Hybris API services

Technical Design Document

# DOCUMENT REVISION HISTORY

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Release Date** | **Revised by** | **Comments/ Indicate Sections Revised** |
| DRAFT | 18/07/2019 | Connector Development Team | Draft version of the Technical Design Document |

Table of Contents

[DOCUMENT REVISION HISTORY 2](#_gjdgxs)

[A.](#_30j0zll) [INTRODUCTION 4](#_30j0zll)

[**1.**](#_1fob9te) **Purpose** 4

[**2.**](#_3znysh7) **Background** 4

[3](#_2jxsxqh) [**HYBRIS** **API Service** 4](#_2jxsxqh)

[**4**](#_2et92p0) **Overview** 5

[B.](#_3dy6vkm) MAPPING – BUSINESS FUNCTIONS CALLS to TECHNICAL OPERATIONS 5

[**1.**](#_1t3h5sf) **List of operations supported by the HYBRIS Connector:** 6

[C.](#_4d34og8) Configurations for Hybris Connector 17

[**1.**](#_35nkun2) **Authentication** [17](#_4d34og8)

D. To connect in design center 18

E. Use cases - Connectivity with Salesforce 21

**APPENDIX A** 29

1.Install Hybris Connector In anypoint studio 29

2.Configure Hybris Connector In anypoint studio 31

1. Authentication 30

2. Configure in Anypoint Studio 31

3. About Connector Namespace and Schema 31

# **INTRODUCTION**

## **Purpose**

This specification document outlines the instructions for Participating Organizations (PO) to access the API service offered in the hybris systems. This document also serves as a reference for the service request parameters, service responses and other information pertinent to the dissemination of Lending information through the API. While there are similar parameters and commands to add, update, and delete services from a Hybris API URL.

## **Background**

Hybris is an e-commerce product platform that is used to address a family of products involving **Customer Experience** and **Management**.

SAP Hybris provides product for e-Commerce, Marketing, Sales and Service, Revenue and cross-functional solutions and they are designed to help organizations to create valuable interactions with their customers and support for different industry types.

Hybris connector enables retailers to integrate their existing applications with Hybris by leveraging the Omni Channel Commerce V2 REST APIs.

## **HYBRIS** **API Service**

HYBRIS APIs provide access to various services for the following workflows:

* + **Products:** This service provides details on total number of product stock levels, product details, reference, list of products and additional data.
  + **Promotions:** This service provides details on list of promotion and specific promotion based on code.
  + **Customer Groups:** This service provides details on all sub groups of customer group, allow creation of new customer group, assign user to customer group, fetch specific group and delete user from customer group. To try out the methods of the Customer Groups controller, you must authorize a user who belongs to the “customer manager group”.
  + **Save Cart:** This service clones a cart, saves cart, restore a saved cart and retrieve saved cart.
  + **Miscs:** This service retrieves list of supported payment card types, list of available currencies, list of supported countries, available languages and all localized titles.
  + **Users:** This service allows all user specific operations like registering, fetching, updating, deleting customer profile.
  + **Catalogs:** This service retrieves a list of catalogs, catalog version and information on category in catalog version.
  + **Export:** This service provides list of product exports.
  + **Vouchers:** This service retrieves a voucher based on code.
  + **Stores:** This service fetches list of store locations and header with number of store locations.
  + **Feeds:** This API gets a list of orders with status updates.
  + **Orders:** This service fetches order details, order history, total number of orders, post and get an order.
  + **Carts:** This API defines all cart related operations like creation and restoration of carts.
  + **Forgotten Passwords:** This API generates a token to restore a customer's forgotten password.

## **Overview**

A Web Service is a method of communication between two electronic devices over a network. Web services allow organizations to communicate data without close knowledge of each other's IT systems. These web services are often called, API, or Application Programming Interface.

A high-level diagram of the architecture is shown in Figure 1.

|  |
| --- |
|  |

Figure 1: Mule Hybris Connector Integration Architecture

# MAPPING – BUSINESS FUNCTIONS CALLS to TECHNICAL OPERATIONS

These operations are available for consumption in a manner which is suitable to the business needs of the Mule application developer. The first column represents the business functions. The next column denotes the corresponding operations.

## **List of operations supported by the HYBRIS Connector:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Sr.No** | **Business**  **Function** | **Operations** | | | **Description** | | **Inbound** | **Outbound** |
| **Products** | | | | | | | | |
| 1 | Get Express Update Products | GET Request | | | Get products added to the express update feed. | | Required field is timestamp. This parameter should be in ISO-8601 format. | Return an array of ProductExpressUpdateElementWsDTO in response. |
| 2 | Search Products | GET Request | | | Get a list of products and additional data, such as available facets, available sorting, and pagination options. | | Required field is baseSiteId. | Return list of products in response. |
| 3 | Get Number of Products | HEAD request | | | Get the details of header with total number of products. | | Required field is baseSiteId. | In the response header, the "x-total-count" indicates the total number of products satisfying a query. |
| 4 | Get Suggestions | GET request | | | Get the list of all available suggestions | | Required field is term, max and baseSiteId. | Returns a list of all available suggestions related to a given term and limits the results to a specific value of the max parameter. |
| 5 | Get Product Details by Product Code | GET request | | | Returns details of a single product according to a product code. | | Required fields are productCode and baseSiteId | Return the response with “success” or “error” body containing a message about success or failure. |
| 6 | Get Products Reference | GET request | | | Returns references for a product with a given product code. Reference type specifies which references to return. | | Required fields are productCode, referenceType and baseSiteId | Returns references for a product in response. |
| 7 | Get Product Review Detail | GET Request | | | Returns the reviews for a product with a given product code. | | Required fields are productCode and baseSiteId | Returns reviews for a product in response. |
| 8 | Create a New Customer Review | POST request | | | Creates a new customer review as an anonymous user. | | Required fields are productCode,review and baseSiteId | Returns response in response. |
| 9 | Get Product's Stock Level | GET Request | | | Returns a product's stock levels sorted by distance from the specified location, which is provided using the free-text "location" parameter, or by using the longitude and latitude parameters. The following two sets of parameters are available: location (required), currentPage (optional), pageSize (optional); or longitude (required), latitude (required), currentPage (optional), pageSize(optional). | | Required fields are productCode and baseSiteId | Returns products stock in response. |
| 10 | Get Product's Stock Level Count | HEAD request | | | In the response header, the "x-total-count" indicates the total number of a product's stock levels. The following two sets of parameters are available: location (required); or longitude (required), and latitude (required). | | Required fields are productCode and baseSiteId | Return the response with “success” or “error” body containing a message about success or failure. |
| 11 | Get Stock Level for a Particular Store | GET Request | | | Returns a product's stock level for a particular store (in other words, for a particular point of sale). | | Required fields are productCode , storeName and baseSiteId | Returns product stock level for a store in response. |
| **Promotions** | | | | | | | | |
| 12 | Get Promotions Details | GET Request | | | Returns promotions defined for a current base site. | | Required fields are type and baseSiteId.  Requests pertaining to promotions have been developed for the previous version of promotions and vouchers and therefore some of them are currently not compatible with the new promotion engine. | Returns list of promotions response in response. |
| 13 | Get Promotion Details by Promotion Code | GET Request | | | Returns details of a single promotion specified by a promotion code. | | Required fields is baseSiteId.  Requests pertaining to promotions have been developed for the previous version of promotions and vouchers and therefore some of them are currently not compatible with the new promotion engine. | Returns a promotion based on code in response. |
| **Customer Groups** | | | | | | | | |
| 14 | Get User Customer Groups | GET request | | | Returns all customer groups that are direct subgroups of a customergroup. | | Required fields is baseSiteId. | Returns all customer groups in response. |
| 15 | Create New Customer Group | POST request | | | Creates a new customer group that is a direct subgroup of a customergroup. | | Required fields are userGroup and baseSiteId. | Return the response with “created” or “error” code. |
| 16 | Get Customer Group by Group Id | GET request | | | Returns a customer group with a specific groupId. | | Required fields are groupId and baseSiteId. | Returns a customer group in response. |
| 17 | Remove User from Usergroup | DELETE request | | | Removes user from a customer group. | | Required fields are groupId, userId and baseSiteId. | Return the response with “OK” or “No Content” code. |
| **Save Cart** | | | | | | | | |
| 18 | Get Saved Cart by Cart Id | GET request | | | Returns a saved cart for an authenticated user. The cart is identified using the "cartId" parameter. | | Required fields are cartId, baseSiteId and userId. | Returns saved cart results object in response. |
| **Miscs** | | | | | | | | |
| 19 | Get Card Types | | GET request | Lists supported payment card types. | | Required field is baseSiteId | | Returns a list of card payments. |
| 20 | Get Currencies | | GET request | Lists all available currencies (all usable currencies for the current store). If the list of currencies for a base store is empty, a list of all currencies available in the system is returned. | | Required field is baseSiteId | | Returns list of available currencies in response. |
| 21 | Get Delivery Countries | | GET request | Lists all supported delivery countries for the current store. The list is sorted alphabetically. | | Required field is baseSiteId | | Returns list of supported countries in response. |
| 22 | Get Languages | | GET request | Lists all available languages (all languages used for a particular store). If the list of languages for a base store is empty, a list of all languages available in the system will be returned. | | Required field is baseSiteId | | Returns list of available languages in response. |
| 23 | Get Titles | | GET request | Lists all localized titles. | | Required field is baseSiteId | | Returns list of all localized titles. |
| **Users** | | | | | | | | |
| 24 | Create User | POST request | | | Registers a customer. | | Required fields are baseSiteId and user. | Returns a user object in response |
| 25 | Get user | GET request | | | Returns customer profile. | | Required fields are baseSiteId and userId. | Returns customer profile in response. |
| 26 | Update User | PUT request | | | Updates customer profile. | | Required fields are baseSiteId,user and userId. | Return “Ok” or “created” code in response. |
| 27 | Get User Addresses by User Id | GET request | | | Returns customer's addresses. | | Required fields are baseSiteId and userId. | Returns customer's addresses. |
| 28 | Create User Address by User Id | POST request | | | Creates a new address. | | Required fields are address, baseSiteId and userId. | Returns an object of new created address in response. |
| 29 | Address Verification by User Id | POST request | | | Verifies address. | | Required fields are address, baseSiteId and userId. | Returns response in response. |
| 30 | Get User Address by Address Id | GET request | | | Returns detailed information about address with a given id. | | Required fields are addressId, baseSiteId and userId. | Returns address in response. |
| 31 | Update User Address | PUT request | | | Updates the address. | | Required fields are address, addressId, baseSiteId and userId. | Returns OK, created and other error codes in response. |
| 32 | Remove User Address by Address Id | DELETE request | | | Removes customer's address. | | Required fields are addressId, baseSiteId and userId. | Returns OK, No content and other error codes in response. |
| 33 | Get User Customer Groups | GET request | | | Returns all customer groups of a customer. | | Required fields are baseSiteId and userId. | Returns customer groups of customers. |
| 34 | Update User Login by User Id | PUT request | | | Changes a customer's login name. Requires the customer's current password. | | Required fields are newLogin,password,baseSiteId and userId. | Returns OK, created and other error codes in response. |
| 35 | Change User Password | PUT request | | | Changes customer's password | | Required fields are new,baseSiteId and userId. | Returns created, accepted and other error codes in response. |
| 36 | Get payment Details | GET request | | | Return customer's credit card payment details list. | | Required fields are saved,baseSiteId and userId. | Returns list of credit card payment. |
| 37 | Get Payment Details by Payment Id | GET request | | | Returns a customer's credit card payment details for the specified paymentDetailsId. | | Required fields are paymentDetailsId,baseSiteId and userId. | Returns credit card payment details for payment Id in. |
| 38 | Update Payment Details by Payment Id | PUT request | | | Updates existing customer's credit card payment info based on the payment info ID. | | Required fields are paymentDetailsId,paymentDetails,baseSiteId and userId. | Returns OK, created and other error codes in response. |
| 39 | Remove Payment Details | DELETE request | | | Removes a customer's credit card payment details based on a specified paymentDetailsId. | | Required fields are paymentDetailsId,baseSiteId and userId. | Returns OK, no content and other error codes in response. |
| **Catalogs** | | | | | | | | |
| 40 | Get Catalogs | GET request | | | Returns all catalogs with versions defined for the base store. | | Required field is baseSiteId. | Returns a list of all catalogs. |
| 41 | Get Catalog by Catalog Id | GET request | | | Returns information about a catalog based on its ID, along with the versions defined for the current base store. | | Required fields are catalogId and baseSiteId. | Returns a catalog based on catalogID. |
| 42 | Get Catalog by Version Id | GET request | | | Returns information about the catalog version that exists for the current base store. | | Required fields are catalogId,catalogVerionId and baseSiteId. | Returns catalog information. |
| **Export** | | | | | | | | |
| 43 | Get Export Products | GET request | | | Used for product export. Depending on the timestamp parameter, it can return all products or only products modified after the given time. | | Required field is baseSiteId. | Returns list of products in response. |
| **Vouchers** | | | | | | | | |
| 44 | Get Voucher Details | GET request | | | Returns details of a single voucher that is specified by its voucher identification code. | | Required fields are baseSiteId and code. | Returns details of voucher in response. |
| **Stores** | | | | | | | | |
| 45 | Store Finder | GET request | | | Lists all store locations that are near the location specified in a query or based on latitude and longitude | | Required field is baseSiteId. | Returns a list of store locations in response. |
| 46 | Store Finder Count | HEAD request | | | In the response header, the "x-total-count" indicates the number of all store locations that are near the location specified in a query, or based on latitude and longitude. | | Required field is baseSiteId. | In the response header, the "x-total-count" indicates the number of all store locations. |
| 47 | Get Store Location by Store Id | GET request | | | Returns store location based on its unique name. | | Required fields are baseSiteId and storeId. | Returns stores information in response. |
| **Feeds** | | | | | | | | |
| 48 | Get Changed Status Orders | GET request | | | Returns the orders that have changed status. Returns only the elements from the current baseSite that have been updated after the provided timestamp. | | Required fields are baseSiteId and timestamp. | Returns a list of orders in response. |
| **Orders** | | | | | | | | |
| 49 | Get Order by Code | GET request | | | Returns details of a specific order based on the order GUID (Globally Unique Identifier) or the order CODE. The response contains detailed order information. | | Required fields are baseSiteId and code. | Returns a object of order in response. |
| 50 | Get order History of User | GET request | | | Returns order history data for all orders placed by a specified user for a specified base store. The response can display the results across multiple pages, if required. | | Required fields are baseSiteId and userId. | Returns order history in response. |
| 51 | Get Order Count By User Id | HEAD request | | | In the response header, the "x-total-count" indicates the total number of orders placed by a specified user for a specified base store. | | Required fields are baseSiteId and userId. | Returns ok, no content codes in response . |
| **Carts** | | | | | | | | |
| 52 | Get Cart | GET request | | | Lists all customer carts. | | Required fields are baseSiteId and userId. | Returns list of all customer carts in response. |
| 53 | Create Cart | POST request | | | Creates a new cart or restores an anonymous cart as a user's cart (if an old Cart Id is given in the request). | | Required fields are baseSiteId and userId. | Returns cart information in response. |
| 54 | Remove User's Cart by Cart Id | DELETE request | | | Deletes a cart with a given cart id. | | Required fields are cartId, baseSiteId and userId. | Returns OK, no content and other error codes in response. |
| 55 | Create Delivery Address of Cart | POST request | | | Creates an address and assigns it to the cart as the delivery address. | | Required fields are address, cartId, baseSiteId and userId. | Returns address object in response. |
| 56 | Set Delivery Address for Cart | PUT request | | | Sets a delivery address for the cart. The address country must be placed among the delivery countries of the current base store. | | Required fields are addressId, cartId, baseSiteId and userId. | Returns OK, created and other error codes in response. |
| 57 | Remove Delivery Address for Cart | DELETE request | | | Removes the delivery mode from the cart. | | Required fields are cartId, baseSiteId and userId. | Returns Ok, No content and other error codes in response. |
| 58 | Get Selected Delivery Mode of Cart | GET request | | | Returns all delivery modes supported for the current base store and cart delivery address. A delivery address must be set for the cart, otherwise an empty list will be returned. | | Required fields are cartId, baseSiteId and userId. | Returns delivery modes object in response |
| 59 | Assign Email to Cart | PUT request | | | Assigns an email to the cart. This step is required to make a guest checkout. | | Required fields are email, cartId, baseSiteId and userId. | Returns Ok, created and other error codes in response. |
| 60 | Get Cart Order Entries | GET request | | | Returns cart entries. | | Required fields are cartId, baseSiteId and userId. | Returns a list of cart entries in response. |
| 61 | Get Cart Entry Details by Entry Number | GET request | | | Returns the details of the cart entries. | | Required fields are entryNumber, cartId, baseSiteId and userId. | Returns cart entries in response. |
| 62 | Remove cart entry | DELETE request | | | Deletes cart entry. | | Required fields are entryNumber, cartId, baseSiteId and userId. | Returns ok, no content and other error codes in response. |
| 63 | Create Payment Details for Cart | POST request | | | Defines the details of a new credit card, and assigns this payment option to the cart. | | Required fields are paymentDetails, cartId, baseSiteId and userId. | Returns payment details object in response. |
| 64 | Set Payment Details for Cart | PUT request | | | Sets credit card payment details for the cart. | | Required fields are paymentDetailsId, cartId, baseSiteId and userId. | Returns ok, created and other error codes in response. |
| 65 | Get Cart Promotions | GET request | | | Returns information about the promotions applied on the cart. | | Required fields are cartId, baseSiteId and userId. | Returns promotions object in response. |
| 66 | Get Cart Vouchers | GET request | | | Returns a list of vouchers applied to the cart. | | Required fields are cartId, baseSiteId and userId. | Returns list of vouchers in response. |
| 67 | Apply Voucher to Cart | POST request | | | Applies a voucher based on the voucherId defined for the cart. | | Required fields are voucherId, cartId,baseSiteId and userId. | Returns ok, created and other error codes in response. |
| 68 | Remove Voucher from Cart | DELETE request | | | Removes a voucher based on the voucherId defined for the current cart. | | Required fields are voucherId, cartId,baseSiteId and userId. | Returns ok, no content and other error codes in response. |
| **Forgotten Passwords** | | | | | | | | |
| 69 | Create Forgotten Password Token | POST request | | | Generates a token to restore a customer's forgotten password. | | Required fields are baseSiteId and userId. | Returns created, accepted and other error codes in response. |
| **Others** | | | | | | | | |
| 70 | Custom Operation | HEAD,POST,PUT,GET,DELETE request | | | It’s used for all request as mentioned. | | NA |  |
| 71 | Get Card Types for Trusted Client | GET request | | | Returns list of card types for trusted client | |  | Returns list of card types for trusted client |
| 72 | Store Finder Count | HEAD request | | | Returns count of stores | |  | Returns count of stores |
| 73 | Set Delivery Mode | PUT  request | | | Sets the delivery mode with a given identifier for the cart. | | Required Parameters are deliveryModeId ,baseSiteId ,userId and cartId | Returns Ok,created and other error codes. |
| 74 | Remove Delivery Mode | DELETE request | | | Removes the delivery mode from the cart. | | Required Parameters are baseSiteId ,userId and cartId | Returns Ok,no content and other error codes. |
| 75 | Create Order Entry | POST  request | | | Creates order entry | |  | Returns order information |
| 76 | Get User Order Details by Order Code | GET request | | | Get order of specified order id | | Required parameter is order id | Returns order information of specified order id. |
| 77 | Update Order Entry | PUT request | | | Updates order information | |  | Returns order information |
| 78 | Selected Delivery Modes of Cart | GET request | | | Returns delivery modes | | Required fields are cartId, baseSiteId and userId. | Returns delivery modes object in response |
| 79 | Place Order | POST request | | |  | |  |  |

# CONFIGURATION FOR HYBRIS CONNECTOR

To use Hybris Connector, you will require the credentials for creating your access and refresh token, which will be required for the connector.

## **Authentication**

The Hybris API uses OAuth 2.0 where a token needs to be generated for communicating with server. This token can be of following two types:

1. **Customer Manager connection**: This connection is used to invoke customer manager operations.It requires below credentials:

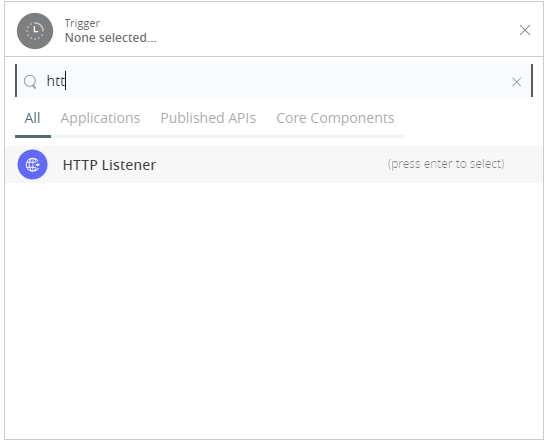
* Uses Password Credentials grant type
* auth url: <url>
* username: <username>
* password: <password>
* clientid: <clientid>
* clientsecret: <clientsecret>

1. **Trusted Client connection**:This connection is used to invoke trusted client operations.It requires below credentials:

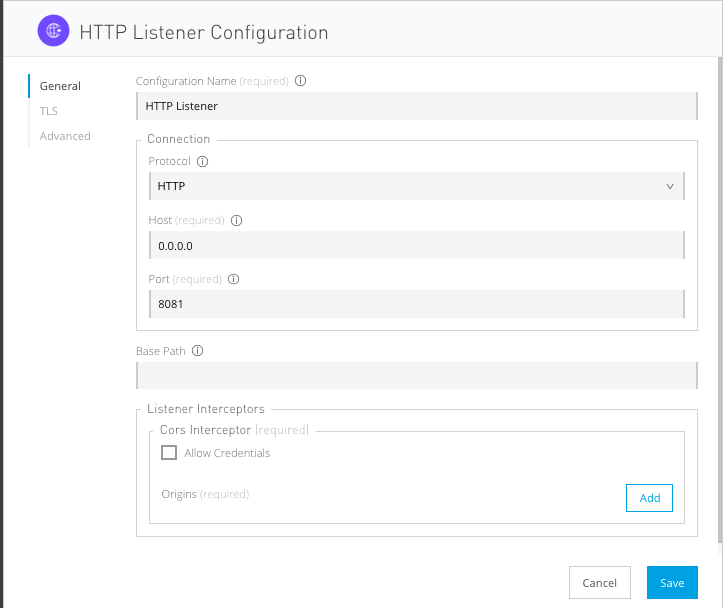
* Uses Client Credentials grant type
* auth url: <url>
* clientid: <clientid>
* clientsecret: <clientsecret>

# TO CONNECT IN DESIGN CENTER

1. In Design Centre, click Create and choose Mule Application.
2. Click a trigger such as an HTTP Listener or the Scheduler trigger.

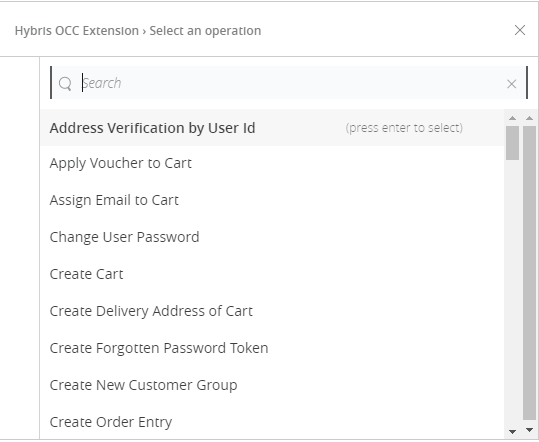


1. To create an HTTP global element for the connector, set these fields:

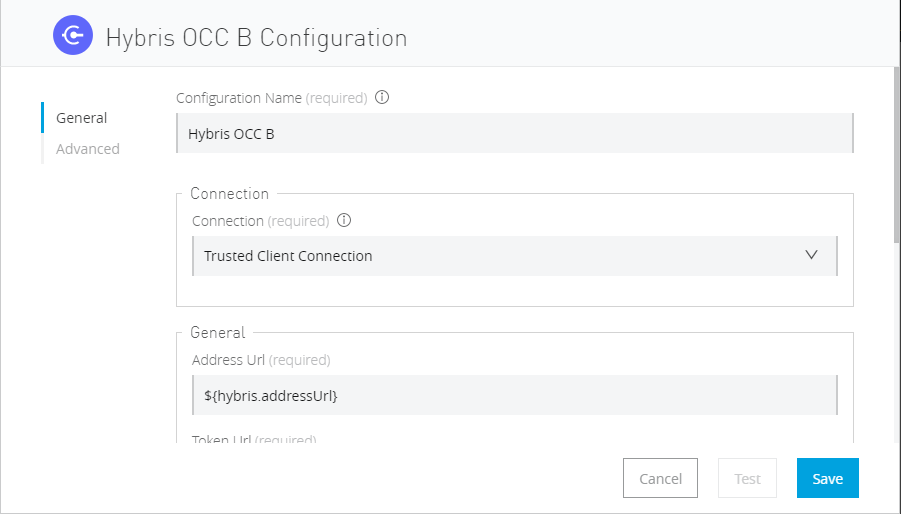


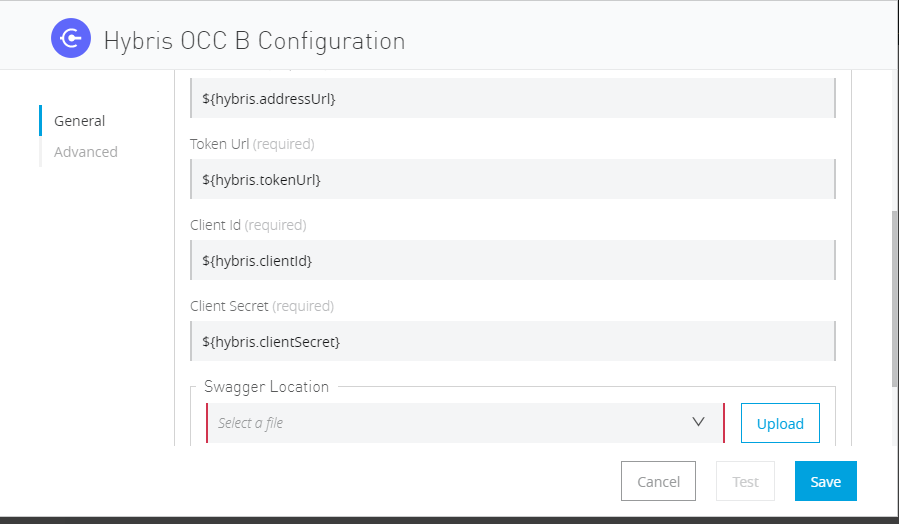
|  |  |
| --- | --- |
| **Field** | **Description** |
| **Protocol** | Protocol selected for the HTTP connector, it can be HTTP or HTTPS (secure). |
| **Host** | IP address where your Mule application listens for requests. |
| **Port** | Port address where your Mule application listens for requests. |
| **Base Path** | Path where your Mule application listens for requests. |

1. Select the plus sign to add a component.
2. Select the Hybris connector as a component.
3. Select an operation:



1. Configure the Global element for the connector:





|  |  |
| --- | --- |
| **Field** | **Description** |
| **Address** | URL to access Hybris API |
| **Token** | Token URL is used to generate access Token for Hybris API |
| **Client Id** | Your username to access your Hybris API. |
| **Client Secret** | Your password to access your Hybris API. |
| **Swagger Location** | Upload swagger file. |

1. Fill the required parameters (if any) for the above selected operation.

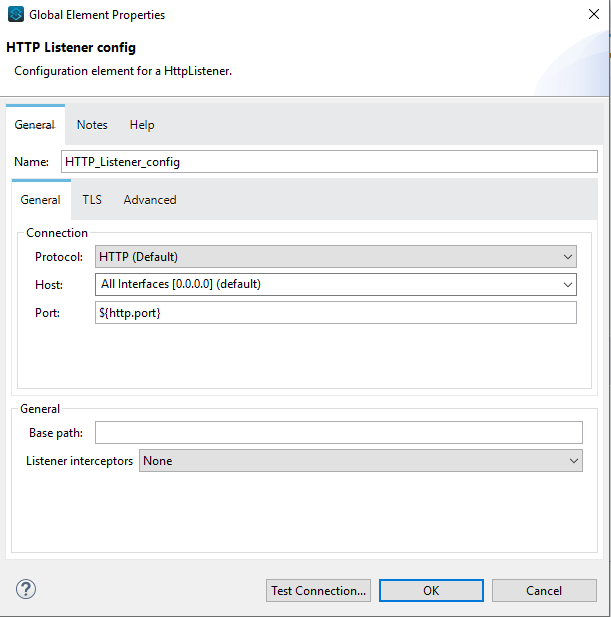
# Use cases – Connectivity with salesforce

This use-case demonstrates the interaction between Hybris and Salesforce systems using Hybris connector. It utilises UPDATE, GET and DELETE operations connectors.

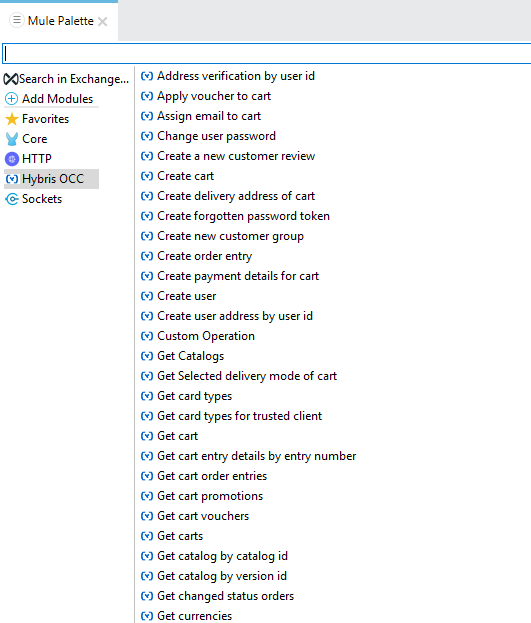
* The flow gets created user using “Get User” operation,then creates cart for the user by using ‘’Create Cart” operation in Hybris systems and also creates account in salesforce.
* Once the Account is created in Salesforce it will fetch the cart for user and update delivery mode in cart.
* Once the cart is updated in hybris system,the cart is deleted from salesforce using Remove User’s Cart by Cart Id operation.

For running this use-case we need following configurations as prerequisites:

1. Drag and drop an HTTP Listener in the canvas.
2. In the Listener properties, give a path you want to use to trigger the listener.
3. Add a new Configuration as follows,



1. Test the connection and click on Okay.
2. Make sure your mule palette has Salesforce and Hybris modules. If you do not have Salesforce module in your palette, go to add module -> Salesforce and drag it to your palette.



1. Now add configurations for Hybris.
2. Go to global-configurations.xml global elements -> create -> Connector Configuration -> Hybris Configuration
3. Add following properties:

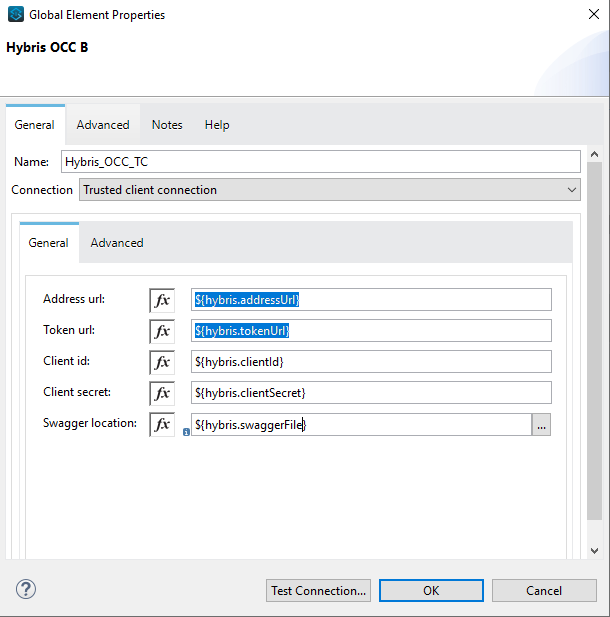
Address Url:

Token Url:

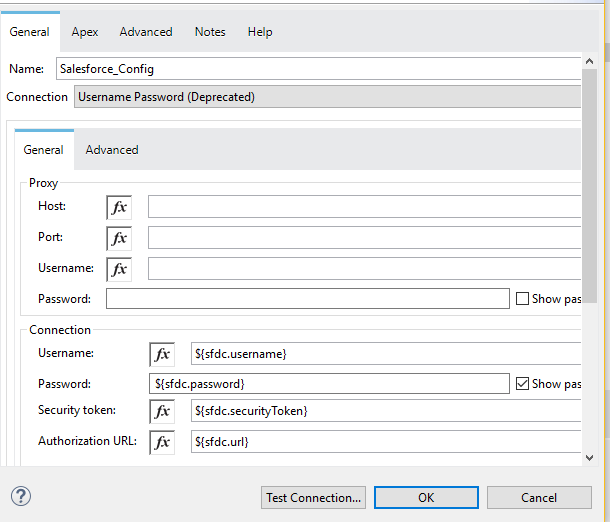
Client Id:

Client secret:

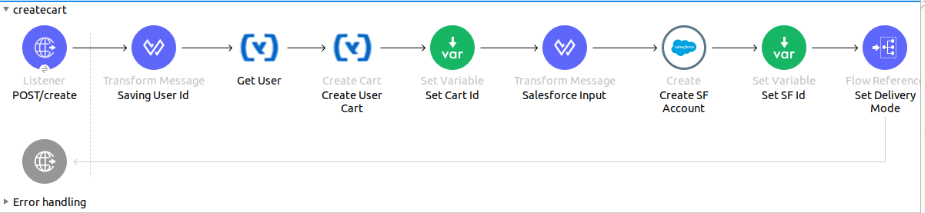
Swagger location:

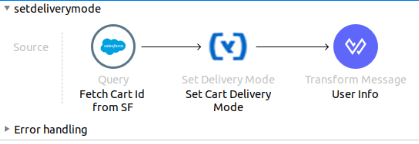


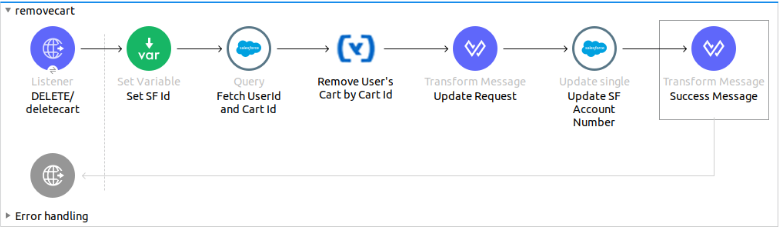
1. Add Salesforce configuration.
2. Go to global-configurations.xml -> global elements -> create -> Connector Configuration -> Salesforce Configuration
3. Add following properties



1. Create a flow with the components displayed in the image below:

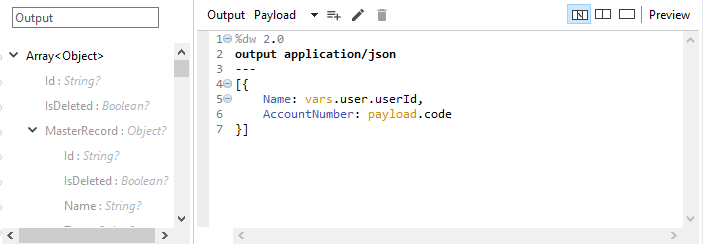




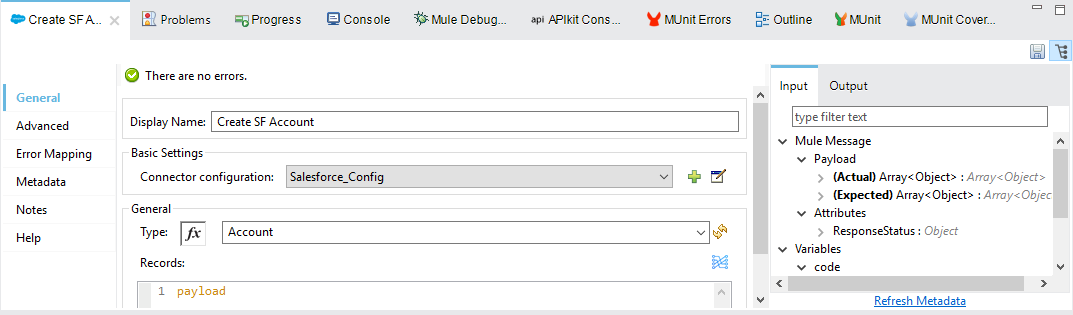


1. Individual mappings for each component are illustrated in below screenshots:

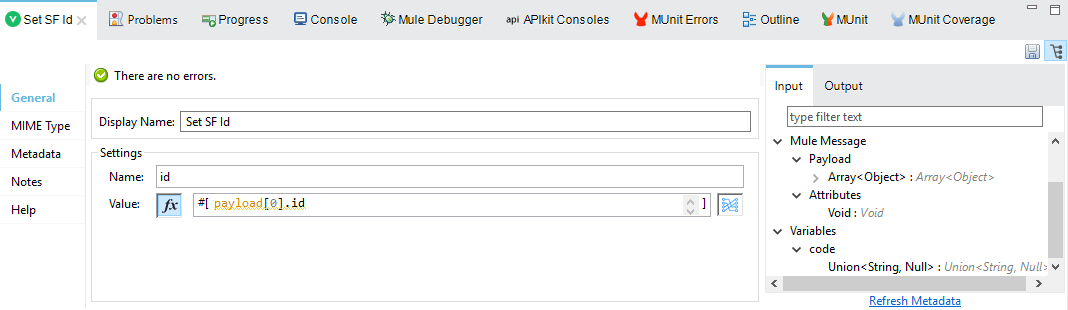
Create account request body:



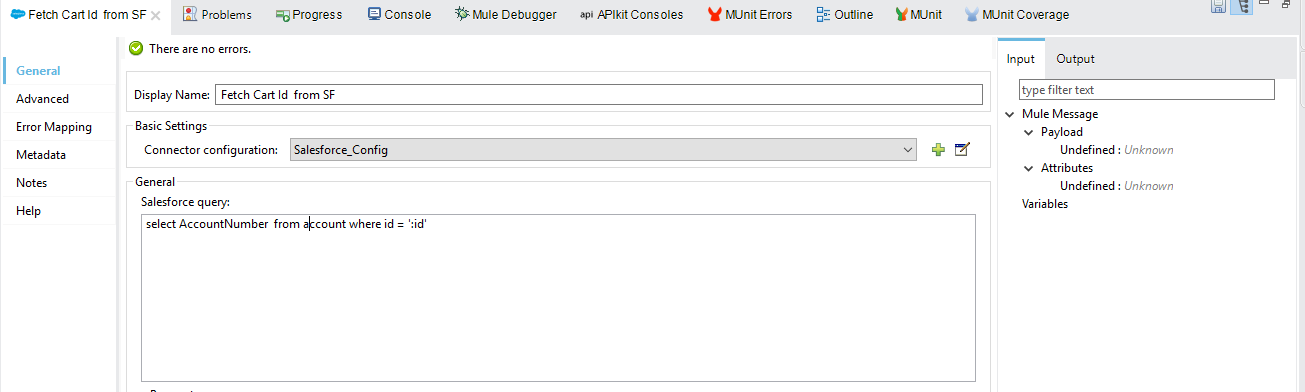
Create account in Salesforce:



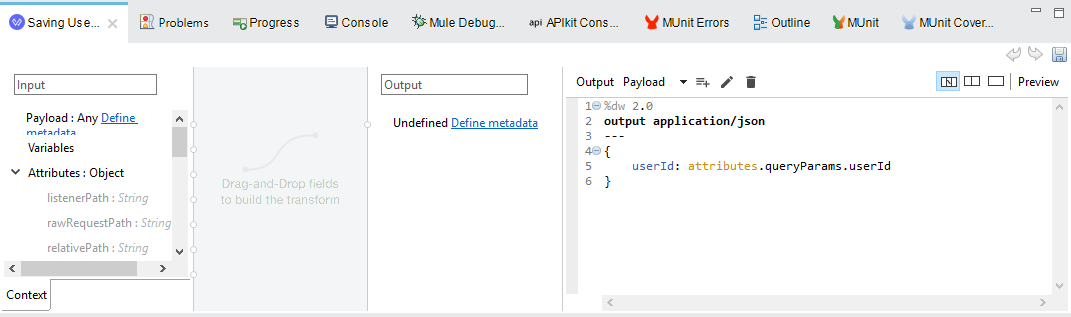
Store Salesforce account id:



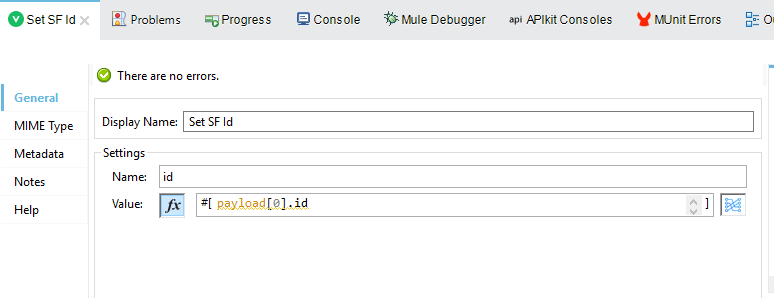
Getting account from SF:



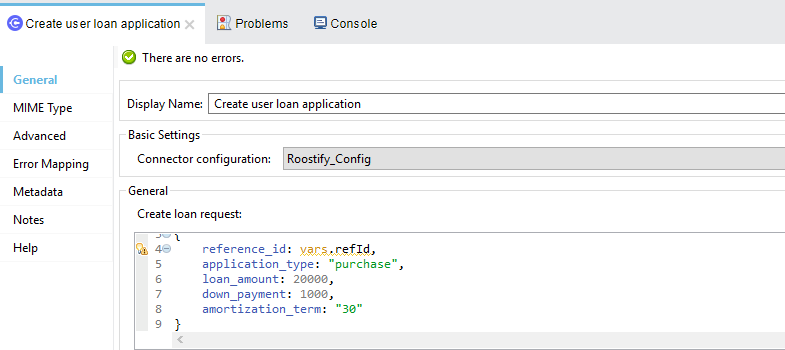
Creating user data:



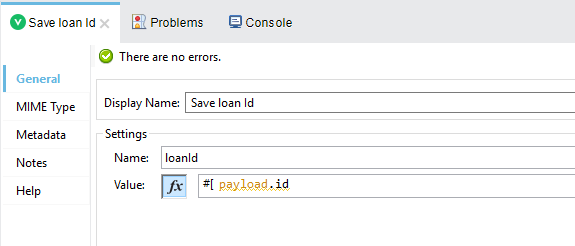
Storing Salesforce Id:



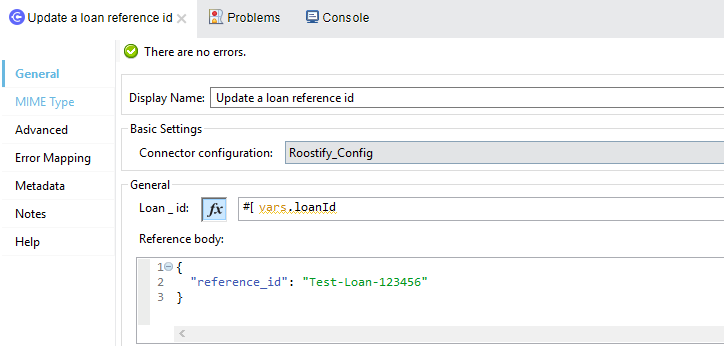
Create Loan application:



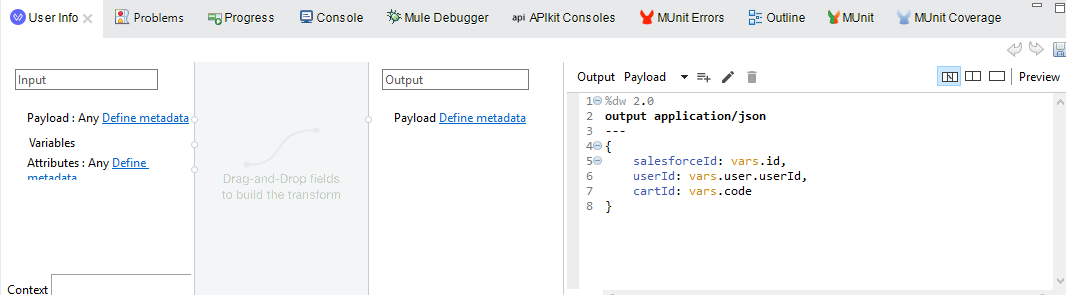
Storing loan application id:



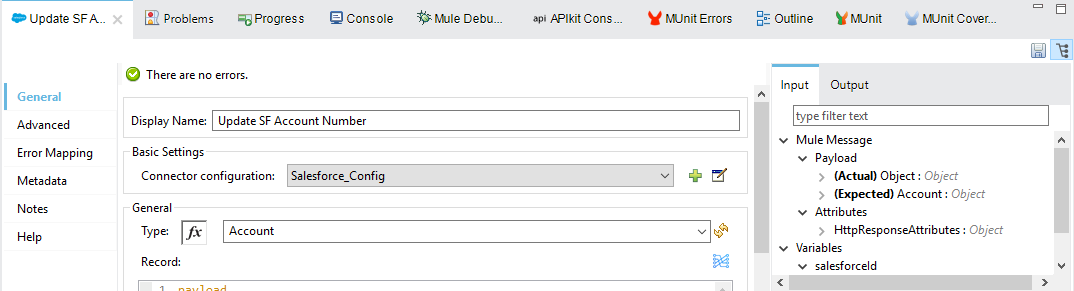
Updating loan reference id:



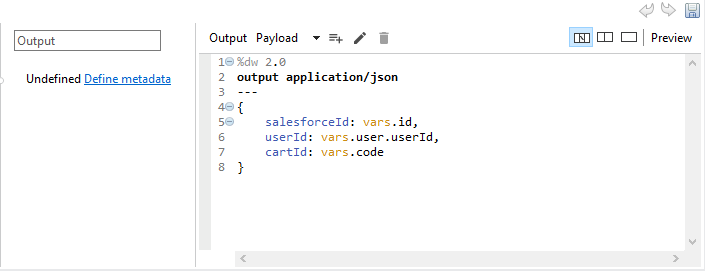
Converting data into SF:



Updating details into SF:



Printing updated response from SF:

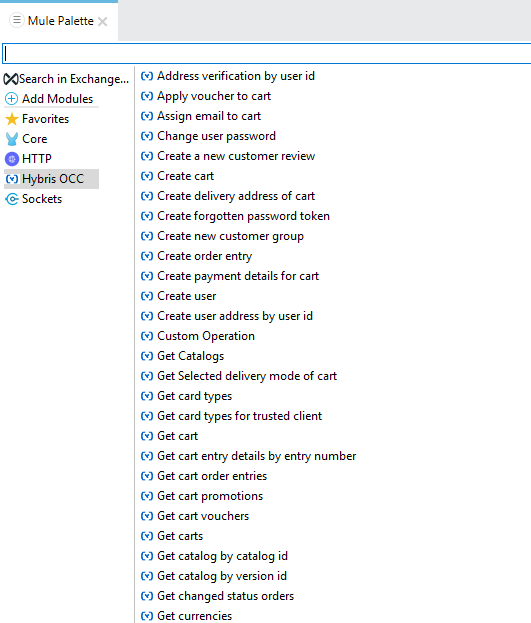


# APPENDIX A

# Install HybRIS Connector in Anypoint Studio

Developers can add the Hybris connector module in Anypoint Studio, by following the steps below:

* Open you Anypoint Studio
* Search for the Hybris Connector in your mule palette



# Configure Hybris Connector In anypoint studio

You will require the credentials for creating your access and refresh token, which will be required for the connector.

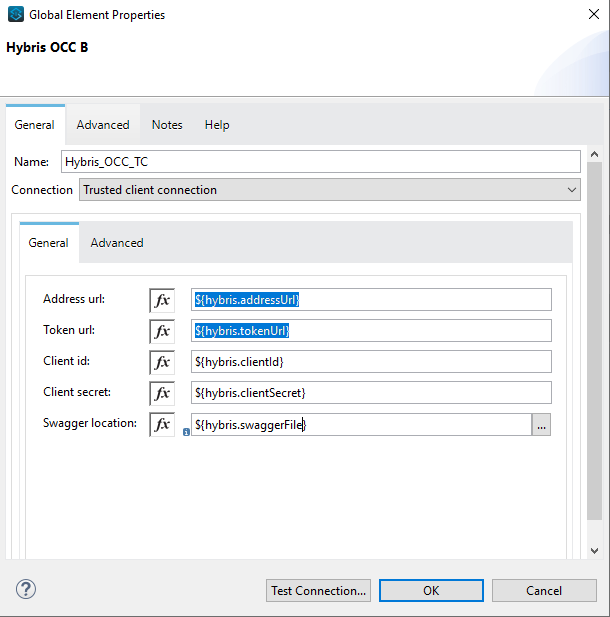
## **Authentication**

Provide following credentials in configuration file

|  |  |
| --- | --- |
| **Field** | **Description** |
| **Address** | URL to access Hybris API |
| **Token** | Token URL is used to generate access Token for Hybris API |
| **Client Id** | Your username to access your Hybris API. |
| **Client Secret** | Your password to access your Hybris API. |
| **Swagger Location** | Upload swagger file. |

## **Configure in Anypoint Studio**

* 1. First thing we need to do to configure the connector, provide above credentials required to access the APIs, as mentioned in this document previously
  2. Once you have your credentials, drag and drop your connector in the mule pallet and make sure you have a listener in the canvas.
  3. In the Global elements, create a new configuration and use one of the following details:
  4. Once these fields are added, you can test the connection



* 1. You don’t have to add any configurations in the XML because, when the connector configurations are created as mentioned in the document below, the xml will automatically get updated as follows:



## **About Connector Namespace and Schema**

When designing your application in Studio, drag and drop the connector in your canvas and the Namespace and schema get populated in the config file as below,

Namespace: [*http://www.mulesoft.org/schema/mule/hybris-occ*](http://www.mulesoft.org/schema/mule/hybris-occ)

Schema Location: [*http://www.mulesoft.org/schema/mule/hybris-occ/current/mule-hybris-occ.xsd*](http://www.mulesoft.org/schema/mule/hybris-occ/current/mule-hybris-occ.xsd)