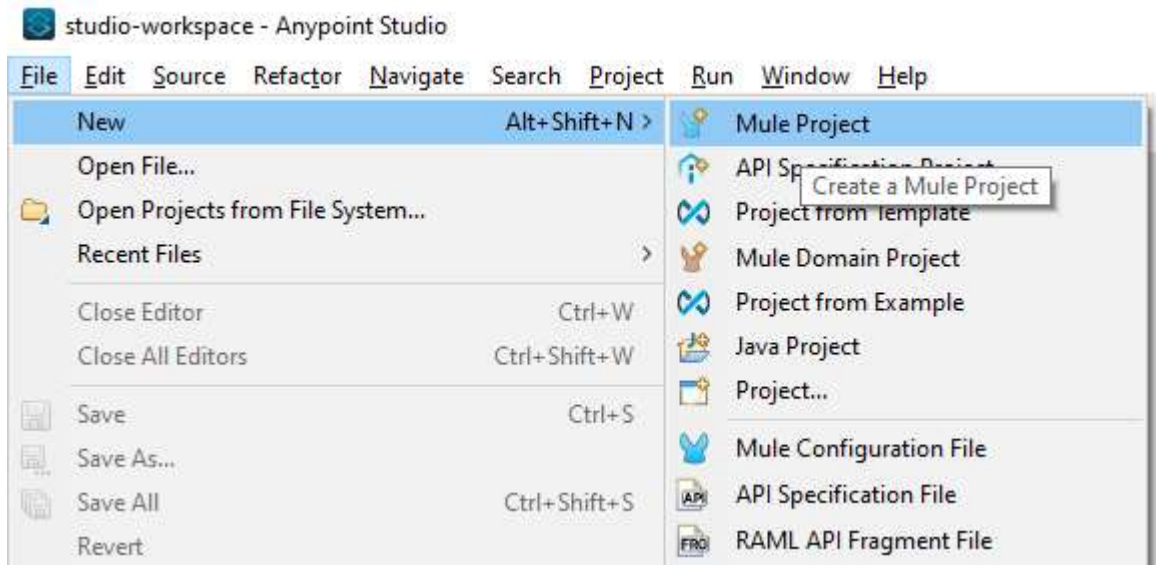
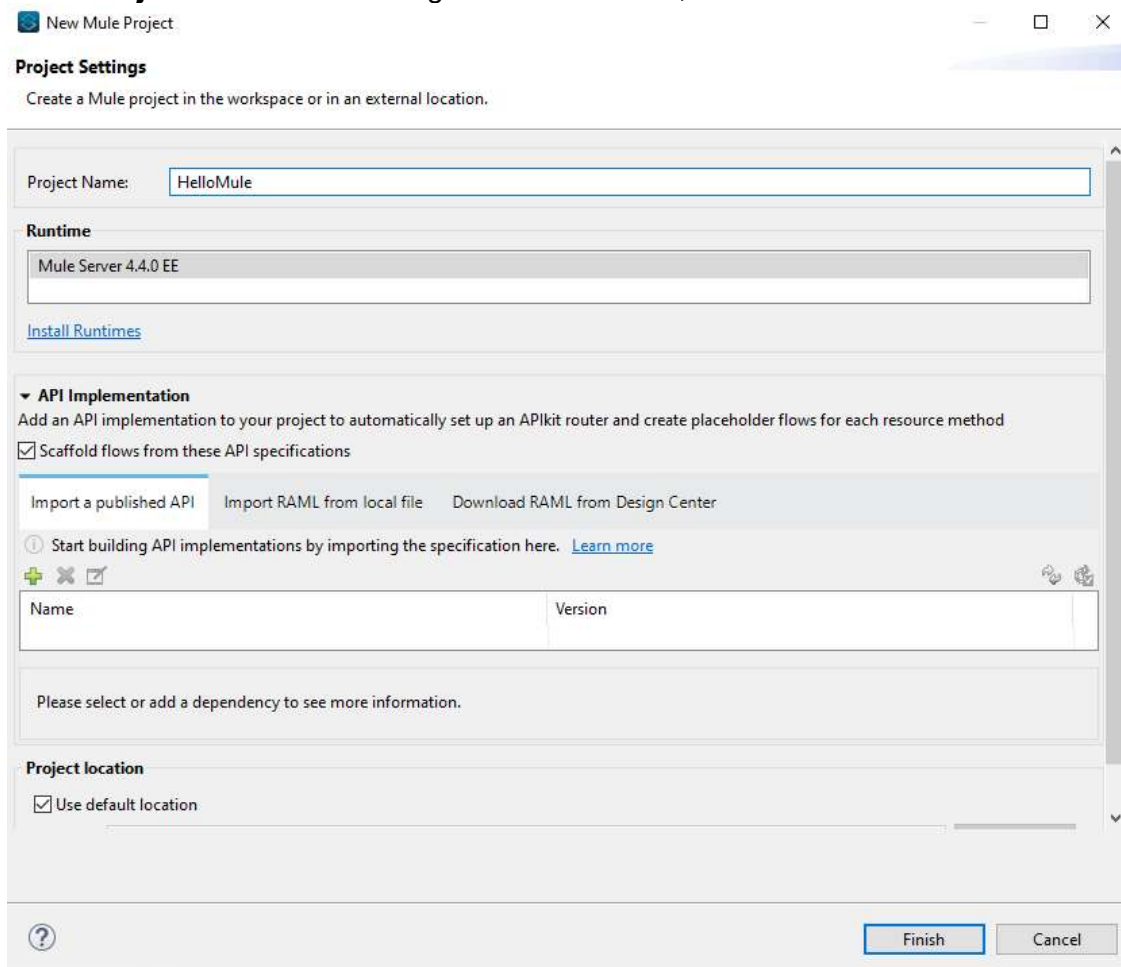


BUILD, TEST, AND DEPLOY YOUR FIRST MULE APP.

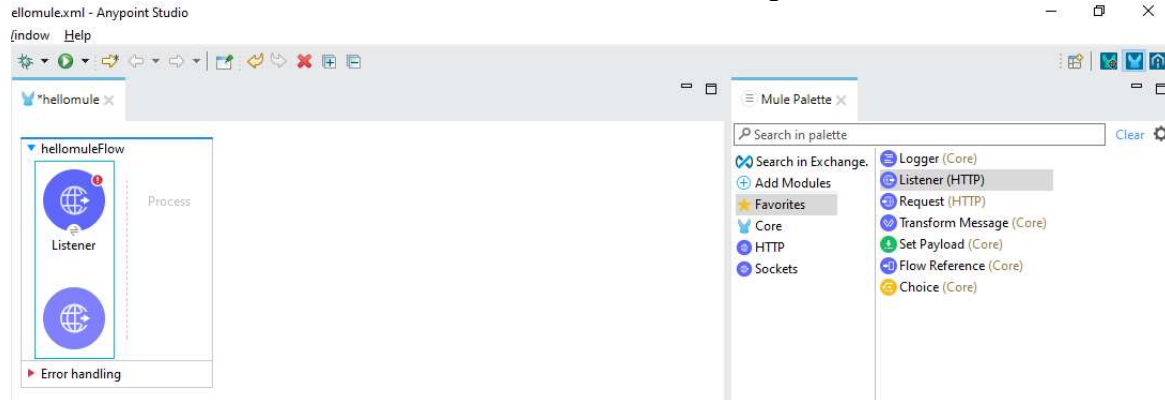
1. First, we are going to open Anypoint Studio and, from there, we are going to **File**, then select **New > Mule Project**.



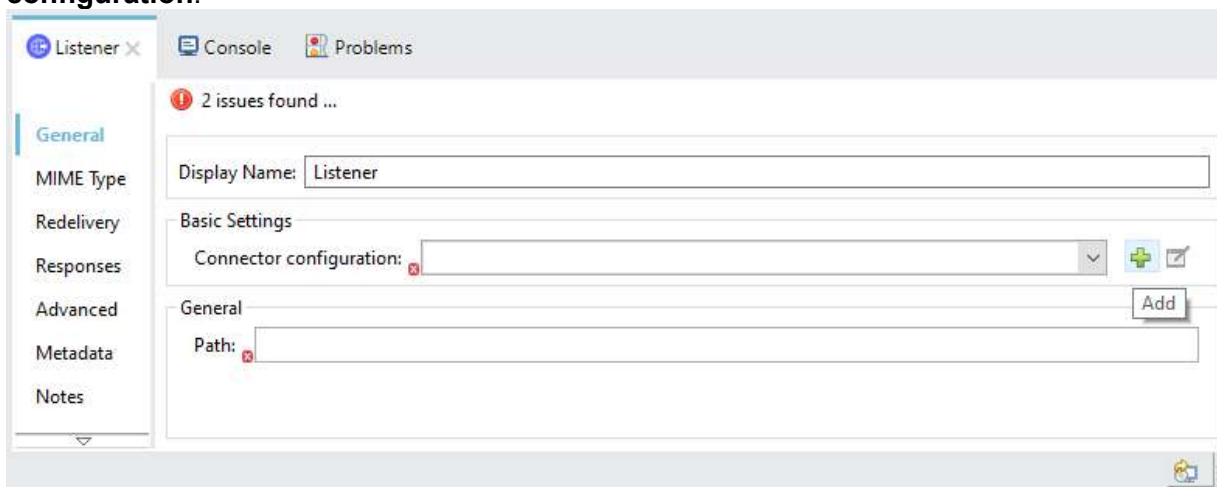
2. In the **Project Name** field is assigned as *Hello Mule*, and then click on **Finish**.



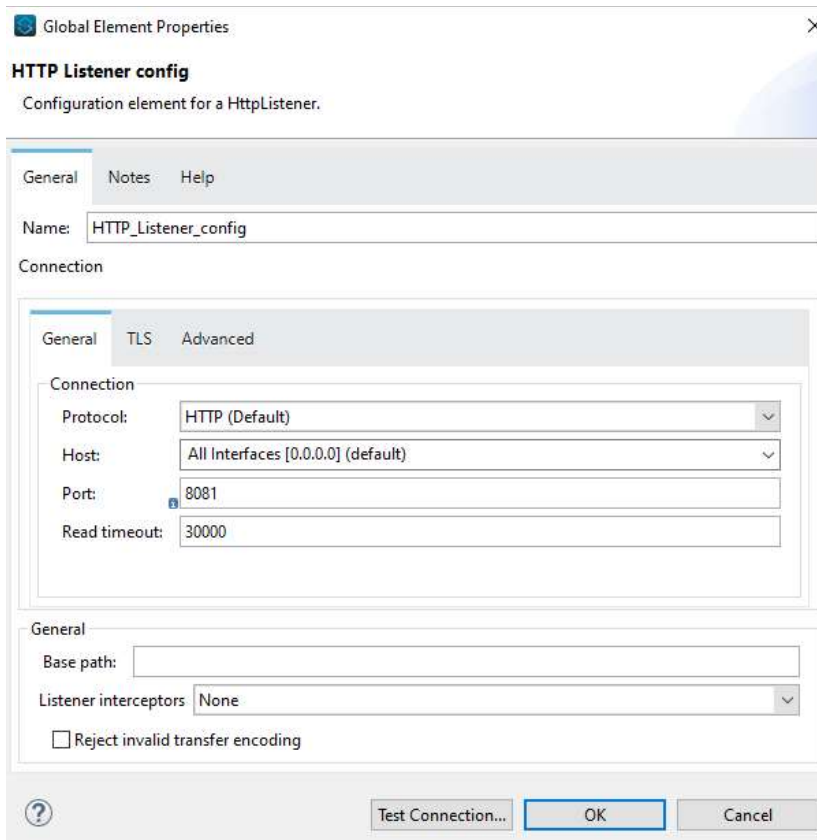
3. Inside the Mule Palette, select the **HTTP Listener** and drag it onto the canvas.



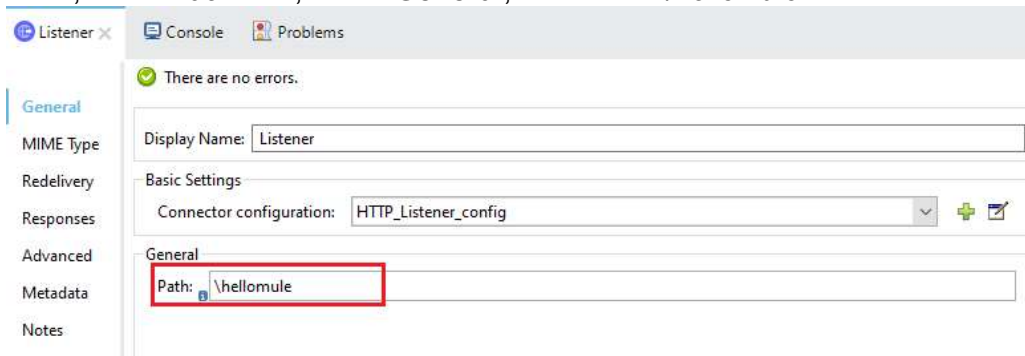
4. Now, within the *Properties Editor*, click on the + symbol in green, which is located next to **Connector configuration**.



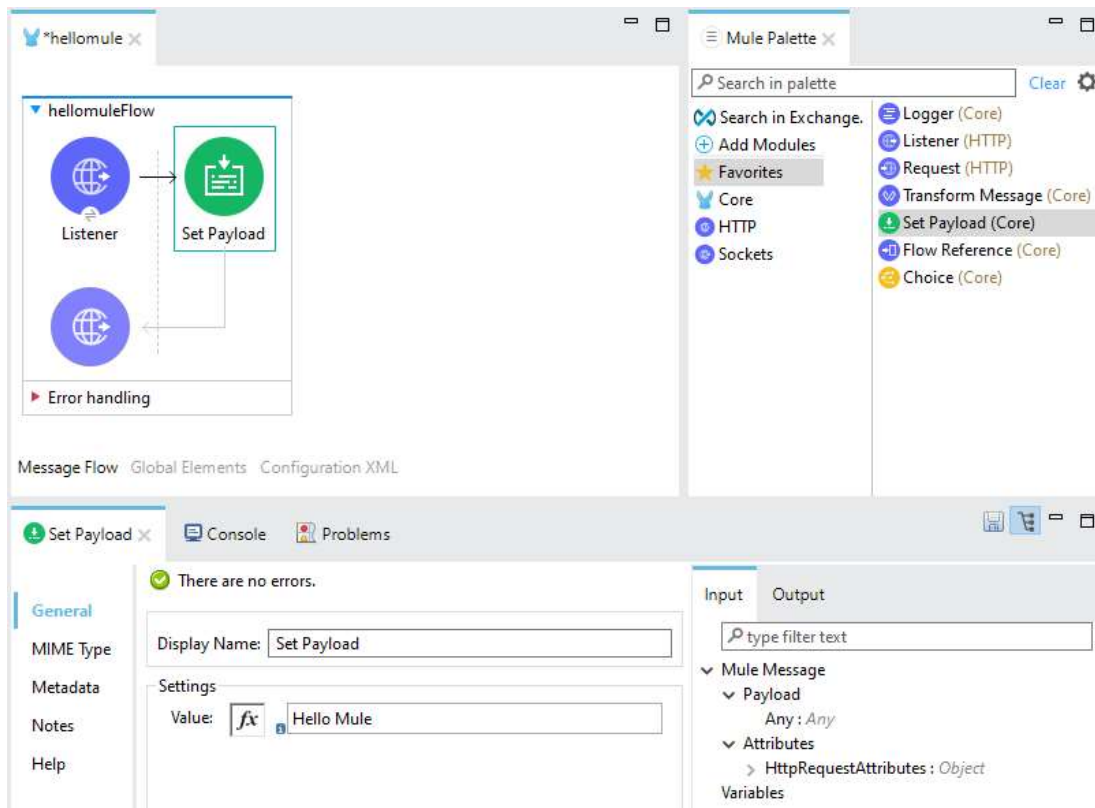
5. Inside the window, the values of **Host**, which should be *0.0.0.0*, and **Port**, which should be *8081*, are checked. Then click on **Ok**.



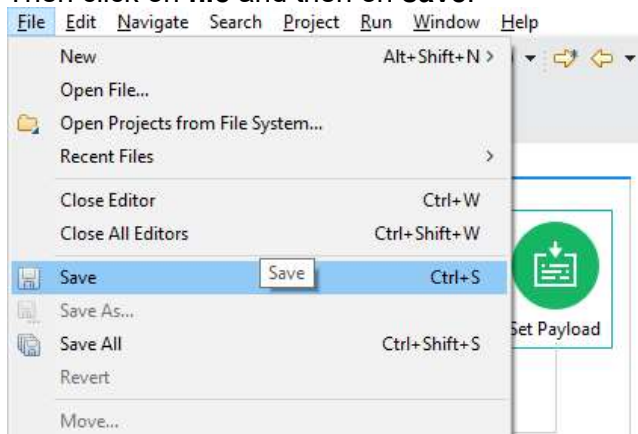
6. Later, in the **Path** field, within **General**, it is written */hellomule*.



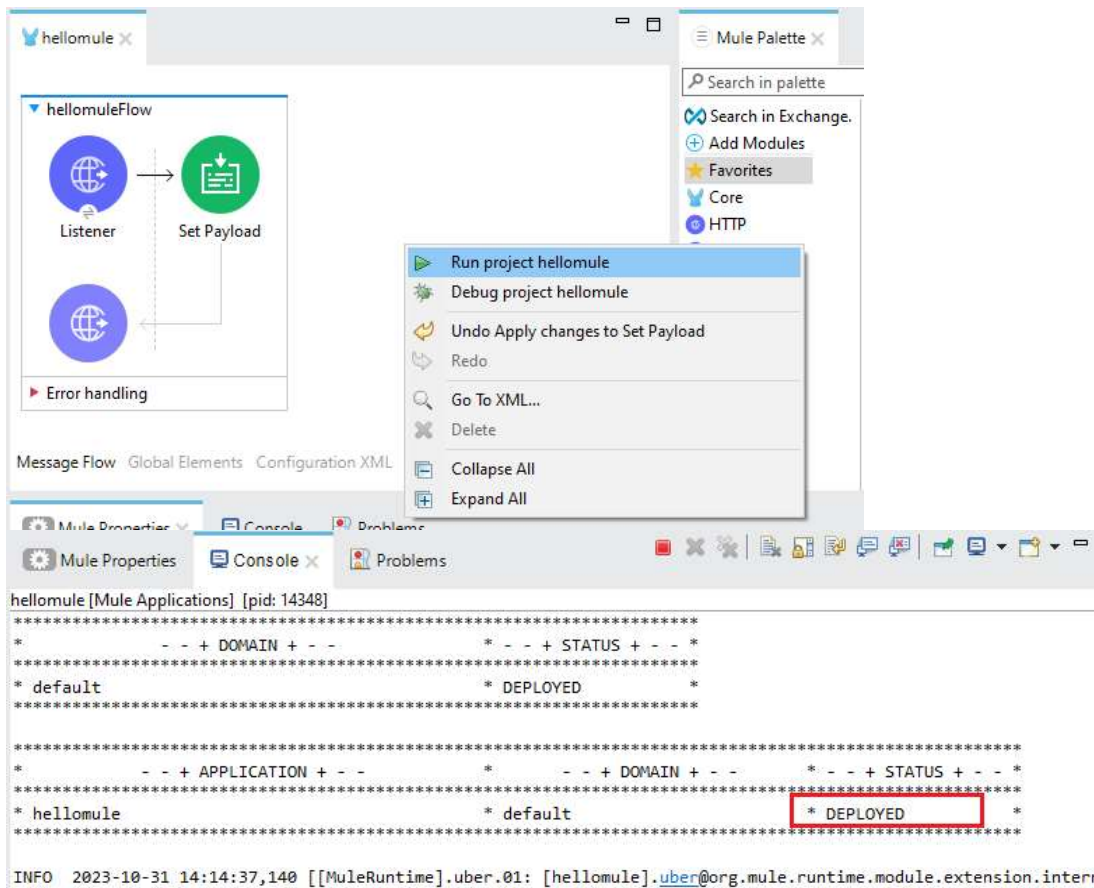
7. Within the *Mule Palette*, select **Set Payload**, and drag it onto the **canvas**. Then, in **Properties Editor**, click on the **fx** formula button and in the **value** field type *Hello Mule*.



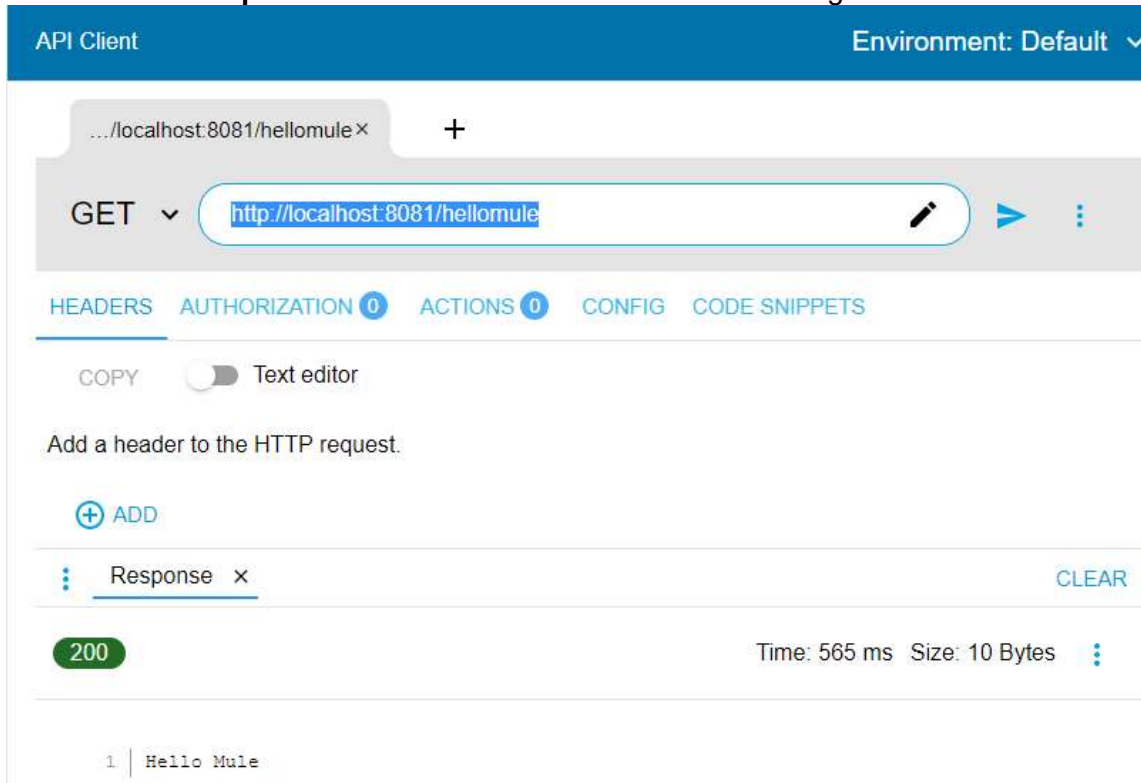
8. Then click on **file** and then on **save**.



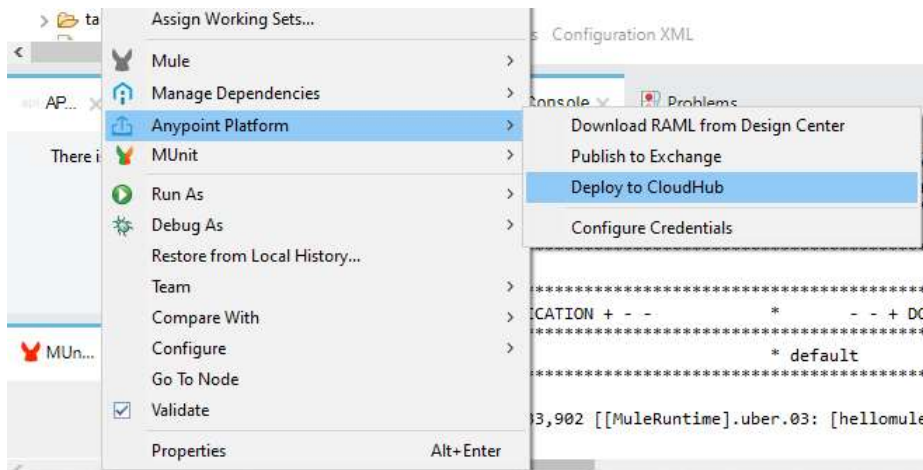
9. Inside canvas, set right-click and then click on **Run project hellomule**, then check in the part of the console that at the bottom right says **DEPLOYED**.



10. Now, we open the Advanced REST Client, and in the **search** field type `http://0.0.0.0:8081/hellomule` and click **Send the request**. You can see in the status a **200 OK** in green.



11. In Anypoint Studio, on the console side, click **Anypoint Platform > Deploy to CloudHub**. Then, it asks us for the access credentials, and then we choose **sandbox** environment.

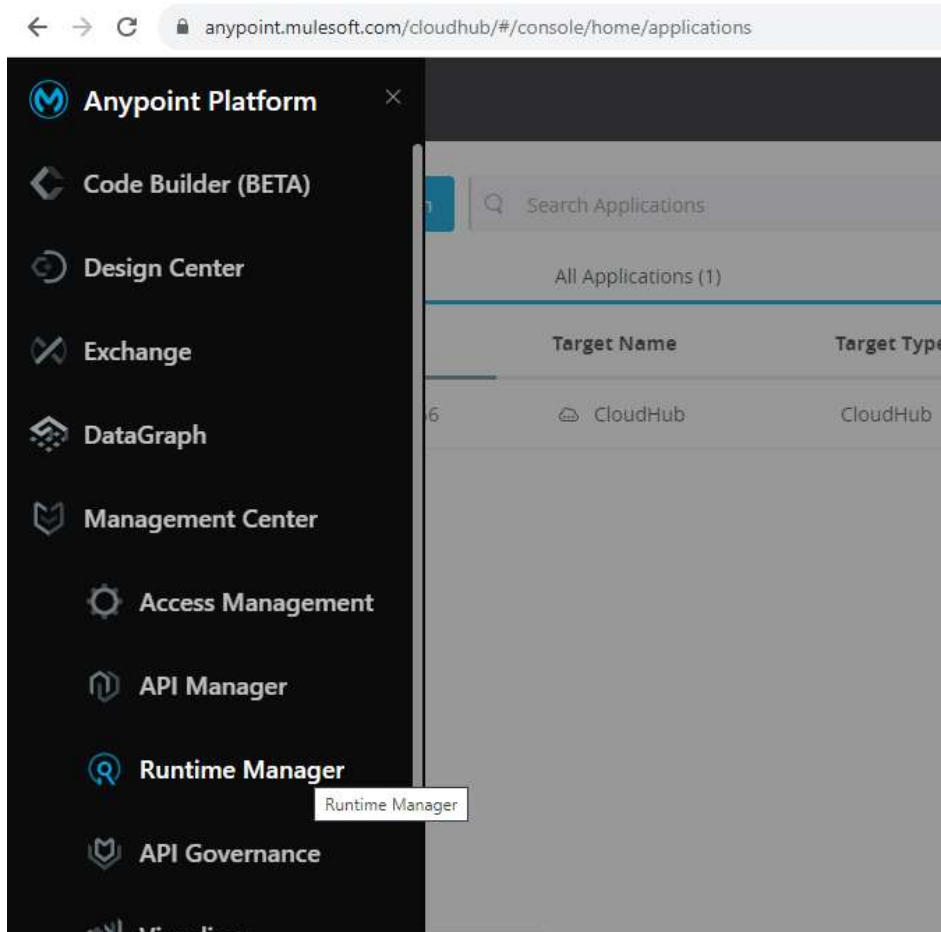


12. Once in the environment, we verify that the name of the application is with a green check. If so, click on **Deploy Application**.

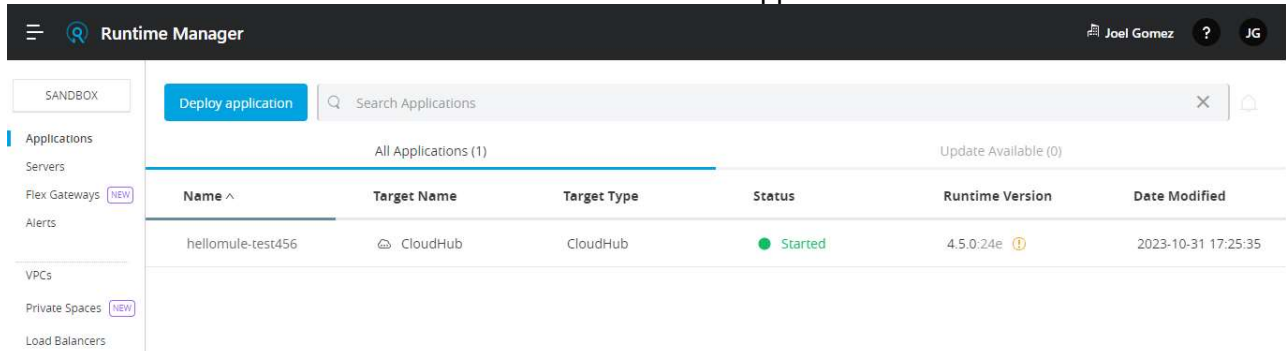


 A screenshot of the 'Deploying Application' dialog in the Anypoint Platform. The dialog shows the application name 'hellomule' with a green checkmark, indicating it is ready for deployment. The deployment target is set to 'Shared Space US East (Ohio) CloudHub 2.0'. The deployment model is set to 'Rolling update'. The runtime options section shows 'Run in Runtime Cluster Mode' and 'Use Object Store V2' as unchecked options. A warning message at the bottom states: 'Deployments with less than 0.1 vCPU available may take up to 10 minutes to start'. The 'Deploy Application' button is highlighted in blue.

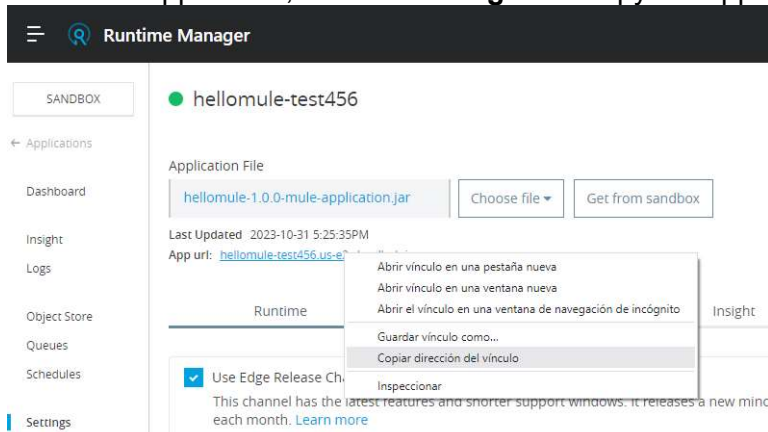
13. Now, through the browser, you can access the website of <https://anypoint.mulesoft.com/login/signin?>, then click on the three-bar menu, and then click on **Runtime Manager**.



14. Then click on the **Sandbox** and click on the name of the application.



15. Within the application, click on **settings** and copy the app's url.

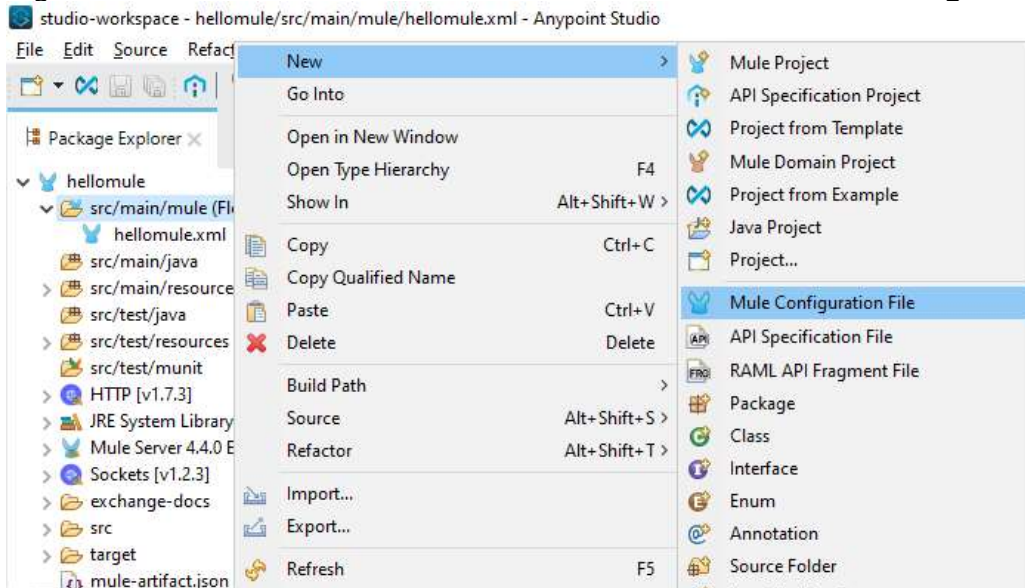


16. Now, within the *Advanced REST Client*, we add a new tab, in the search field the url copied is pasted and click on **Send the request**.

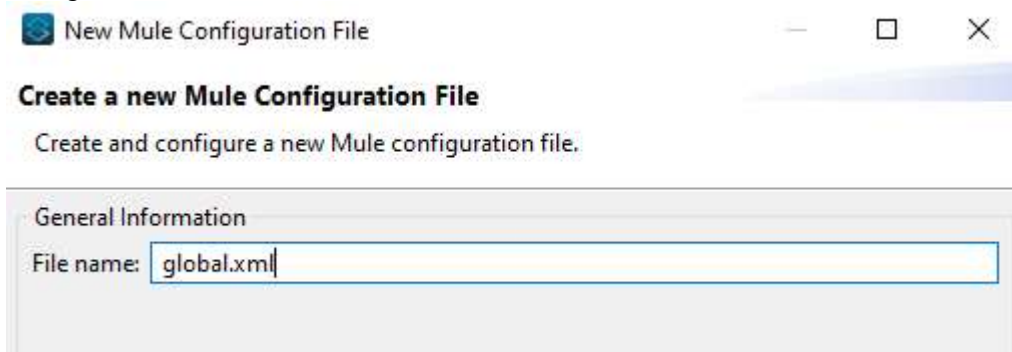
The screenshot displays the Advanced REST Client interface. At the top, the title bar reads "API Client" and "Environment: Default". Below this, there are two tabs: "...est456.us-e2.cloudhub.io x" and "...e2.cloudhub.io/hellomule x", with a "+" icon to add more. The main area shows a "GET" method selected from a dropdown, followed by the URL "http://hellomule-test456.us-e2.cloudhub.io/hellomule". To the right of the URL are icons for editing (pencil), sending (blue arrow), and a menu (three dots). Below the URL bar are tabs for "HEADERS", "AUTHORIZATION 0", "ACTIONS 0", "CONFIG", and "CODE SNIPPETS". Under the "HEADERS" tab, there is a "COPY" button and a "Text editor" toggle switch. Below this, the text "Add a header to the HTTP request." is displayed, followed by a "+ ADD" button. A "Response" tab is active, showing a status of "200" in a green circle, "Time: 485 ms", "Size: 10 Bytes", and a "CLEAR" button. The response body is displayed as "1 | Hello Mule".

USE THE PROPERTY FILE TO MAINTAIN AND REFERENCE SENSITIVE DATA SEPARATELY FROM THE GENERATED CODE.

1. Right-click on the **src/main/mule** folder. And click on **New > Mule Configuration File**.



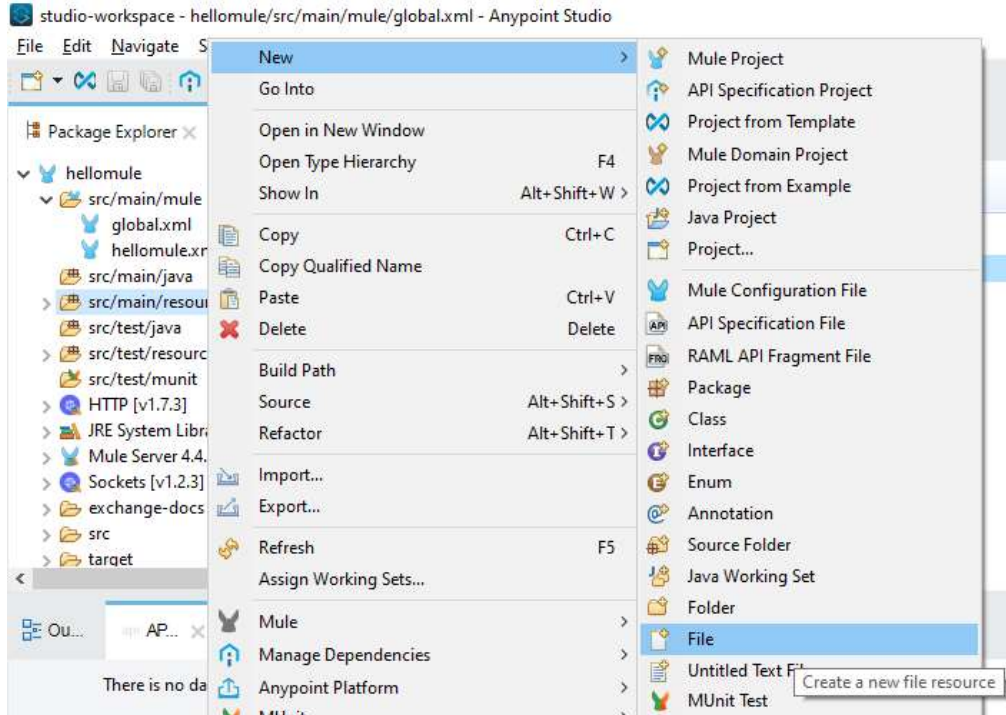
2. Add *global.xml* as a name inside the **File name** field, and then click on **Finish**.



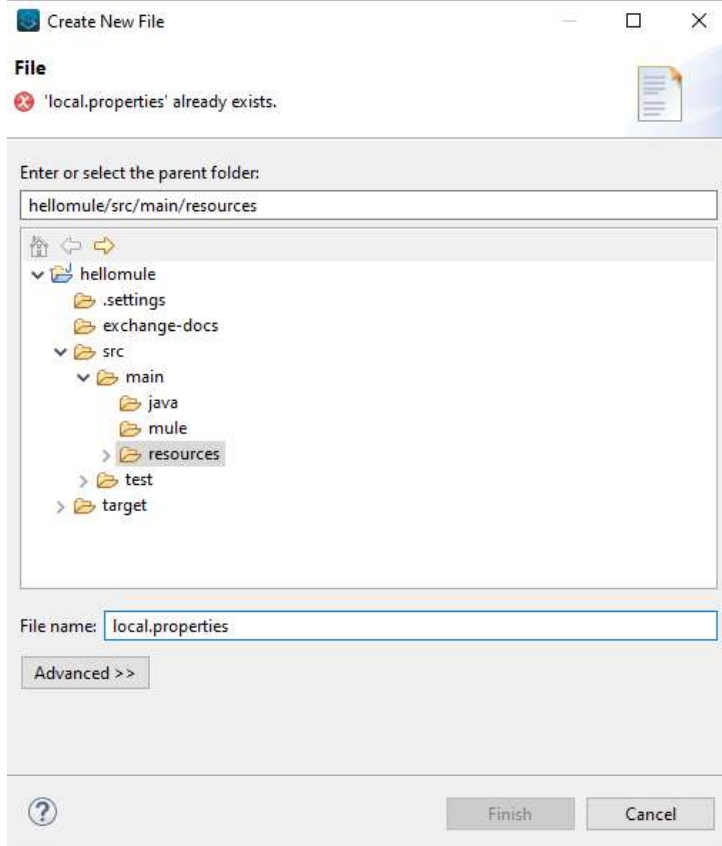
3. Now in *hellomule.xml* within the **Configuration XML** tab, we select everything that includes within *http:listener-config* and then it is cut.



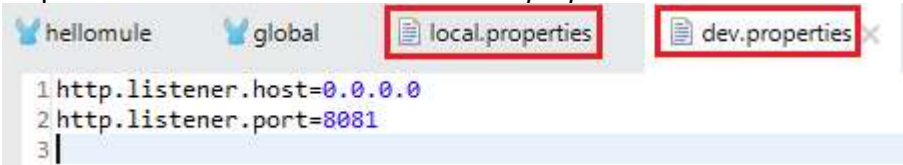
8. To avoid having *hardcoded values*, they are going to be outsourced. To do this, right-click on the **src/main/resources** folder and select **New > File**.



9. Inside the window, type in the **File name** field *local.properties*.

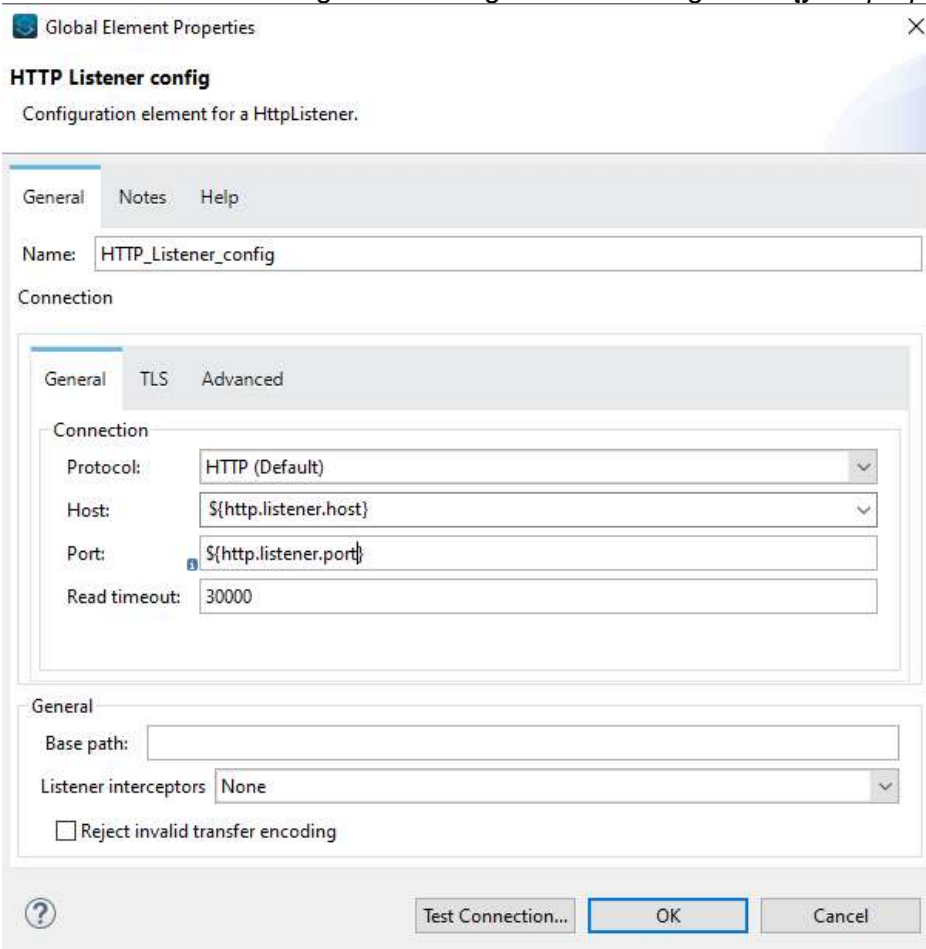


10. The code is written into the file (as shown in the image below) and saved. The same is repeated from steps 8 and 9 but with the file name *dev.properties*.



```
1 http.listener.host=0.0.0.0
2 http.listener.port=8081
3 |
```

11. Then, in global.xml within global elements tab, click on **edit** from the **HTTP Listener** and there the **Host** and **Port** fields are changed according to the following text: *\${your.property.name}*. Then click **Save all**.



Global Element Properties

HTTP Listener config
Configuration element for a HttpListener.

General Notes Help

Name: HTTP_Listener_config

Connection

General TLS Advanced

Connection

Protocol: HTTP (Default)

Host: \${http.listener.host}

Port: \${http.listener.port}

Read timeout: 30000

General

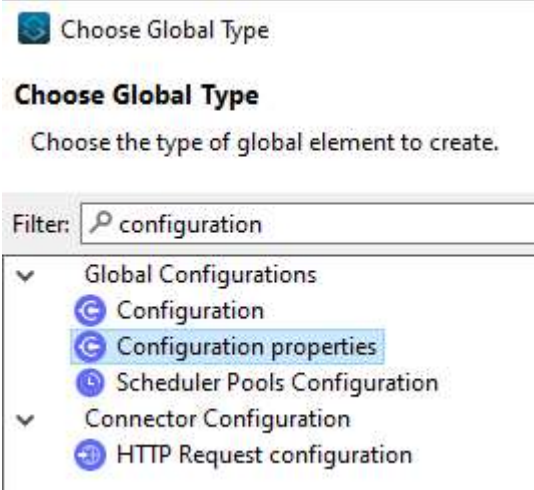
Base path:

Listener interceptors: None

☐ Reject invalid transfer encoding

Test Connection... OK Cancel

12. Now within global, in the **Global elements** tab, click on **create**. In the window, enter **configuration properties** in the search field, and then click on **Ok**.



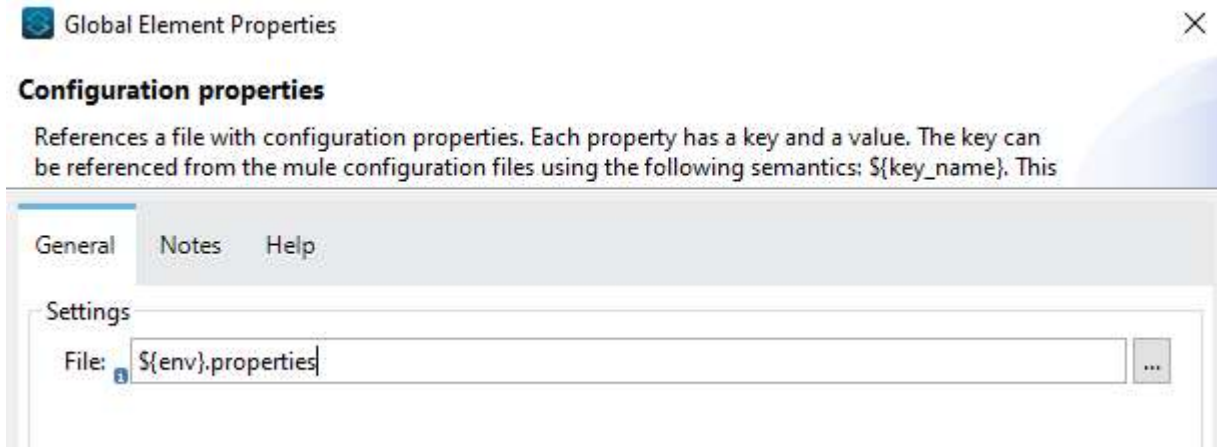
Choose Global Type

Choose the type of global element to create.

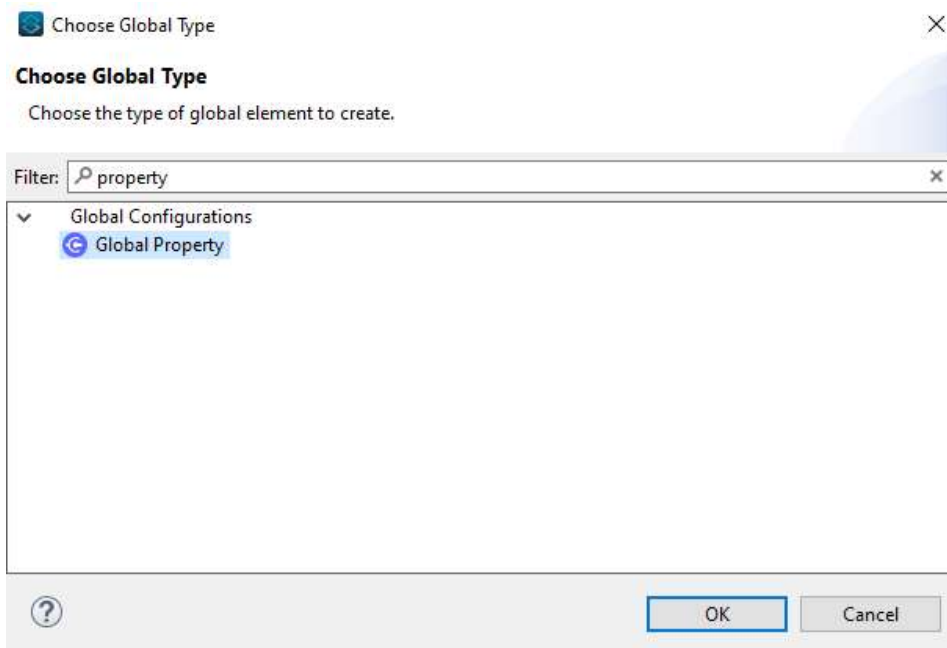
Filter: configuration

- Global Configurations
 - Configuration
 - Configuration properties
 - Scheduler Pools Configuration
- Connector Configuration
 - HTTP Request configuration

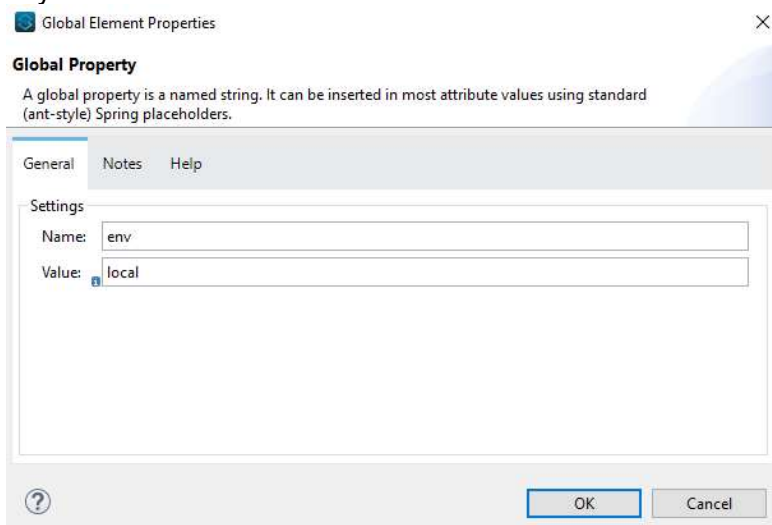
13. Now, in the window, type `${env}.properties` in the **File** field, and then click on **OK**. Then click on **Save all**.



14. Now in `global.xml`, in the **Global elements** tab, click **create**. In the window, enter *Global Property* in search field and then click on **ok**.



15. Now, in the window, in the **Name** field, type `env` and also in the **Value** field we type `local`, and then click on **Ok**. Then click on **Save all**. Then it goes to the `hellomule.xml` tab, in the **Message flow** part there right-click inside the canvas and then click on **Run project hellomule**. And then, wait for the console to say **DEPLOYED**.

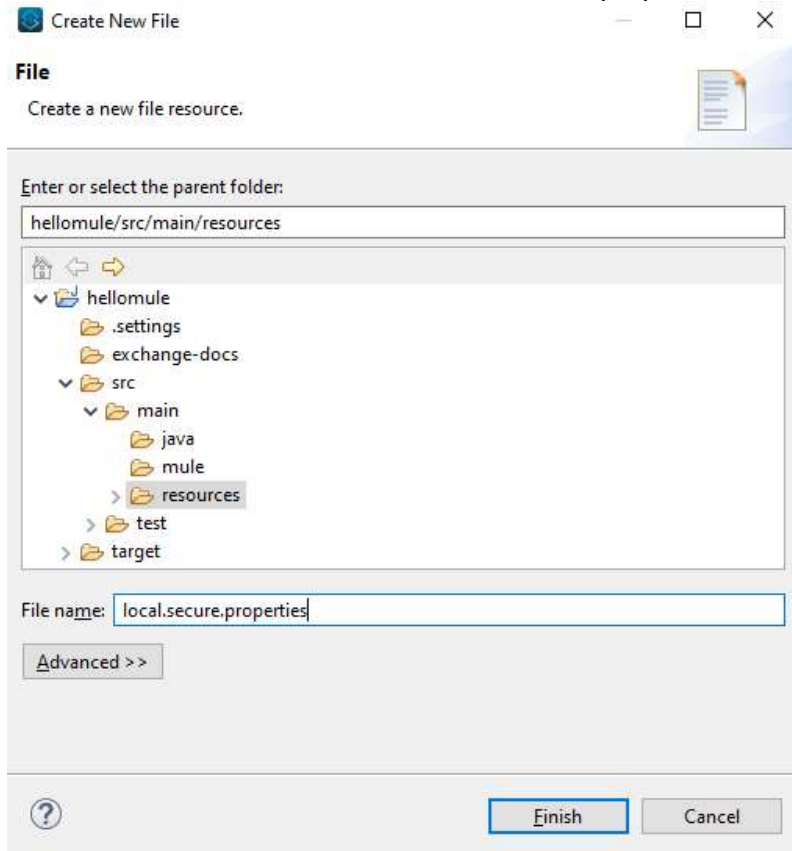


16. Within the Advanced REST Client, type *localhost:8081/hellomule* into the search field and click **send**. Always validating that the status comes out **OK 200**.
17. Go back to Anypoint Studio and click on **Terminate**.

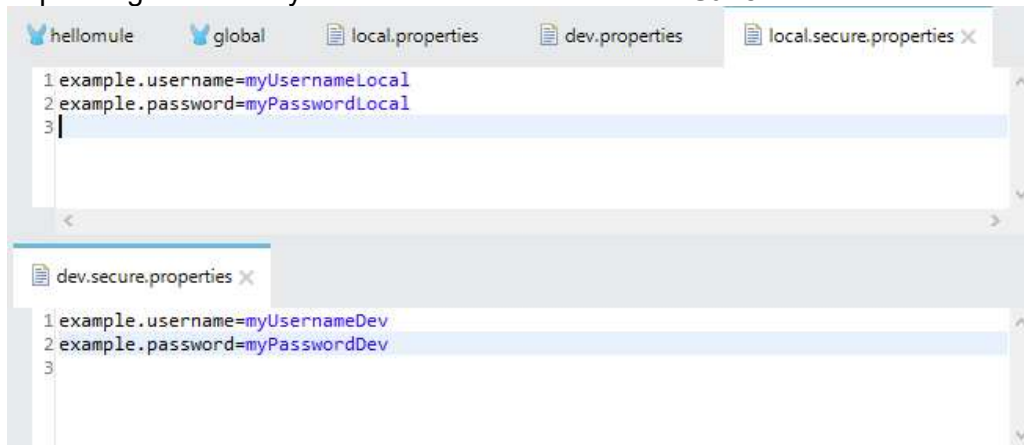


CREATE A SECURE PROPERTY FILE TO PROTECT SENSITIVE DATA THAT POSES A RISK TO KEEP IT CLEAR.

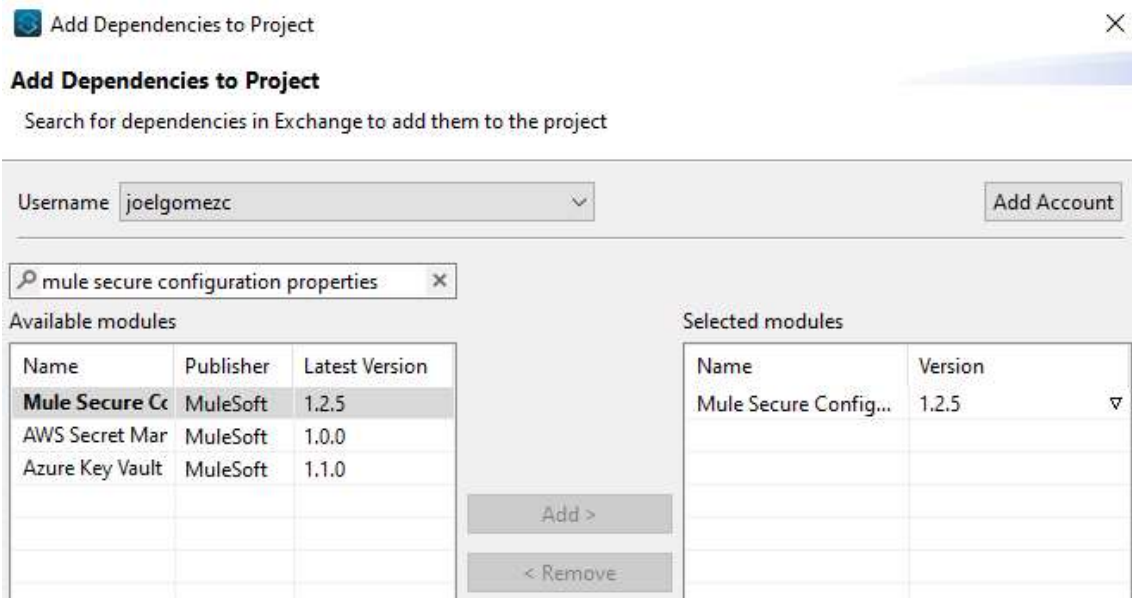
1. Right-click on the **src/main/resources** folder and click on **New > File**. And in that same window, in the **File** field, enter in the **name** field *local.secure.properties*.



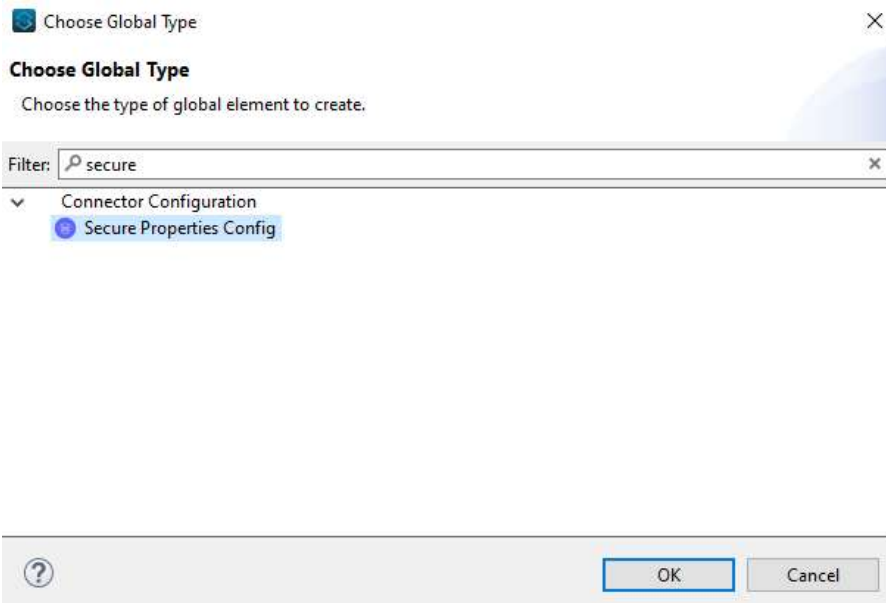
2. The previous step is repeated, but it is entered in the **File name** field *dev.secure.properties*.
3. Now, in both created files, the following lines of code are entered:
`example.username=myUsernameLocal`
`example.password=myPasswordLocal`
depending on the file you want to edit. Then click on **Save All**.



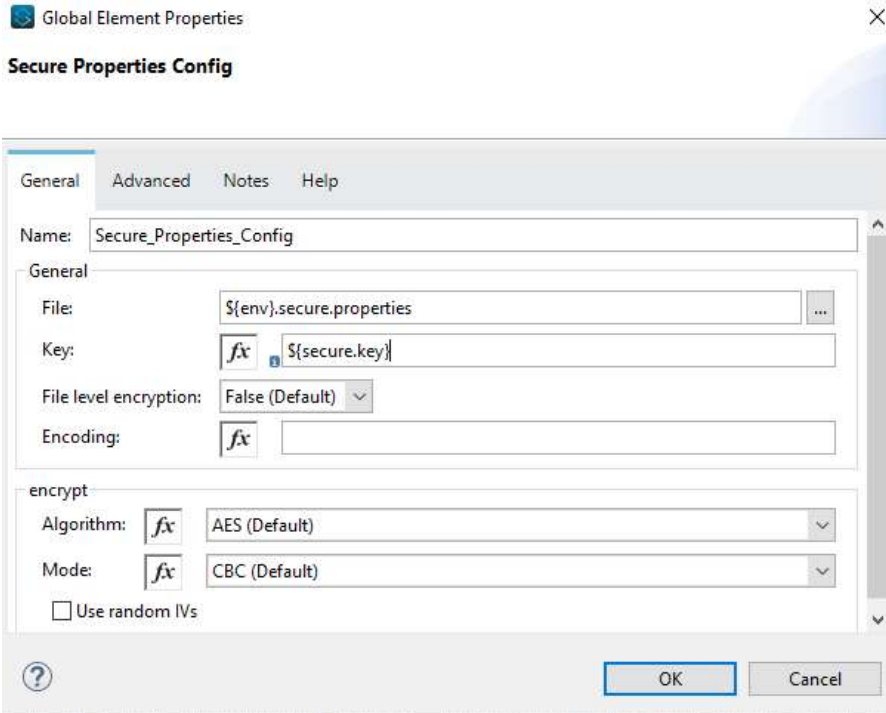
4. Within `hellomule.xml`, click on **Search in Exchange** and from there search for **Mule Secure Configuration Properties**, select the one highlighted in bold and click on **Add**, and then click on **Finish**.



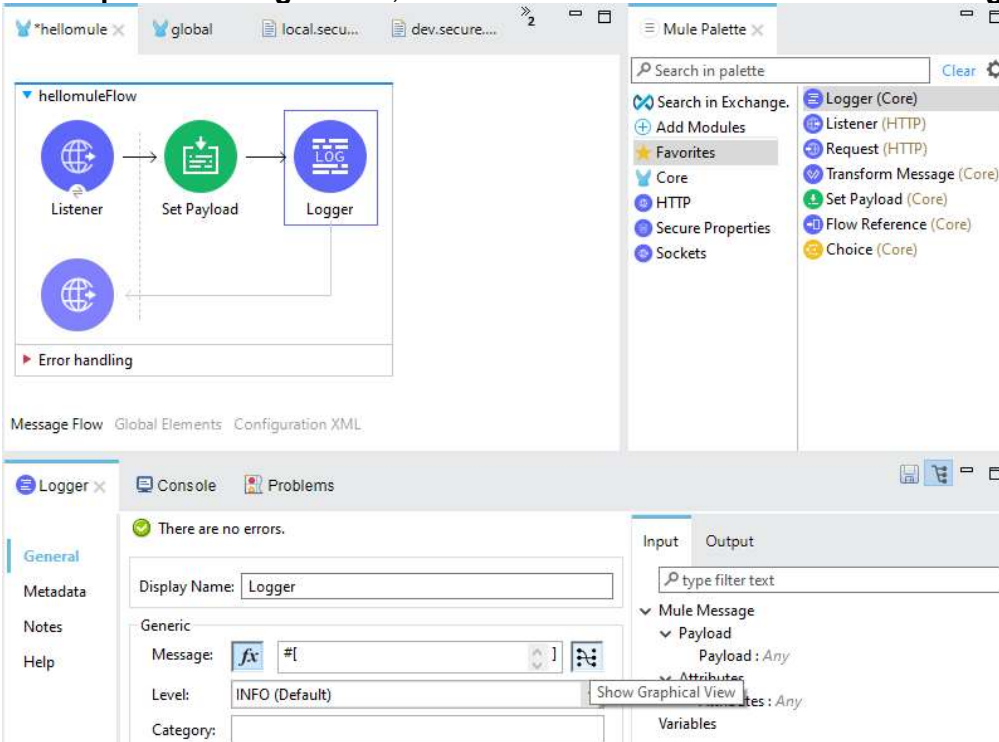
5. Now in `global.xml`, in the **Global elements** tab, click on **create**, in the window type in the search field **Secure properties config**, select it and click on **OK**.



6. Inside the window, enter in the **File** and **Key** fields, `${env}.secure.properties` and `${secure.key}`. It is put into **Algorithm** as *Blowfish*. It is confirmed by clicking **Ok**. Then click on **Save all**.



7. Returning to `hellomule.xml`, the **Mule Palette Logger (core)** is searched and dragged to the canvas. In the **Properties configuration**, click on the **fx** button and then on the **Show graphical view** button.



8. Then click on the **source only** view and enter the lines of code according to the image, and then click on **Done**.

```
output application/java
```

```
---
```

```
"Username: " ++ Mule::p("secure::example.username")
```

```
++ " - " ++
```

```
"Password: " ++ Mule::p("secure::example.password")
```



9. Now, from the Mulesoft documentation, we should download the **Secure properties Tools Jar** file.
10. Open Command Prompt, search for the folder where the file was downloaded, and then enter the next line of code.

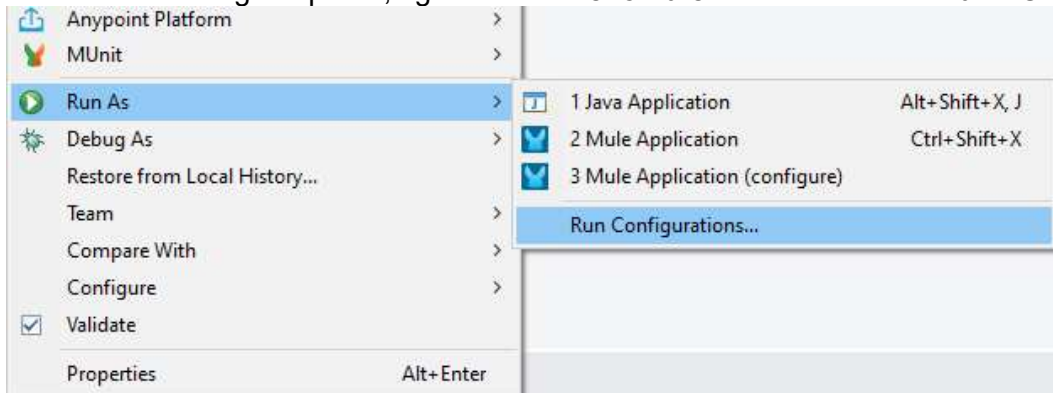
```
java -cp secure-properties-tool.jar com.mulesoft.tools.SecurePropertiesTool  
string encrypt Blowfish CBC MyMuleSoftKey "myUsernameLocal"  
Do the same thing, but instead of "myUsernameLocal", switch to "myPasswordLocal",  
"myUsernameDev" and "myPasswordDev".
```

```
C:\Users\dell\Downloads>java -cp secure-properties-tool.jar com.mulesoft.tools.SecurePropertiesTool string encrypt Blowfish CBC MyMuleSoftKey "myUsernameLocal"  
HbsuWJRjiubchmzQREGdsA==  
  
C:\Users\dell\Downloads>java -cp secure-properties-tool.jar com.mulesoft.tools.SecurePropertiesTool string encrypt Blowfish CBC MyMuleSoftKey "myPasswordLocal"  
zhoritn2db6Ud+9QuhyViQ==  
  
C:\Users\dell\Downloads>java -cp secure-properties-tool.jar com.mulesoft.tools.SecurePropertiesTool string encrypt Blowfish CBC MyMuleSoftKey "myUsernameDev"  
HbsuWJRjiuZZis+2oogkdQ==  
  
C:\Users\dell\Downloads>java -cp secure-properties-tool.jar com.mulesoft.tools.SecurePropertiesTool string encrypt Blowfish CBC MyMuleSoftKey "myPasswordDev"  
zhoritn2db6oII22158LPQ==
```

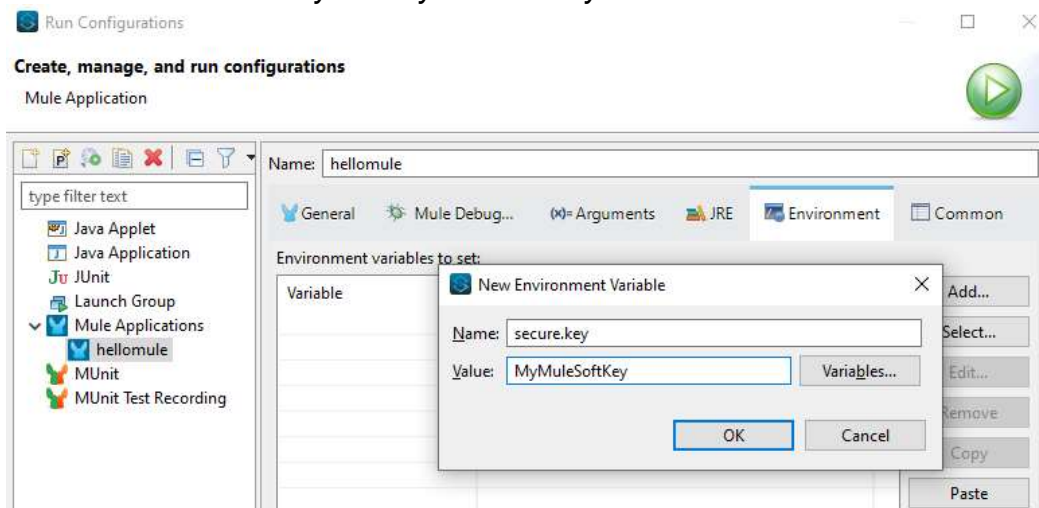
11. Now, in Anypoint Studio, we're going to add the values generated from each command prompt execution, inside the **local.secure.properties** files and also **dev.secure.properties**. They are changed as follows: **![encryptedValue]**.



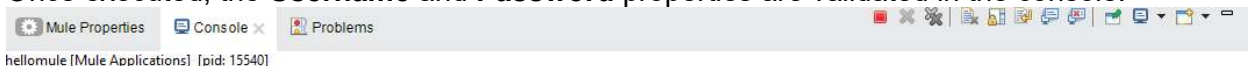
12. Now within Package Explorer, right-click on **hellomule** and then click on **Run As > Run Configurations**.



13. Now in the window, the **Environment** tab opens, click on **Add** and enter in the **Name** and **value** fields these values: **secure.key** and **MyMuleSoftKey**. Click on **OK**.



14. Then click on **Apply** and then on **Run**. Proceed to wait for it to say **DEPLOYED**.
15. Within the Advanced REST Client, type **localhost:8081/hellomule** into the search field and click on **send**. Always validating that the status comes out **200**.
16. Once executed, the **Username** and **Password** properties are validated in the console.

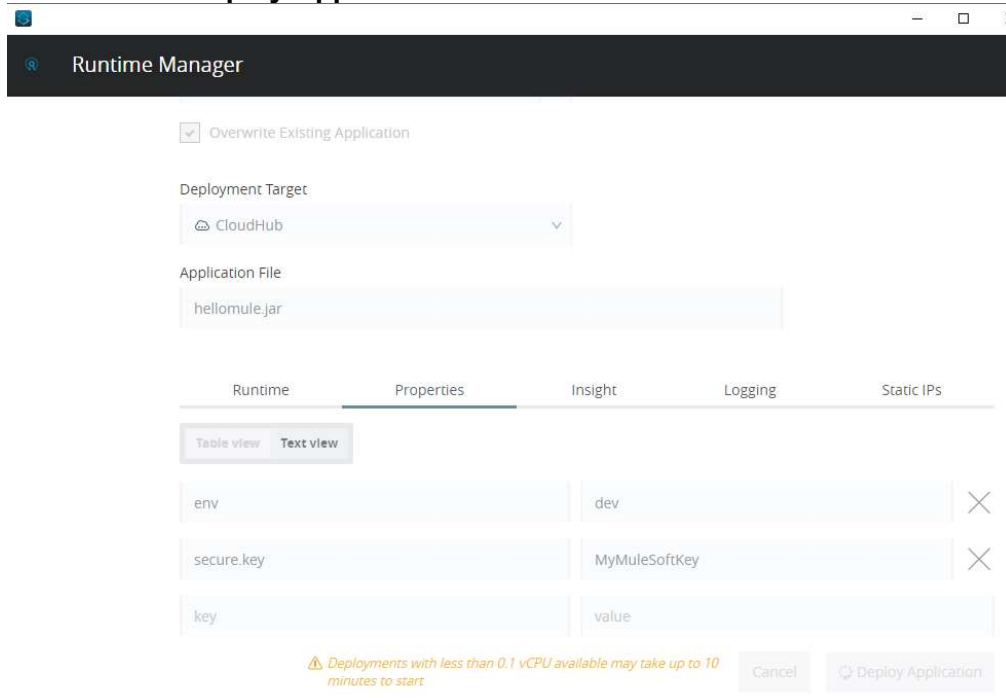


17. Go back to Anypoint Studio and click on **Terminate**.
18. Right-click on the project name, then click on **Anypoint Platform > Deploy to CloudHub**.

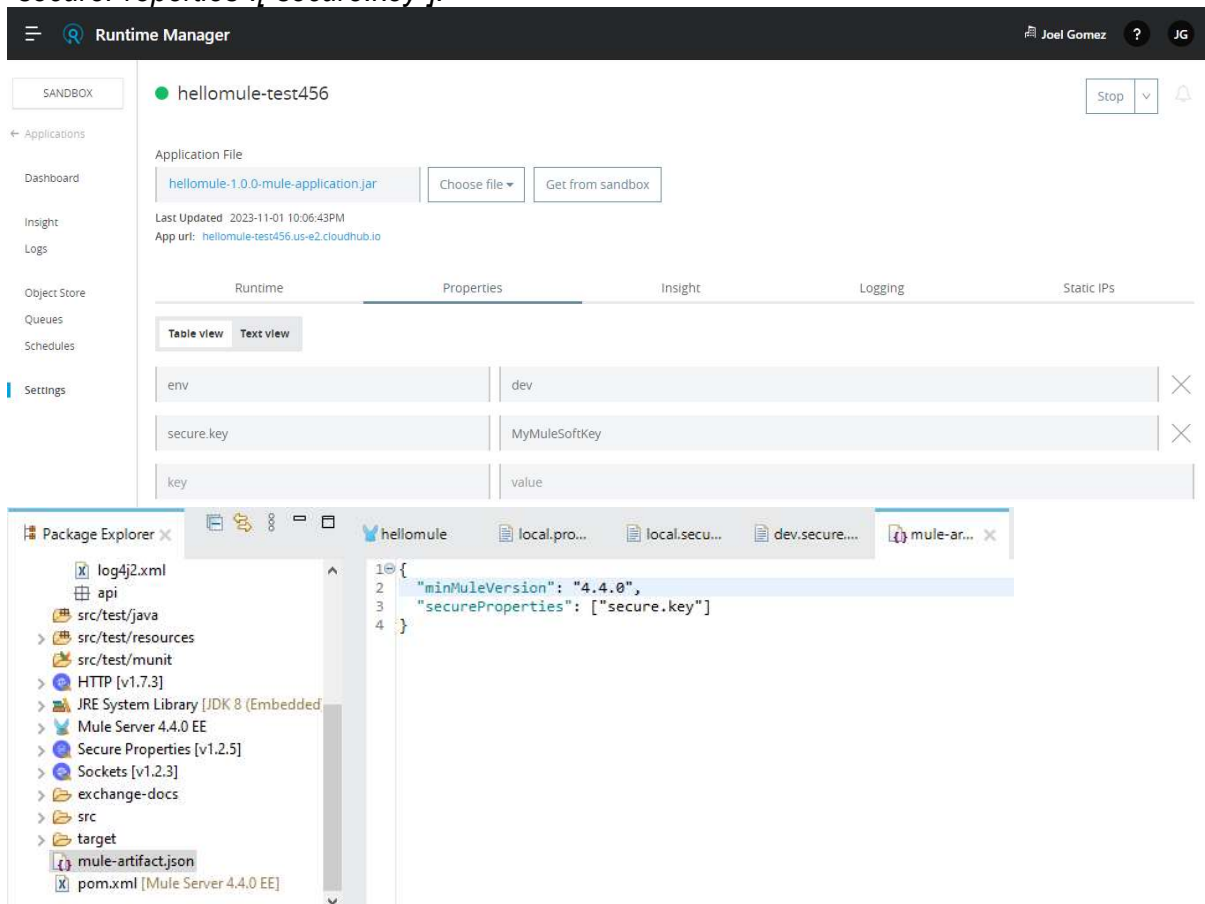
19. Inside that window, click on properties, and enter the following values:

- env = dev
- secure.key = MyMuleSoftKey

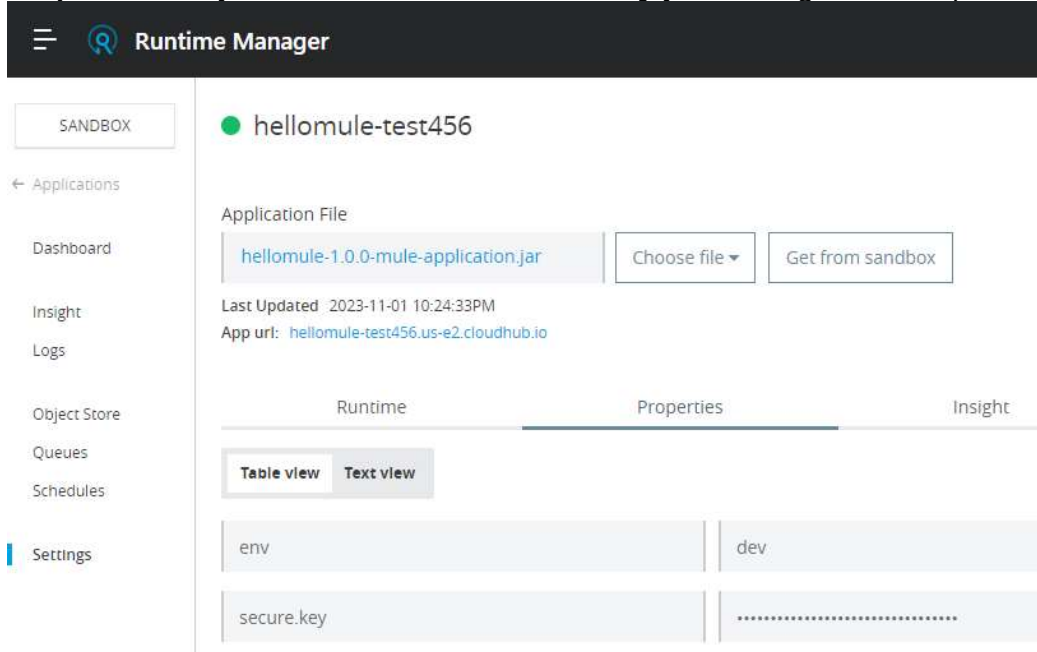
Then click on **Deploy Application**.



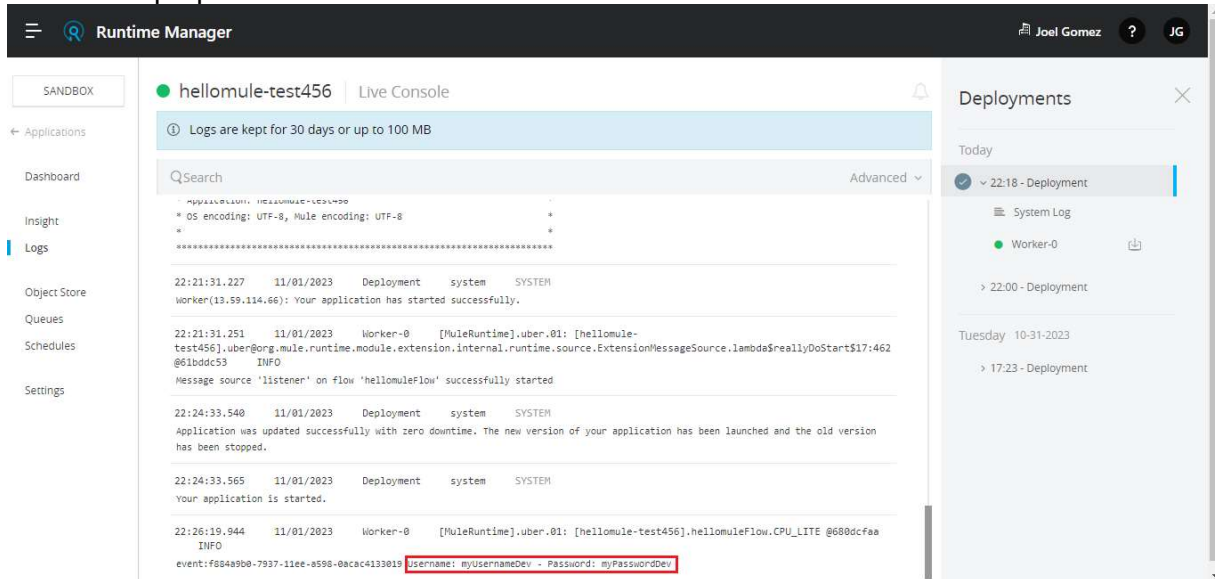
20. Now, entering the CloudHub site, inside **Runtime Manager**, click on the **Settings** tab and then on **Properties** tab and you can notice that in **secure.key** you see the password. To do this, we go to **Anypoint Studio**, open the **mule-artifact.json** file and add the following line of code: **"secureProperties":["secure.key"]**.



21. Then click on **Save all**. Then, click on **Anypoint Platform > Deploy to CloudHub** and click on **Deploy Application** again.
22. Now, entering the CloudHub site, within Runtime Manager, click on the **Settings** tab and then on **Properties** and you can notice that in **secure.key** you no longer see the password.

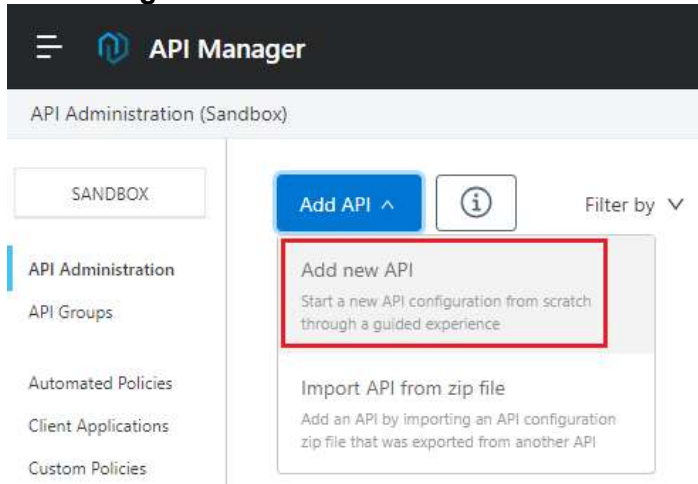


23. Now, within the *Advanced REST Client* we add a new tab, in the search field paste the url copied and click on **Send the request**.
24. Once executed, we should go to logs (within the runtime manager) and check for the **Username** and **Password** properties.



CREATE AN API IN API MANAGER TO ALLOW APPLY POLICY.

1. On the Anypoint Platform platform, click on the menu, and from there click on **Management Center > API Manager**. Then click on **Add API** and then **Add new API**.



2. We select **Mule Gateway**, as well as **Connect to existing application** and **Mule 4** checks.

Select runtime

☐ Flex Gateway **NEW**
Ultrafast API gateway designed to manage and secure APIs running anywhere.

☒ Mule Gateway
API gateway embedded in Mule runtime. Connect directly to an existing Mule app or deploy a new proxy app.

☐ Service Mesh
Manage Kubernetes-based non-Mule microservices with Anypoint Service Mesh.

Proxy type *

☒ Connect to existing application (basic endpoint)
Connect your API to a Mule application using Autodiscovery.

☐ Deploy a proxy application
Select a deployment target and deploy a new Mule application to serve as a proxy.

Mule version *

☒ Mule 4 (recommended)

☐ Mule 3 or below

Cancel Next

3. Then select the **Create New API** check, and *hellomule* is assigned to the **Name** field and *HTTP API* is assigned to **Asset types**.

API

Select the API you want to manage.

☐ Select API from Exchange

☒ Create new API

Once the API is created it will be published in Exchange in stable state.

Name

Asset types

Advanced >

Cancel Previous Next

4. Click on **next** in the next downstream, upstream and review windows and click on **Save**.

5. Within the API manager, click on the name of the API, and then click (in the right-side panel) on **API Summary**. The number which is on **API Instance ID** property is considered.

ndbox) hellomule (v1) - API Summary

APIs / hellomule / API Summary

i To complete the registration process, you need to connect this API to your Mule application using Autodiscovery. [Learn more](#)

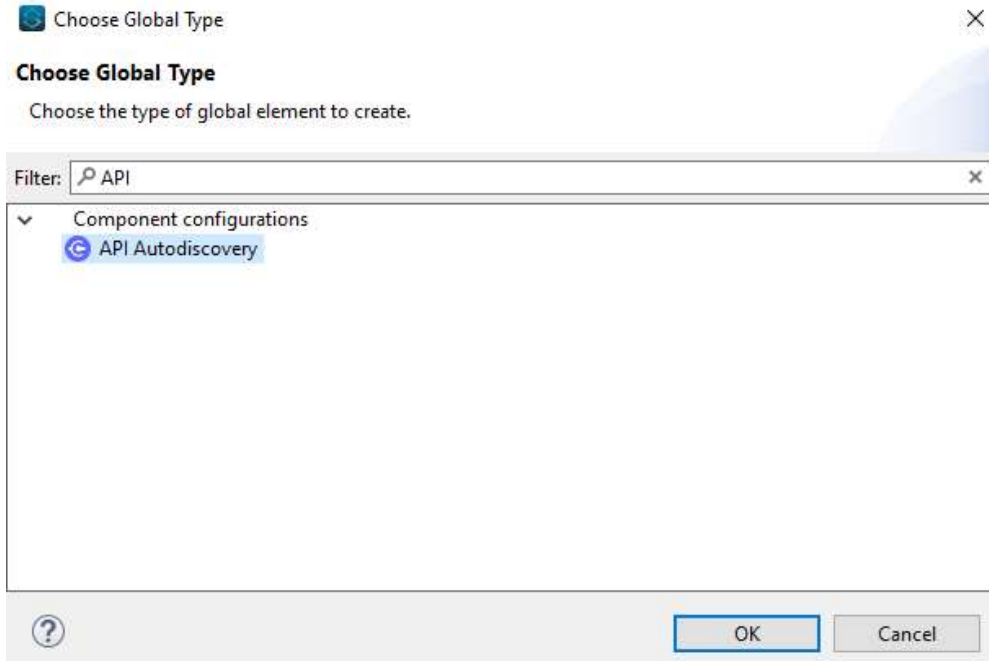
Type	Asset Version	Implementation URI ⁱ
HTTP	1.0.0 (Latest)	N/A
API Label ⁱ	API Version	API Status
-	v1	● Unregistered
Consumer Endpoint	API Instance ID ⁱ	
N/A	19081864	
Tags		
ADD A TAG		

6. Within Anypoint Studio, in the **local.properties** and **dev.properties** files, we add the line of code *api.id* with the value from the previous step.

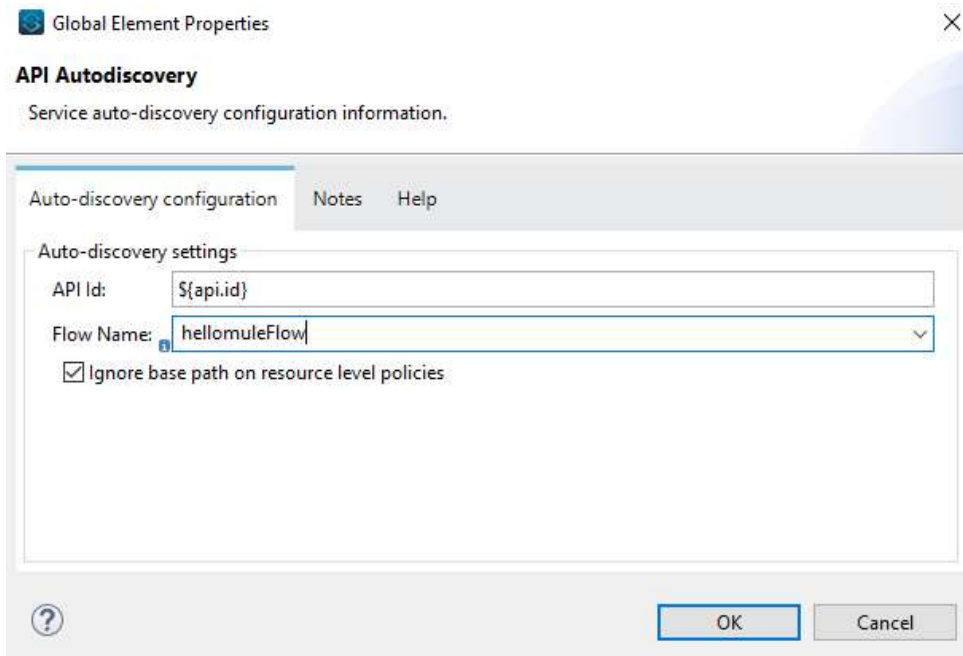
The screenshot shows the Anypoint Studio interface. On the left, the Package Explorer displays the project structure for 'hellomule', including 'src/main/resources' with files like 'dev.properties', 'local.properties', and 'log4j2.xml'. On the right, two tabs are open: 'local.properties' and 'dev.properties'. Both files contain the following content:

```
1 http.listener.host=0.0.0.0
2 http.listener.port=8081
3
4 api.id=19081864
```

7. Now we go to **global.xml** and from there in **Global Elements**, click on **create**. And look in the **API Autodiscovery** field and click on **OK**.



8. In the next window, enter the *`${api.id}`* and *`hellomuleFlow`* values in the **API Id** and **Flow Name** fields. Then click on **OK**.



9. Now, click on the menu (of Anypoint Platform) and then click on **Management Center > Access Management**. And inside the window click on **Enviorenements** and then to **Business Groups > Username > Environments > Sandbox**. And the window is shown as shown in the following image. We copy the **Client ID** and **Client Secret** properties.

Edit environment

Name

Client ID

Client Secret

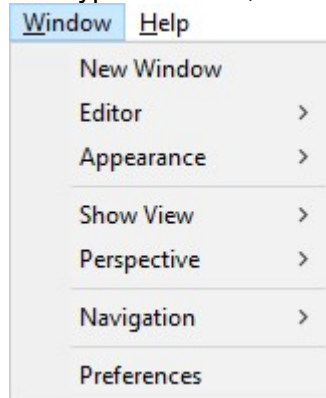
[Hide](#)

Delete Environment

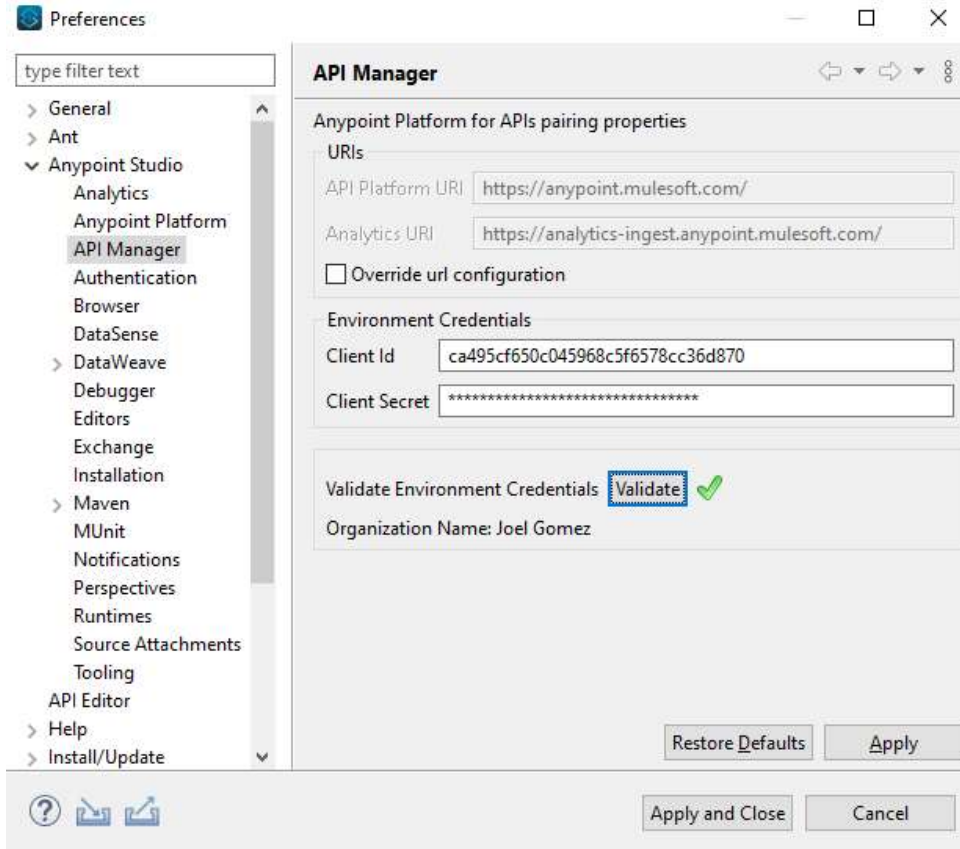
Cancel

Update

10. In Anypoint Studio, click on **Window > Preferences**.

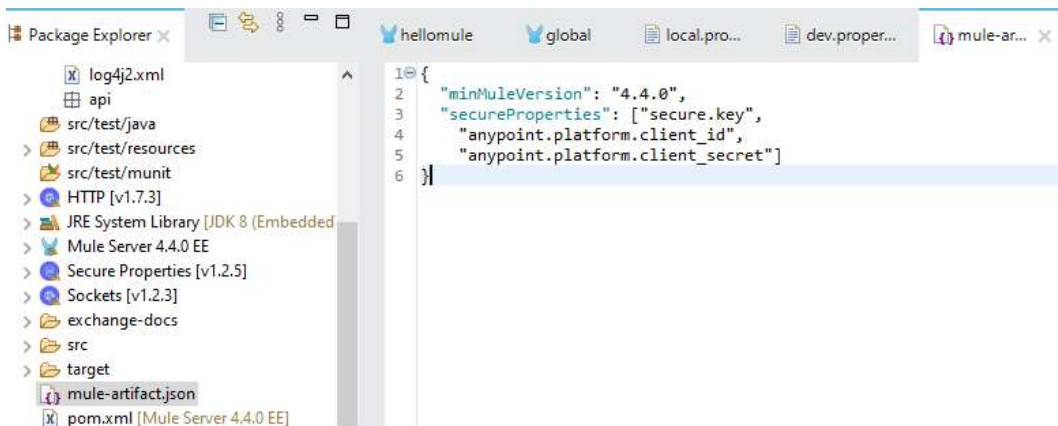


11. Now click on **Anypoint Studio > API Manager** and paste the copied credentials. Then click on **validate** and click on **Apply and close**.



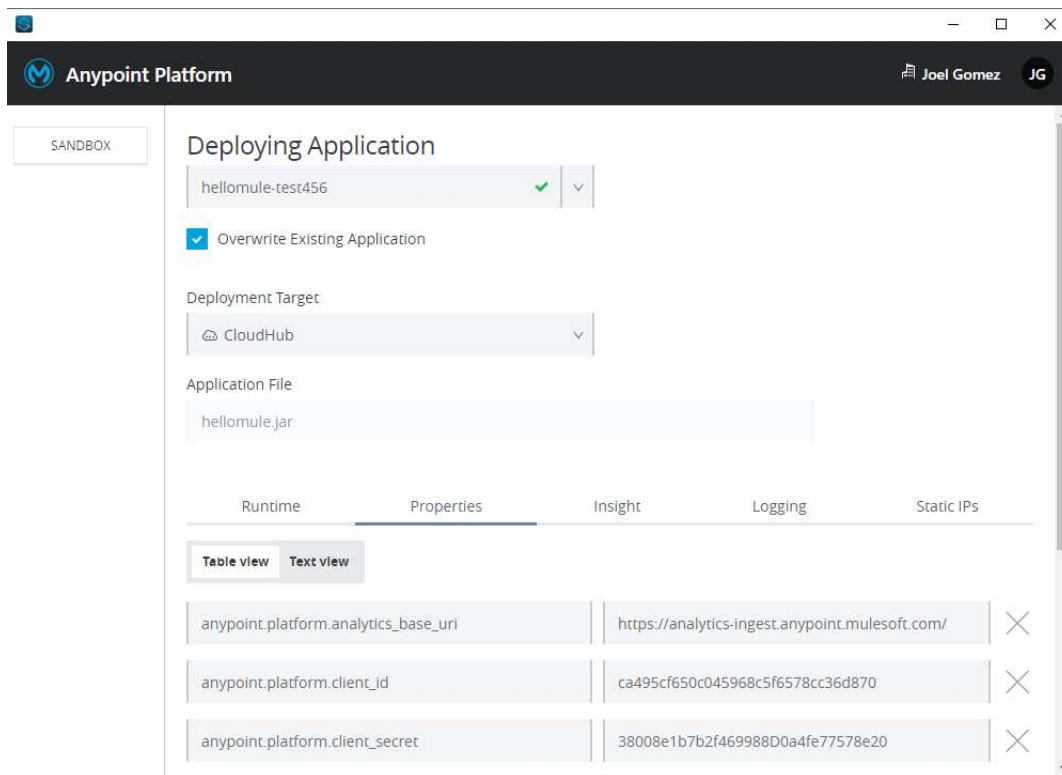
12. Then we have to execute the application, inside the canvas, always verifying that at the end it says in the console **DEPLOYED**.
13. Now, we have to go to the Advanced REST Client, and in the **search** field, you type *http://0.0.0.0:8081/hellomule* and click on **Send the request**. You can see in the status a **200 OK** in green.
14. Now, we should open the **mule-artifact.json** file and we should add the following two properties

```
"anypoint.platform.client_id",  
"anypoint.platform.client_secret"
```



15. Then click on **Save all** and right-click on **hellomule** and from there click on **Anypoint Platform > Deploy to CloudHub**.

16. Now, we verify that the **Overwrite Existing application** box is checked, as well as the data mentioned in 9 are included and match. Then click **Deploy Application**.



Deploying Application

hellomule-test456 ✓

☒ Overwrite Existing Application

Deployment Target: CloudHub

Application File: hellomule.jar

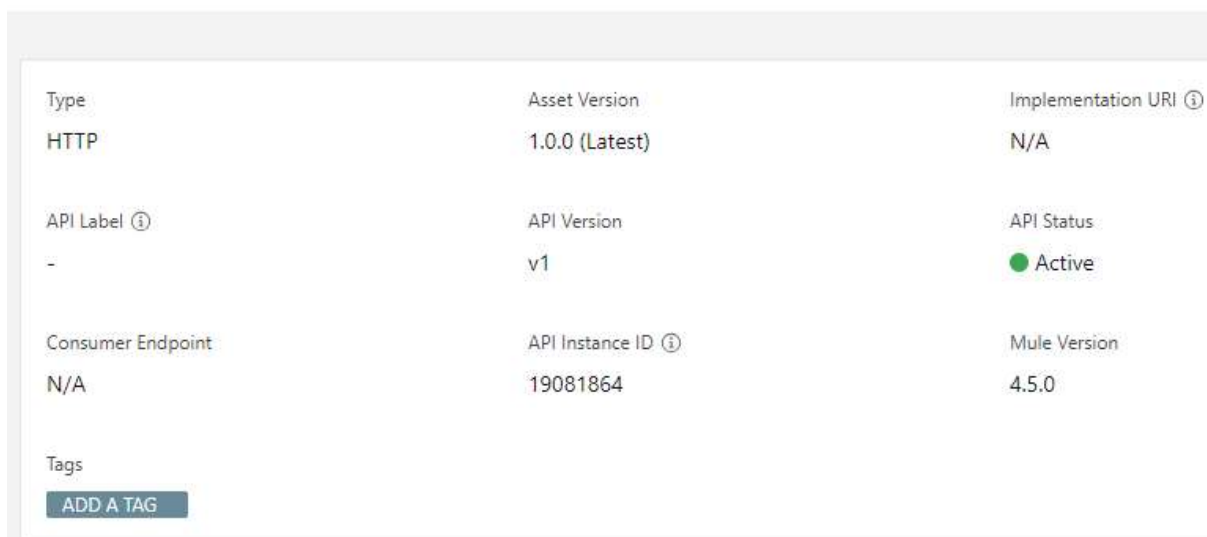
Runtime | **Properties** | Insight | Logging | Static IPs

Table view | Text view

anypoint.platform.analytics_base_uri	https://analytics-ingest.anypoint.mulesoft.com/	✕
anypoint.platform.client_id	ca495cf650c045968c5f6578cc36d870	✕
anypoint.platform.client_secret	38008e1b7b2f469988D0a4fe77578e20	✕

17. Now, within the Advanced REST Client a new tab is added, in the search field the url copied is pasted and click on **Send the request**.
18. Now, click on the **Anypoint Platform menu** and then click on **Management Center > API Manager** and then click on **hellomule**. And we validate that the API status be active.

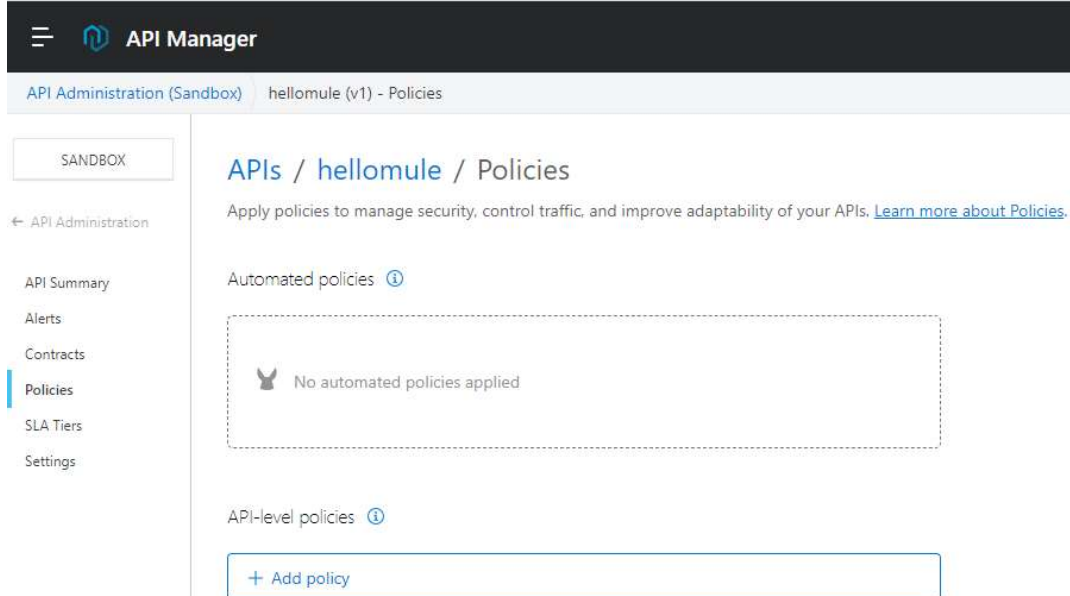
[APIs](#) / [hellomule](#) / API Summary



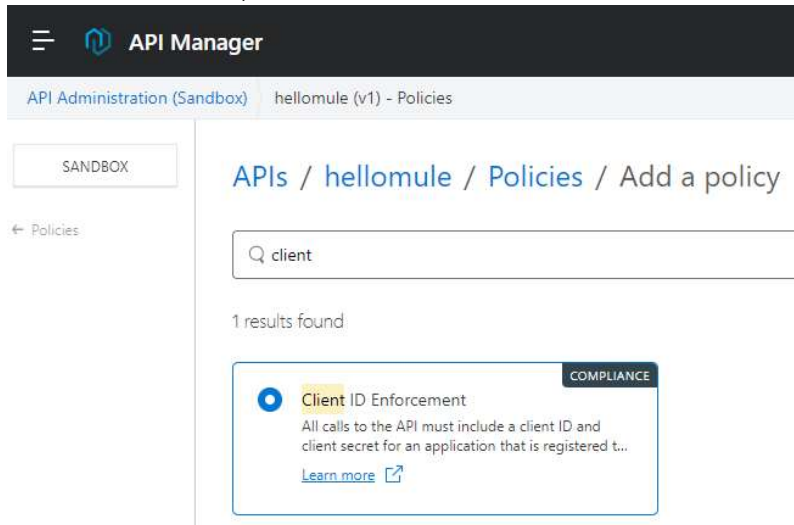
Type	Asset Version	Implementation URI ⓘ
HTTP	1.0.0 (Latest)	N/A
API Label ⓘ	API Version	API Status
-	v1	● Active
Consumer Endpoint	API Instance ID ⓘ	Mule Version
N/A	19081864	4.5.0
Tags	ADD A TAG	

APPLY CLIENT ID ENFORCEMENT POLICY TO API IN API MANAGER

1. Now, click on the **Anypoint Platform** menu and click on **Management Center > API Manager** and then click on **hellomule**. And then we go to where it says **Policies**. And click on **Add policy**.



2. Inside the window, search for **Client ID Enforcement** in the field. And then you click on **Next**.



3. Now within this window, click on the **HTTP Basic Authentication Header** check, expand **Advanced options** and select the latest version, and also click on the **Apply configuration to all API method & resources** check, and then click on **Apply**.

API Manager

API Administration (Sandbox) | hellomule (v1) - Policies

SANDBOX

← Policies

Credentials origin
Origin of the Client ID and Client Secret credentials.

☒ HTTP Basic Authentication Header
☐ Custom Expression

Advanced options ▾
Configure policy version, methods and resources

Policy version *
1.3.2 (latest) ▾ ⓘ

Method & resource conditions

☒ Apply configuration to all API method & resources
☐ Apply configuration to specific API method & resources

Previous Apply

4. Now, within **Advanced REST Client**, we add a new tab, we paste in the search field the url copied and click on **Send the request**. As you can see in the image below, the status result is **401 Unauthorized** and the error is **"Authentication denied"**.

API Client Environment: Default ▾

.../localhost:8081/hellomule × ...e2.cloudhub.io/hellomule × +

GET ▾ http://hellomule-test456.us-e2.cloudhub.io/hellomule ✎ ▶ ⋮

HEADERS AUTHORIZATION 0 ACTIONS 0 CONFIG CODE SNIPPETS

COPY ☐ Text editor

Add a header to the HTTP request.

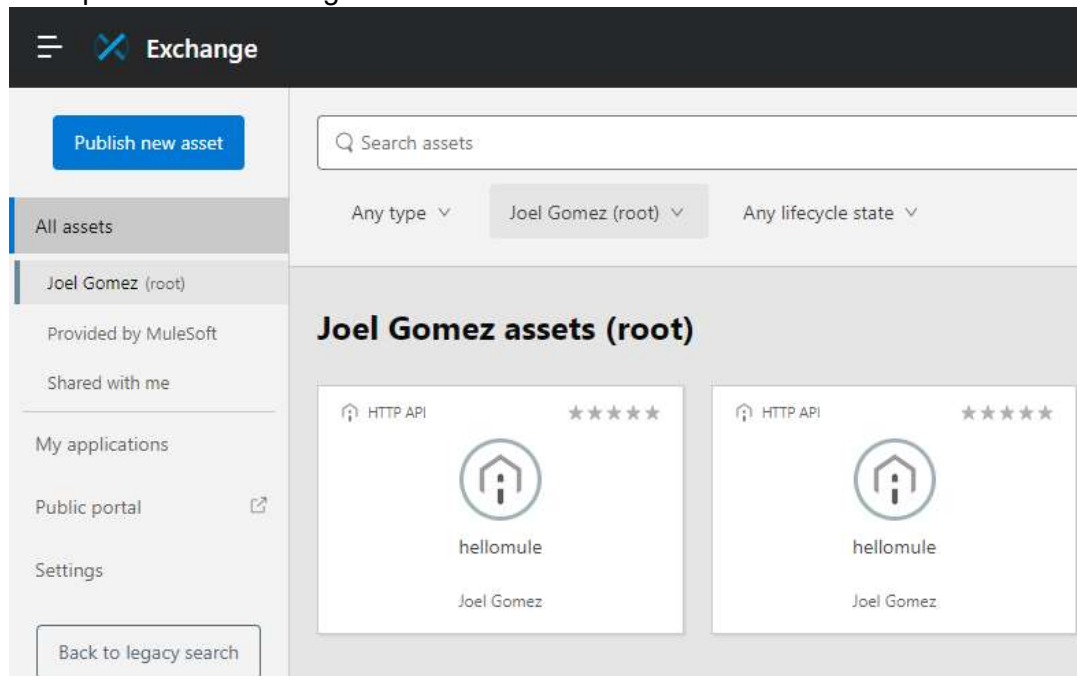
+ ADD

⋮ Response × CLEAR

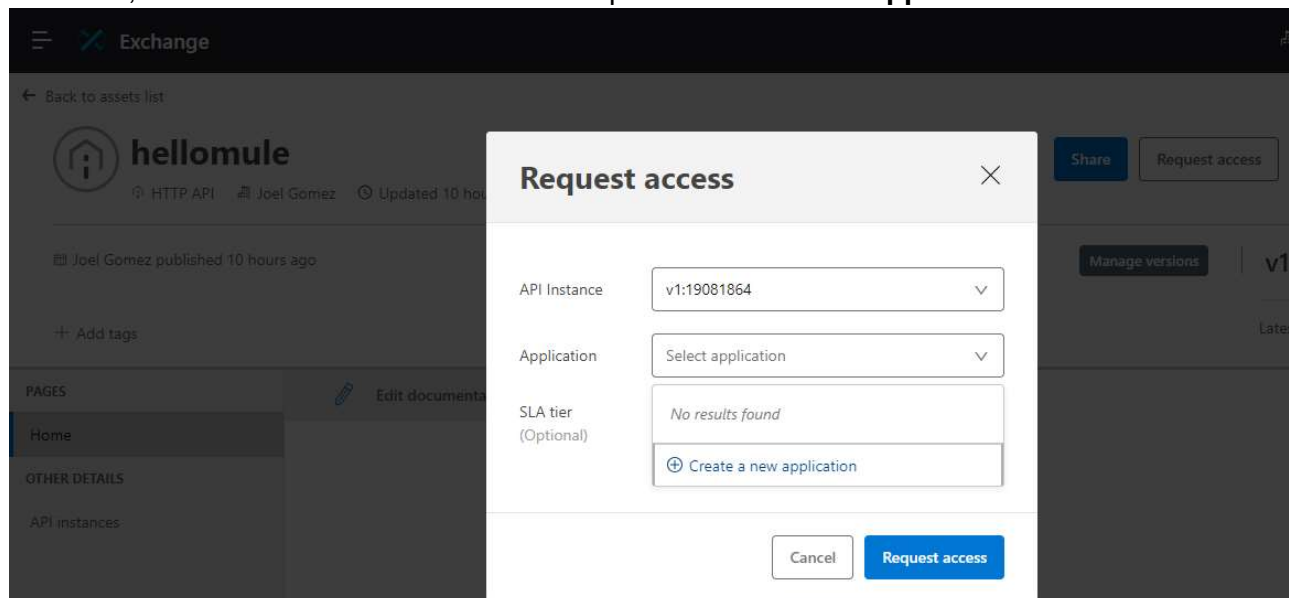
401 Unauthorized Time: 1528 ms Size: 39 Bytes ⋮

```
1 {
2   "error": "Authentication denied."
3 }
```


5. Now, on Anypoint Platform, we are heading to **Exchange**. We can see that the HTTP API has already been published to the organization's assets.



6. We select the corresponding API, within that window click on **Request Access**, it is selected within **API Instance**, and within **SLA Tier** we select the option **Create a new application**.



7. In the **Application Name** field, we enter *hellomuleapp*, click on **Create**, and then click on **Request Access**.

Create new application

Application Name

hellomuleapp

Description
(Optional)

Write a description here...

Application URL
(Optional)

https://yourcompany.com/applicationURL

OAuth 2.0 redirect URIs ⓘ
(Optional)

https://yourcompany.com/callback

Cancel

Create

Request access



API Instance

v1:19081864



Application

hellomuleapp



SLA tier
(Optional)

Instance does not have SLA tiers




Cancel

Request access

8. Those login credentials are considered when these are requested in the REST client.

Request API access



Your request has been received and approved.

Use this client ID and client secret to access the requested API instance. You can always find these values at the [application page](#).

Client ID: 0d67dda2969b4b92ac84937a2f54d41c




Client Secret: a99F13Aec4b74747a0fA29998475123d

Close

9. Now in Advanced REST Client, go to the **AUTHORIZATION** tab and paste the username and password from the previous step, click on **send the request** again where the link is requested, and you see that the application is run with **200 OK** as status.

API Client Environment: Default ▾

.../localhost:8081/hellomule × ...e2.cloudhub.io/hellomule × +


GET ▾ http://hellomule-test456.us-e2.cloudhub.io/hellomule   

HEADERS AUTHORIZATION 0 ACTIONS 0 CONFIG CODE SNIPPETS

Select authorization ▾
Basic

Basic authorization allows to send a username and a password in a request


User name
0d67dda2969b4b92ac84937a2f54d41c

Password
a99F13Aec4b74747a0fA29998475123d 

⋮ Response ×

CLEAR

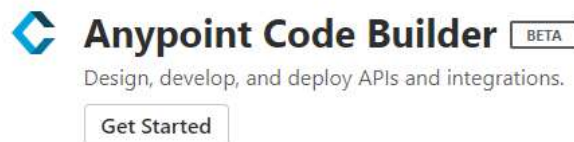
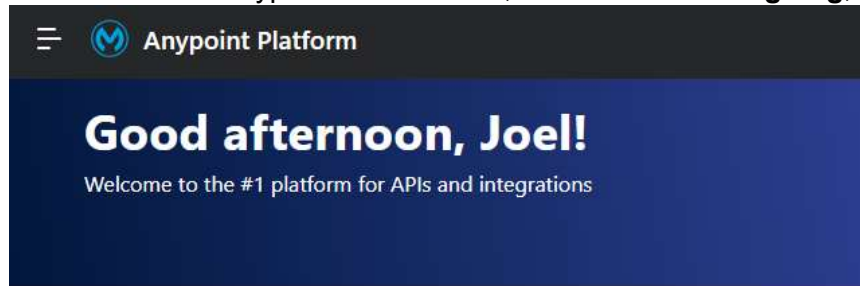
200 OK

Time: 250 ms Size: 10 Bytes 

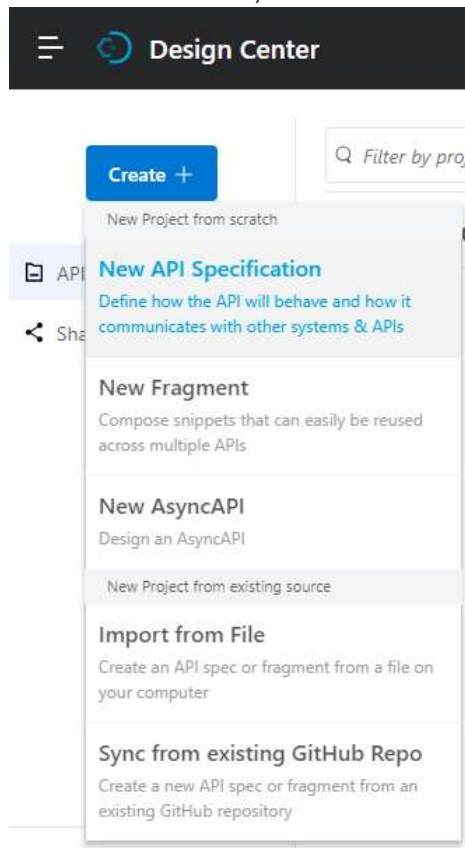
1 | Hello Mule

LEARN ABOUT THE FIRST API SPECIFICATION AND BUILD AN API SPECIFICATION IN API DESIGNER.

1. Within the main Anypoint Platform site, click on **Start Designing**, in the **Design Center** section.



2. Now in the window, click on **Create** and from there click on **New API Specification**.



3. In the **Project Name** field, we enter *NTO Customer Database API 1* and check the **Guide me through it** box, then click on **Create API**.

New API specification

Project name (required)

NTO Customer Database API

How do you want to draft the API Spec?



I'm comfortable designing it on my own

A complete code editing experience with interactive documentation

Specification Language

RAML 1.0



Guide me through it

Use a visual interface scaffolding the API Specification (can generate both RAML & OAS)

4. We have to configure the fields in the window, according to the image.

Design Center

Undo Redo Saved a few seconds ago

NTO Customer Database API 1

Q Filter

API Summary

Add a new Security Scheme, Resource or Data type by clicking on the + buttons or import an existing asset from Exchange.

Import from Exchange

SECURITY SCHEMES

RESOURCES

DATA TYPES

GROUPS

API Summary

Title NTO Customer Database API 1 Version 1.0.0

Protocols HTTP x HTTPS x Media type application/json x

Base URI api.samplebaseurl.com Base URI Parameters (0) >

Description

B I » </> ☰ ☷ H

Markdown Visual

NTO Customer Database API 1, to access NTO's customer records.

5. Then, click on **+** next to **Data Types**. Then, we type *adress* in the **Name** field, and now in **Description** field we type *Adress data type*. They are added as properties: *Street*, *city*, *postalCode*, *state*, and *country*. All as a **String** type, if the **Required** boxes are not checked.

Design Center

Undo Redo Saved 4 minutes ago NTO Customer Database API 1

Filter

API Summary

- SECURITY SCHEMES +
- RESOURCES +
- DATA TYPES +
- adress**
- GROUPS +

Create groups to semantically group resources and data types in your API specification.

Data types

Name	Type	Union
adress	Object	<input type="checkbox"/>

Description

B I » </> |≡ ≡≡ H **Markdown** Visual

Adress data type

Property Name	Required	Type	Union	⌵
street	<input type="checkbox"/>	Stri...	<input type="checkbox"/>	
city	<input type="checkbox"/>	Stri...	<input type="checkbox"/>	

6. We write the following lines of code into the **Example** field, as long as we click on **Edit**.

```
{  
  "street": "44 Shirley Ave.",  
  "city": "West Chicago",  
  "postalCode": "60185",  
  "state": "IL",  
  "country": "USA"  
}
```

Property Name

country

Required

Type

Stri...

Union

Add Property

Example

Inherited **Edit**

```
{  
  "street": "44 Shirley Ave.",  
  "city": "West Chicago",  
  "postalCode": "60185",  
  "state": "IL",  
  "country": "USA"  
}
```

7. Step 5 is repeated, but now in the **Name** field we type *Contact*, and we modify on the **Type** and **Description** fields with the values of *Object* and *Contact Data Type*.

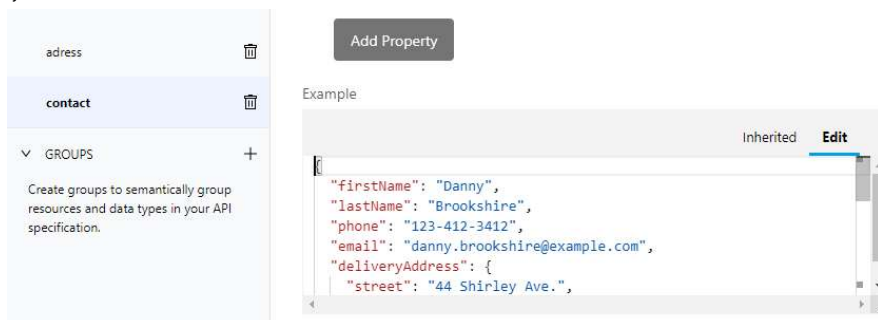
The screenshot shows the 'Design Center' interface for 'NTO Customer Database API 1'. On the left, a sidebar contains a search bar and a list of API elements: 'API Summary', 'SECURITY SCHEMES', 'RESOURCES', 'DATA TYPES' (expanded), 'adress', and 'contact'. The 'contact' data type is selected. The main area is titled 'Data types' and contains a form for 'contact'. The 'Name' field is 'contact', the 'Type' is 'Object', and the 'Description' field contains 'Contact data type' in a rich text editor with 'Markdown' and 'Visual' tabs.

8. We should add as properties *FirstName*, *lastName*, *phone*, *email*, *deliveryAddress*, and *postalAddress*. The only ones that are of type **Address** are *deliveryAddress* and *postalAddress*, and the only one that is marked in **Required** is *lastName*.

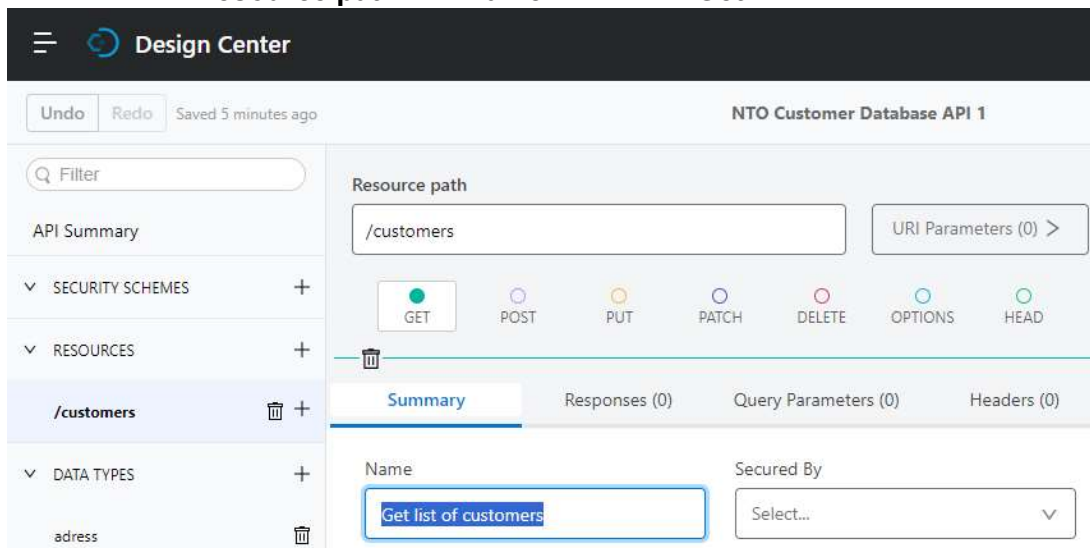
The screenshot shows the 'Design Center' interface for 'NTO Customer Database API 1'. On the left, a sidebar contains a search bar and a list of API elements: 'API Summary', 'SECURITY SCHEMES', 'RESOURCES', 'DATA TYPES' (expanded), 'adress', 'contact', and 'GROUPS'. The 'contact' data type is selected. The main area is titled 'Data types' and contains a list of properties for 'contact'. The properties are: 'lastName' (Required, Type: String), 'phone' (Not Required, Type: String), 'email' (Not Required, Type: String), 'deliveryAddress' (Not Required, Type: Address), and 'postalAddress' (Not Required, Type: Address). An 'Add Property' button is at the bottom.

9. Now, we type the following lines of code into the **Example** field, if we click on **Edit**.

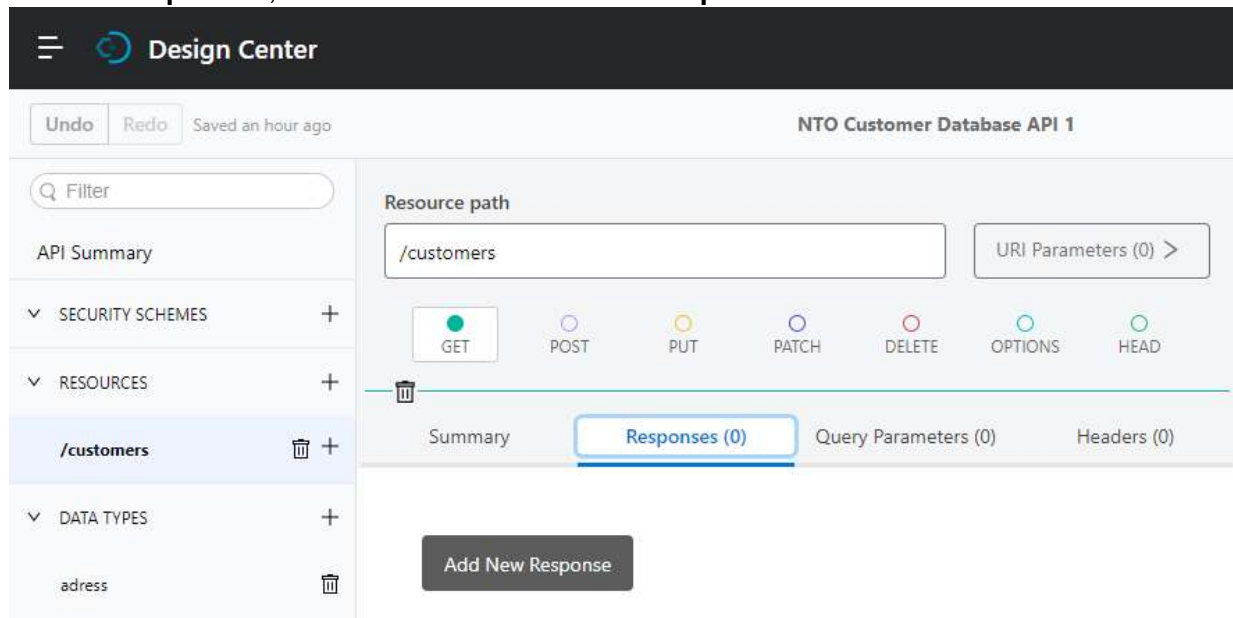
```
{
  "firstName": "Danny",
  "lastName": "Brookshire",
  "phone": "123-412-3412",
  "email": "danny.brookshire@example.com",
  "deliveryAddress": {
    "street": "44 Shirley Ave.",
    "city": "West Chicago",
    "postalCode": "60185",
    "state": "IL",
    "country": "USA"
  },
  "postalAddress": {
    "street": "44 Shirley Ave.",
    "city": "West Chicago",
    "postalCode": "60185",
    "state": "IL",
    "country": "USA"
  }
}
```



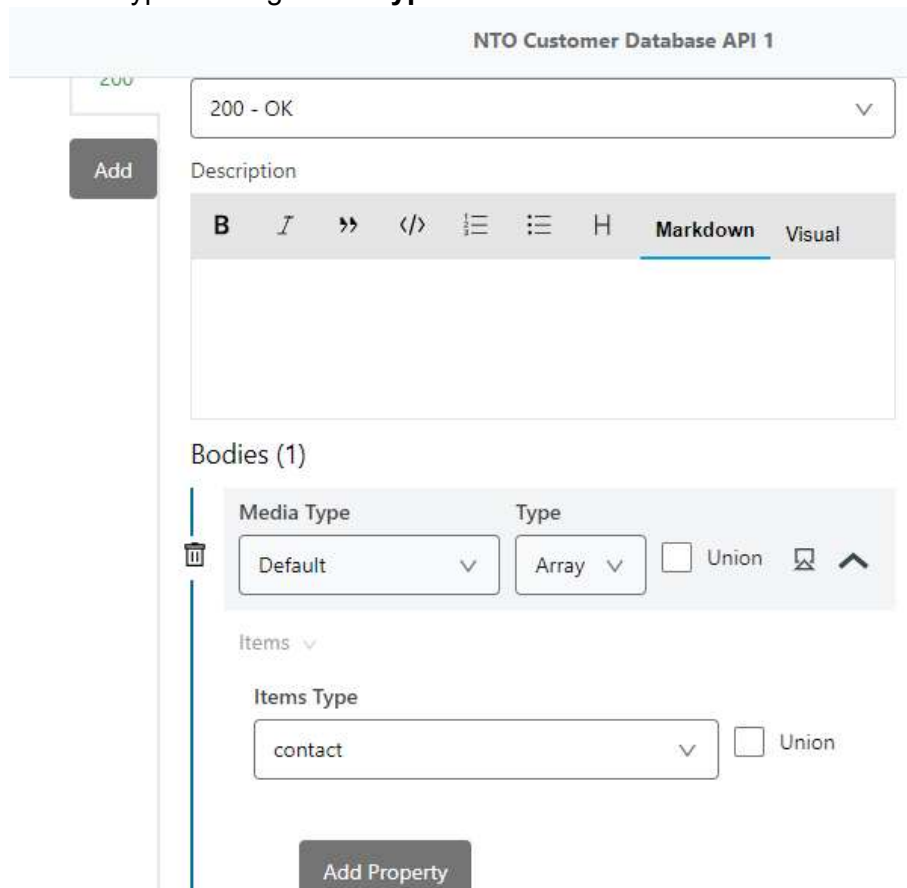
10. On the **Resources** tab, click on **+**. Subsequently, the values of `/customers` and *Get list of customers* are entered in the **Resource path** and **Name** fields. The **Get** box is checked.



11. Click on **responses**, and then click on **Add New Response**.



12. Inside the window, we check that the status is 200 OK, click on **Add Body**. **Media Type** is set to **Default** (**application/json**), and **Type** to **Array**. Click on the downward-pointing arrow on the right, and the **contact** type is assigned to **Type**.



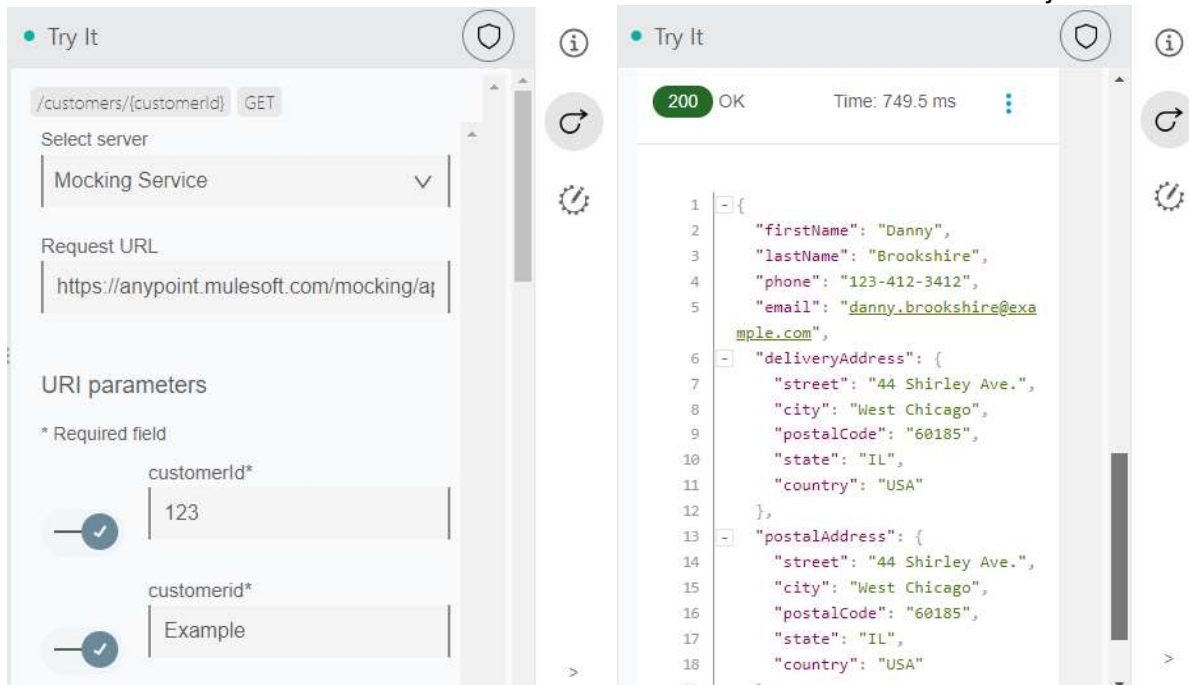
13. Then click on **+** next to **RESOURCES**. And in the window we set the properties of the **Resource path** as `/customers/{customerId}`, and in **name** we assign the value of *Get customer by ID* and check the **Get** box. And click on **URI Parameters**, and click on **URI Parameters** and we assign the values of *customerId* and *String* on the **URI Parameter Name** and **Type** fields. The **Required** box is checked.

The screenshot shows the API design tool interface. On the left is a sidebar with a search bar and a tree view containing 'API Summary', 'SECURITY SCHEMES', 'RESOURCES', 'DATA TYPES', and 'GROUPS'. The 'RESOURCES' section is expanded, showing a list of resources: '/customers', '/customers/{customerId}', 'adress', and 'contact'. The resource '/customers/{customerId}' is selected. The main panel displays the configuration for this resource. The 'Resource path' is set to '/customers/{customerId}'. Below it, the 'URI Parameters (1)' section shows a table with one parameter: 'customerId' of type 'Str...' (String), which is marked as 'Required'. Below the table is an 'Add URI Parameter' button. A row of HTTP methods (GET, POST, PUT, PATCH, DELETE, OPTIONS, HEAD) is shown, with 'GET' selected. At the bottom, the 'Summary' tab is active, showing the 'Name' as 'Get customer by ID' and 'Secured By' as 'Select...'. The 'Responses (0)', 'Query Parameters (0)', and 'Headers (0)' tabs are also visible.

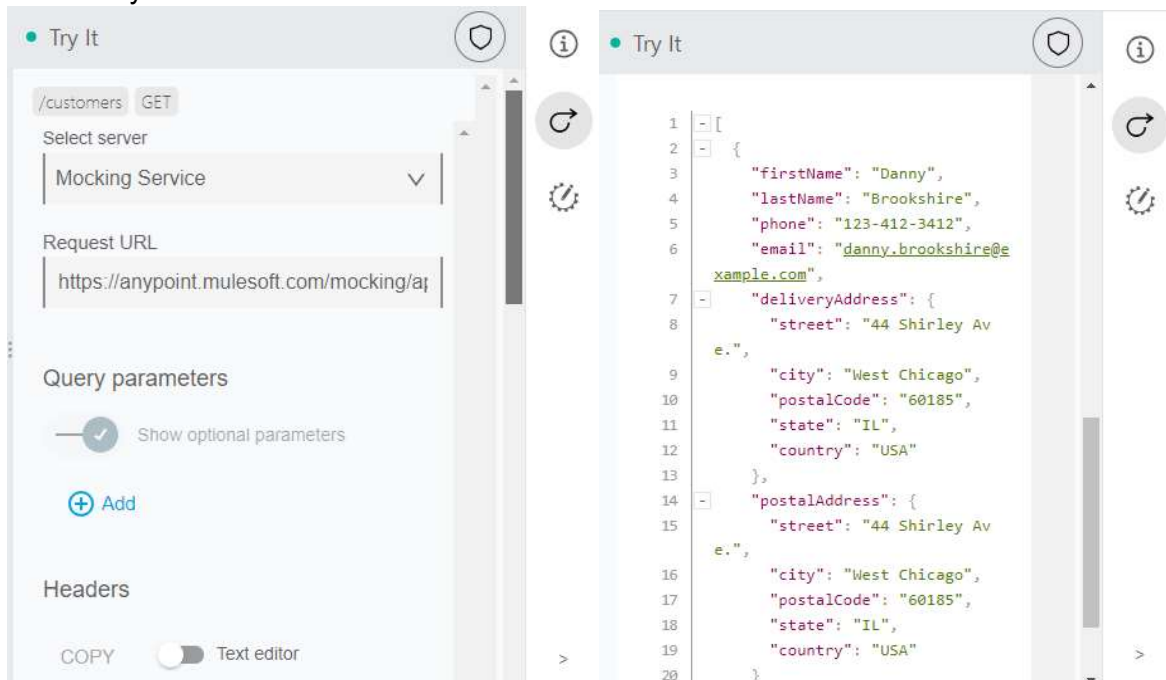
14. On the **Responses** tab, click on **Add new response** and then click on **Add body**, and we assign *Default* in **Media Type**, we assign *contact* in the **Type** field.

The screenshot shows the API design tool interface with the 'Responses (1)' tab selected. A new response is being added with a status of '200'. The 'Status' dropdown is set to '200 - OK'. The 'Description' field is empty. Below the description, the 'Bodies (1)' section shows a table with one body: 'Default' of type 'cont...' (contact), which is marked as 'Required'. Below the table is an 'Add Body' button. The 'Summary', 'Query Parameters (0)', and 'Headers (0)' tabs are also visible.

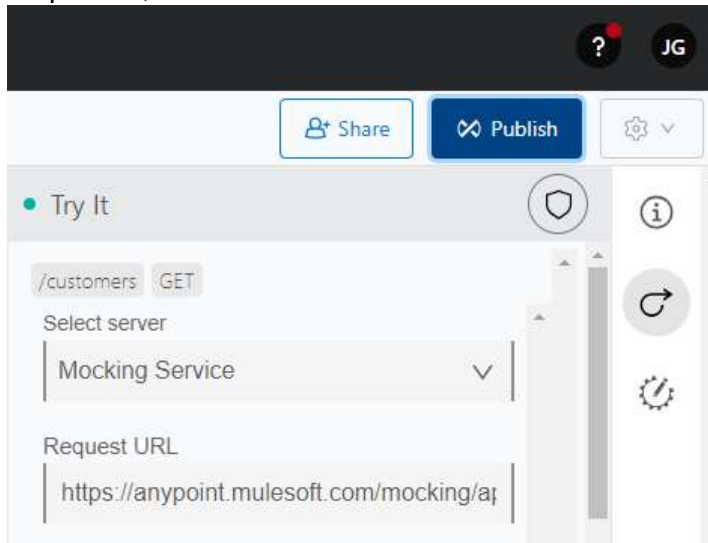
15. Now, we have to test. To do this, click on **/customers/{customerId}**, then click on **Try It** (from the bar to the right of the interface), once in the window, configure **Select server** as **Mocking service**, and enter **123** in **URI Parameters**. Then click on **send**. We can see that the result is an object.



16. Then click on **/customers**, then click on **Try It** (from the bar to the right of the interface), once in the window, we configure **Select server** as **Mocking service**. Then click on **send**. We can see that the result is an array.



17. To publish, click on the **Publish** button.

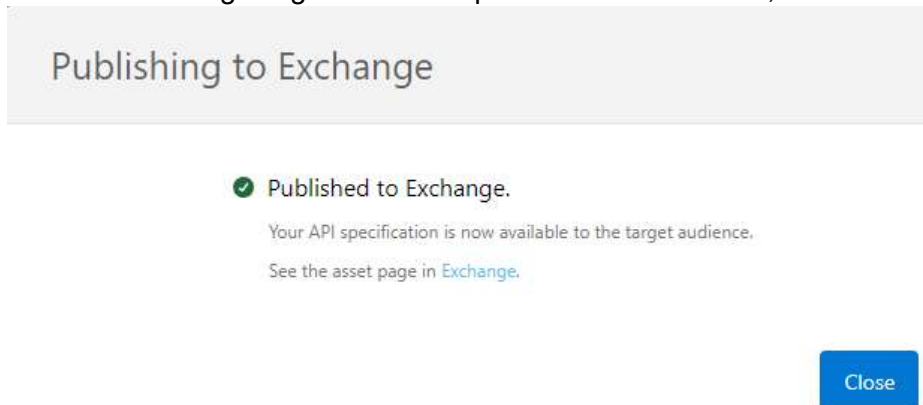


18. Then, we make sure that in **Asset Version** is **1.0.0**. Then, click on **Publish to Exchange**.

Publishing to Exchange

Asset version (required)	About asset versioning
<input type="text" value="1.0.0"/>	To publish to Exchange, please, use Semantic Versioning. Examples of good versions are 1.0.0 or 4.3.1.
API version (required)	More help
<input type="text" value="v1"/>	<ul style="list-style-type: none">• Changing a project's main/root file• What is an API version?
LifeCycle State	The lifecycle state of an asset shows its status in the software development lifecycle, from development to stable releases to deprecation. Learn more
<input checked="" type="radio"/> Stable State of release, ready to consume	
<input type="radio"/> Development	
<input type="button" value="Cancel"/>	<input type="button" value="Publish to Exchange"/>

19. Once the message is given that the publication is available, click on **Exchange**.



20. There in the window, click on **Summary** of the right panel. And we can see that inside of **API Endpoints** are **/customers** and **/customers/{customerId}**.

PAGES

Home

SPECIFICATION

Summary

Endpoints

/customers

Overview

GET Get list of customers

/customerId

Types

OTHER DETAILS

Conformance Status

API instances

API specification summary

API title: NTO Customer Database API 1

Version: 1.0.0

NTO Customer Database API 1, to access NTO's customer records.

`http://api.samplebaseurl.com`

Supported protocols

HTTP HTTPS

API endpoints

[/customers](#)

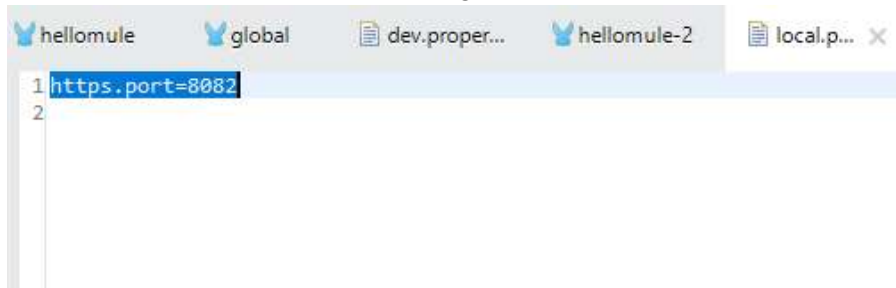
GET

[/customers/{customerId}](#)

GET

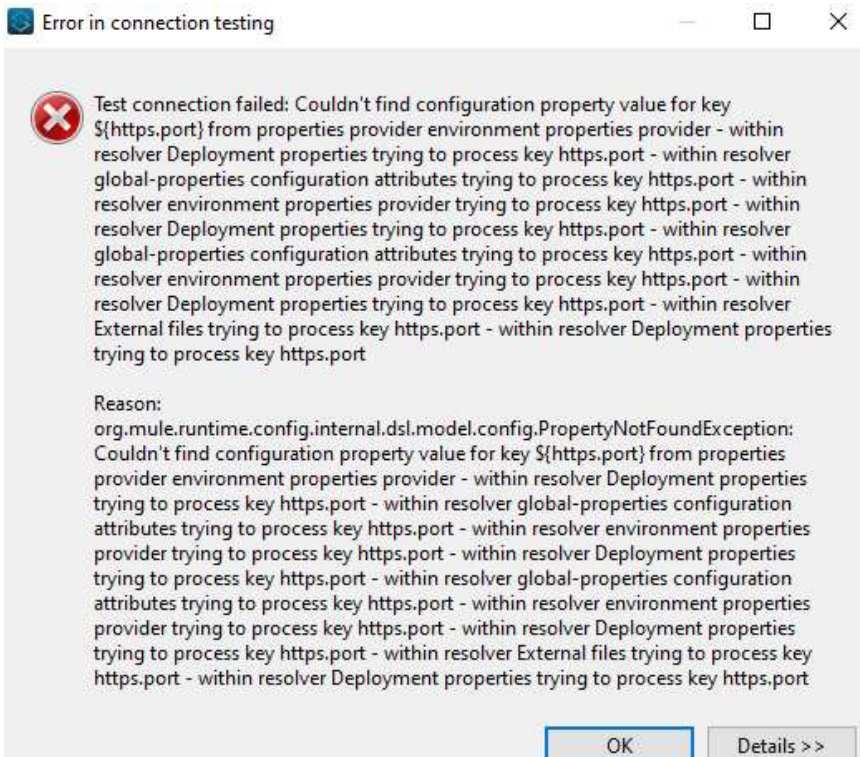
ENABLE MULE APPLICATION CONSUMPTION THROUGH HTTPS.

1. Repeat the first exercise, but with the difference that the project will be named *hellomule-2*.
2. From the left panel of Anypoint Studio, right click on **src/main/resources**, and then **New > File**.
3. Then we assign on the **File name** field as *local.properties*.
4. Inside the file, we write the following: `https.port=8082`.



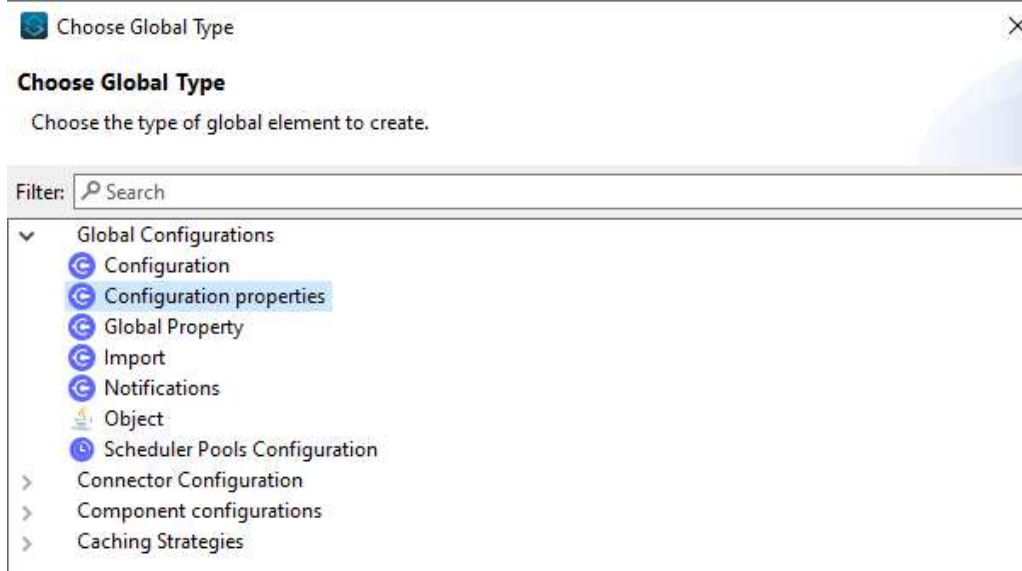
5. Now in `hellomule-2.xml` we will go to the **Global Elements** tab, and from there click on **Edit**. Later, we edit the **Protocol** and **Port** fields, as `HTTPS` and `${https.port}`. Then click on **Test connection**.

The screenshot shows the 'Global Element Properties' dialog box for the 'HTTP Listener config'. The 'General' tab is selected. The 'Name' field is 'HTTP_Listener_config'. The 'Connection' section is expanded, showing the 'General' sub-tab. The 'Protocol' is set to 'HTTPS', the 'Host' is 'All Interfaces [0.0.0.0] (default)', and the 'Port' is set to the property placeholder '\${https.port}'. The 'Read timeout' field is empty. Below the 'Connection' section, the 'General' sub-tab is also visible, showing the 'Base path' field and the 'Listener interceptors' dropdown set to 'None'. There is an unchecked checkbox for 'Reject invalid transfer encoding'. At the bottom, there is a 'Test Connection...' button, which is highlighted with a blue dashed border, and 'OK' and 'Cancel' buttons.

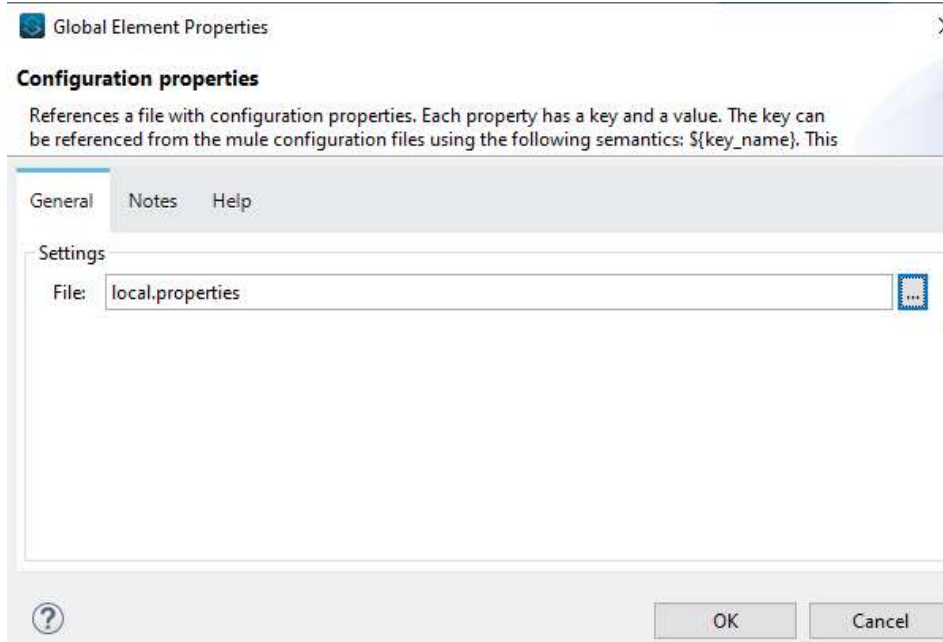


Then click **OK**.

6. Then click on **Create**, within **Global Elements**, and consider **Configuration Properties**.

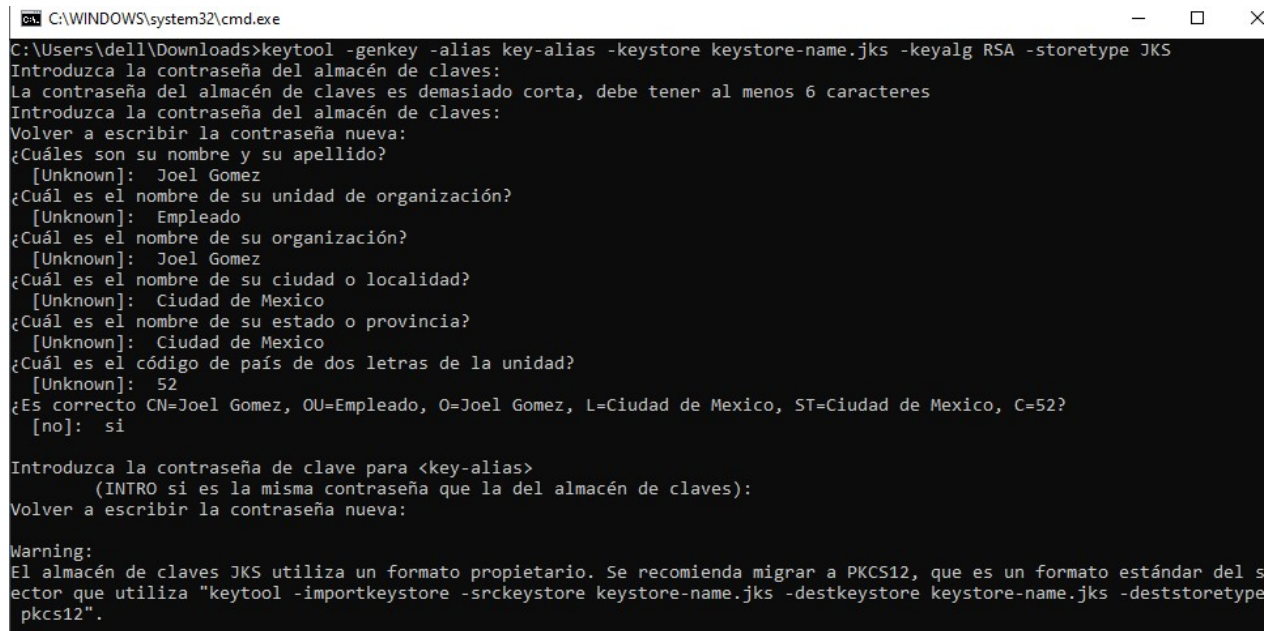


7. Once we clicked on **OK**, in the **File** field we enter *local.properties*. And click on **OK**.



8. Now at Command Prompt, the following line of code is typed. Fill in the corresponding fields and consider that we have saved the password typed.

```
keytool -genkey -alias key-alias -keystore keystore-name.jks -keyalg RSA -storetype JKS
```

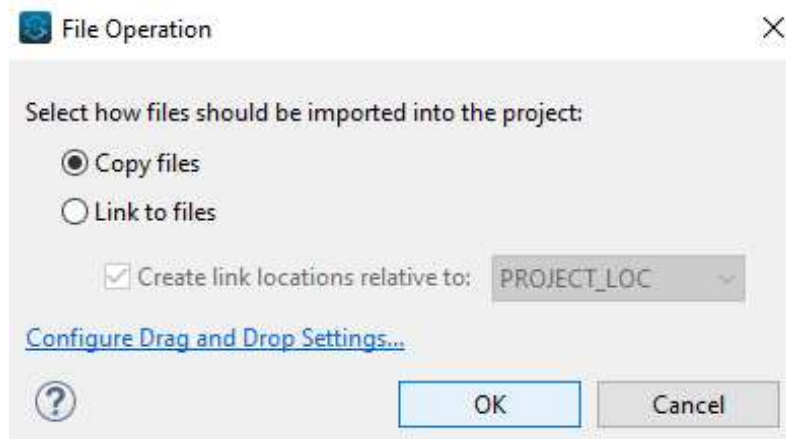


```
C:\WINDOWS\system32\cmd.exe
C:\Users\dell\Downloads>keytool -genkey -alias key-alias -keystore keystore-name.jks -keyalg RSA -storetype JKS
Introduzca la contraseña del almacén de claves:
La contraseña del almacén de claves es demasiado corta, debe tener al menos 6 caracteres
Introduzca la contraseña del almacén de claves:
Volver a escribir la contraseña nueva:
¿Cuáles son su nombre y su apellido?
[Unknown]: Joel Gomez
¿Cuál es el nombre de su unidad de organización?
[Unknown]: Empleado
¿Cuál es el nombre de su organización?
[Unknown]: Joel Gomez
¿Cuál es el nombre de su ciudad o localidad?
[Unknown]: Ciudad de Mexico
¿Cuál es el nombre de su estado o provincia?
[Unknown]: Ciudad de Mexico
¿Cuál es el código de país de dos letras de la unidad?
[Unknown]: 52
¿Es correcto CN=Joel Gomez, OU=Empleado, O=Joel Gomez, L=Ciudad de Mexico, ST=Ciudad de Mexico, C=52?
[no]: si

Introduzca la contraseña de clave para <key-alias>
(INTRO si es la misma contraseña que la del almacén de claves):
Volver a escribir la contraseña nueva:

Warning:
El almacén de claves JKS utiliza un formato propietario. Se recomienda migrar a PKCS12, que es un formato estándar del s
ector que utiliza "keytool -importkeystore -srckeystore keystore-name.jks -destkeystore keystore-name.jks -deststoretype
pkcs12".
```

9. Now, the **keystore-name.jks** file is copied to Anypoint Studio, in the **src/test/resources** folder. Then click on **OK**.



10. Now you go back to **HTTP Listener Config**, and from there we click on **Edit**. We change to the **TLS** tab and within **TLS Configuration** we assign **Edit Inline**. Scroll down to the **Key Store Configuration** section, and edit the **Path**, **Alias**, **Key Password**, and **Password** fields. And then click on **Test Connection**.

Global Element Properties

HTTP Listener config
Configuration element for a HttpListener.

General Notes Help

Algorithm:

☐ Insecure

Key Store Configuration

Type:

Path:

Alias:

Key Password: ☒ Show password

Password: ☒ Show password

Algorithm:

Advanced

Enabled Protocols:

Enabled Cipher Suites:

General

Base path:

Test Connection... OK Cancel

11. When the **Connection successful** pop-up appears, click on **OK**.
12. Then, click on **save all**, and from there we go to **Message Flow**, right-click on the canvas and click on **Run Project**. Wait until the console says **DEPLOYED**.
13. Now in the REST client, we click on add new tab, and we type `https://0.0.0.0:8082/hellomule` in the **search field**, and at the end we click on **send**.

API Client Environment: Default

.../localhost:8081/hellomule × ...e2.cloudhub.io/hellomule × ...//0.0.0.0:8082/hellomule × +

GET

HEADERS AUTHORIZATION 0 ACTIONS 0 CONFIG CODE SNIPPETS

COPY ☐ Text editor

Add a header to the HTTP request.

+ ADD

Response × CLEAR

200 Time: 560 ms Size: 12 Bytes

1 | hello mule 2