is the default tab to open for the Date Formatter while configuring the component.

a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

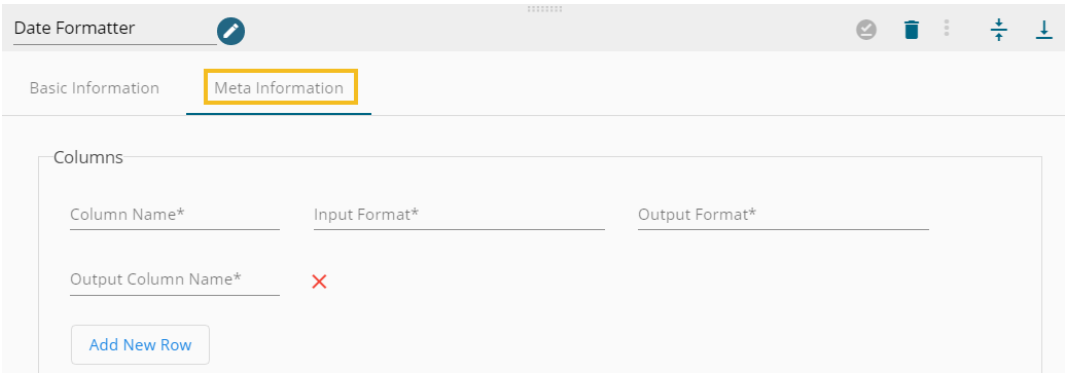


vi) Open the **Meta Information tab** and provide the connection-specific details.

a. Fill in the Column Name.
b. Select the input format.
c. Select the output format.
d. Provide a name for the Output Column.

(This component transforms date type for selected input column to provided output data type.)

e. To perform multiple Date Formatter transformations, click the 'Add New Row' option, and follow the above steps.

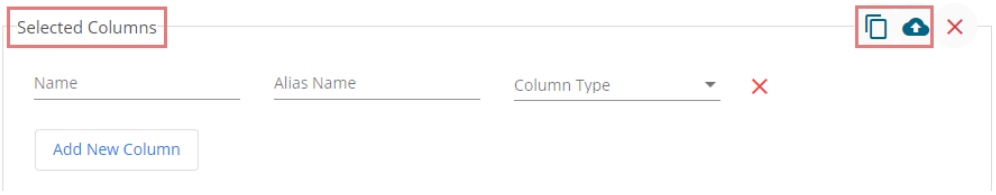


vii) **Selected Columns:** The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.

or

Use 'Download Data' and 'Upload File' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.



viii) Limit: Provide a limit of the records.

ix) Click the '**Save Component in Storage**' icon to save the configured properties of the Date Formatter component.

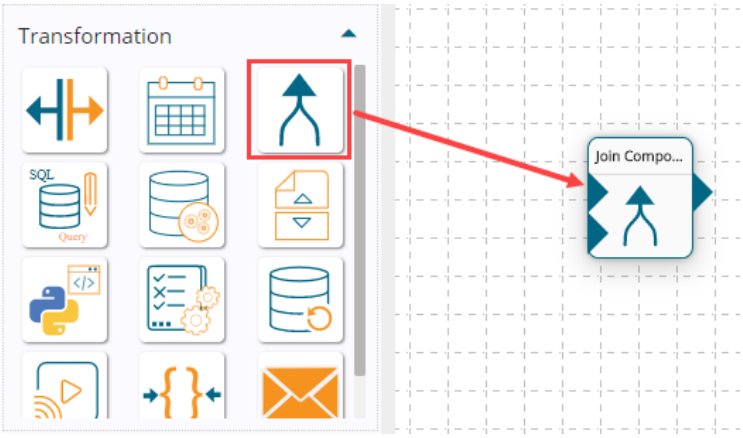
x) A message appears to notify the successful update of the component.

xi) The Date Formatter component is ready to read the data coming from the input event, transform the data, and send the newly formatted Date column(s) to the output event.

9.3.3. Join Component

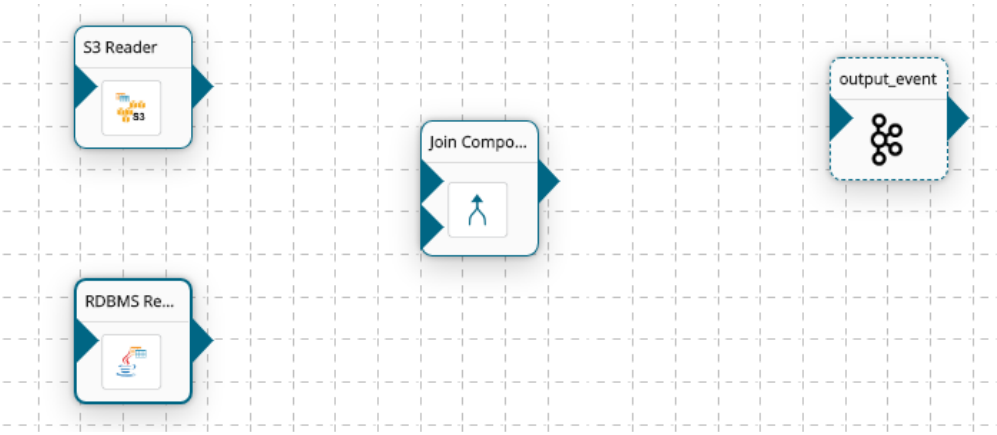
The Join transform allows users to join two or more input data sets as per the user-defined join conditions.

i) Drag and Drop the Join component to the Workflow Editor.

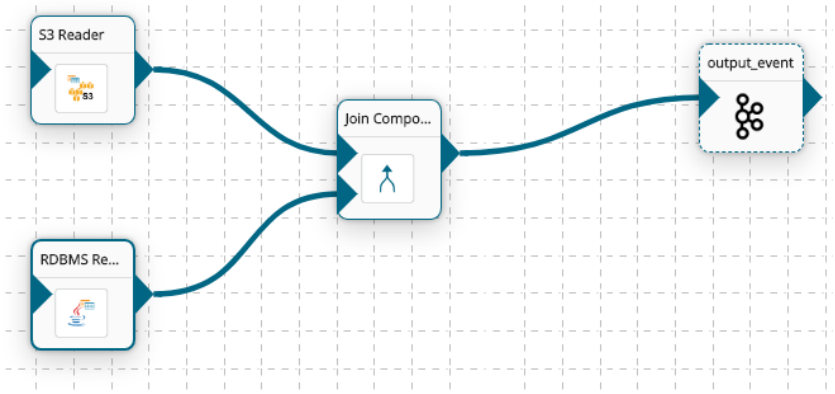


ii) This transformation component requires two input data sources and sends the data to an output event.

iii) Create two input data sources and one output Event and drag them to the workspace.

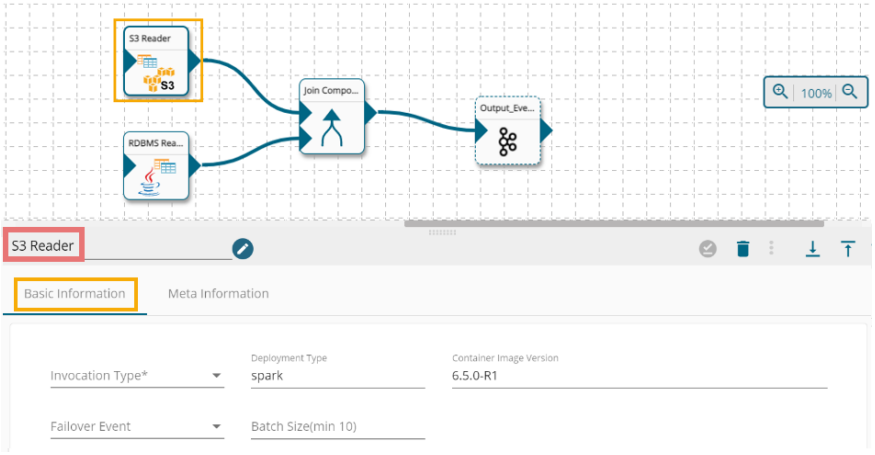


iv) Connect the input data sources and the output event. The data in input data sources can select only from Readers.



v) Configure the Input Data sources.

Note: Refer to the Reader section of the document for more information.



vi) Click the Join component to get the component properties field for the Join Component.

vii) The **Basic Information tab** opens by default.

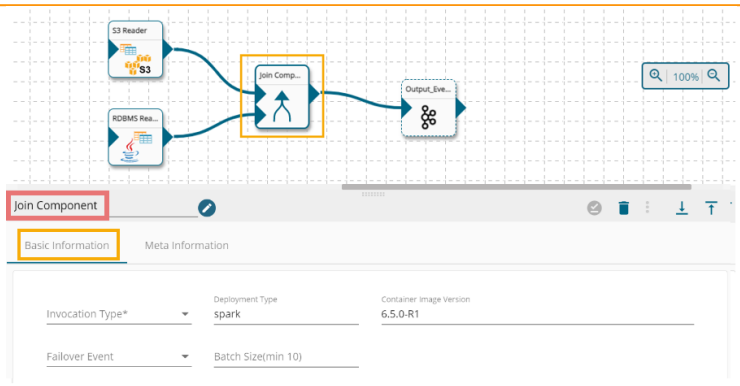
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover Event from the drop-down menu.

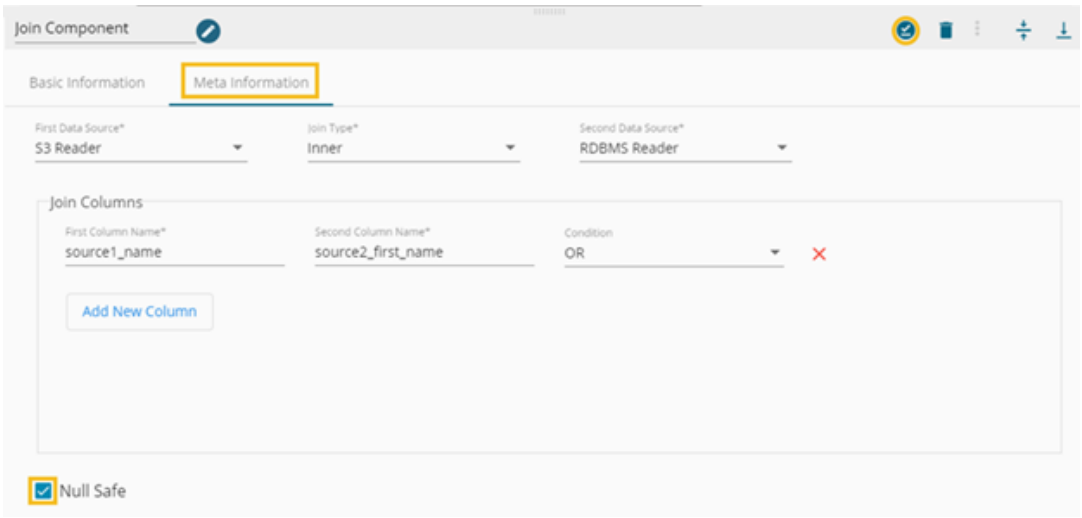
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



viii) Open the **Meta Information** tab and provide the connection-specific details.

a. Select the first data source.
b. Select a Join Type from the drop-down menu.
c. Select the Second Data Source.
d. Configure the Join Columns- Enter the First Column Name, Second Column Name, and select a Condition from the drop-down menu.
Note: The Condition field can be ignored for the single join.
e. To use multiple joins, click on the '**Add New Column**' option.
f. Use a checkmark in the box beside the '**Null Safe**' option to include the 'null' values for join columns.

ix) Click the '**Save Component in Storage**' icon to save the Join component properties.



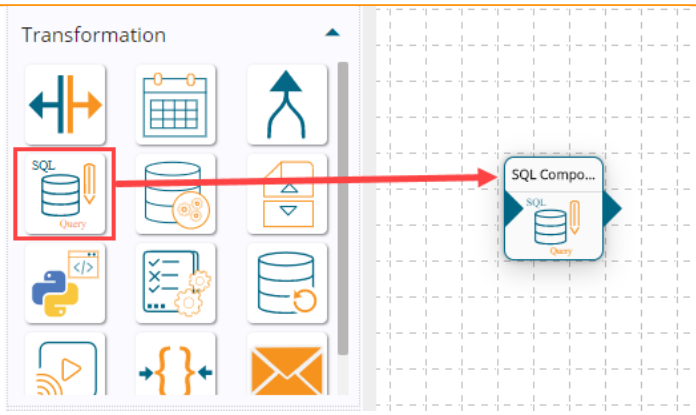
x) A message appears to notify the successful update of the component.

xi) The Join component is ready to read the data coming from the provided inputs, transform the data, and send the joined data to the output event.

9.3.4. SQL Component

This component helps users to get data as per the entered query.

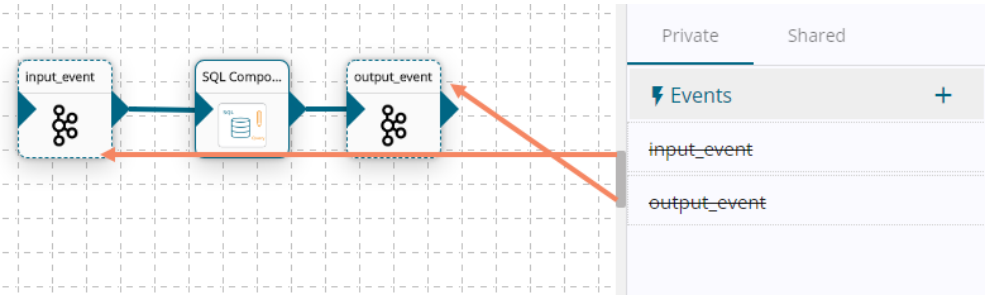
i) Drag and Drop the SQL component to the Workflow Editor.



ii) The transformation component requires an input event (to get the data) and sends the data to an output event.

iii) Create two Events and drag them to the Workspace.

iv) Connect the input event and the output event (The data in the input event can come from any Ingestion, Reader, or shared events).



v) Click the SQL component to get the component properties tabs.

vi) The **Basic Information** tab opens by default.

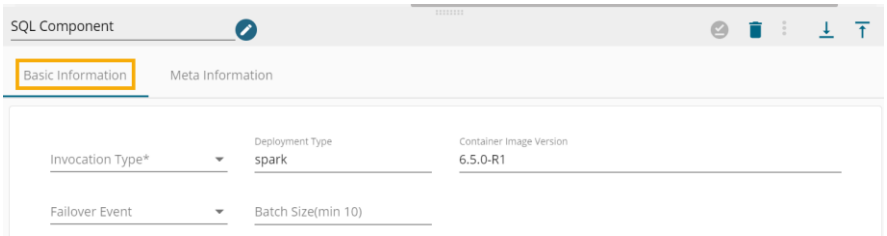
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover Event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

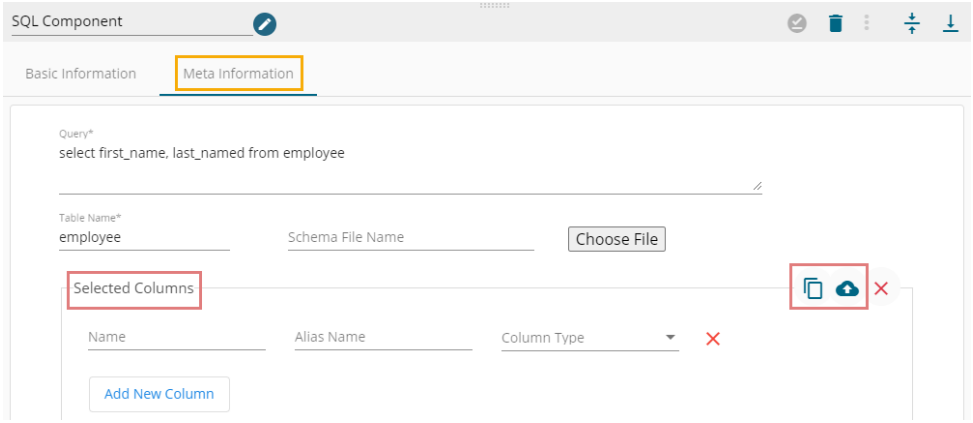


vii) Open the **Meta Information tab** and provide the connection-specific details**.**

a. Enter a valid data query to fetch data.

b. Provide the Table Name.

c. Schema File Name: This option helps for schema conversion.
d. Choose File: The user can browse a file and select it.
e. **Selected Columns**: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section. Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise keep the alias name the same as of column name and then select a Column Type from the drop-down menu.
or
Use 'Download Data' and 'Upload File' options to select the desired columns.
1. Upload File: The user can upload the existing system files (CSV, JSON) using the 'Upload File' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the 'Download Data' icon.



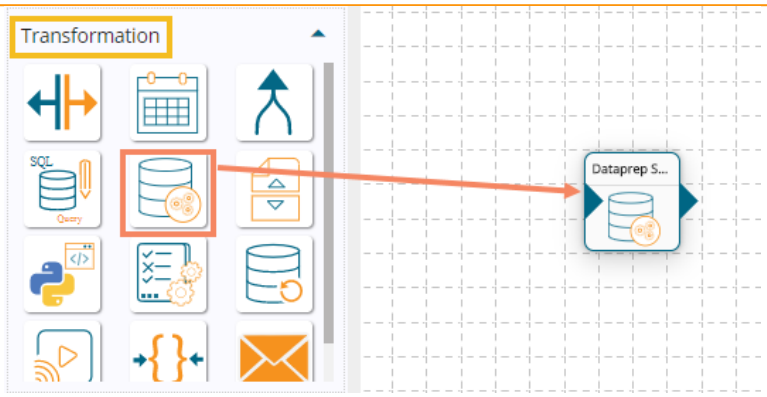
viii) Click the '**Save Component in Storage**' icon to save the component properties.
ix) A message appears to notify the successful update of the component.

x) The SQL component gets configured to read the data coming from the input event, transform the data, and return it to the output events.

9.3.5. Dataprep Script Runner Component

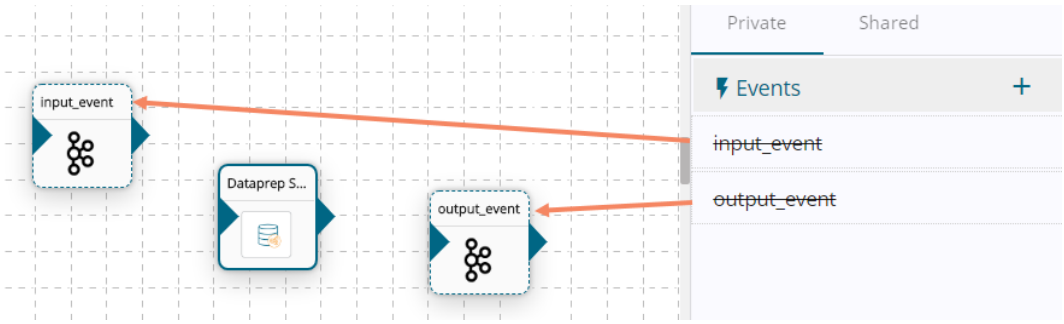
This transform component enables the users to use the exported Dataprep scripts in the Data Pipeline.

i) Drag and Drop the Dataprep Script Runner component to the Workflow Editor.

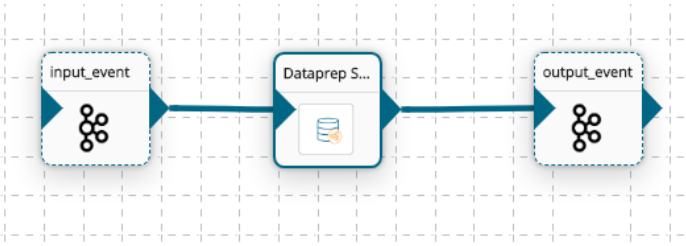


ii) The Transformation component requires an input event (to get the data) and sends the processed data to an output event.

iii) Create two Events and drag them to the Workspace.



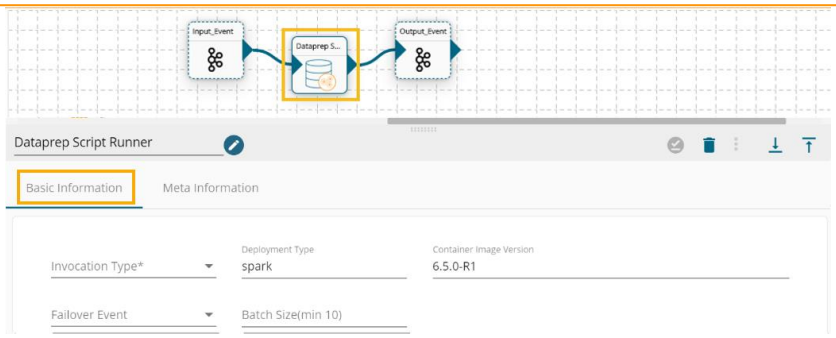
iv) Connect the input event and the output event. (The data in input event can come from any Ingestion, Reader, or shared events)



v) Click the Dataprep Script Runner component to get the component properties tabs.

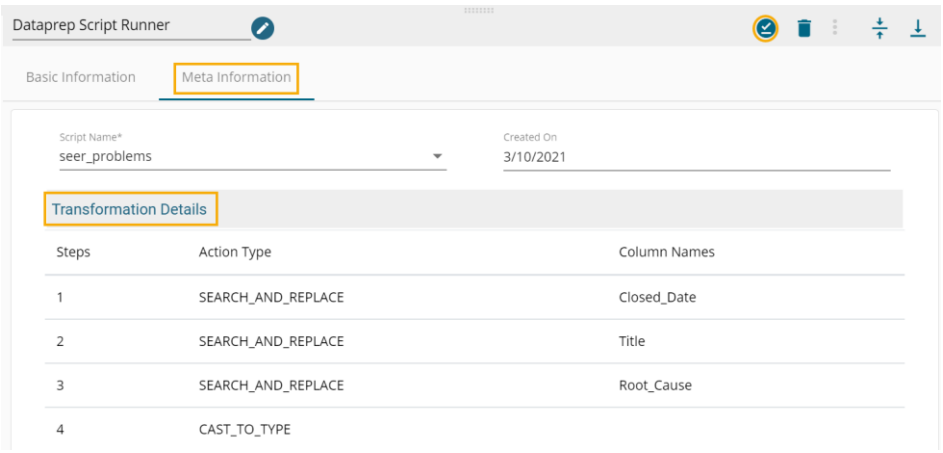
vi) **Basic Information**: It is the default tab to open for the Dataprep Script Runner while configuring the component.

a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



vii) Select the '**Meta Information' tab** and fill all the required inputs fields.
   a. Script Name: Select a script that is exported from the Dataprep Application.
   b. Created On: By selecting a script the date of creation reflects by default.
   c. Transformation Details: The transformation details get displayed below.

viii) Click the '**Save Component in Storage**' icon to save the component properties.



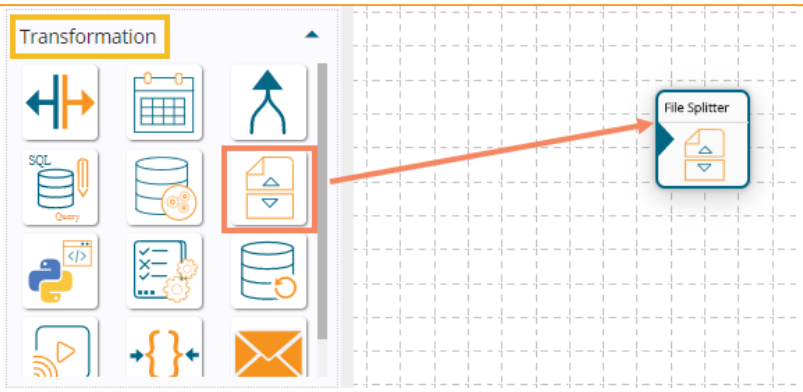
ix) A message appears to notify the successful update of the component.

x) The Dataprep Script Runner component gets configured to read the data coming from the input event, transform the data, and return it to the configured output nodes.

9.3.6. File Splitter

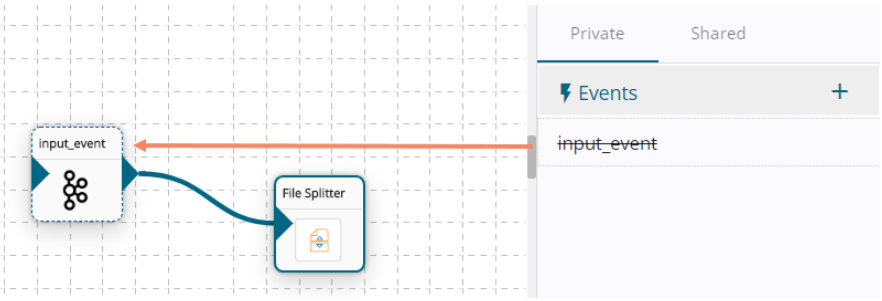
This transform component enables the users to split files.

i) Drag and Drop the File Splitter component to the Workflow Editor.



ii) The Transformation component requires an input event (to get the data). The dragged File Splitter will not have the output connection. The user needs to configure it from the Meta Information tab.

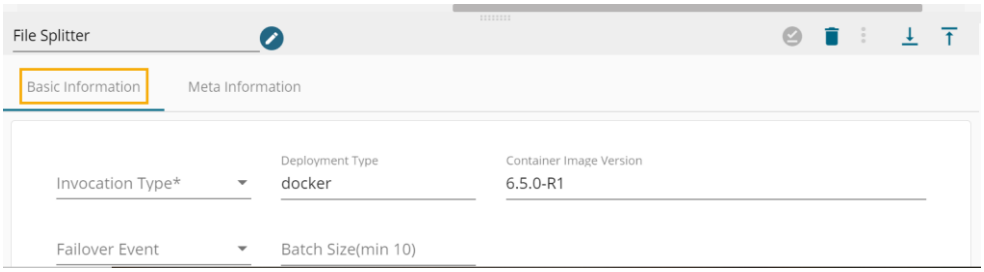
iii) Create one input event, drag it to the Workflow editor, and connect it with the File Splitter component.



iv) Click the File Splitter component to get the component properties tabs.

v) **Basic Information**: It is the default tab to open for the File Splitter component while configuring the component.

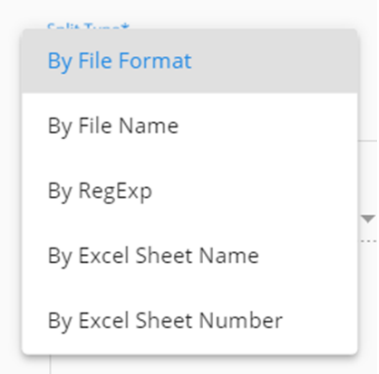
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation type for this component is Real-time.
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



vi) Select the '**Meta Information'** tab and fill all the required properties fields.

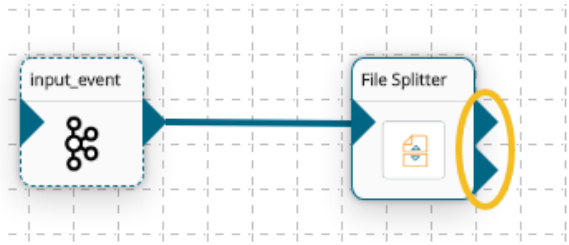
a. Split Type: Select a split type from the drop-down.
The supported Split Types are given below:

1. By File Format
2. By File Name
3. By RegExp
4. By Excel Sheet Name
5. By Excel Sheet Number

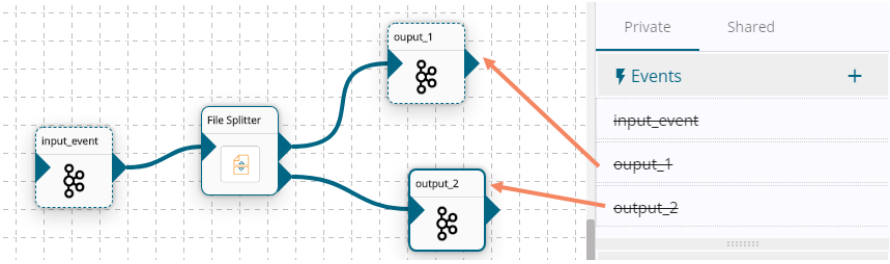


b. No. of Outputs: Select a number from the drop-down menu to create the Output nodes.

vii) By selecting the '**No. of Outputs**' for the File Splitter component the dragged File Splitter component gets those many output nodes.



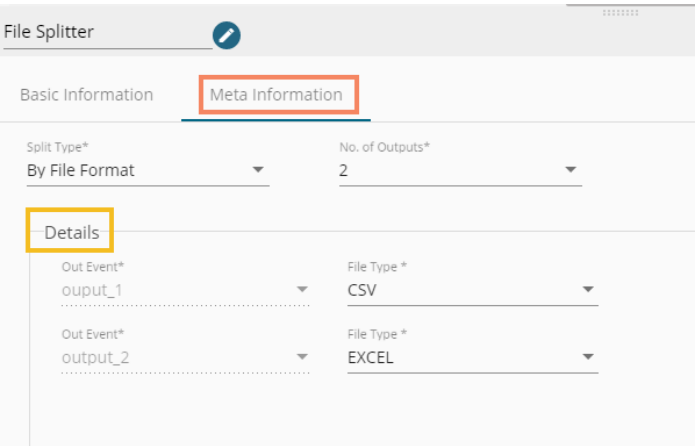
viii) Create and connect the output events to the output nodes of the File Splitter component (E.g., In the present case, 2 output events are created and connected to the File Splitter component).



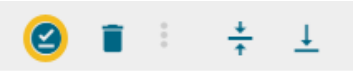
ix) Open the **Meta Information** tab and provide the connection-specific details.

x) The Details field displays both the connected output events.

xi) Select File Type for the given output event as displayed in the following image (the supported file types are PDF, CSV, Excel, Others).



xii) Click the '**Save Component in Storage**' icon to save the component properties.



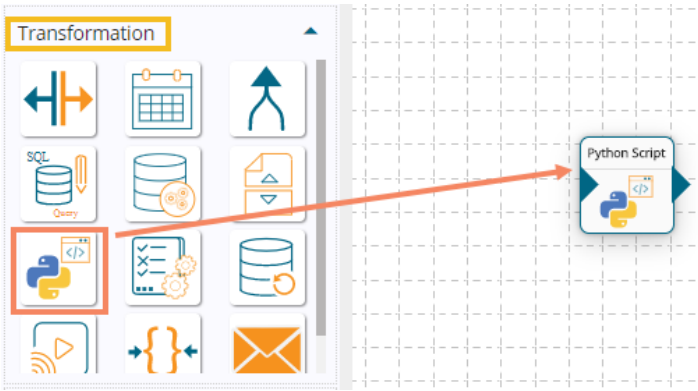
xiii) A message appears to notify the successful update of the component.

xiv) The File Splitter component gets configured to read the data coming from the input event, transform the data, and return it to the configured output nodes.

9.3.7. Python Script (Custom Python Script)

Python Script component works as a normal python compile.

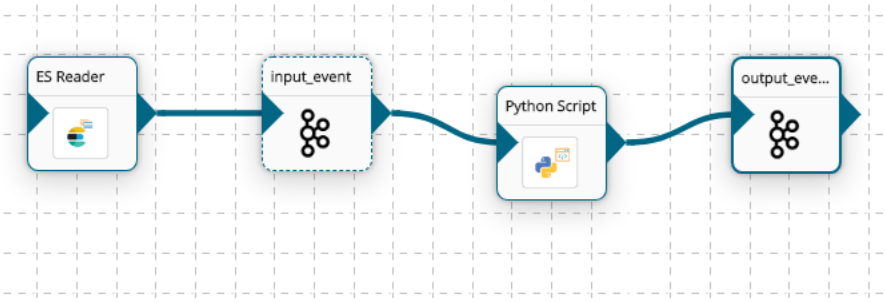
i) Drag and drop the Python Script to the Workflow Editor.



ii) The Python script will read data from an input event and will pass the processed data into an output event, so create and connect two events to the Python Script component.

iii) One reader is needed to pass the data to the input event. (in this case, the ES Reader component is used).

iv) Connect the ES Reader, the Kafka Events, and Python Script components as displayed below:



v) Click the dragged Python Script component to get the component properties tabs:

vi) **Basic Information**: It is the default tab to open for the Python Script component while configuring the component.

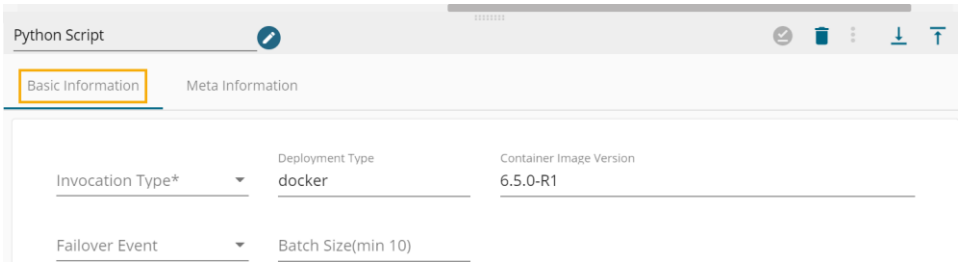
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation type for this component is Real-time.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover Event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



vii) Open the '**Meta Information tab** to open the fields and configure them.

a. Component Name: Provide a name for the Python Script component.
Note: The component name should be without space and special characters. Use the underscore symbol to show space in between words.

b. Python Script: Insert the Python script containing at least one function. The function should not have an argument, data frame argument, or custom argument.

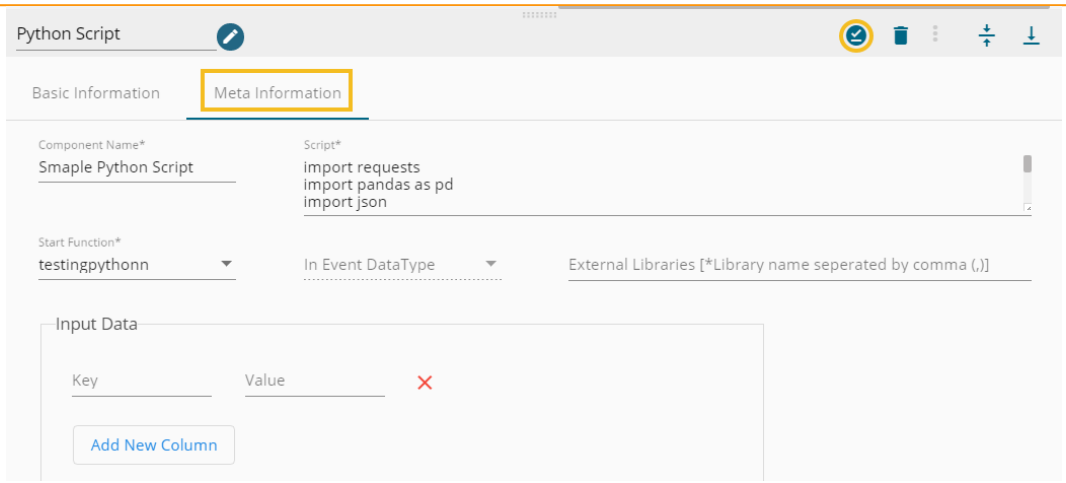
c. Start Function Name: It displays all the function names used in the python script in a drop-down menu. Select one function name with which you want to start.

d. In Event Data Type: Provide input data type as a data frame or list.

e. External Library: Provide the external library name in this field. Insert multiple library names separated by commas.

f. Input Data: Use custom argument names as keys and provide the required value.

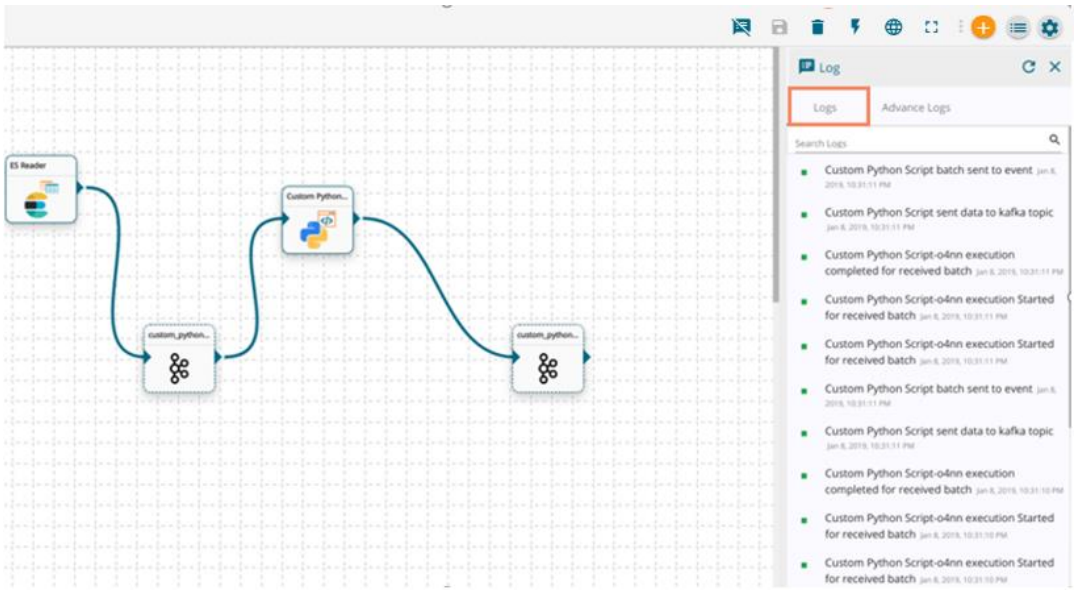
viii) Click the '**Save Component in Storage'** icon.



ix) Click the '**Update Pipeline**' icon to save the Pipeline workflow. (After getting the success message)

x) Activate the Pipeline workflow.

xi) Open the Log section to see the logs.



xii) The Python Script component is ready to read the data coming from the input event, it transforms the data and returns output data.

Note:

a. The below-given instructions should be followed while writing a Python script in the Data Pipeline:

- The Python script needs to be written inside a valid Python function.
  E.g., The entire code body should be inside the proper indentation of the function (Use 4 spaces per indentation level.)
- The Python script should have at least one main function. Multiple functions are acceptable, and one function can call another function.

- It should be written above the calling function body (if the called function is an outer function)
- It should be written above the calling statement (if called function is an inner function)

- Spaces are the preferred indentation method.
- Do not use "type" as the function argument as it is a predefined keyword.
- The code in the core Python distribution should always use UTF-8.
- Single-quoted strings and double-quoted strings are considered the same in Python.
- All the packages used in the function need to import explicitly before writing the function.
- The Python script should return data in the form of a data frame or list only. The form of data should be defined while writing the function.
- If the user uses some Kafka event data for transformation, then the first argument of the function should be a data frame or list.
- If the user needs to use some external library, the user needs to mention the library name in the external libraries field. If the user wants to use multiple external libraries, the library names should be separated by a comma.
- If you need to pass some external input in your main function, then you can use the input data filed. The key name should be the same according to the variable's name and value that is put as per the requirement.
- We can use that component as a reader, transformation, and writer.

b. Python Script Examples

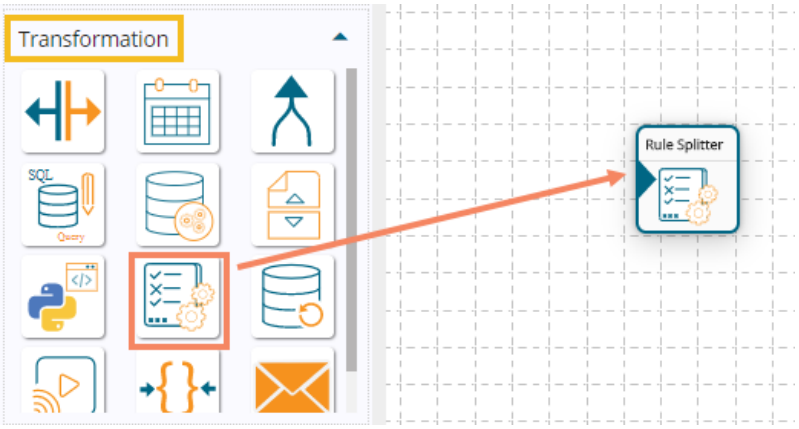
The Custom Python Script transform component supports 3 types of scripts in the Data Pipeline.

1. Without argument (Like Read Data): If you don't have any in Event then you can use no argument function. For Example,
   i. import json
   ii. import requests
   iii. import pandas as pd
   iv. def getmovies_result():
   v. data = requests.get("http://www.omdbapi.com/?s=water&apikey=ba5d53d4")
   vi. loaded_json = json.loads(data.content)
   vii. data = loaded_json['Search']
   viii. df = pd.DataFrame.from_dict(data, orient='columns')
   ix. return df
2. With argument (Like Transformation): If you have data frame to execute some operation. Then use first argument as data frame. For Example,
   i. def getcsvdata(df):
   ii. cond1 = df['Unit Price'] > 450
   iii. filter_df = df[cond1]
   iv. return filter_df
3. Custom Argument with Data frame: If there is custom argument with data frame means Input Event then first argument is always data frame variable. For Example,
   i. def getcsvdata(df, range):
   ii. cond1 = df['Unit Price'] > range
   iii. filter_df = df[cond1]
   iv. return filter_df

9.3.8. Rule Splitter

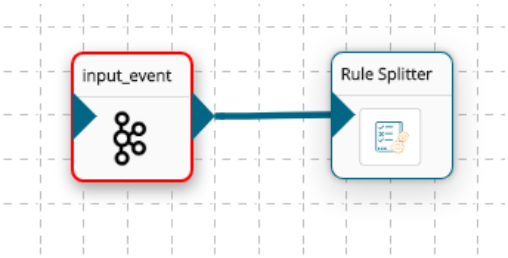
This component in the Pipeline helps the users to apply a filter on the data as per the condition and move to separate events.

i) Drag and drop the Rule Splitter component to the Workflow Editor.



ii) The transformation component requires an input event (to get the data) and sends the data to an output event.

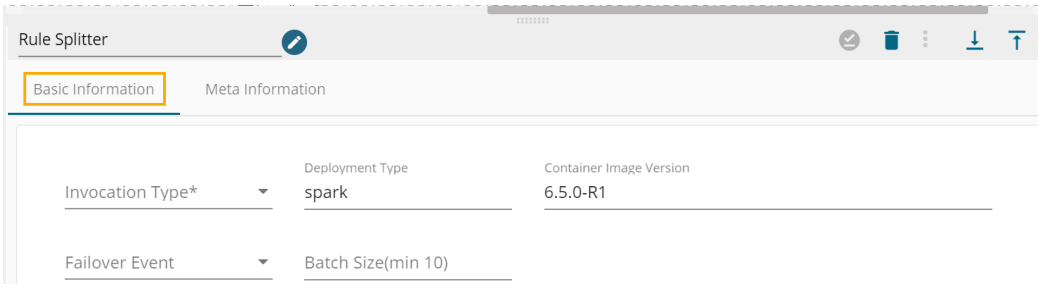
iii) Create an input event and connect it to the Rule Splitter component.



iv) Click the Rule Splitter component to get the tabs containing the component properties.

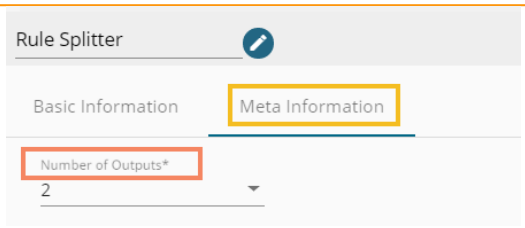
v) The **Basic Information tab** opens by default.

a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



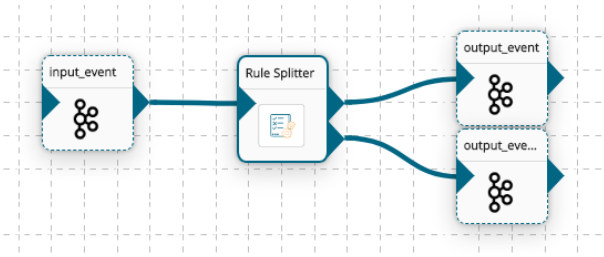
vi) Open the **Meta Information tab** and configure the required information:

vii) Select the Number of Outputs from the drop-down menu.



viii) Based on the No. of Outputs the Rule Splitter component gets the output nodes.

ix) Connect the output events to the Rule Splitter and create a workflow as given below:

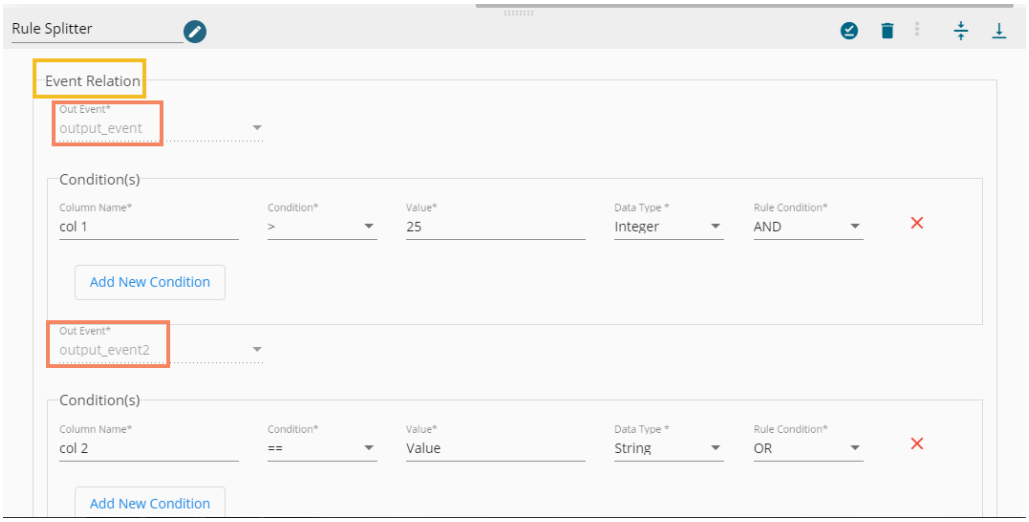


x) Open the '**Meta Information** tab and provide the connection-specific details.

xi) Configure the '**Event Relation**' fields.

i. **Column Name**- Provide a name for the column
ii. **Condition**- Select a condition from the provided options
iii. **Value**- Enter a defining value for the column
iv. **Data Type** (String/Integer/Double/Date)
v. **Rule Condition** (AND/OR)

xii) Click the '**Save Component in Storage**' icon after providing the required information.



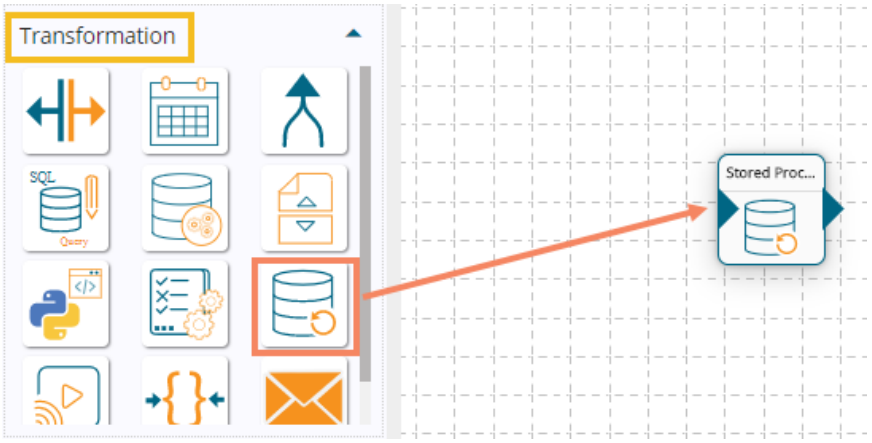
xiii) A message appears to notify the successful update of the component.

xiv) The Rule Splitter component is ready to read the data coming from the input event, transform the data, and return output data (based on the configured output nodes).

9.3.9. Stored Procedure Runner

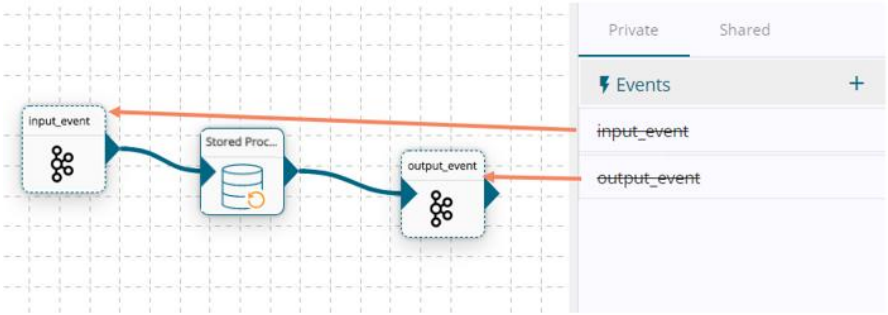
Stored Procedure Runner Component can be used to run a stored procedure using a Pipeline. It is a **Batch** component. As soon as the data comes from the previous components the Stored Procedure Runner gets triggered and performs its task only once for that batch. This Transformation component cannot be used with Stream data because the velocity of data in the case of streaming is very high.

i) Drag and drop the Stored Procedure Runner component to the Workflow Editor.



ii) The transformation component requires an input event (to get the data) and sends the data to an output event.

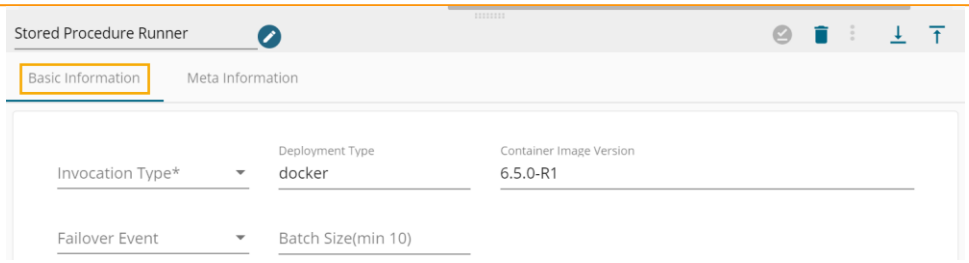
iii) Create one input event and an output event, connect them to the Stored Procedure Runner component.



iv) Click the Stored Procedure Runner component to get the tabs containing the component properties.

v) The **Basic Information tab** opens by default.

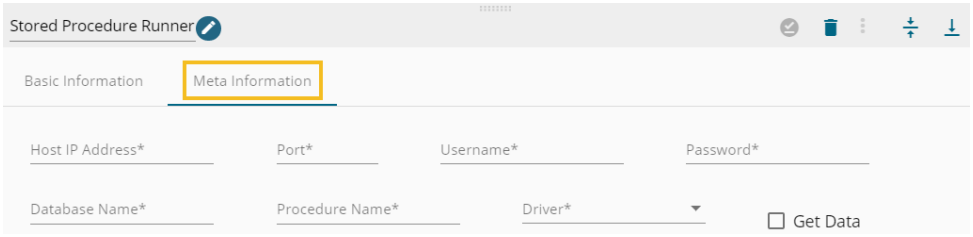
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component (Only the '**Batch**' invocation type is supported).
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



vi) Open the **Meta Information tab** and provide the connection-specific details.

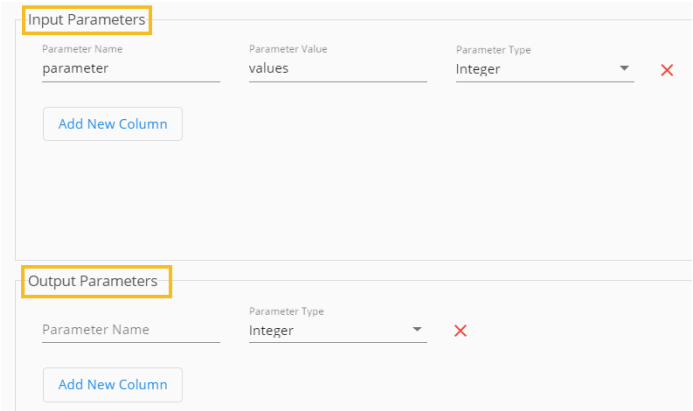
vii) Configure the mandatory fields by providing the required information:

a. Host (JDBC Host)
b. Port
c. Username
d. Password
e. Database Name
f. Procedure Name (Name of the Stored Procedure)
g. Driver: Select a driver using the drop-down menu
h. Get Data: Enable this option using a checkmark to get returned data from the Stored Procedure



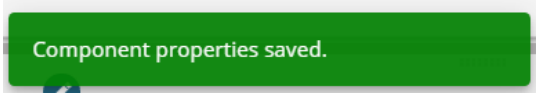
viii) Configure the Stored Procedure specific Input and Output parameters:

a. Input Parameters
   i. Parameter Name
   ii. Parameter Value
   iii. Parameter Type
b. Output Parameters
   i. Parameter Name
   ii. Parameter Type



ix) Click the '**Save Component in Storage**' icon.

x) A message appears to notify the successful update of the component properties.



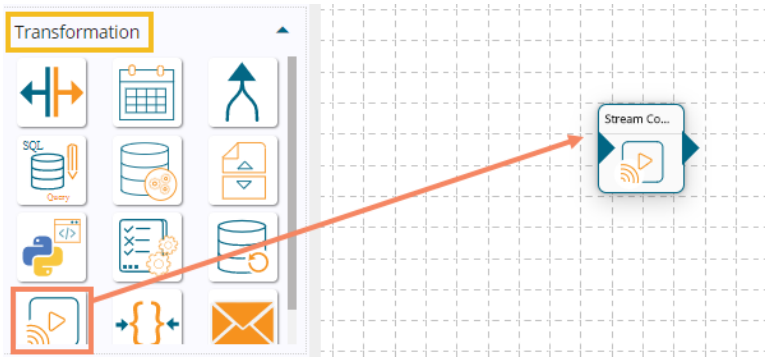
xi) The Stored Procedure Runner component is ready to read the data coming from the input event, transform the data, and return output data to the output events.

Note: Data from an Input Event cannot be taken as Input Parameter for the Stored Procedure Runner component.

9.3.10. Stream Component

The steps to configure the Stream component is described in this section.

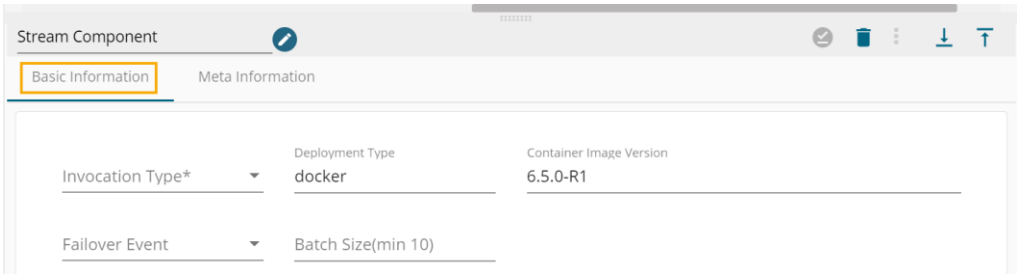
i) Drag and drop the Stream Component to the Pipeline Workflow Editor.



ii) Click the dragged Stream component to get the Configuration fields:

iii) The **Basic Information tab** opens by default.

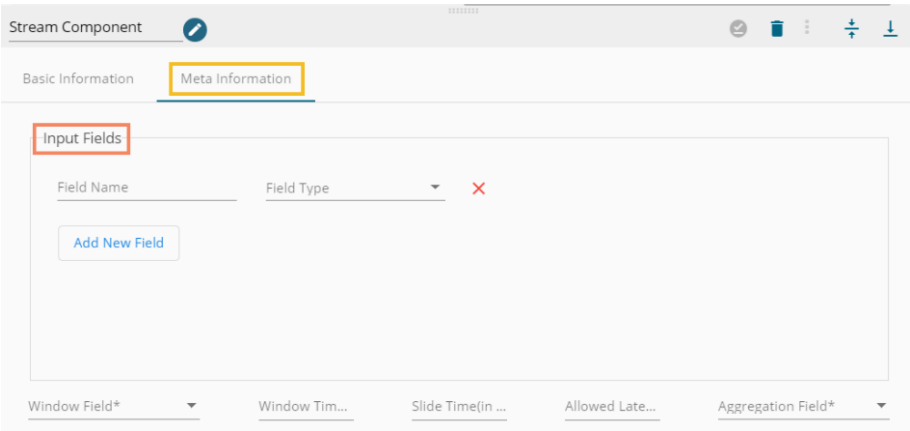
a. Select the invocation type (Real-Time/Batch)
b. Deployment Type: It comes preselected based on the component.
c. Container Image Version: It comes preselected based on the component.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



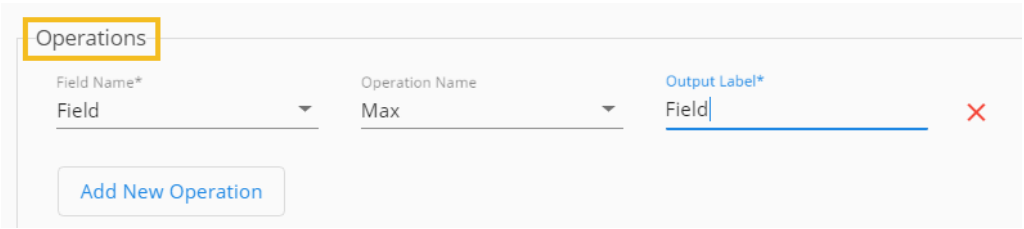
iv) Open the **Meta Information tab** and provide the connection-specific details.

a. Input Fields: The user needs to provide all/selected columns that the IoT device is producing along with their Data Type.
   1. Field Name: Provide a name for the field.
   2. Filed Type: Select a field type from the drop-down menu (The supported field types are String, Long, Integer, Float, Double, Date).

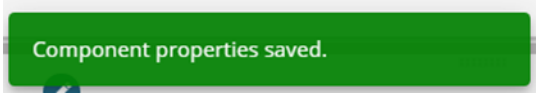
b. Window Field: The user needs to provide the field using which we want to do bucketing of data. This field must be a timestamp value.
c. Window Time: Time duration of Window (Bucket).
d. Allowed Lateness: Allowed time up to which the component accepts late data.
e. Aggregation Field: The user needs to provide a field using which component performs aggregation operations on operation fields.
f. Operations: The user needs to select the field name, operation name, and provide the output field label. Multiple operations are also supported.



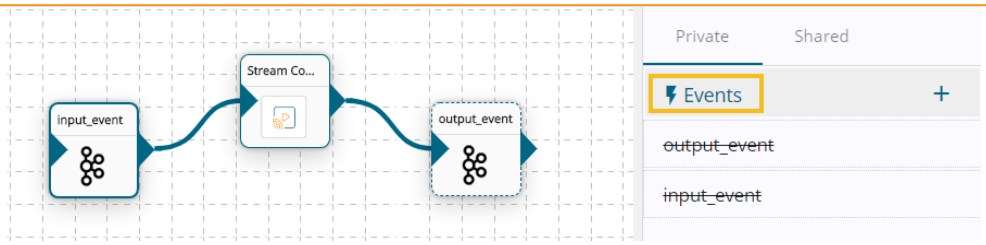
g. Operations: Configure the following details.
   1. Field Name: The configured input fields appear in the drop-down. Select a field from the drop-down menu.
   2. Operation Name: Select an option to select the Operation name.
   3. Output Label: Provide an output label.



v) Click the '**Save Component in Storage**' icon.
vi) A message appears to notify the action.



vii) The Stream Component is ready to be used in the Pipeline workflow. It can read data from an input event and pass the processed streaming data to an output event, so accordingly connect the required event components to create a pipeline workflow.



Note:

a. The user can add a new field and new operation by clicking the '**Add New Field'** and '**Add New Operation**' options, respectively.
b. Windowing: Windowing is an operation in spark structured streaming where the stream of data gets divided into a small amount of windows and then performs aggregation operations on the data.

For detailed information, please visit this link:
https://spark.apache.org/docs/2.3.0/structured-streaming-programming-guide.html#window operations-on-event-time

9.3.11. Flatten JSON

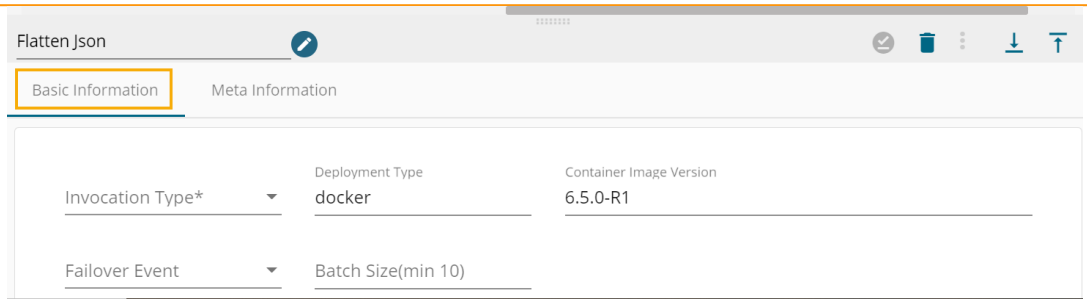
The pipeline provides the Flatten JSON component for flatting a nested JSON as well as filtering data from a nested JSON. It can filter and remove the unnecessary dumping of data on a Kafka topic.
It can run in the following scenarios:

1. Real-Time
2. Batch Process

The steps to configure the Flatten JSON component are described in this section.

i) Drag and drop to the Flatten Json component to the pipeline Workflow Editor.

ii) Click the dragged Flatten JSON component to get the Configuration fields:
iii) The **Basic Information** tab opens by default.
   a. Select the invocation type (Real-Time/Batch)
   b. Deployment Type: It comes preselected based on the component.
   c. Container Image Version: It comes preselected based on the component.
   d. Failover Event: Select a failover Event from the drop-down menu.
   e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the **Meta Information** tab to provide the connection-specific details.

a. Name: The user must provide the full path of data by using dot(.) which he wants to retrieve from a nested JSON.

For instance - We have a nested JSON of the following format, and we want to retrieve the device type of the Device in dashboard_1 we must give the name **dashboard_1. Device.devicetype.**


```
"dashboard_1": {
    "Device": [
      {
        "devicetype": "Motor-N1",
        "Vibration": 53,
        "Temperature": 23
      },
      {
        "devicetype": "Motor-N2",
        "Vibration": 48,
        "Temperature": 40
      },
      {
        "devicetype": "Motor-S1",
        "Vibration": 98,
        "Temperature": 33
      },
      {
        "devicetype": "Motor-S2",
        "Vibration": 77,
        "Temperature": 37
      },
      {
        "devicetype": "Gearbox-LS",
        "Vibration": 56,
        "Temperature": 43
      },
      {
        "devicetype": "Gearbox-SS",
        "Vibration": 97,
        "Temperature": 50
      }
    ],
    "Brake": [
      {
        "devicetype": "Brake-N",
        "Thickness": 19,
        "Temperature": 414
      },
      {
        "devicetype": "Brake-S",
        "Thickness": 11,
        "Temperature": 397
      }
    ],
    "RPM": 675,
    "LOAD(T)": 40,
    "MTBF": 562,
    "MTTR": 62,
    "RUN": 3609,
    "RTBM": 317
  },
  "dashboard_2": {
    "Cast_Spd_St1": 0.013,
    "Cast_Spd_St2": -0.01,
```


v) Selected **Columns**: The users can select some specific columns from the table to read data instead of selecting a complete table; this can be achieved via the 'Selected Columns' section.

Select the columns which you want to read and if you want to change the name of the column, then put that name in the alias name section otherwise **keep the alias name the same as of column name** and then select a Column Type from the drop-down menu.

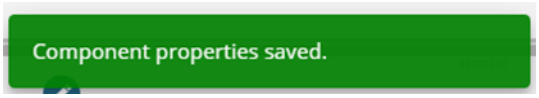
or

Use '**Download Data**' and '**Upload File**' options to select the desired columns.

1. Upload File: The user can upload the existing system files (CSV, JSON) using the '**Upload File**' icon (file size must be less than 2 MB).
2. Download Data (Schema): Users can download the schema structure in JSON format by using the '**Download Data**' icon.

vi) Click the '**Save Component in Storage**' icon to save the configuration.

vii) A message appears to notify the action.

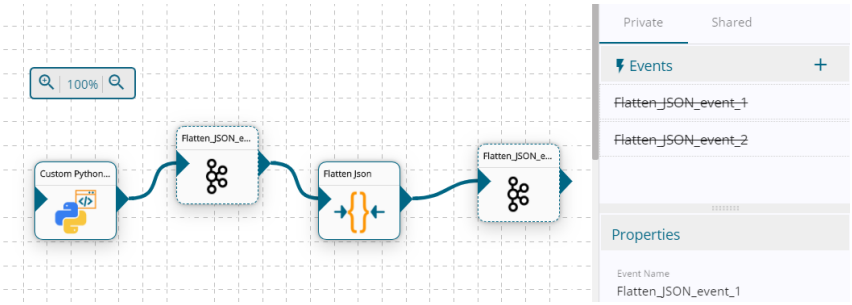


viii) The Flatten JSON is ready to be used in the Pipeline workflow.

Note:

a. The Flatten JSON component cannot generate data in rows.

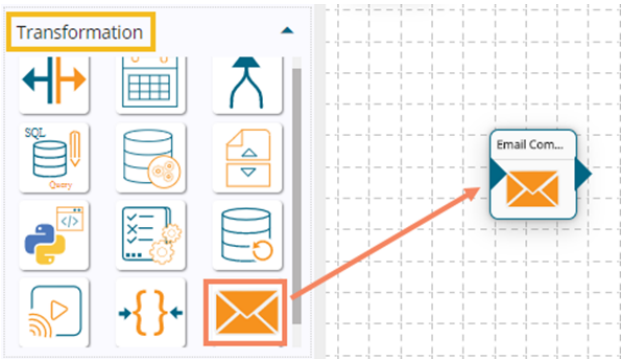
b. The Flatten JSON component must have an in-event and out-event in the workflow, as displayed below:



9.3.12. Email Component

The Email Component helps to configure the Email content together with the Email server and recipients.

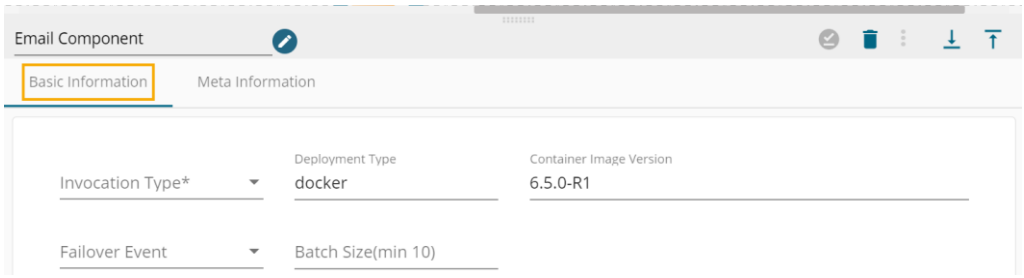
i) Drag and drop the Email Component to the Pipeline Workflow Editor.



ii) Click the dragged Email component to get the Configuration fields:

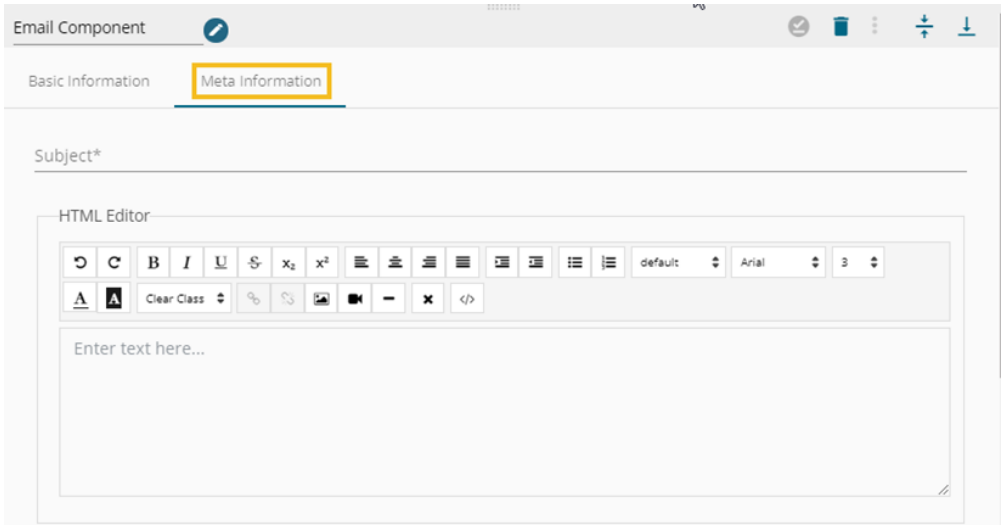
iii) The **Basic Information tab** opens by default.

a. Select the invocation type (Real-Time/Batch)
b. Deployment Type: It comes preselected based on the component.
c. Container Image Version: It comes preselected based on the component.
d. Failover Event: Select a failover Event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

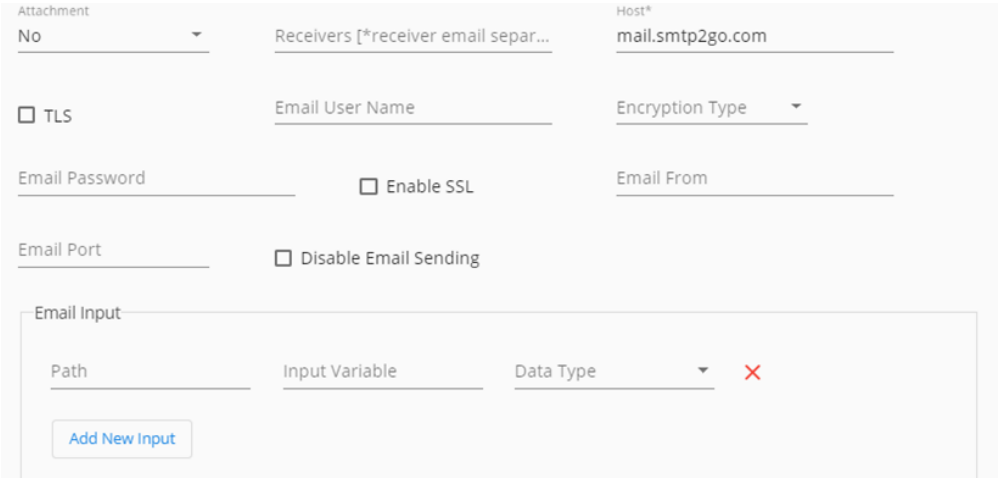


iv) Open the **Meta Information tab** and provide the connection-specific details.

a. Subject (*): Subject of the Email
b. Text Message: Provide the email content in the given space.

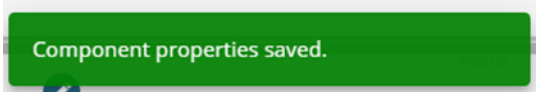


c. Attachment: Select an option out of 'Yes' or 'No'.
d. Receivers (*): Provide receivers' email ids separated by commas.
e. Host (*): Provide host details.
f. TLS: Enable or disable TLS by putting a checkmark in the given box.
g. Email Username: Provide the name of the email user.
h. Encryption Type: Select an encryption type from the drop-down menu.
i. Email Password: Provide the password of the email
j. Enable SSL: Enable SSL by putting a checkmark in the given box.
k. Email From: Provide the email address of the senders.
l. Email Port: Provide the Port number of the email.
m. Disable Email Sending: Put a tick mark to disable the sending of the notification emails.
n. Email Input: Provide the following details.
    1) Path
    2) Input Variable
    3) Data Type



v) Click the '**Save Component in Storage**' icon.

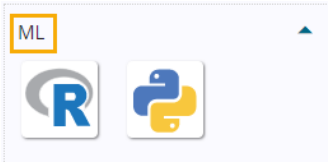
vi) A message appears to notify the action.



vii) The Email Component is ready to be used in the Pipeline workflow. It can read data from an input event and pass the processed data to an output event, so accordingly connect the required event components to create a pipeline workflow.

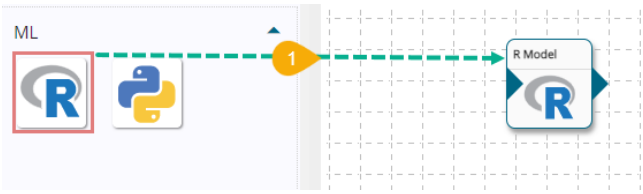
9.4. ML

ML Model Runner Components allow us to use the models created on R and Python Workspaces of the Data Science Workbench inside the Data Pipeline.



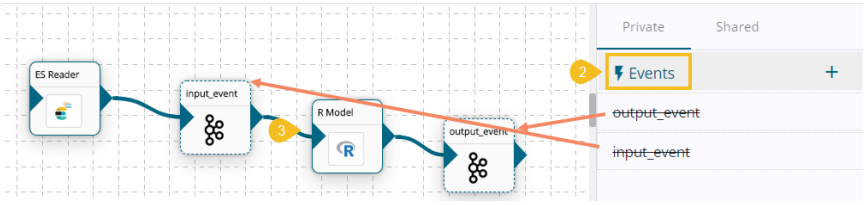
9.4.1. R Model Runner

i) Drag and Drop R Model (runner) component to the Workflow Editor.

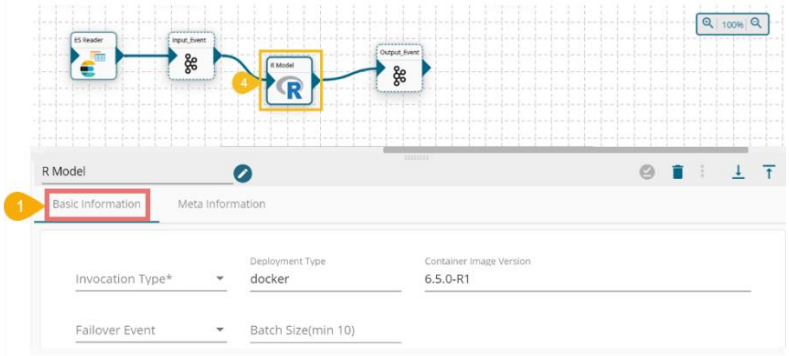


ii) Model Runner requires data input from an Event and sends the processed data to another Event, so create two Events.

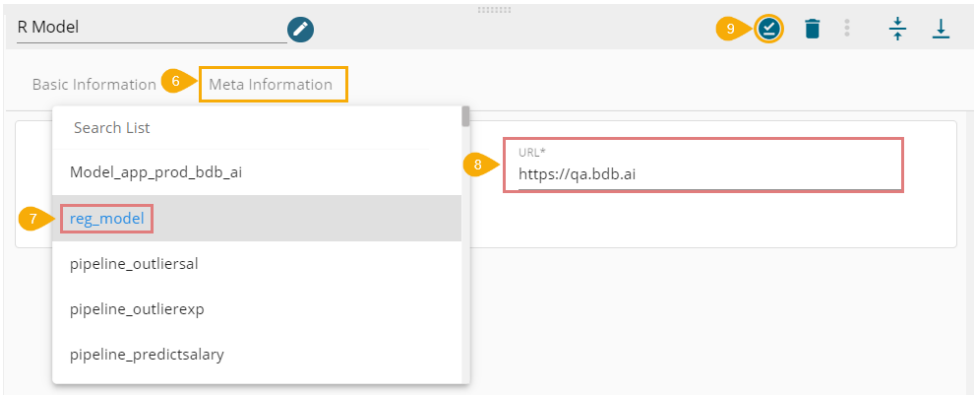
iii) Drag them to the workflow editor and connect to the dragged R Model (runner) component as displayed below:
Note: The data in Input Event can come from any Ingestion, Reader, or shared events.



iv) Click on the R Model Component to get the Component Properties tabs.
v) **Basic Information**: It is the default tab to open for the component.
   a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.
   b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   d. Failover Event: Select a failover Event from the drop-down menu.
   e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



vi) Click the **Meta Information tab** to open.
vii) All the exported R models appear under the '**Model Name**' menu. Search and select the R model you want to use.
viii) Configure the URL link (The URL link comes predefined and can be changed by the user).
ix) Click the '**Save Component in Storage**' icon.

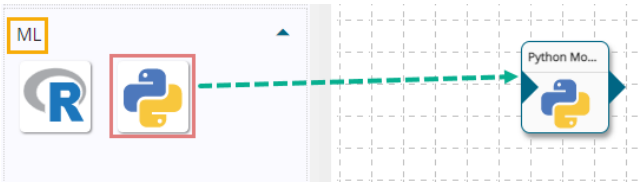


x) A message appears to confirm the action.

xi) The R Model runner is configured. The component reads the data coming to an input event, runs the model, and gives the output data with processed columns to the output event.

9.4.2. Python Model Runner

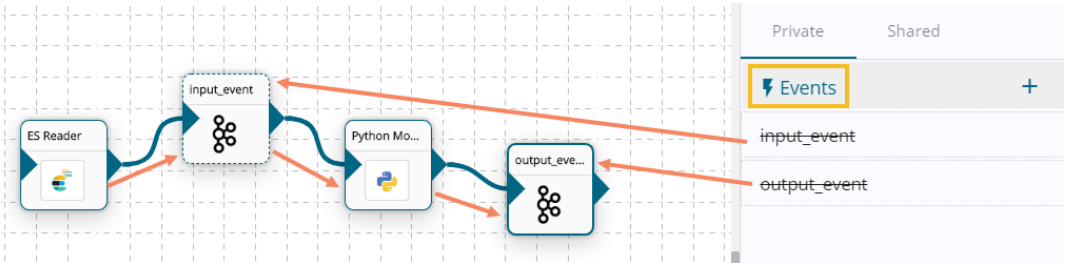
i) Drag and drop the Python Model (runner) component to the Workflow Editor.



ii) The Python Model runner requires input data from an Event and sends the processed data to another Event. So, create two Events and drag them onto the Workspace.

iii) Connect the input and output events with the Python Model runner component as displayed below.

iv) The data in the input event can come from any Ingestion, Readers, or shared events.



v) Click the Python Model runner component to get the component properties tabs below.

vi) **Basic Information**: It is the default tab to open for the component.

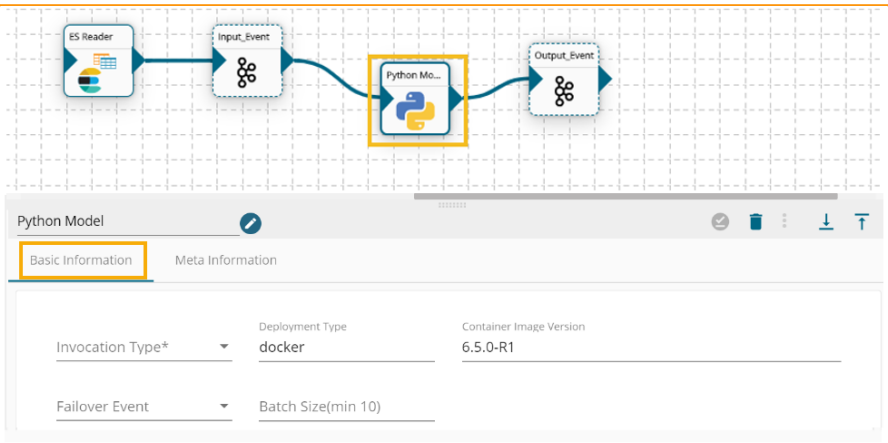
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select '**Real-Time**' or '**Batch**' from the drop-down menu.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

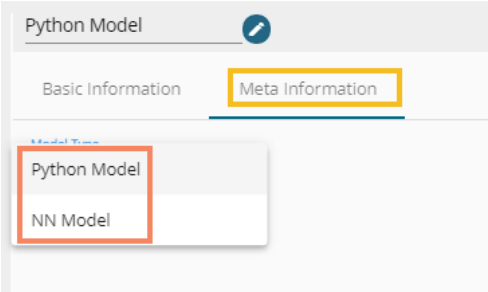
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover Event from the drop-down menu.

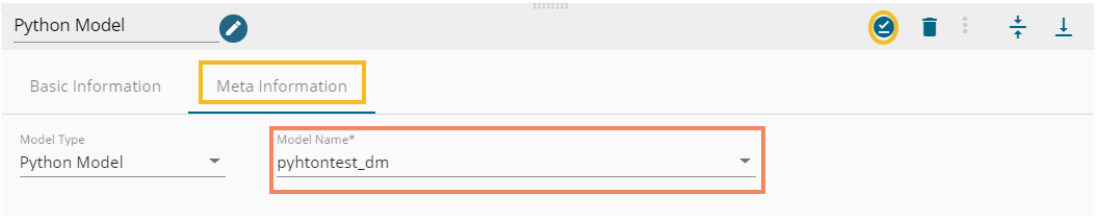
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



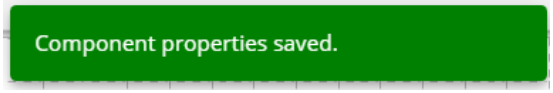
vii) Open the **Meta Information tab** and click on the Model name field.
viii) The model type displays the Python Model and NN Model options.
ix) Select a model Type option.



x) Search the model you want to use and select the model.
xi) Click the '**Save Component in Storage**' icon.



xii) A notification message appears.



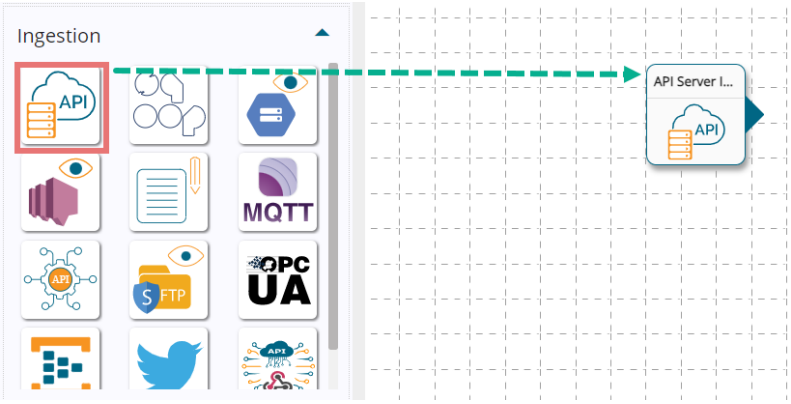
xiii) The Python Model runner component reads the data coming to the input event, runs the model, and gives the output data with predicted columns to the output event.

9.5. Ingestion

Ingestion components allow the users to ingest data in the pipeline from outside the pipeline as in files from some SFTP location or manual ingestion of data using a service in real-time.

9.5.1. API Server Ingestion

i) Drag and Drop the **API Server Ingestion** component to the Workflow Editor.



ii) Click on the dragged ingestion component to get the component properties tabs.

iii) Configure the **Basic Information** tab.

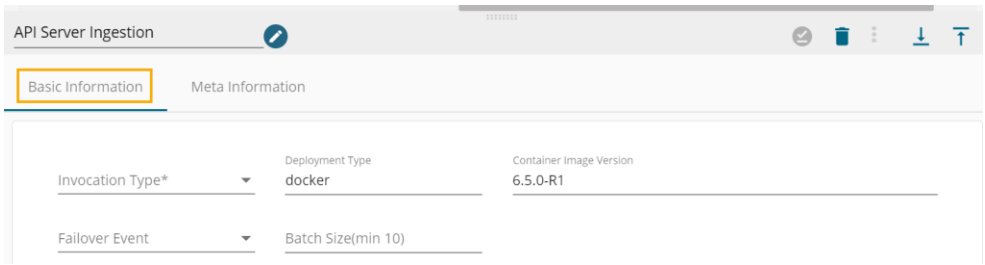
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation type is '**Real-Time**'.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the '**Meta Information**' **tab.**

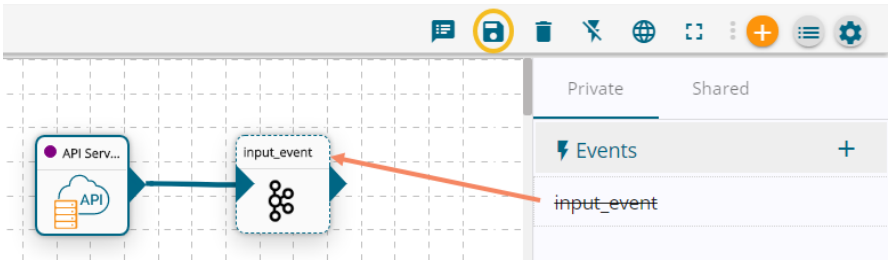
a. Provide the Ingestion Id

b. Provide the Ingestion Secret (Key)

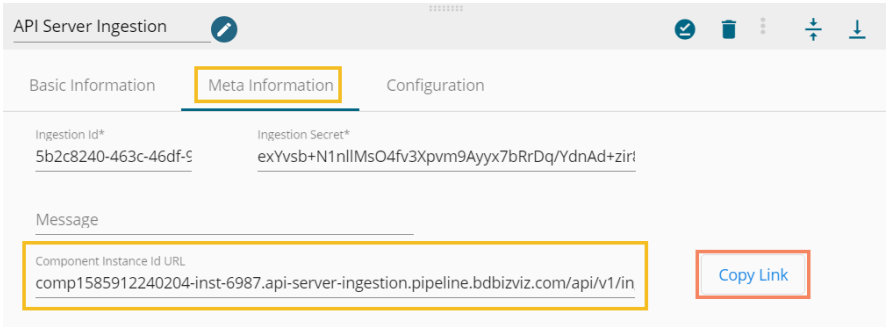
c. Write a Message (optional)
d. The Component Instance Id URL generates (after providing the Ingestion Id and Secret Key).

v) Click the '**Save Component in Storage**' icon to save the API Server Ingestion.

vi) A notification message appears to inform that the component has been saved.
vii) Connect the API Ingestion component to an Event and save the Pipeline.



viii) As we save the pipeline, an auto-generated Component **Ingestion URL** appears inside the Meta Information tab to ingest the data.

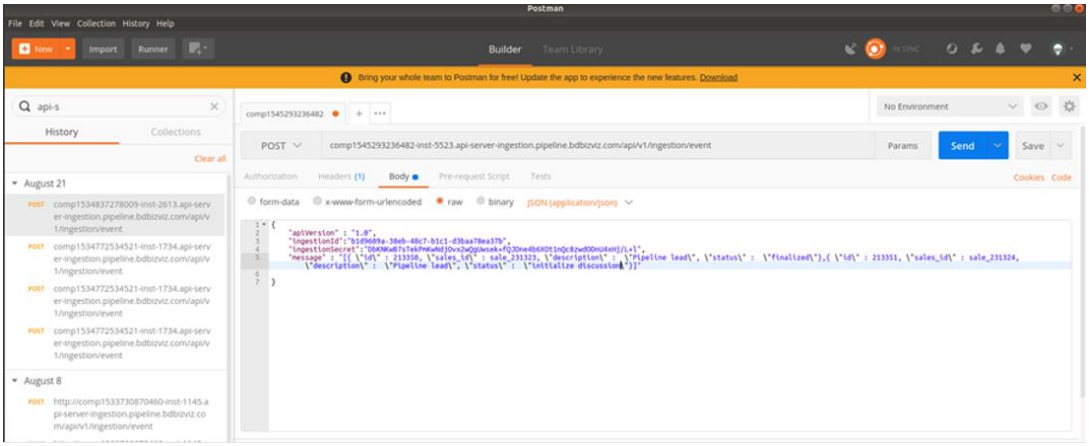


ix) Click the '**Save Component in Storage**' icon.
x) A notification message appears to inform about the action.

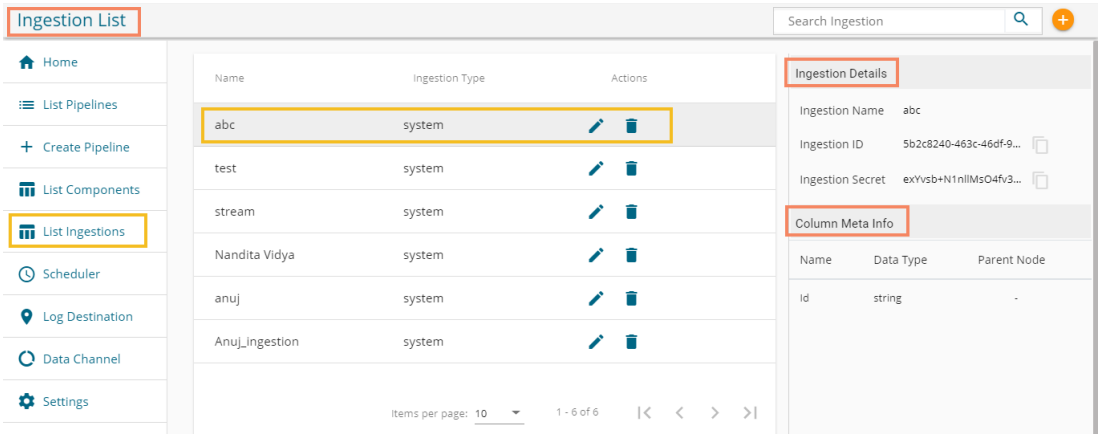
xi) The Ingestion component can be used in a pipeline workflow as an input component after component configuration.

Note:

a. The users can use the Component Ingestion URL with the following format in the program or add it anywhere in the third-party portal.



b. Click the List Ingestions option to open the list of ingestions. Select an Ingestion from the list. The Ingestion Details opens on the top-right side of the page with Ingestion ID and Ingestion Secret (Key).



Service Info:

- IngestionID and Ingestion Secret (key) must be the same as filled in a pipeline component.
- A message accepting data in Array as String format appears.
- In the message, all the objects must follow the same column info as declared in the settings page at the time ingestion gets created.

URL → comp1545293236482-inst-5523.api-server-ingestion.pipeline.bdbizviz.com/api/v1/ingestion/event (will be Auto-generated Component Ingestion URL)
Type → POST
Content-Type → application/json
Body →

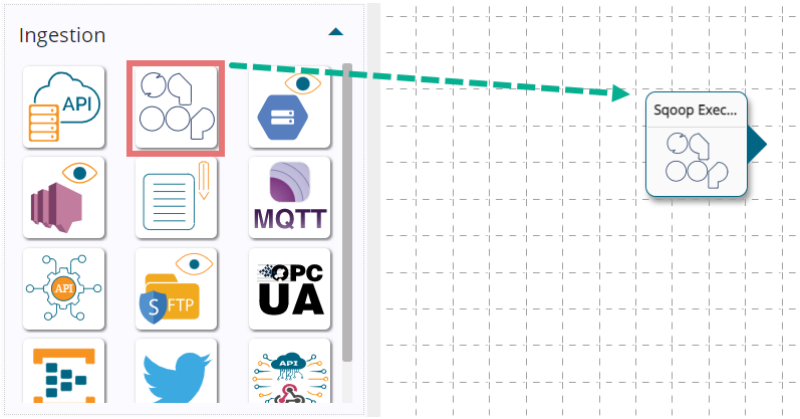
```
{
 "apiVersion": "1.0",
 "ingestionId": "b1d9609a-38eb-48c7-b1c1-d3baa78ea37b",
 "ingestionSecret":
"DbKNKwB7sTekPmKwNdjOvx2wQgUwsek+fQJDne4b6XOt1nQc8zwdODnU4xHj/L+l",
```

"message": "[{\"id\" : 213350, \"sales_id\" : sale_231323, \"description\" : \"Pipeline lead\", \"status\" : \"finalized\"},{\"id\" : 213351, \"sales_id\" : sale_231324, \"description\" : \"Pipeline lead\", \"status\" : \"initialize discussion\"}]"
}

9.5.2. Sqoop Executer

The Sqoop component in BDB Pipeline helps the users to transfer tables and databases from RDBMS (Supported by Sqoop) to HDFS (Hadoop System Distributed File).

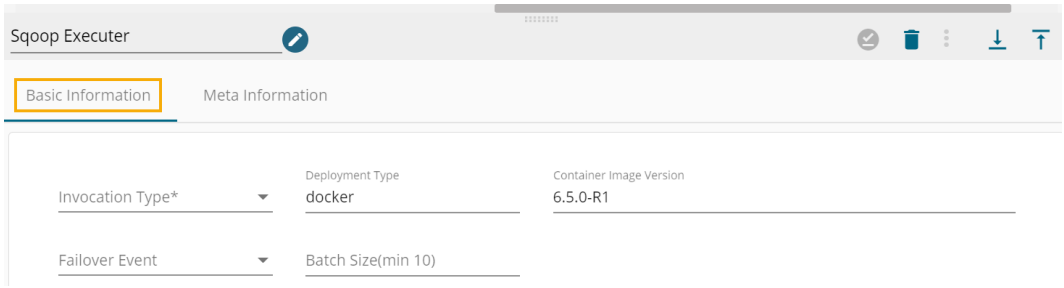
i) Drag and drop the Sqoop component to the pipeline Workflow Editor.



ii) Click the Sqoop Executer to see the component properties tabs below.

iii) The **Basic Information tab** opens by default by clicking the Sqoop Executer component.

a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation type is '**Real-Time**'.
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



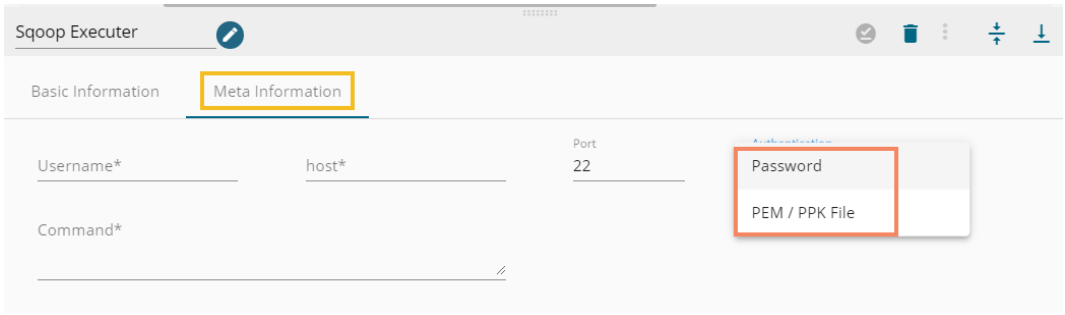
iv) Open the **Meta Information tab** and provide the following information:

a. Username: Provide the username.
b. Host – Provide a host or IP address.
c. Port- Provide a Port number (the default number for these fields is 22).
d. Authentication- Select an authentication type from the drop-down menu out of the given choices

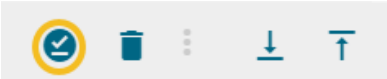
i. Password- Provide the correct password for this authentication option

ii. PEM/PPK File-choose a file and provide the file name if the user selects this authentication option.

e. Command-Enter the relevant Sqoop command.

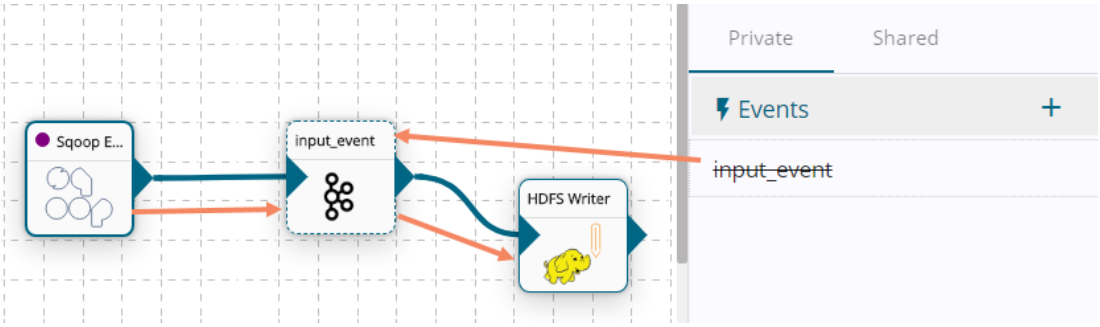


v) Click the '**Save Component in Storage**' icon to save the component.



vi) A notification message appears to inform about the action.

vii) Create one Event and connect it to the Sqoop component for receiving the success message. Connect an HDFS writer to the input event to create a workflow. Once the data gets transferred to the HDFS, the user can use it for further processing in the pipeline.



9.5.3. GCS Monitor

9.5.3.1. Creating GCS (Google Cloud Storage) Credential

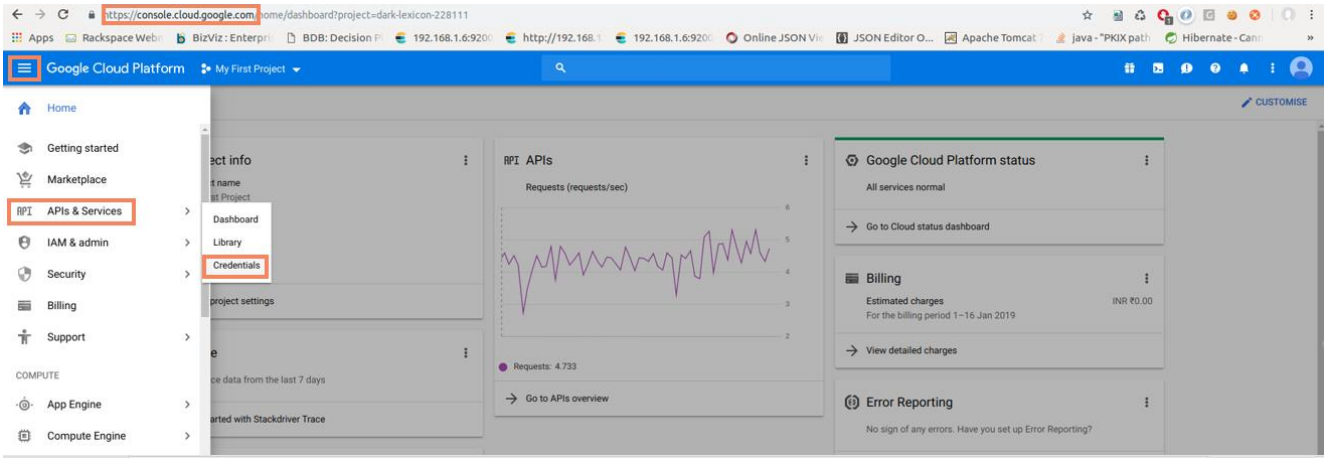
The users require Google credentials to access the GCS bucket via the GCS monitor. The steps to create a service account key are described below:

i) Open **Google Cloud Console** or open the link- https://console.cloud.google.com/home

ii) Click on the menu button (from the top left side) to open the list of options

iii) Select and click the **APIs & Services** option from the menu list

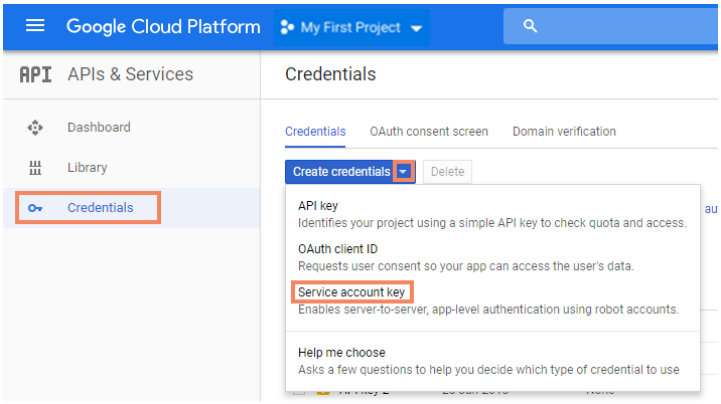
iv) A context menu opens as displayed in the image below
v) Select the **Credentials** option from the context menu



Note: The user can ignore the above-given steps to create credentials if a Service Account key is already created.

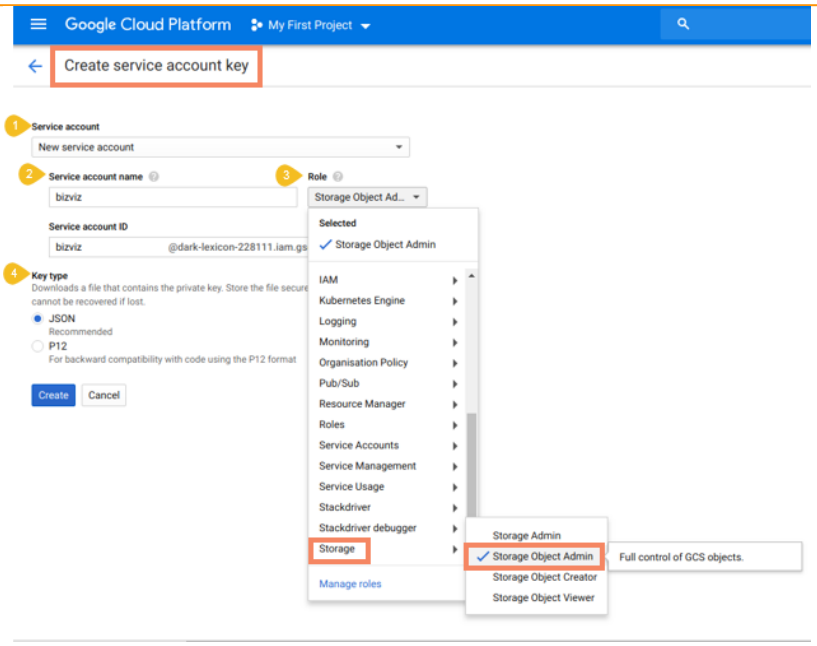
vi) The '**Credentials**' opens as displayed in the below image.
vii) Click the '**Create Credentials'** drop-down option.
viii) **A menu opens with 3 types of authorization options:**
1. API Key
2. OAuth Client ID
3. Service account key

ix) Click the '**Service Account Key'** authorization option.



x) The '**Create Service Account Key**' page opens.
xi) The users need to provide the following parameters:
1. Services account: Select 'New Service Account' from the drop-down menu.
2. Service account name: Provide a user-defined name for the Service Account. (E.g., 'bizviz' in the image below)
3. Role: Select the '**Storage'** option from the drop-down menu.
   a. A new context menu opens.
   b. Select **'Storage Object Admin'** from the context menu.
4. Key type: Select JSON by choosing the checkmark provided next to the option.

xii) The Service Account Key gets created.



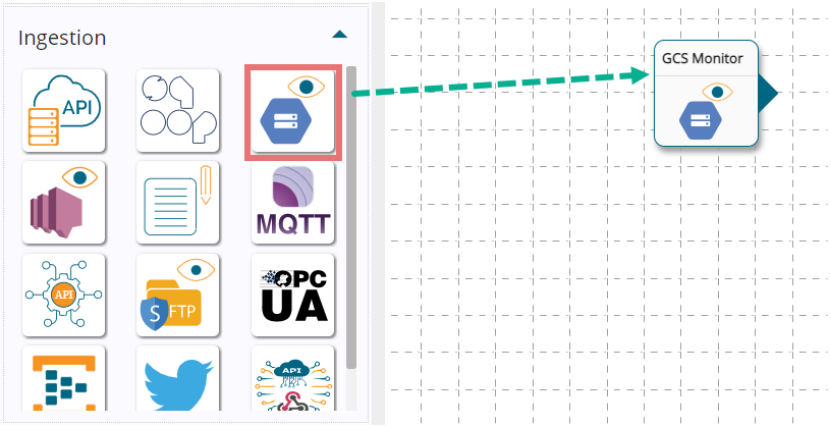
Note: After the successful creation of the Service Account, the users should get the Credential JSON file name according to their project name it indicates that the Service Account Key file got successfully created.

9.5.3.2. GCS Monitor in Data Pipeline

GCS Monitor monitors one folder all the time. If any file has been uploaded, the GCS Monitor reads the file name and sends it to the event. The GCS Monitor also copies that file to the copied (defined) location and deletes it from the monitoring folder. It repeats the same process for all the files.

Steps to create one pipeline using the GCS Monitor component.

i) Drag and drop the GCS Monitor component to the Workflow Editor.



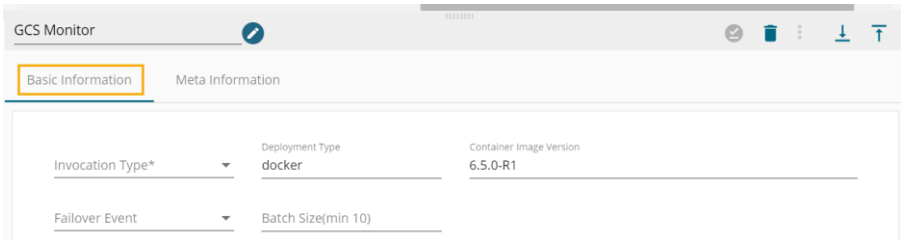
ii) Click the dragged GCS Monitor component to access the component properties.

iii) The **Basic Information tab** opens by default.

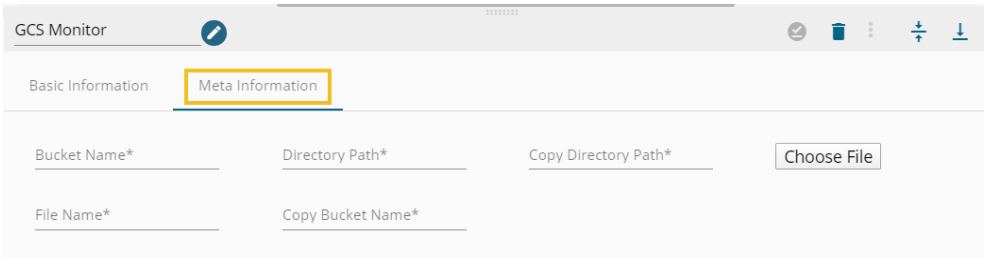
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation type is '**Real-Time**'.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

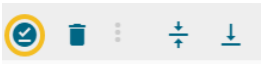
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the **Meta Information tab** and configure it.
1. Bucket Name: Enter the source bucket name
2. Directory Path: Fill monitor folder path using forward-slash (/). E.g., monitor**/**
3. Copy Directory Path: Fill in the copy folder name where you want to copy the uploaded file. E.g., **monitor_copy/**
4. Choose file: Upload a **Service Account Key(s)** file.
5. File Name: After the Service Account Key file is uploaded, the file name gets auto-generated based on the uploaded file.
6. Copy Bucket Name: Fill in the destination bucket name where you need to copy.

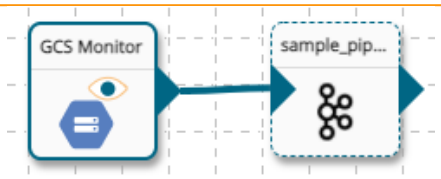


v) Click the '**Save Component in Storage'** icon to save all the Meta Information for the GCS Monitor component (A message appears to notify completion of the action).



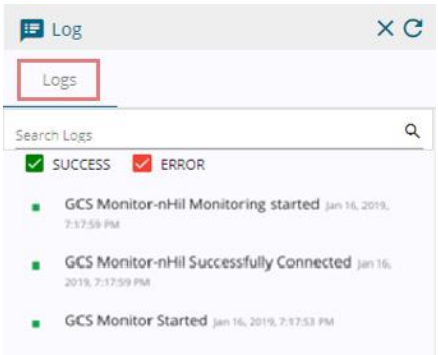
vi) The GCS Monitor component gets configured, and a notification message appears to confirm the action.

vii) The users need to create an event and connect it with the GCS Monitor component.
(Monitor sends filename to the event when any file gets uploaded on monitor folder.)



viii) Click the Update Pipeline icon (A message appears to notify completion of the action).

ix) Open the Log toggle panel to see the process of the GCS Monitor.



x) The GCS Monitor component can be used in a pipeline workflow.

Note: The GCS monitor can work with GCS Reader. BDB Pipeline supports CSV and JSON files at present.

9.5.4. AWS SNS Monitor

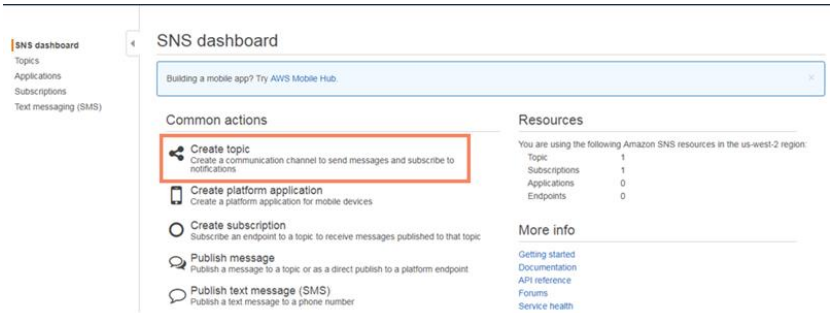
The user needs to create an SNS topic as a precondition for SNS Monitor in BDB Data Pipeline.

9.5.4.1. Steps to Configure SNS and SQS for SNS Monitor

i) Log in to an AWS account.

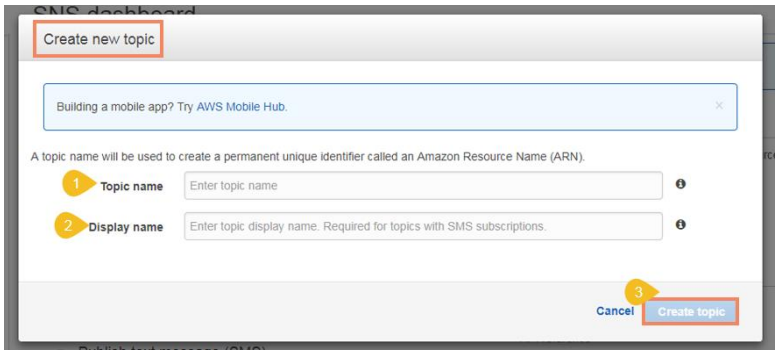
ii) Navigate to the SNS console using the below-given link:
https://us-west-2.console.aws.amazon.com/sns/v2/home?region=us-west-2#/home

iii) Select the '**Create topic**' option.

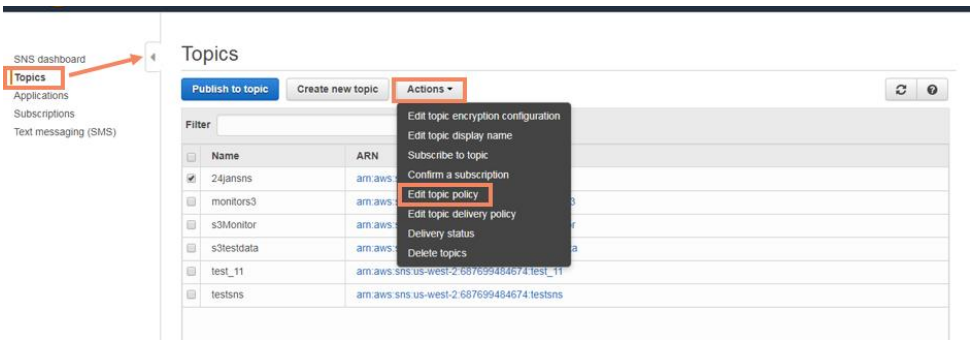


iv) The '**Create new topic**' window opens.

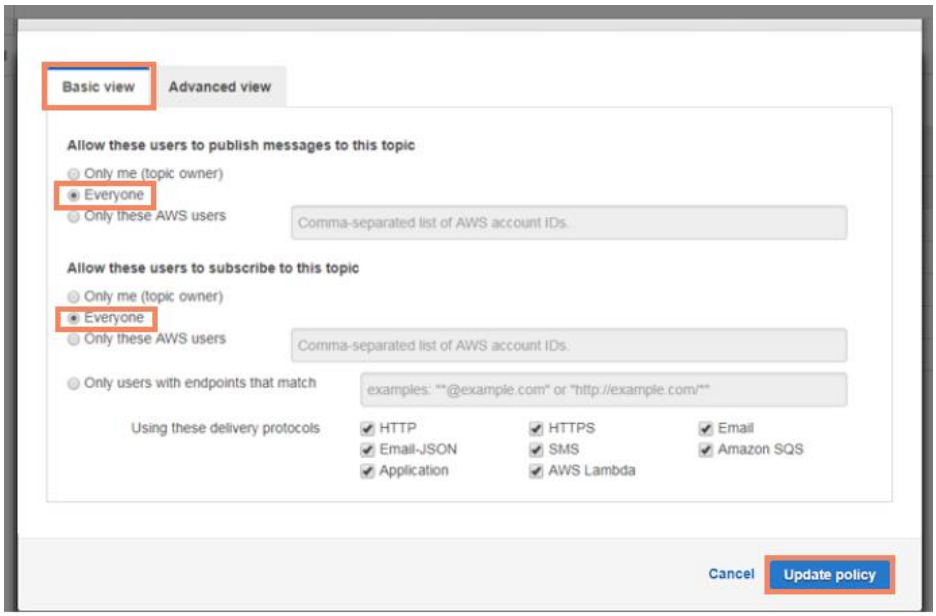
a. Provide the following information:
   i. Topic Name: Enter a name for the topic
   ii. Display Name: Enter the topic name (Required for topics with SMS subscription)
   iii. Click the '**Create topic**' option.



v) Assign permission after the topic gets created.
vi) Navigate to the '**Topics**' page.
vii) Click the '**Actions**' drop-down menu.
viii) Select the '**Edit topic policy**' option from the menu.



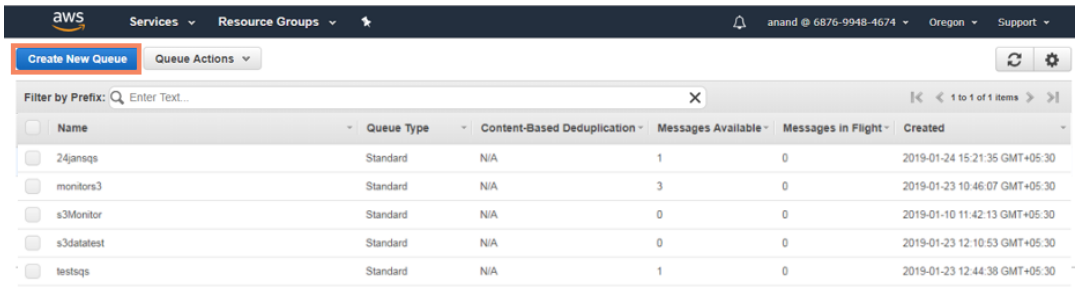
ix) The '**Basic view**' tab opens.
x) Update the policy from '**Only me**' to '**Everyone**' using the radio button.
xi) Click the '**Update policy**' option.



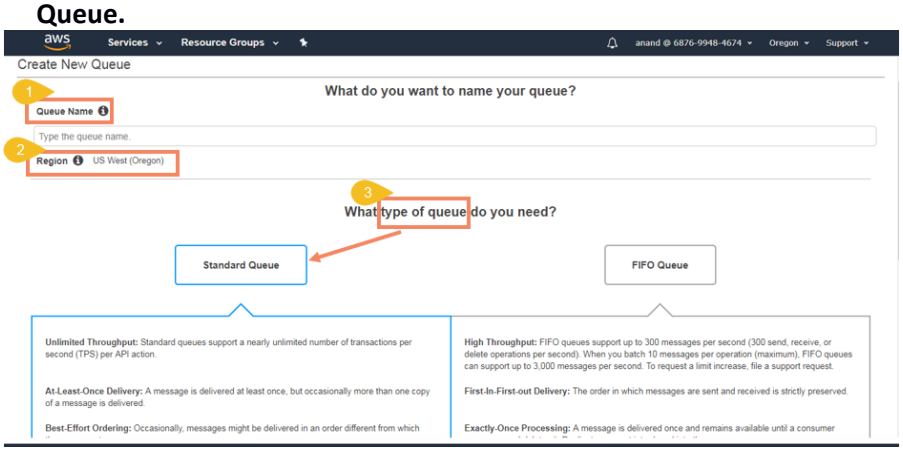
9.5.4.2. Create the SQS Queue

The steps to create an SQS Queue are described below:

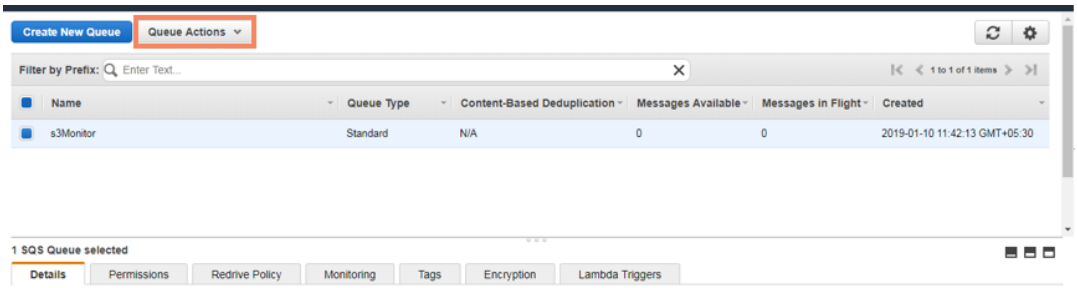
i) Log in to an AWS account.
ii) Navigate to the SQS console. https://console.aws.amazon.com/sqs/home?region=us-west-2#queue-browser:selected=https://sqs.us-west-2.amazonaws.com/687699484674/s3Monitor;prefix=
iii) Click the '**Create New Queue**' option.



iv) The '**Create New Queue**' page opens.
   a. Provide a '**Queue Name**' in the given space.
   b. Provide the '**Region**'.
   c. Select the type of queue from the provided options: **Standard Queue** or **FIFO Queue.**



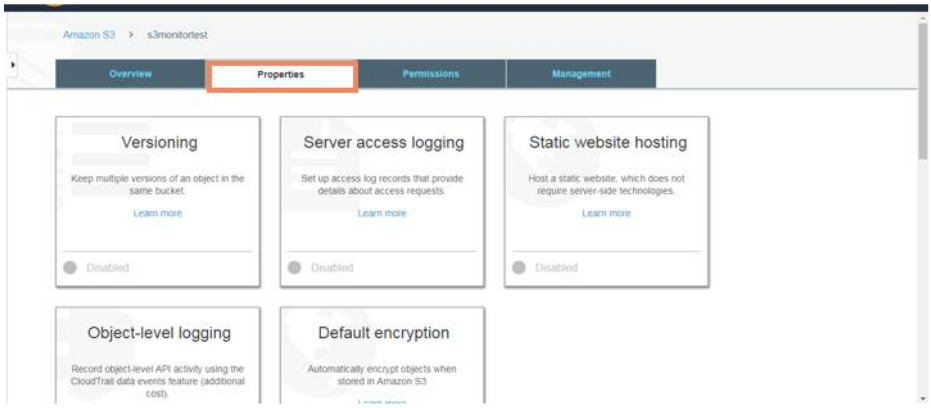
v) The users need to subscribe to the queue to an SNS topic.
vi) Navigate to the '**Queue Actions**' drop-down menu.
vii) Select the '**Subscribe Queue to SNS topic**' option.



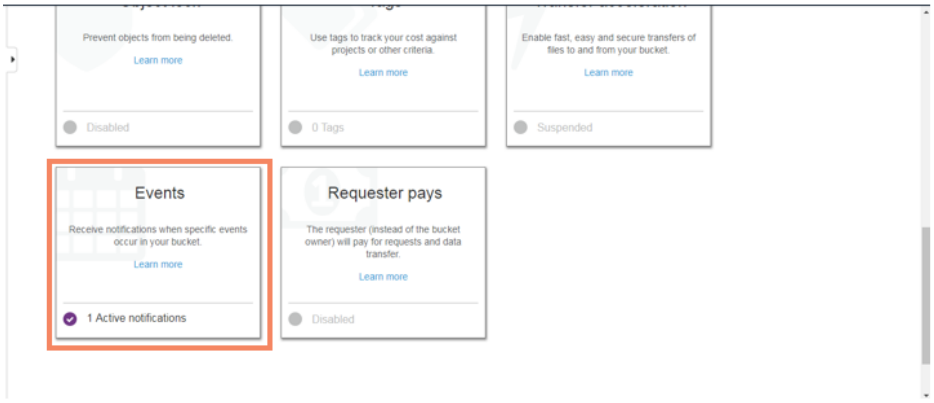
Note: Once the subscription is done with our SNS topic, the users need to create the event in S3 so that the pipeline can receive the uploaded message that comes to the S3 bucket.

9.5.4.3. Creating an S3 Event

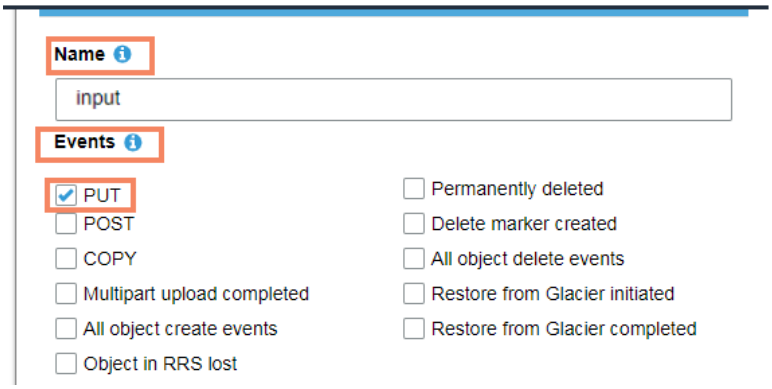
i) Log in to an AWS account.
ii) Create the S3 bucket.
iii) Select the '**Properties**' tab.



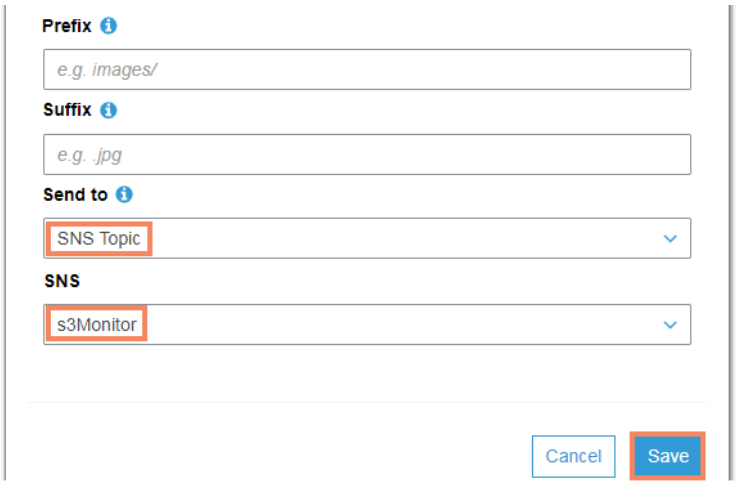
iv) Select the '**Events**' option under properties.



v) Create a new Event for the respective S3 bucket by providing the following information:
   a. Provide '**Name**' for the Event.
   b. Select an option using the checkmark in the given box from the provided list of **Events.**



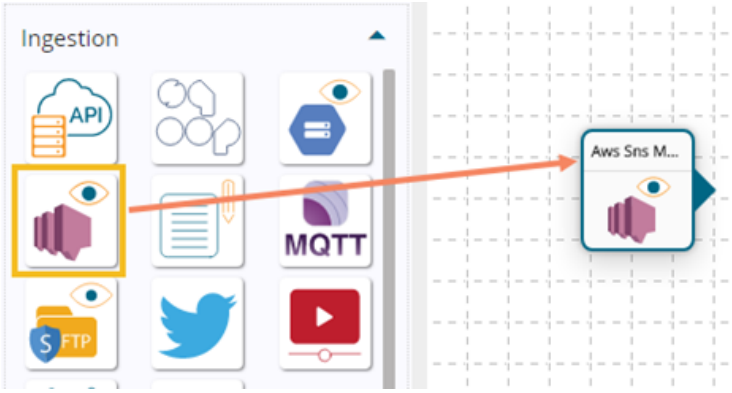
c. Select the **SNS topic** as an option from the '**Send to'** drop-down menu.
d. Select the '**S3 Monitor**' from the SNS drop-down menu.
e. Click the '**Save**' option.



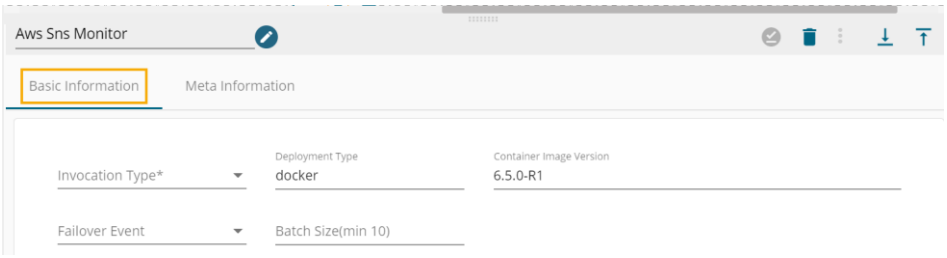
After the Event gets created, the users can access the complete setup to use the Data Pipeline SNS Monitor to monitor the S3 location.

9.5.4.4. Using the AWS SNS Monitor Component in the Data Pipeline

i) Drag and drop the AWS SNS Monitor component to the Workflow Editor.



ii) Click on the component to get the component properties tabs below.
iii) The **Basic Information tab** opens by default.
   a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation type is '**Real-Time**'.
   b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   d. Failover Event: Select a failover event from the drop-down menu.
   e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

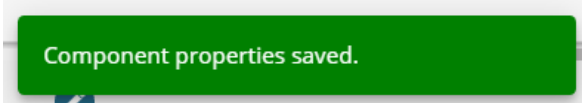


iv) Open the **Meta Information tab** and provide all the required details:

a. Access Key
b. Secret Key
c. Region
d. SQS URL

v) Click the '**Save Component in Storage**' icon to save the configuration details.

vi) The AWS SNS Monitor component gets configured and a notification message appears to confirm.

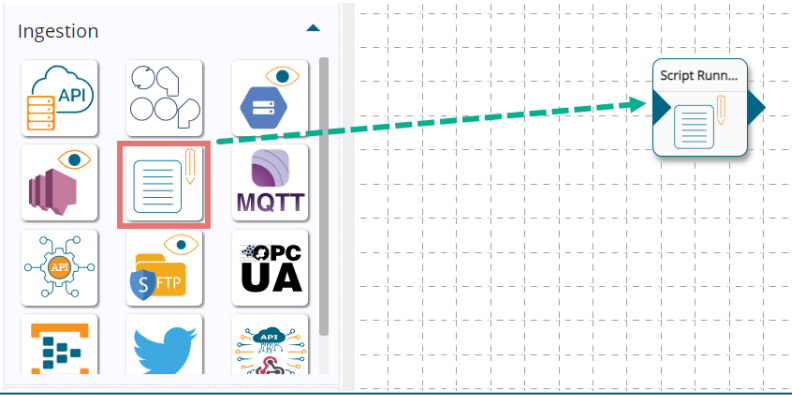


vii) The AWS SNS Monitor component can be used in a pipeline workflow now.

9.5.5. Script Runner

This component can be used for connecting it to a remote server/machine and running script files present there based on some events.

i) Drag and drop Script Runner Component to the Workflow Editor.



ii) Open the dragged Script Runner component to open the component properties tabs.

iii) The **Basic Information tab** opens by default.

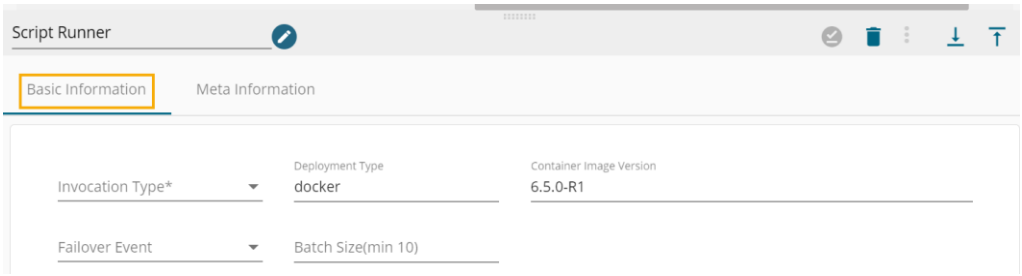
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation types are '**Real-Time**' and '**Batch**'.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Open the **Meta Information tab** and configure the required information.

a. **Host:** Host IP of the remote server/machine

b. **Username:** Username of the remote server/machine.

c. **Port:** Provide machine Port number.

d. **Authentication:** Select an authentication option from the drop-down menu.

1. Password: By selecting this option the user needs to pass the password.
2. PEM/PPK File: By selecting this option the user needs to pass the authentication file to connect to the server.

e. **Script type**: Choose the type of script file that You want to run out of SSH/ PERL/command options.

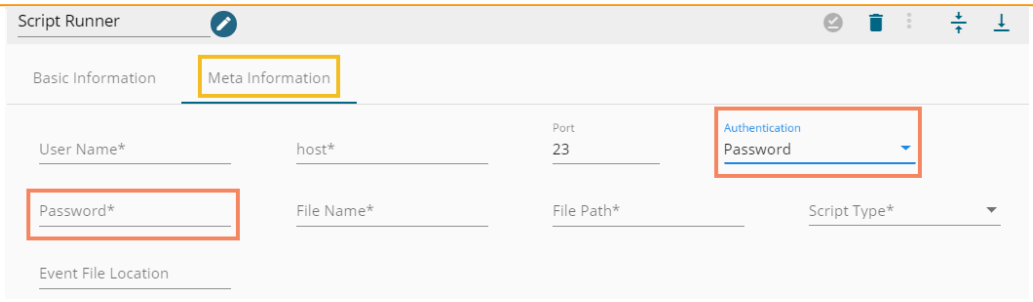
f. **File path:** Path of the file that is stored at the remote server.

g. **File Name:** The script file that you want to execute.

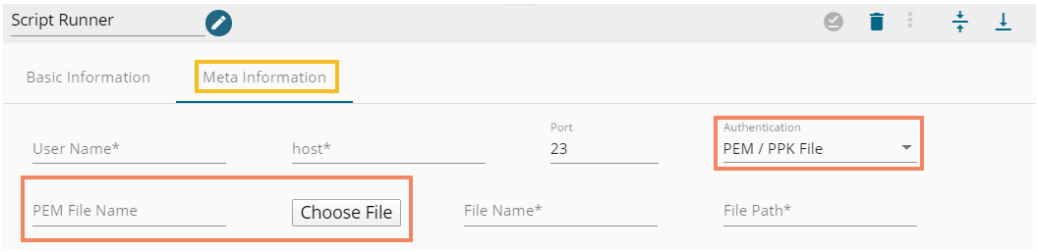
h. **Event File Location:** this is the location of the file sent through the file monitor (Non-mandatory).

Note: The displayed fields may vary based on the selected Authentication option.

Component Properties when the Authentication option is Password.

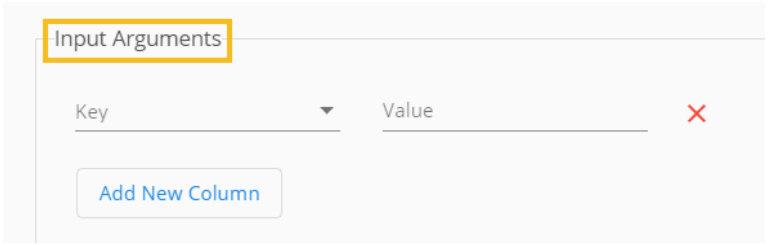


Component Properties when the Authentication option is PEM/PPK File.



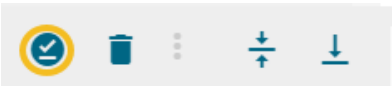
i. Input Arguments

1. Manual Arguments (Optional): These are the arguments to the parameter of the script that the user can provide manually.
2. Event Arguments (Optional): These are the arguments to the parameter coming from the previous event/Kafka topic.



Note: The user need not do anything if the arguments to be passed are coming from the previous event. By selecting Even Arguments, the value is set to true by default.

v) Click the '**Save Component in Storage**' icon (A notification message appears to confirm the action completion).



vi) The Script Runner component gets configured, and the notification message appears to inform the same.

Note: The component can connect to the remote machine using the details provided. It will pick the file from the location in that machine using the file name and file path respectively and finally execute the script after passing arguments (if any).

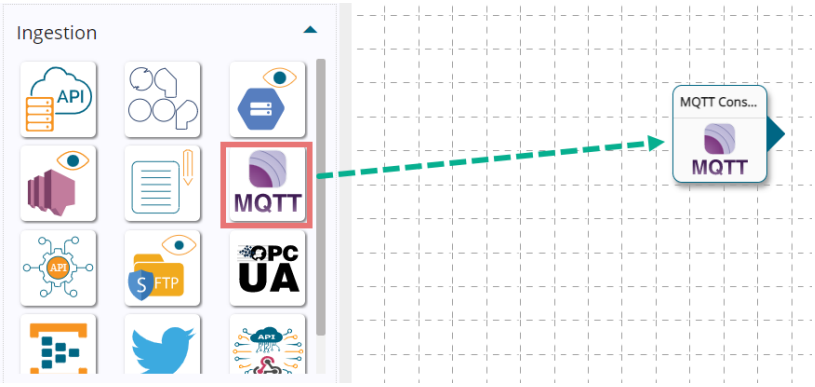
- **Limitations**

a. It accepts only lists as input i.e. the in-event data should be a list.
b. It sends data on the out-event only when there is a print statement as output in the script if not there will be no data on the out event.
c. The data produced from the script is of a list type.

9.5.6. MQTT Consumer

The MQTT consumer is instrumental in consuming data from IoT devices. It consumes data from provided MQTT broker and ingests the data in our data pipeline for further processing.

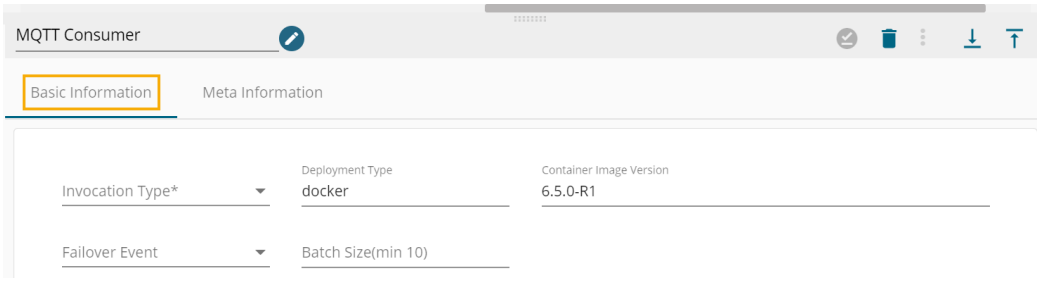
i) Drag and drop the MQTT Consumer to the Workflow Editor.



ii) Click the dragged MQTT Consumer component to get the component properties tabs.

iii) The **Basic Information tab** opens by default.

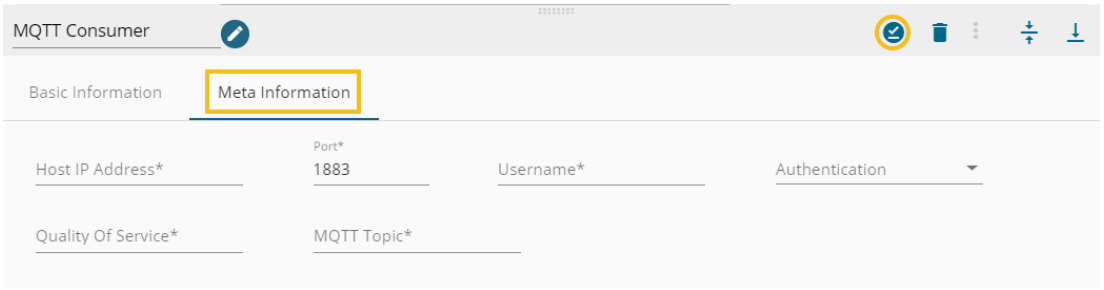
a. Select the invocation type (At present only the '**Real-Time**' option is provided)
b. Deployment Type: It comes preselected based on the component.
c. Container Image Version: It comes preselected based on the component.
d. Failover Event: Select a failover event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Click on the **Meta Information tab** to open the properties fields.

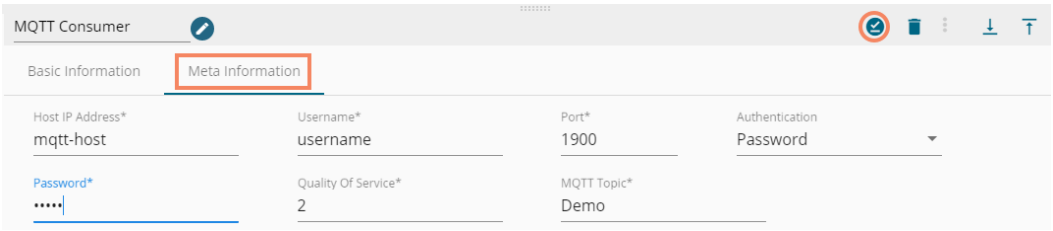
a. Host: Broker IP or URL
b. Username: If authentication is required then give username
c. Broker Port Number
d. Authentication: Select an authentication option using the drop-down list.
   i. Password: Provide a password to authenticate the MQTT component.
   ii. PEM/PPK File: Choose a file to authenticate the MQTT component
e. Quality of Service: The Quality of Service (QOS) level is an agreement between the sender of a message and the receiver of a message that defines the guarantee of delivery for a specific message.
f. MQTT Topic: Provide MQTT topic name where the user is producing data.

v) Click the '**Save Component in Storage**' icon (A notification message appears to confirm the action).

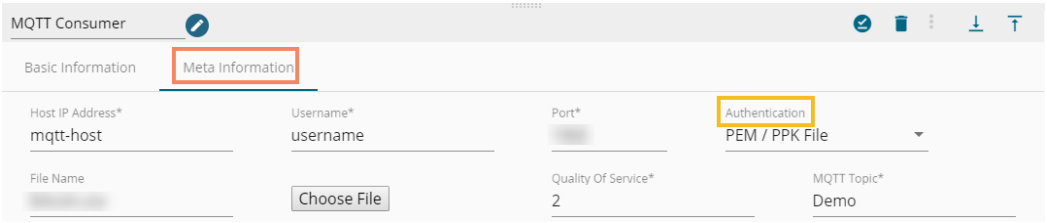


Note: The displayed fields may vary based on the selected Authentication option.

Configuration Fields when the Authentication option is Password

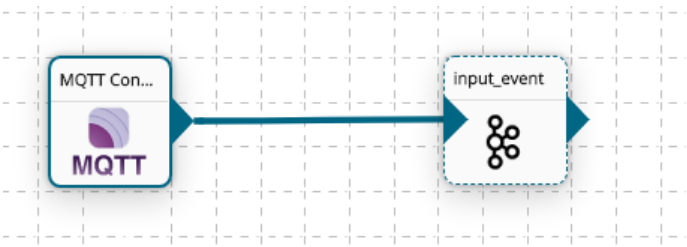


Configuration Fields when the Authentication option is PEM/PPK File



vi) The MQTT gets configured, and the notification message appears to inform the same.

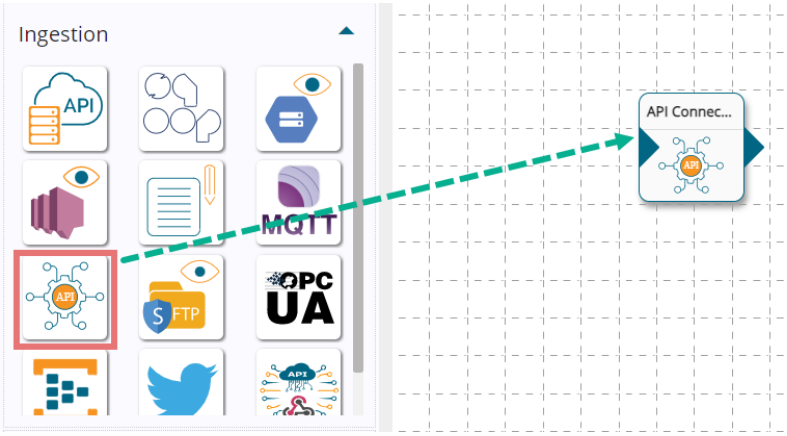
Note: The MQTT Consumer should be connected to an input event to pass the data and create a Pipeline workflow.



9.5.7. API Connector

The API connector helps to ingest data from the Google Analytics and Service Now API connectors.

i) Drag and Drop the **API Connector** component to the Workflow Editor.



ii) Click on the dragged ingestion component to get the component properties tabs.

iii) Configure the **Basic Information** tab.

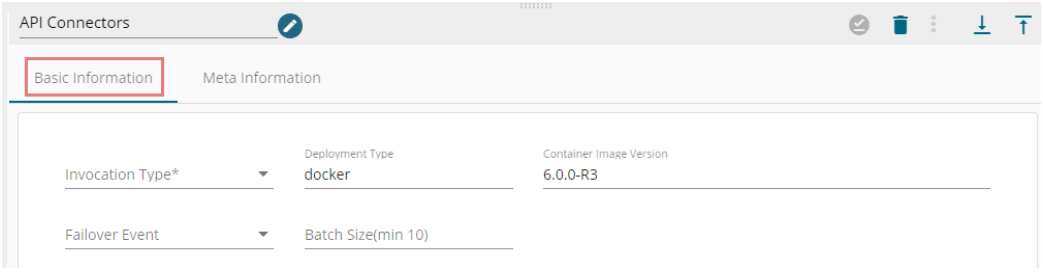
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select one option out of '**Real-Time**' and '**Batch**'.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Click on the **Meta Information tab** to open the component properties fields.

a) Customer Type: Select one option from the drop-down menu. The given choices are:

i. Service Now

ii. Google Analytics

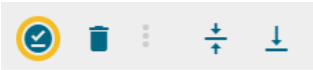
Meta Information fields when the Connector Type is Service Now

1. Connector Type: Select the Service Now option from the drop-down menu.
2. Service Now DataSet: Select a Service Now Data Set from the drop-down menu.
3. Domain: it gets selected based on the chosen Data Set
4. Query: It gets reflected in the query statement based on the selected Data Set
5. Columns: It gets reflected based on the selected Data Set
6. Dataset ID: It gets reflected based on the selected Data Set

Meta Information fields when the Connector Type is Google Analytics

1. Connector Type: Select the Google Analytics option from the drop-down menu.
2. Data Connector: Select a Google Analytics data connector from the drop-down menu.
3. Start Date: It gets reflected based on the chosen Data Connector.
4. End Date: It gets reflected based on the chosen Data Connector.
5. Metrics: The selected metrics get reflected based on the choice of the data connector.
6. Dimensions: The selected dimensions from the data connector get reflected.
7. Domain: The domain address appears based on the selected data connector.
8. View id: It gets reflected from the selected data connector.
9. Data Set ID: It gets reflected from the selected data connector.

v) Click the '**Save Component in Storage**' icon to save the component properties.



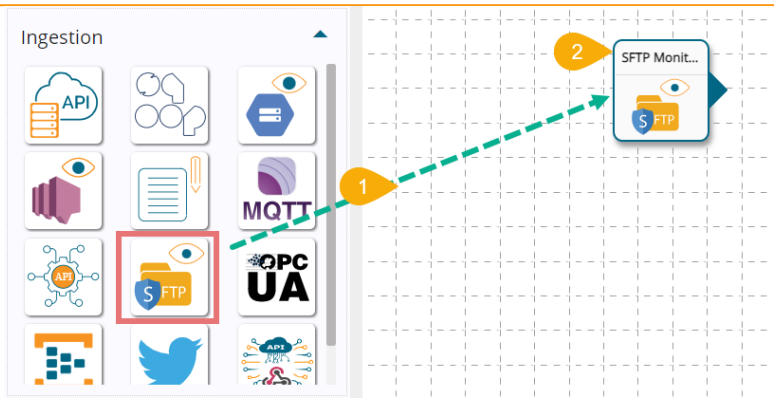
vi) A notification message appears to inform about the action.

vii) The API Connector component gets configured.

9.5.8. SFTP Python Monitor

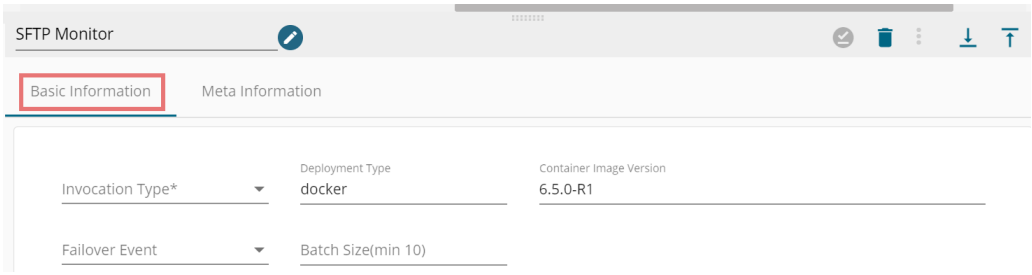
i) Drag and Drop the SFTP Python Monitor ingestion component to the Workflow Editor.

ii) Click the dragged ingestion component to get the component properties tabs.



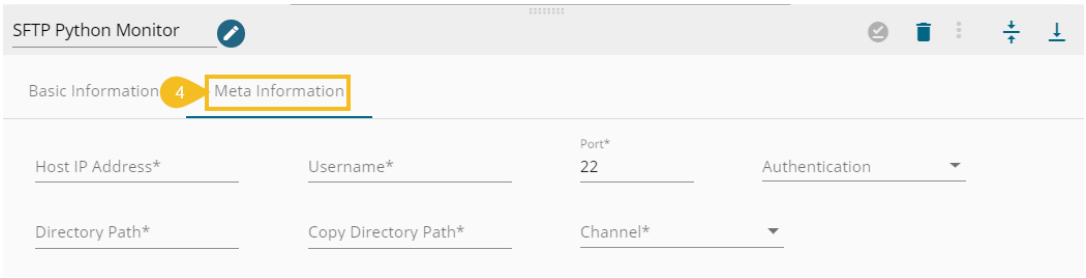
iii) Configure the **Basic Information Tab**.

a. Select the invocation type (at present only the '**Real-Time**' option is provided).
b. Deployment Type: It comes preselected based on the component.
c. Container Image Version: It comes preselected based on the component.
d. Failover Event: Select a failover event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



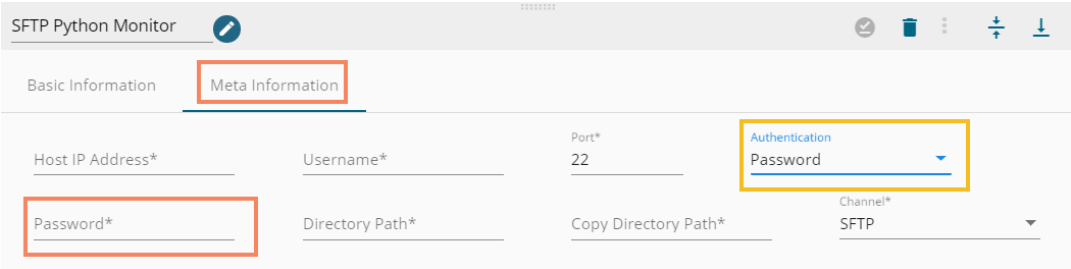
iv) Configure the **Meta Information tab** for the dragged SFTP Python Monitor component.

a. Host: Broker IP or URL
b. Username: If authentication is required then give username
c. Port: Provide the Port number
d. Authentication: Select an authentication option using the drop-down list.
   i. Password: Provide a password to authenticate the MQTT component.
   ii. PEM/PPK File: Choose a file to authenticate the MQTT component
e. Directory Path: Fill monitor folder path using forward-slash (/). E.g., **monitor/**
f. Copy Directory Path: Fill in the copy folder name where you want to copy the uploaded file. E.g., **monitor_copy/**
g. Channel: Select a channel option from the drop-down menu (the supported channel is SFTP).

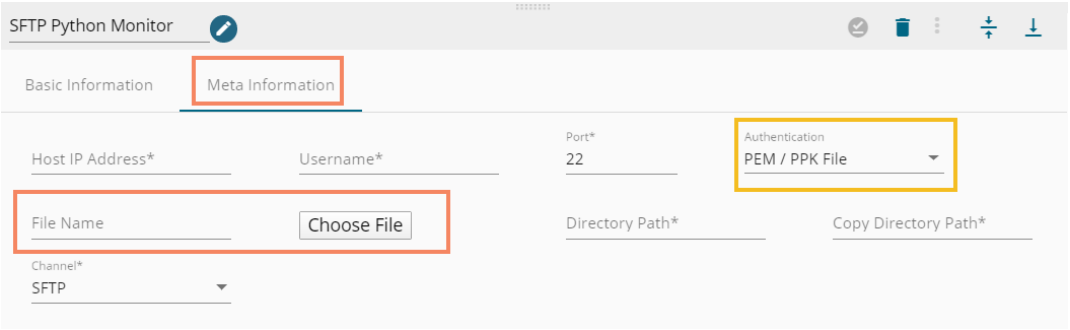


Note: The SFTP Python Monitor may have Meta Information fields based on the selected Authentication option.

Meta Information Fields when the Authentication option is Password



Meta Information Fields when the Authentication option is PEM/PPK

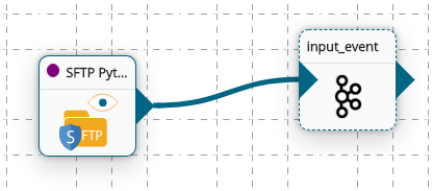


v) Click the '**Save Component in Storage**' icon to save configured details of the SFTP Python Monitor component.

vi) A notification message appears to confirm the same.

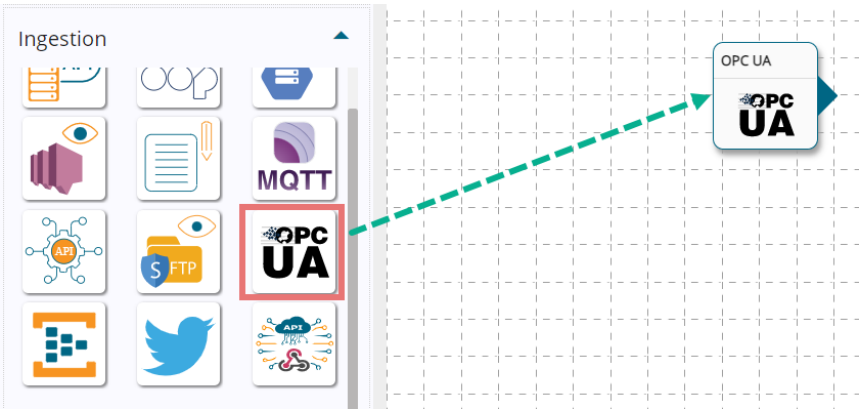
Note:

a. The SFTP Python Monitor component monitors the file coming to the monitored path and copies the file in the Copy Path location for SFTP Reader to read.
b. The SFTP Python Monitor component requires an Event to send output.



9.5.9. OPC UA

i) Drag and Drop the **OPC UA** ingestion component to the Workflow Editor.



ii) Click on the dragged ingestion component to get the component properties tabs.

iii) Configure the **Basic Information tab**.

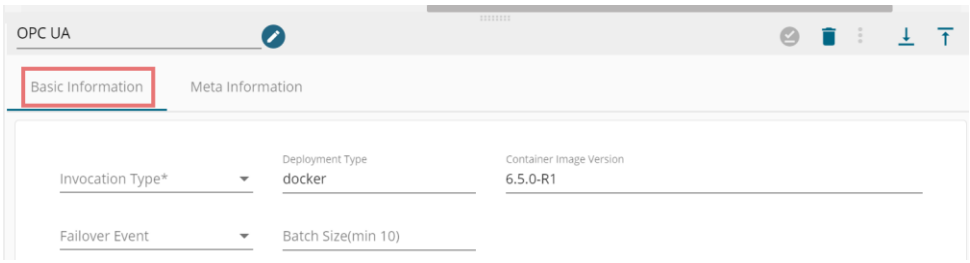
a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. Select one option out of '**Real-Time**' and '**Batch**'.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

d. Failover Event: Select a failover event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Configure the **Meta Information tab** by providing the required fields.

a. URL (*)- Provide URL link

b. Message Security Mode- Select a message security mode from the drop-down menu (The supported options are '**Sign**' and '**SignAndEncrypt**')

c. Security Policy- Select a policy using the drop-down menu.

d. Certificate File Name- This name gets reflected based on the Choose File option provided for the Certificate file.

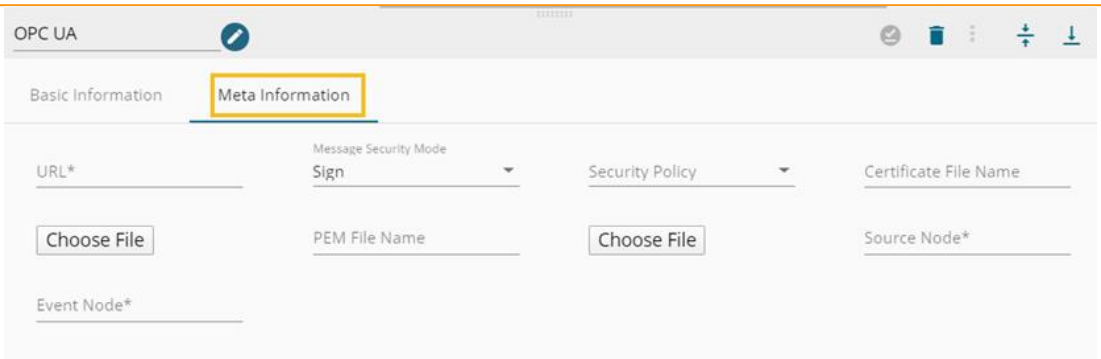
e. Choose File- Browse a certificate file by using this option.

f. PEM File Name: This name gets reflected based on the Choose File option provided for the PEM file.

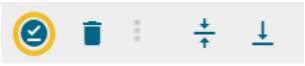
g. Choose File: Browse a PEM file by using this option.

h. Source Node: Provide the source node.

i. Event Note: Provide the event node.



v) Click the '**Save Component in Storage**' icon.



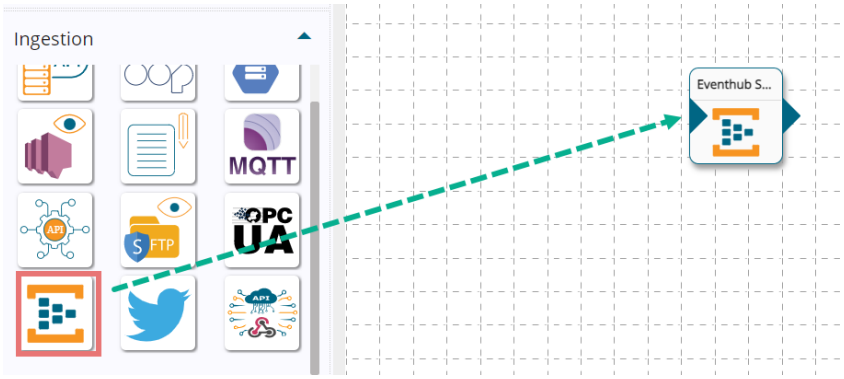
vi) A notification message appears to inform about the action.

vii) The OPC UA ingestion component gets configured, and a notification message appears to inform the same.

9.5.10. Eventhub Subscriber

This Component consumes the event ingestion service (event hub) provided by azure. The user can directly subscribe to the azure event hub and ingest the events into a data pipeline. This comes under the ingestion component group.

i) Drag and drop the Eventhub Subscriber component to the Workflow Editor.



ii) Click the dragged Eventhub component.

iii) Click on the dragged ingestion component to get the component properties tabs.

iv) Configure the **Basic Information tab**.

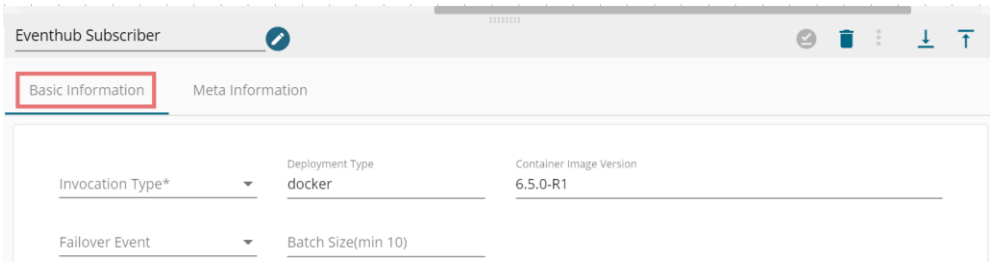
a. Select an Invocation type from the drop-down menu to confirm the running mode of the component. The supported invocation option is '**Real-Time**'.

b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.

c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.

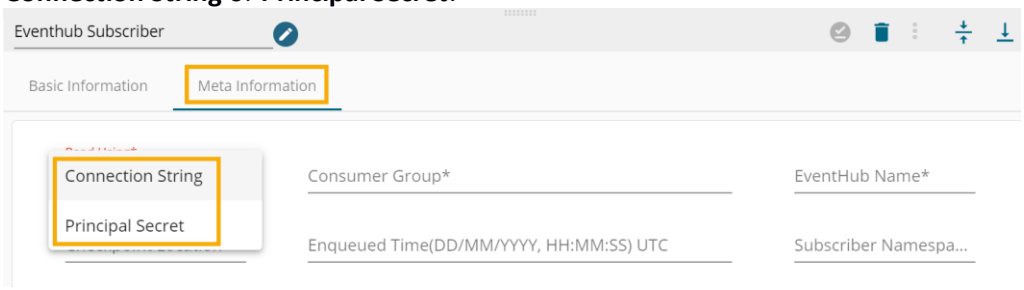
d. Failover Event: Select a failover event from the drop-down menu.

e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



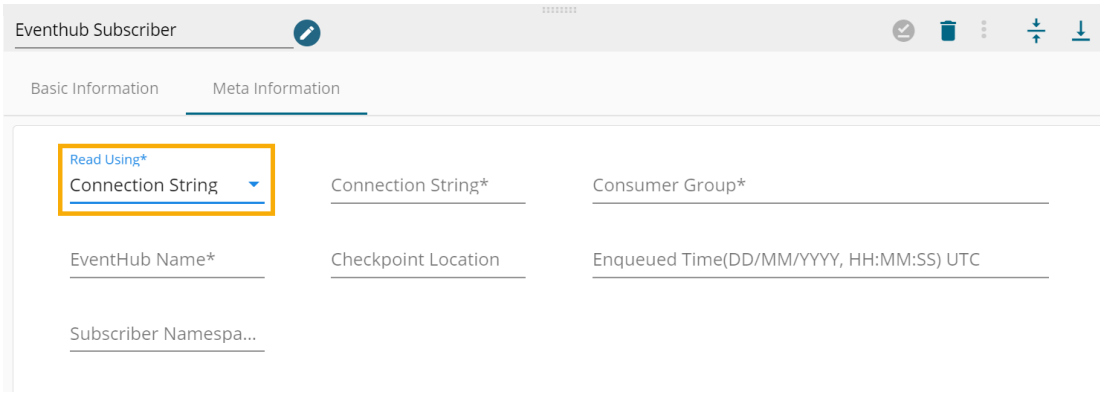
v) Configure the **Meta Information tab** by providing the required fields.

vi) The user can configure the Eventhub Subscriber meta-Information by reading the data using either **Connection String** or **Principal Secret**.



The Meta Information fields with **Connection String** as a Reading option.

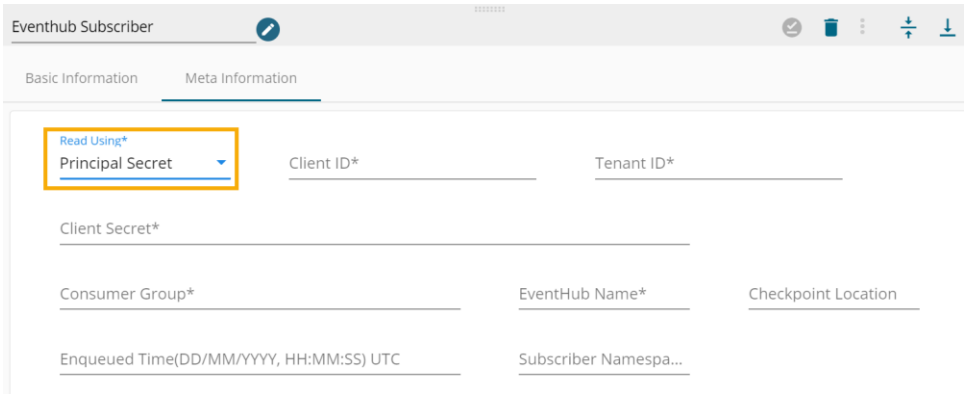
a. **Reading Using**: Select the '**Connection Sting**' to connect to the event hub service (this can be found in Azure platform access manages].
b. **Consumer Group**: A consumer group is a view (state, position, or offset) of an entire event hub.
c. **Eventhub name**: Name of the subscribed event hub.
d. **Checkpoint location:** it is an HDFS location URL ("hdfs://<ip>/user/Cloudera/Test"). It can be used when a component fails, and you want to read from the exact event where you left.
e. **Enqueued time:** This is to be used if the user wants to read the events from a specific time.
f. **Subscriber Namespace:** Subscriber Namespace of the subscribed Eventhub.



The Meta Information fields with **Principal Secret** as a Reading option.

a. **Reading Using**: Select the '**Principal Secret'** to connect to the event hub service.
b. **Client ID**: Provide the Client ID.
c. **Configure Tenant ID**: Provide the Tenant ID.
d. **Client Secret**: Provide the Client Secret ID.
e. **Consumer Group**: A consumer group is a view (state, position, or offset) of an entire event hub.

f. **Eventhub name**: Name of the subscribed event hub.
g. **Checkpoint location:** it is an HDFS location URL ("hdfs://<ip>/user/Cloudera/Test"). It can be used when a component fails, and you want to read from the exact event where you left.
h. **Enqueued time:** This is to be used if the user wants to read the events from a specific time.
i. **Subscriber Namespace:** Subscriber Namespace of the subscribed Eventhub.



vii) Click the '**Save Component in Storage**' icon.

viii) A notification message appears to inform about the action.

ix) The Eventhub Subscriber ingestion component gets configured, and a notification message appears to inform the same.

9.5.11. Twitter Scrapper

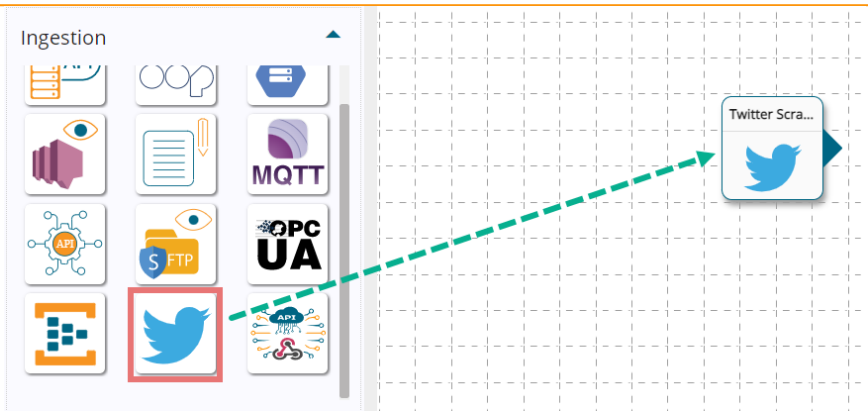
Twitter Scrapper helps in scrapping Twitter data. Twitter Scrapper provides two types of scrapping:

1. Real-Time
2. History Data

Real-Time – By using this option, the user can fetch the live Tweets by using Twitter API along with specific search filters.

History Data – By using this option, the user can provide or fetch Twitter data for the past 7 days (provided limit is up to 1000 tweets). It also provides the users with a facility to fetch data according to specific
tags.

i) Drag them and drop Twitter Scrapper to the Workflow Editor.



ii) Click the dragged Twitter Scrapper component to get the component properties tabs.

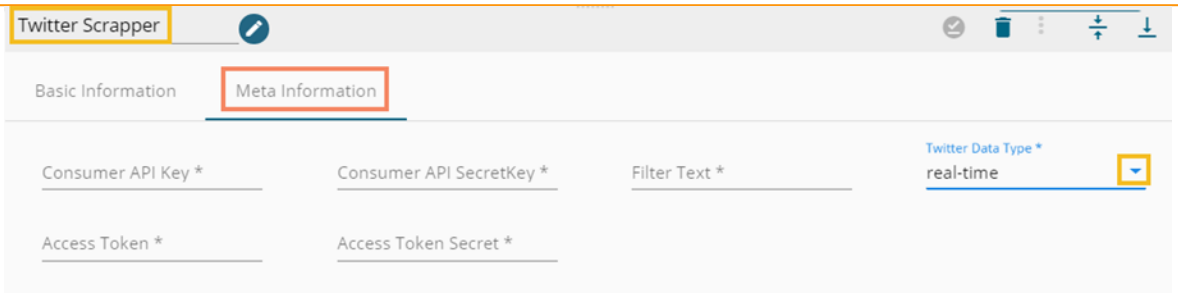
iii) The **Basic Information tab** opens by default.

a. Select the invocation type (at present only the '**Real-Time**' option is provided)
b. Deployment Type: It comes preselected based on the component.
c. Container Image Version: It comes preselected based on the component.
d. Failover Event: Select a failover event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

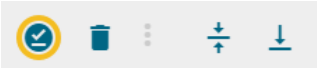
iv) Click on the **Meta Information tab** to open the configuration fields for the selected columns.

a. Consumer API Key: This is going to be unique depending upon the user's account. It is essential in both the cases of scrapping Real-time and History Data.
b. Consumer API Secret Key: This is going to be unique depending upon the user's account. It is essential in both the cases of scrapping Real-time and History Data.
c. Filter Text: This field is used to specify separate tags that the user wants to fetch from Twitter. It provides the choice to the user for fetching multiple tags data at a time, to accomplish this user needs to provide the tags name with a comma separator. For instance, #India, #Cricket
d. Twitter Data Type: This field provides a dropdown menu from where the consumer can select the scrapping strategy(real-time/history).
e. Access Token: This field is consumable only when the user wants to fetch real-time data from a scrapper.
f. Access Token Secret: This field is consumable only when the user wants to fetch real-time data from a scrapper.

Note: The Access Token and the Access Token Secret (key) fields appear only when the selected Twitter Data Type is 'real-time'.



v) Save the Twitter Scrapper component.



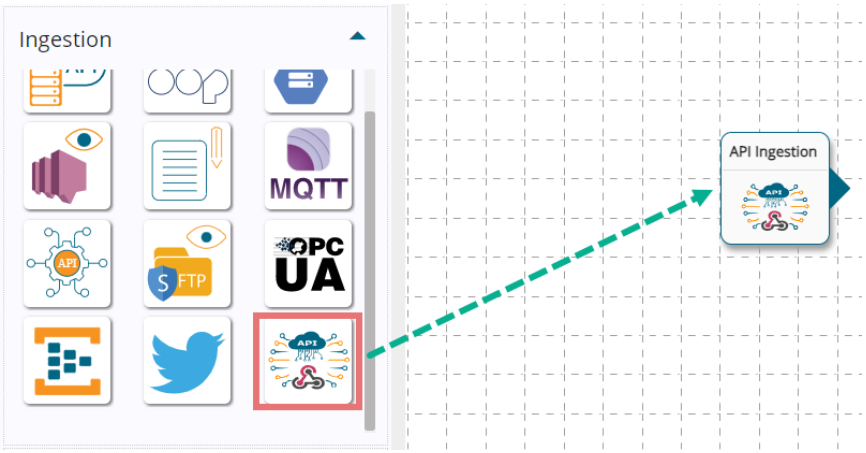
vi) The Twitter Scrapper component properties get saved and a notification message appears to inform the same.

Note: The user should know the Consumer API Key, Consumer Secret Key, Access Token, Access Token Secret from a developer account of Twitter.

9.5.12. API Ingestion

The API Ingestion component is provided in the Component to ingest external API content.

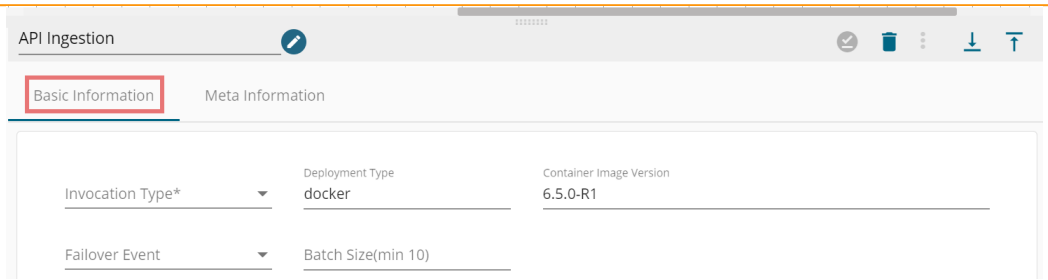
i) Drag them and drop API Ingestion to the Workflow Editor.



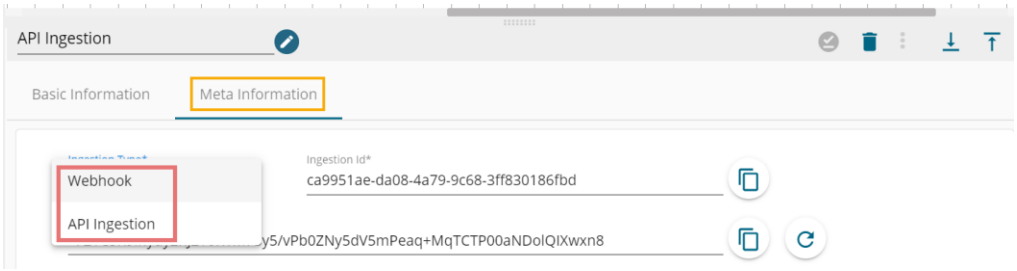
ii) Click the dragged API Ingestion component to get the component properties tabs.

iii) The **Basic Information tab** opens by default.

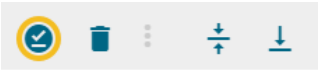
a. Select the invocation type (at present only the '**Real-Time**' option is provided)
b. Deployment Type: It comes preselected based on the component.
c. Container Image Version: It comes preselected based on the component.
d. Failover Event: Select a failover event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Click on the **Meta Information tab** to open the configuration fields for the selected columns.
   a. The Ingestion Type field provides two options:
      i. Webhook
      ii. API Ingestion
   b. Provide an Ingestion Id.
   c. Provide an Ingestion Secret Key.
   d. Provide Component Instant Id URL.



v) Save the API ingestion component.



vi) The API Ingestion component properties get saved and a notification message appears to inform the same.

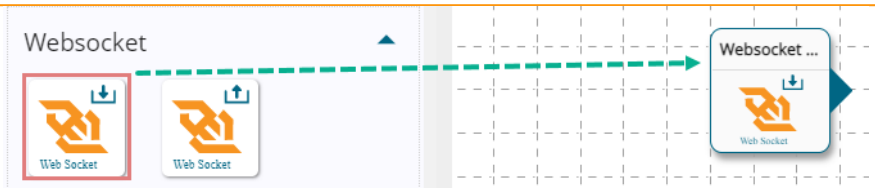
9.6. WebSocket

The WebSocket component provided in the Components Pallet of the Data pipeline contains WebSocket Listener and WebSocket Writer components to produce and consume the live streaming data using socket.

9.6.1. WebSocket Listener

The WebSocket Listener component listens to and accepts the incoming WebSocket connection requests.

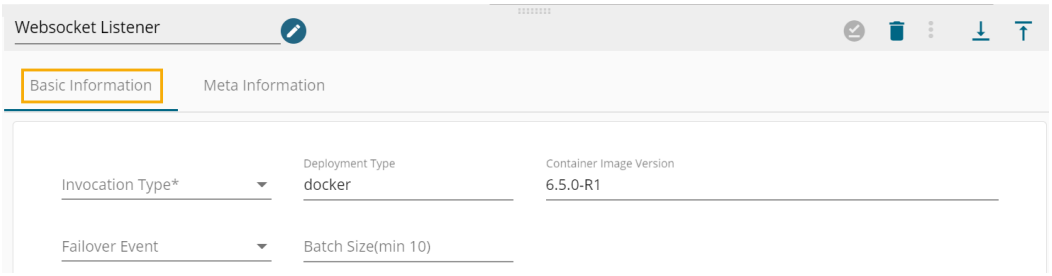
i) Drag the WebSocket Listener component to the Workflow Editor.



ii) Open the WebSocket Listener component to get the Component Properties tab below.

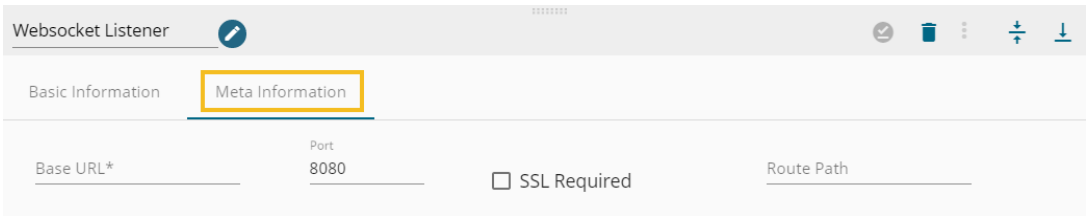
iii) The **Basic Information tab** opens by default.

a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation type is '**Real-Time**'.
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

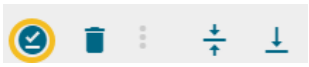


iv) Configure the **Meta information tab** by providing the required fields.

a. Base URL
b. Port
c. SSL Required- Enable the
d. Route Path



v) Click the '**Save Component in Storage**' icon.



vi) A notification message appears to inform about the action.

vii) The WebSocket Listener component gets configured, and a message appears to inform the same.

9.6.2. WebSocket Producer

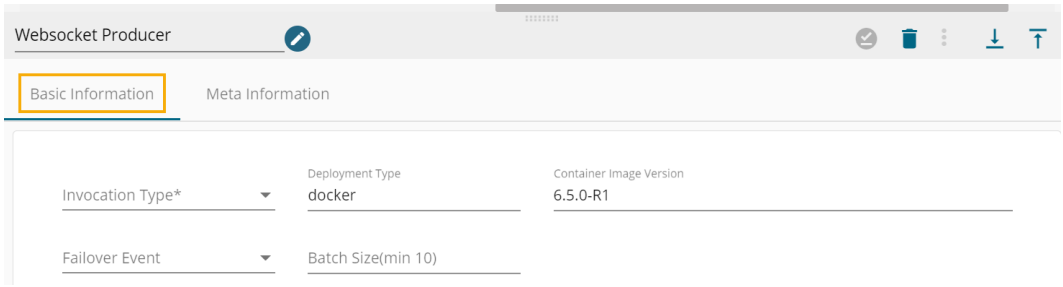
The WebSocket Producer helps the user to get the message received by the Kafka channel.

i) Drag the WebSocket Producer component to the Workflow Editor.

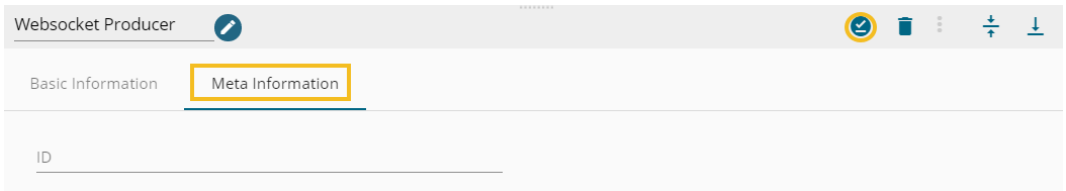
ii) Open the WebSocket Producer component to get the Component Properties tab below.

iii) The **Basic Information tab** opens by default.

a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation type is '**Real-Time**'.
b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
d. Failover Event: Select a failover event from the drop-down menu.
e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).



iv) Provide the WebSocket ID field to configure the **Meta Information tab**.



v) Click the '**Save Component in Storage**' icon (A message appears to confirm the same).

vi) A notification message appears to inform about the action.

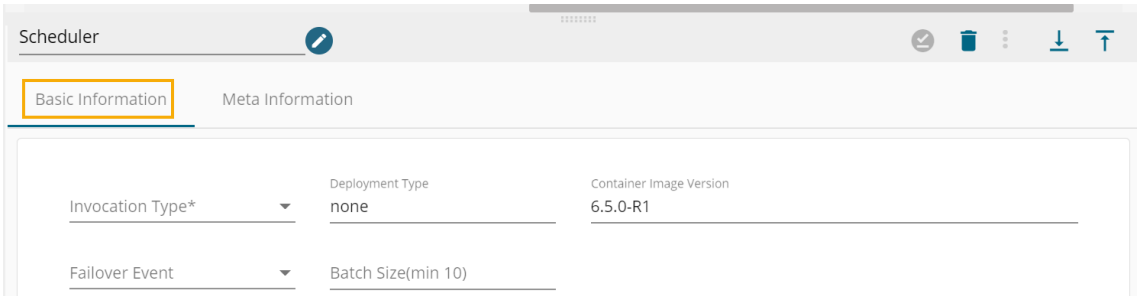
vii) The WebSocket Producer component gets configured, and a message appears to inform the same.

9.7. Scheduler

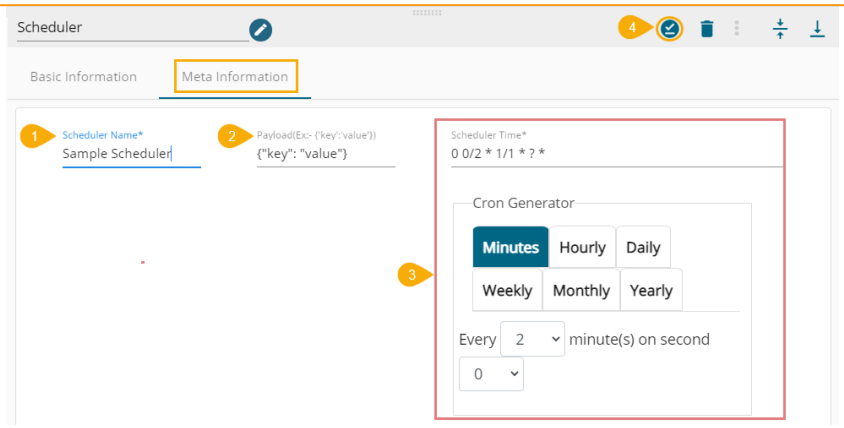
The scheduled pipelines are listed together with the scheduler details. It displays the meta-information filled in the scheduler component for the respective pipeline. The page also contains information on how many times the pipeline has been triggered and when the next time the scheduled component will get deployed.

i) Drag and drop the Scheduler component to the Workflow Editor.
ii) Connect it with a reader or Data Loading component (Input event).

iii) Click on the Scheduler component to get the configuration details.
iv) The Basic tab opens by default.
   a. Select an Invocation type from the drop-down menu to confirm the running mode of the reader component. The supported invocation type is '**Real-Time**'.
   b. Deployment Type: It displays the deployment type for the component. This field comes pre-selected.
   c. Container Image Version: It displays the image version for the docker container. This field comes pre-selected.
   d. Failover Event: Select a failover event from the drop-down menu.
   e. Batch Size (min 10): Provide the maximum number of records to be processed in one execution cycle (Min limit for this field is 10).

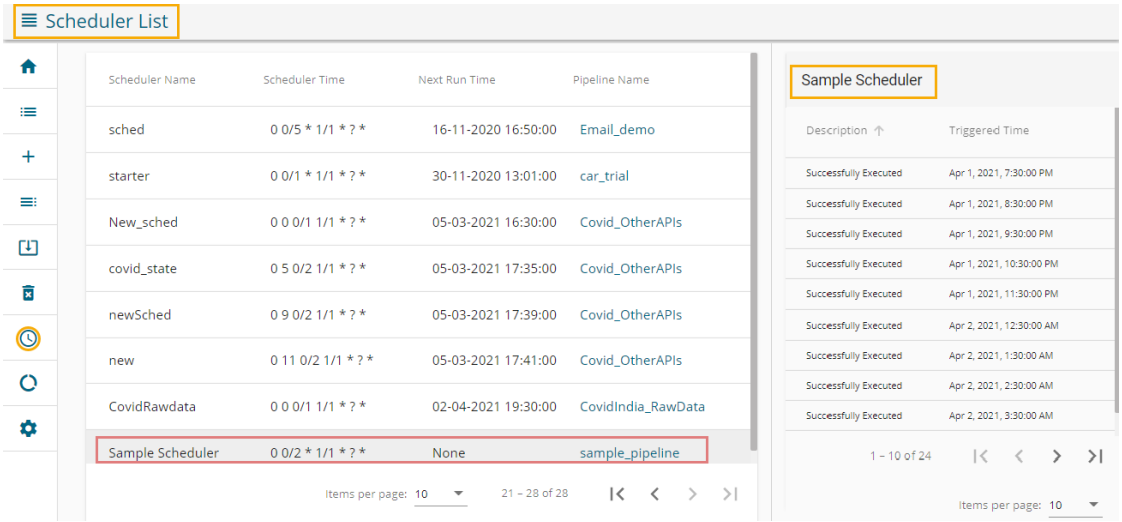


v) Open the Meta Information tab and configure the required fields.
vi) Provide Scheduler name
vii) Provide Payload value i(n the predefined format)
viii) Set Scheduler Time using the Crone Generator (the displayed fields may vary based on the selected time).
ix) Click the '**Save Component in Storage**' icon to save the component configuration.



x) A notification message appears to confirm the action.

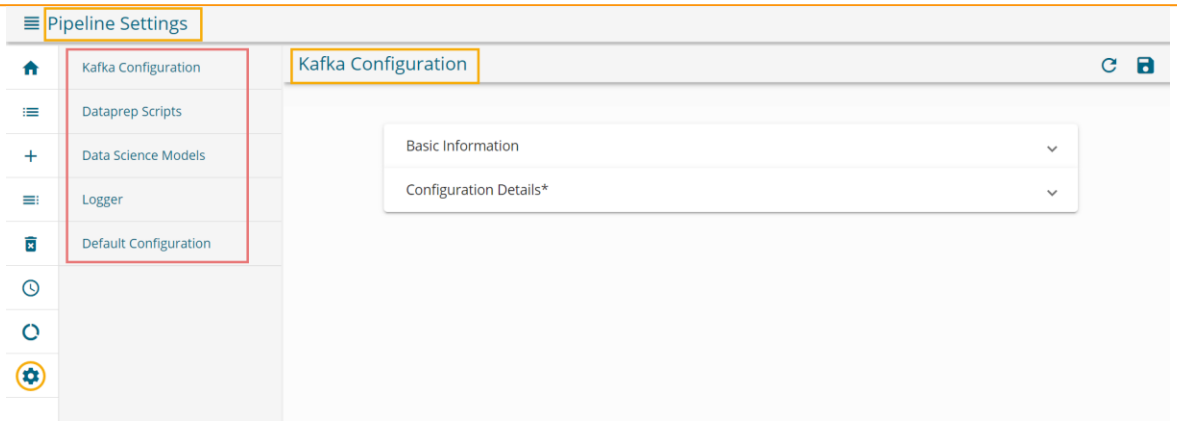
xi) Update and activate the pipeline (notification messages appear to confirm the same).
xii) Open the Scheduler tab from the landing page screen.
xiii) The scheduled pipeline gets listed displaying the triggered time and Next Run Time details.



Note: The scheduler supports only reader components at present.

10. Settings

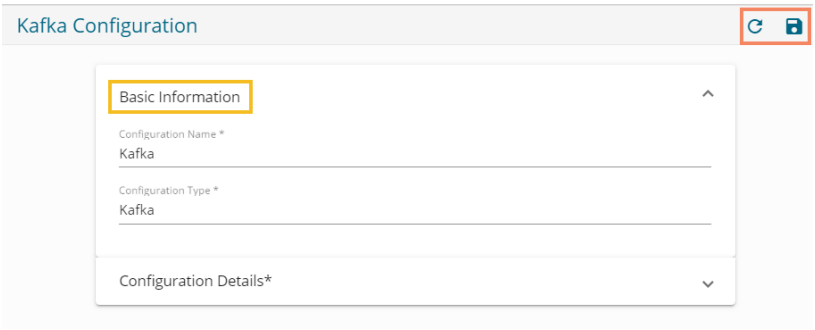
The user gets settings information about various pipeline functionalities. The Settings page provides information about Kafka Configuration, Data Prep Scripts, Data Science Models, and Logger. The Kafka Configuration details open by default while selecting the '**Settings**' page.



10.1. Kafka Configuration

The Kafka Configuration page displays editable Basic Information and Configuration Details for Kafka.

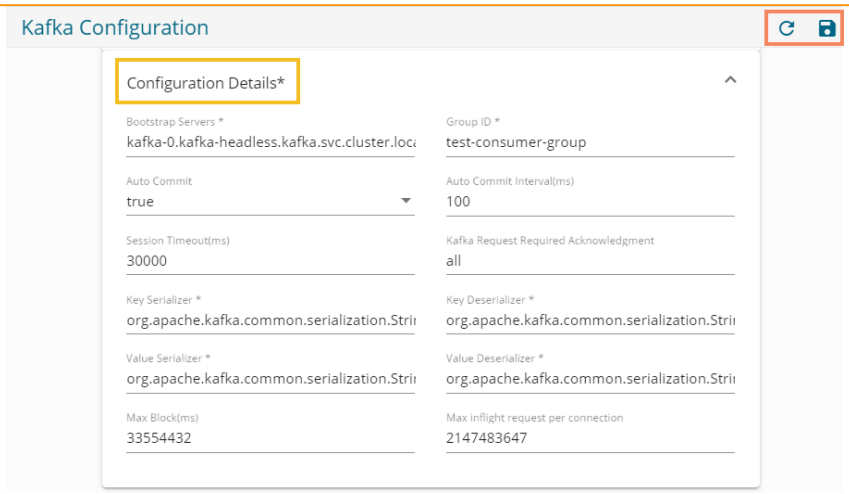
Basic Information contains Configuration Name and Configuration Type. The basic information details can be refreshed and saved by using the icons given for the same.



The Configuration Details tab for Kafka displays the following information:

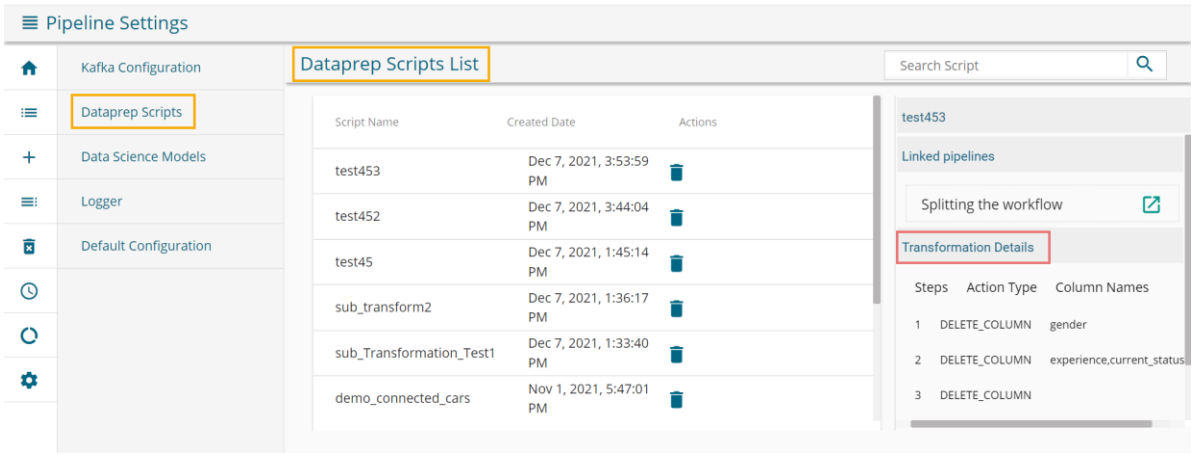
a. Bootstrap Servers
b. Group ID
c. Auto Commit
d. Auto Commit Interval(ms)
e. Session Timeout(ms)
f. Kafka Request Required Acknowledgment
g. Key Serializer
h. Key Deserializer
i. Value Serializer
j. Value Deserializer
k. Max Block(ms)
l. Max Inflight request per connection

The user can refresh the configuration and save the configuration by using the provided icons for the same.



10.2. Dataprep Scripts

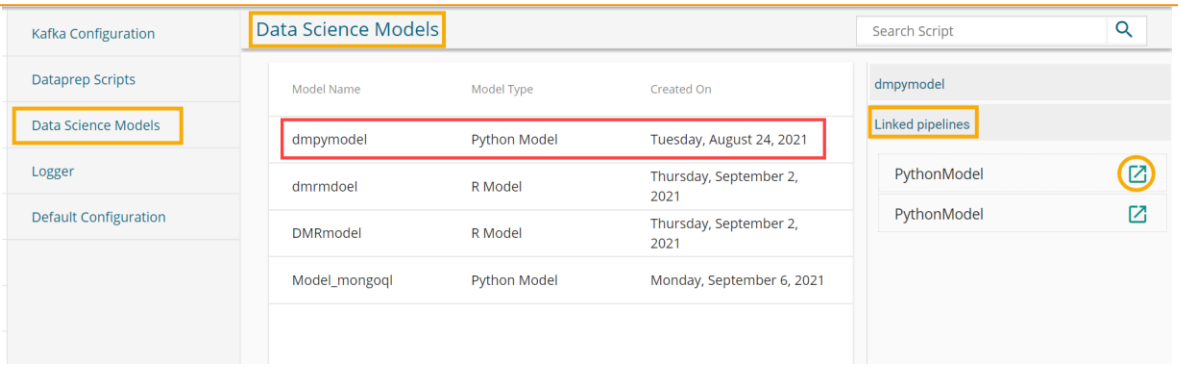
The user can see a list containing all the exported Dataprep Scripts from the Data Preparation module under the Dataprep Scripts tab provided on the Settings page. The user can also view the Transformation Details provided with the sequence of Steps, Action Type, and Column Names for the selected Dataprep Script.



Note: The user can delete a Dataprep Script by clicking the '**Delete Script**' icon provided next to it.

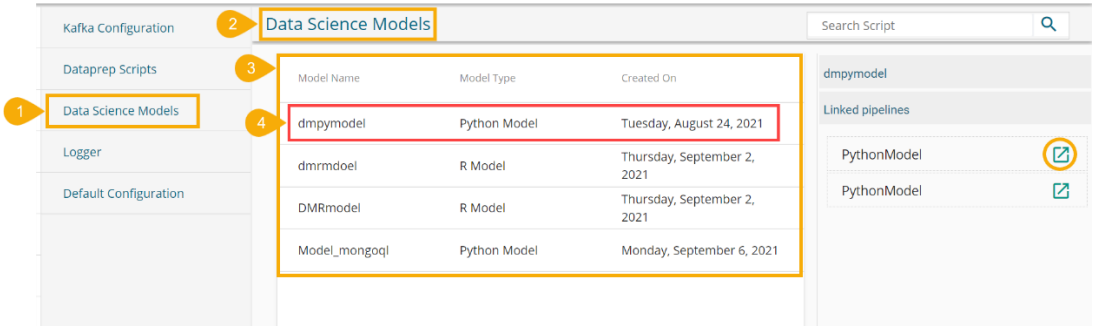
10.3. Data Science Models

The exported models from Data Science Workbench gets listed on this page. The linked pipeline to the selected model is also mentioned on the right side of the page. The user can navigate to the related pipeline by clicking the '**Open Pipeline**' icon from this page.

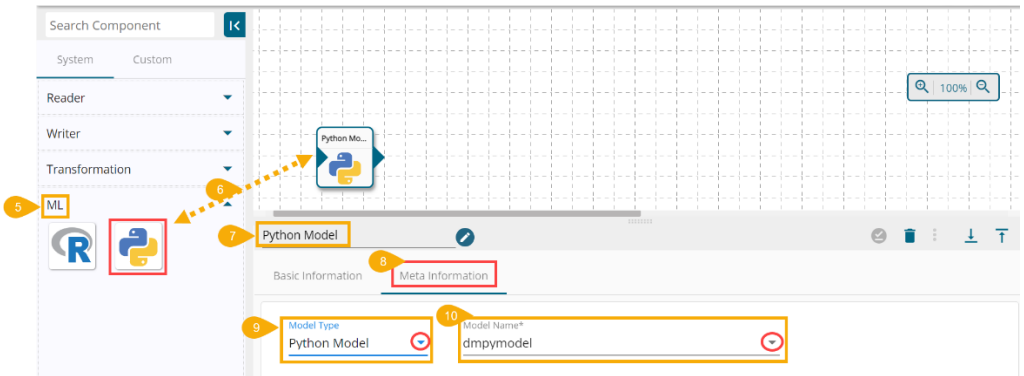


10.3.1. Using a Deployed Model in Data Pipeline

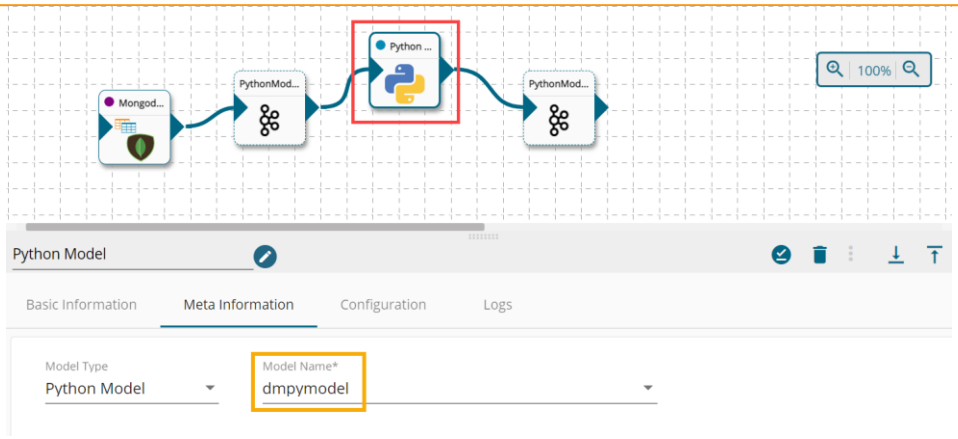
i) Navigate to the Pipeline Settings page.
ii) Select the '**Data Science Models**' option.
iii) The deployed Data Science Models display in a list.
iv) Select a Data Science Model from the list.



v) Navigate to the ML section on the Pipeline Editor page.
vi) Drag the Python Model component to the workspace.
vii) Click on the dragged Python Model component to display the configuration fields.
viii) Select the '**Meta Information**' tab.
ix) Select the '**Python Model**' using the Model Type drop-down menu.
x) Select a model using the '**Model Name**' drop-down menu.

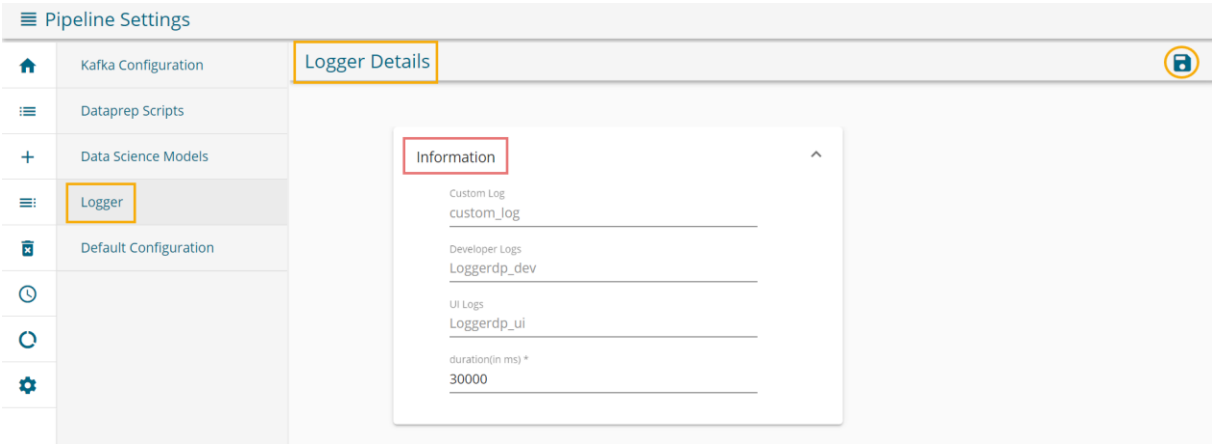


xi) The user can drag and connect the Data Science Model with other relevant components to create a pipeline workflow (as displayed in the following image).



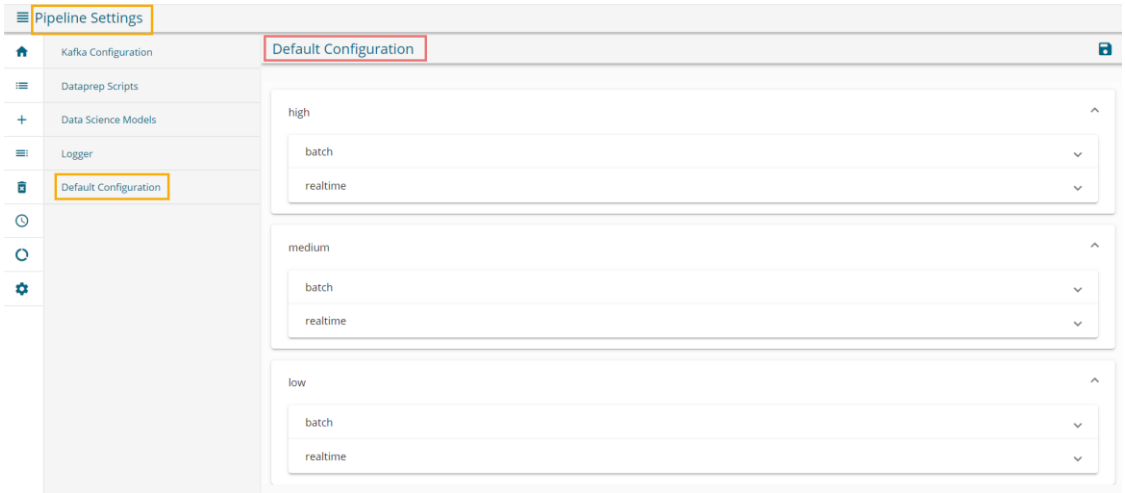
10.4. Logger

This page displays the Logger configuration details for the UI Logs that appear on the log panel of the workflow editor.



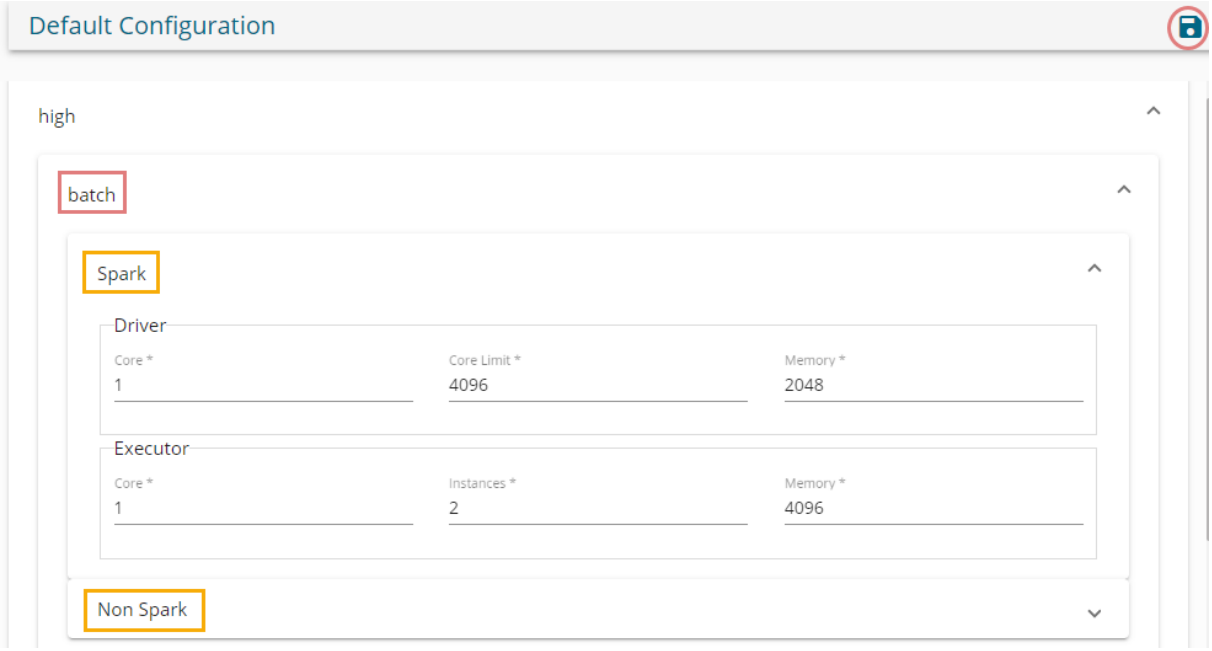
10.5. Default Configuration

The Default Configuration displays pipeline default configuration information categorized as High, Medium, and Low level. The users can edit the information and click the '**Save Configuration**' icon.



Each category displays further division as batch and real-time:

Each batch and real-time category further display Spark and Non-Spark categories:



A message appears to confirm the updates.

The configuration gets saved.

11. Third-Party Data Ingestion using Data Pipeline

The Python Script component in Data Pipeline can be used in a very flexible way as a reader, ingestion, transform component, and data writer component.
All the user needs to do is to write a custom script according to the use case.

Below is an example of listening data from an MQTT topic and calling a machine learning model API. this API return data after applying the model to it.

11.1. Data Ingestion from an MQTT Consumer

Sample Code:

```
from threading import Thread
import paho.mqtt.client as mqtt
import os,json,requests
from confluent_kafka import Producer

def get_out_event():
    componentInsId = os.getenv('COMPONENT_INST_ID')
    url = str(os.getenv('MDS_BASE_URL')) + "/api/v1/componentInst/" + str(componentInsId)
    uuid = os.getenv('UUID')
    if uuid is not None:
      url = url + "?uuid=" + str(uuid)
    resp = requests.get(url)
    component_json = json.loads(resp.content)
    data = component_json['data']
    meta_data = data['metaData']
    out_event = meta_data['outEvent'][0]
    return out_event

def kafka_config():
    url = str(os.getenv('MDS_BASE_URL')) + "/api/v1/configuration/kafka"
    resp = requests.get(url)
    component_json = json.loads(resp.content)
    config = component_json['data']['confDetails']['bootstrap.servers']
    return config



    config = kafka_config()
    # config = "192.168.1.32:9092"
    out_event = get_out_event()
    # out_event = "adnoc_wesocket_test"
    p = Producer({'bootstrap.servers': config})

def sent_mqtt_data(broker_address,username,password,topic):
    client = mqtt.Client("P1")  # create new instance
    #client._client_id="lens_2q4aSVgeMiSkFw6fpTUbEJx7G0B"
    client.username_pw_set(username, password)
    client.connect(broker_address,port=1883,keepalive=120)  # connect to broker
    def on_message(client, userdata,message):
      p.produce(out_event,message.payload.decode("utf-8"))
    print("connected")
    client.subscribe(topic=topic,qos=0)
    client.on_message = on_message

    client.loop_forever()
    # return client
```

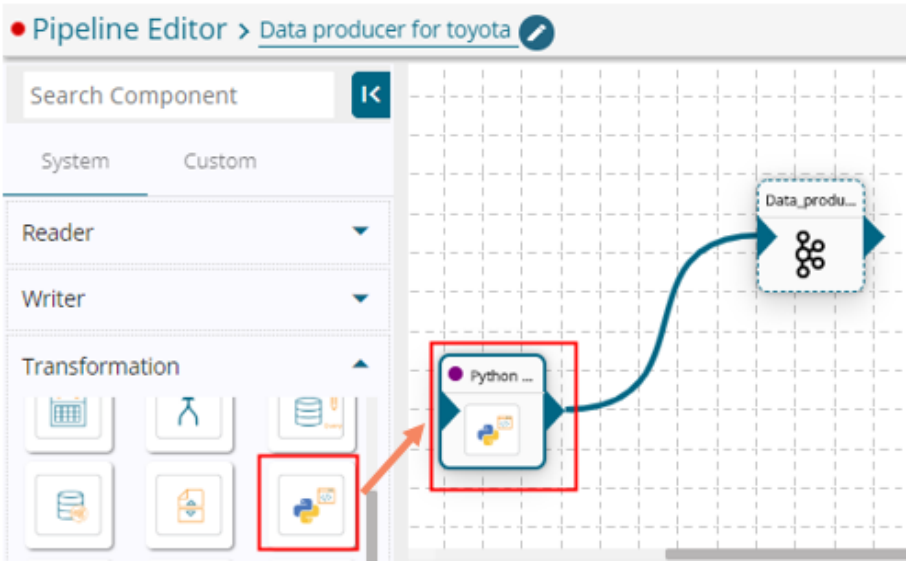
Corresponding File:

sampleMQTT.py

11.2. Configuring a Data Pipeline According to the Script

i) Navigate to the Transformation section of the component pallet.

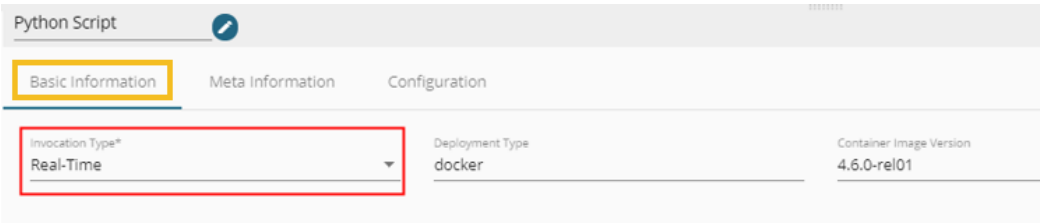
ii) Select and drag the Python Script component to the canvas (Workflow Editor).



iii) Configure the Python Script component.

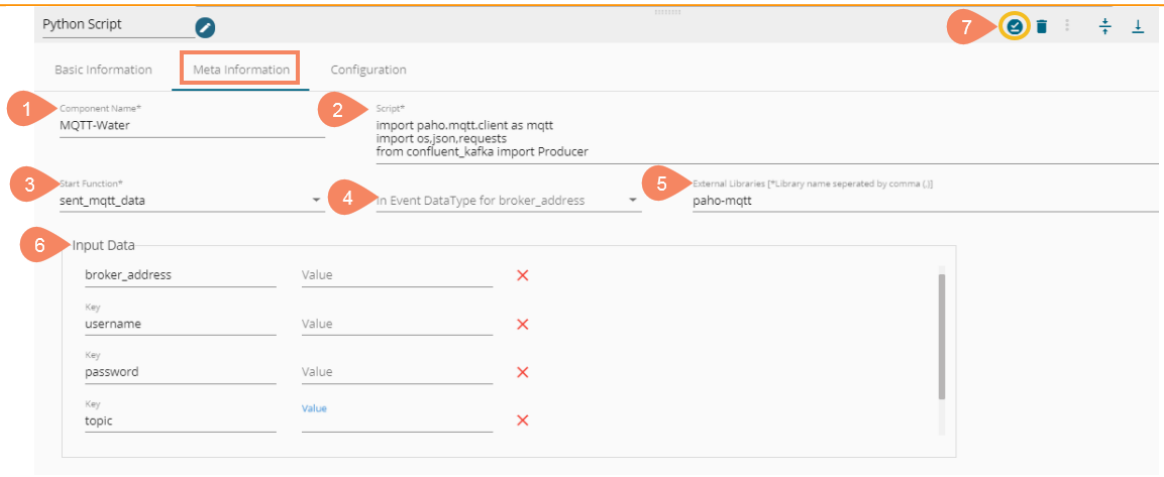
iv) The **Basic Information** tab opens by default.

v) Select the '**Real-Time**' as invocation type so it can keep listening to the MQTT topic in real-time.



vi) Open the Meta Information tab and configure as displayed in the below image:

1. Component Name: Provide a name for the component.
2. Script: Insert the custom python script in this space.
3. Start Function: Activate the function to run the Python script.
4. In Event Data Type: Select a data frame or list.
5. External Libraries: Provide names of the libraries separated by a comma
6. Input Date: Fill in the values of the input parameters of the start function.
7. Click the 'Save' option for the component.



Note : List of already installed libraries are as given below:

certifi==2019.6.16
chardet==3.0.4
confluent-kafka==1.1.0
idna==2.8
numpy==1.17.1
pandas==0.25.1
PyMySQL==0.9.3
python-dateutil==2.8.0
pytz==2019.2
requests==2.22.0
six==1.12.0
urllib3==1.25.3

11.2.1. Making a POST request along with the data

Sample code:

```
# In[]: arguments for OCR
import requests
import time


def main(df):
    try:
        print("Process Started")

        # In[]: Creating data to post

        url = 'API URL'
        response = requests.post(url,data=data_to send)

        # In[]: Reading the returned information
        res = response.json()
        # results_text = res['lttsText']
        print(res)
        return [res]
```

Corresponding File:

SampleApiPostReq.py

12. Signing Out

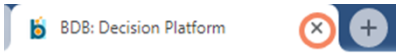
The users can Sign-out from the Data Pipeline tab at any given stage, but preferable is that the users should complete all the tasks they wish to perform and save them before closing the tab or signing out from the Platform.

The Signing Out process for the Data Pipeline has two steps:

1. Closing the BDB Data Pipeline

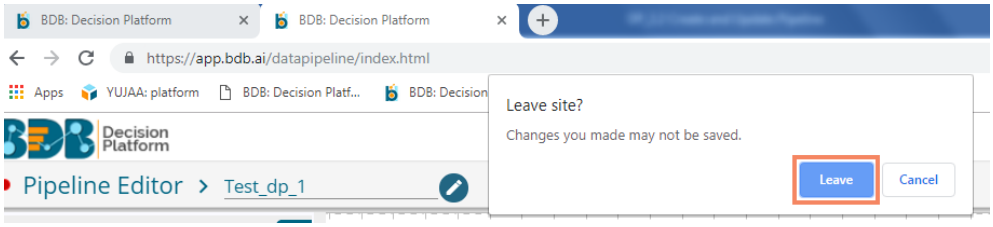
Once you have completed the Data Pipeline tasks, save your work, and close the Data Pipeline tab.

i) Click the **'Close'** button (the 'X' on the right edge) from the Data Pipeline tab.



ii) A window appears to confirm the action of closing the BDB Pipeline.

iii) Click the '**Leave**' option to close the selected pipeline.

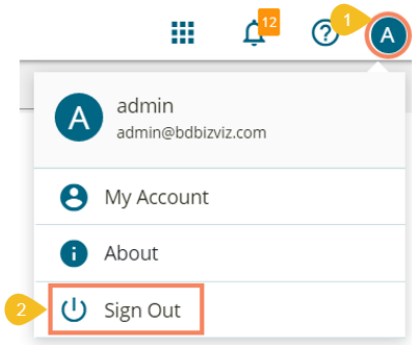


2. Sign Out from the BDB Platform

i) After closing the Data Pipeline plugin, the users get redirected to the Platform homepage.

ii) Click the '**User Profile**' icon on the Platform homepage.

iii) Click the '**Sign Out**' option.



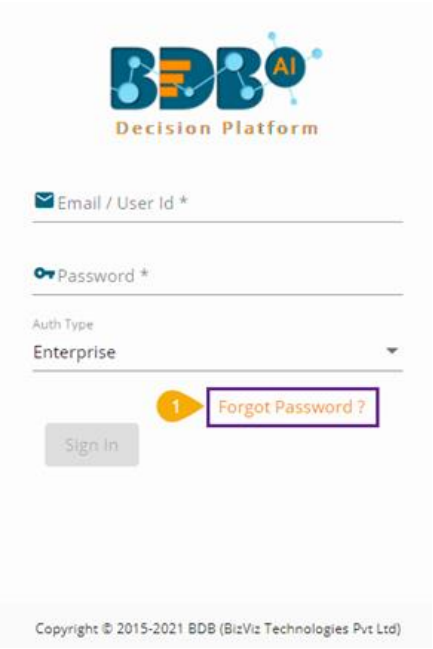
iv) The user successfully signs off from the **BDB Platform**.

Note: Clicking on the '**Sign Out**' option redirects the user back to the login page of the BDB platform.

12.1. Forgot Password Option

The users are provided with a choice to change the password on the Login page of the platform.

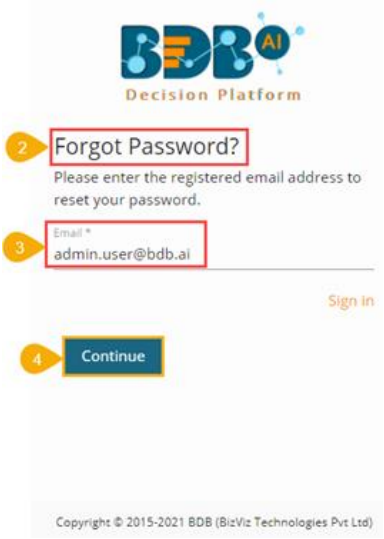
i) Click the '**Forgot Password?**' option from the Sign In page.



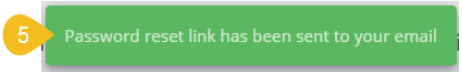
ii) The '**Forgot Password?**' page opens.

iii) Provide the email id that is registered with BDB to send the reset password link.

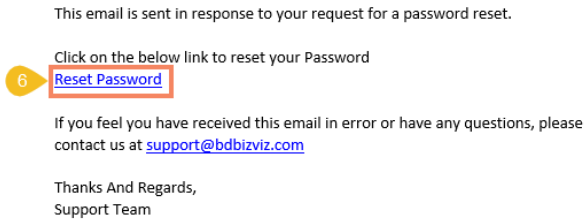
iv) Click the '**Continue**' option.



v) The user may be redirected to select a space in case of multiple spaces under one server link (The user needs to select a space and click the '**Continue**' option once again). If a user does not have multiple spaces, a message appears to notify the user that the password reset link (The users receive the reset link via their registered email.)

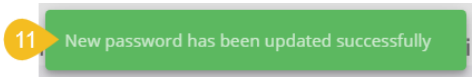


vi) Click the link from your registered email.



vii) The user gets redirected to the '**Reset Password**' page to set a new password.
viii) Set a new password.
ix) Confirm the newly set password.
x) Click the '**Continue**' option.

xi) The password for the selected BDB account gets reset and a message appears to inform the user.

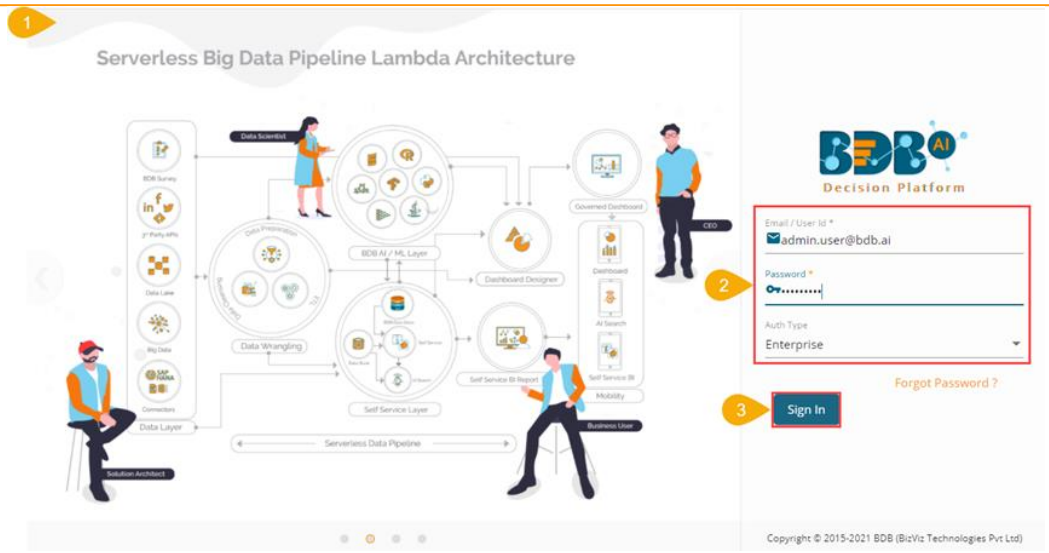


Note: The user gets redirected back to the Sign In page after successfully resetting the password.

12.2. Force Login

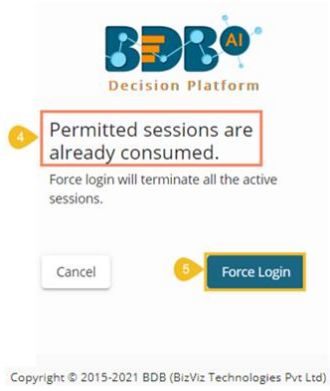
The '**Force Login**' functionality has been introduced to control the number of active sessions up to three. The users can access only 3 sessions at a time when they try to access the 4th session, a warning message displays to inform that the user has consumed the permitted sessions and, a click on the '**Force Login**' would kill all those active sessions.

i) Navigate to the BDB Platform Login page.
ii) Enter the valid credentials to log in.
iii) Click the '**Sign In**' option.



iv) The user gets the following message if the permitted active sessions (3 sessions at a time) are consumed.

v) Click the '**Force Login**' option.



vi) A warning message appears the currently active sessions get killed, and the user gets redirected to the BDB Platform Sign In page.

vii) The user needs to provide valid credentials once again and click the '**Continue**' option to access the platform.