

Introduction:

The Customer Data Retrieval API enables programmatic access to the company's customer database such as name, lastname, email, and the company they are associated with, which can be used for both internal and external use.

The Customer Data Retrieval API delivers information in JSON format ready for deserialization and quick understanding, which enables it to be used in multiple programming languages.

Authorization:

Authorization is required for the use of the API, this is implemented by using a username and password

All requests to the API must include a "Basic Auth" authorization header with the following data:

Username: 5161aa31f9b94a7b82b5e4d28bbdbd65

Password: eB07357Fb01E4b9594C91748CFe0F062

Customer:

-The customer resource

The "customer" resource is a JSON formatted version of a customer's data, the /customer endpoint responds with this format

Get Customer Data:

GET <https://practica-tecnica-daniel-gonzalez-garcia-24-f49kwq.5sc6y6-2.usa-e2.cloudhub.io/api/v1/sps/customers>

Response: Array of Customer

MuleSoft Technical Practice Trainee

We will start the process by creating the project within Anypoint Studio with the following configuration

New Mule Project

Project Settings
Create a Mule project in the workspace or in an external location.

Project Name:

Runtime
Mule Server 4.7.0 EE
[Install Runtimes](#)

API Implementation
Add an API implementation to your project to automatically set up an APIkit router and create placeholder flows for each resource method
☒ Scaffold flows from these API specifications

Import a published API | Import RAML from local file | Download RAML from Design Center

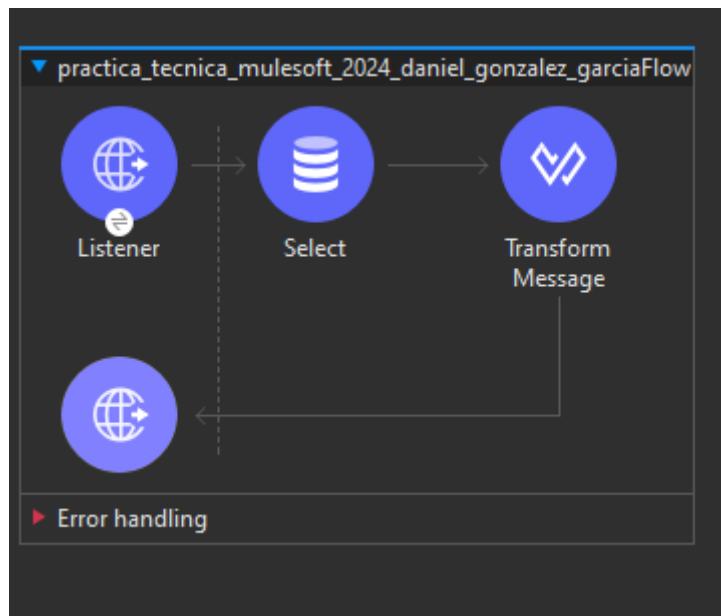
① Start building API implementations by importing the specification here. [Learn more](#)

+ ✕ 📄

Name	Asset type	Version
Please select or add a dependency to see more information.		

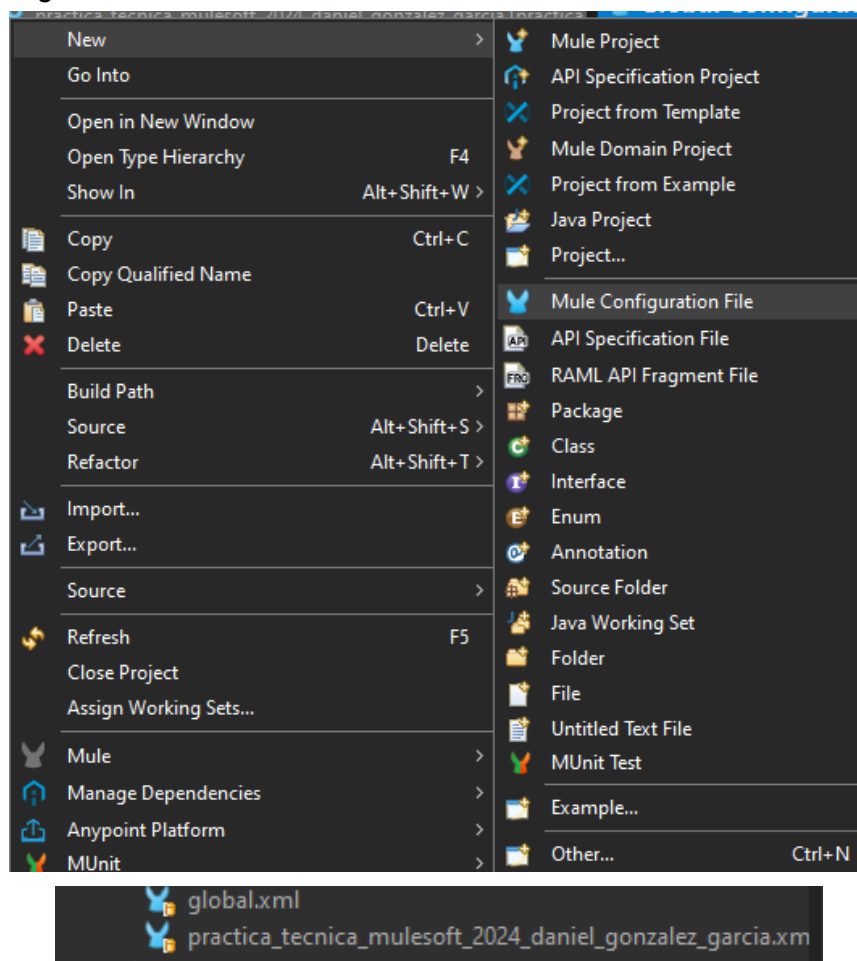
Project location
☒ Use default location
Location:

Once created, we will add three modules: "Listener", "Select", "Transform Message"

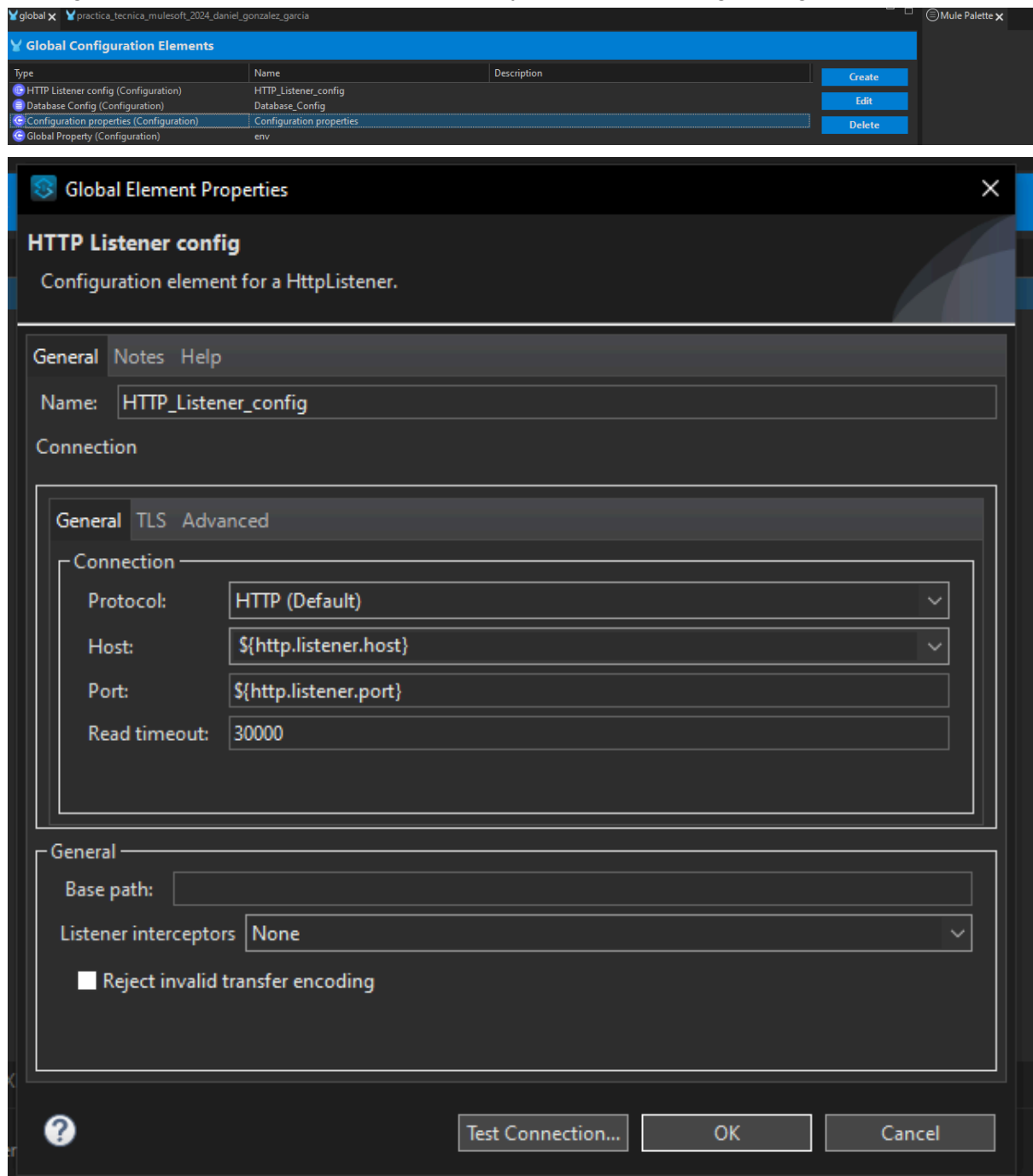


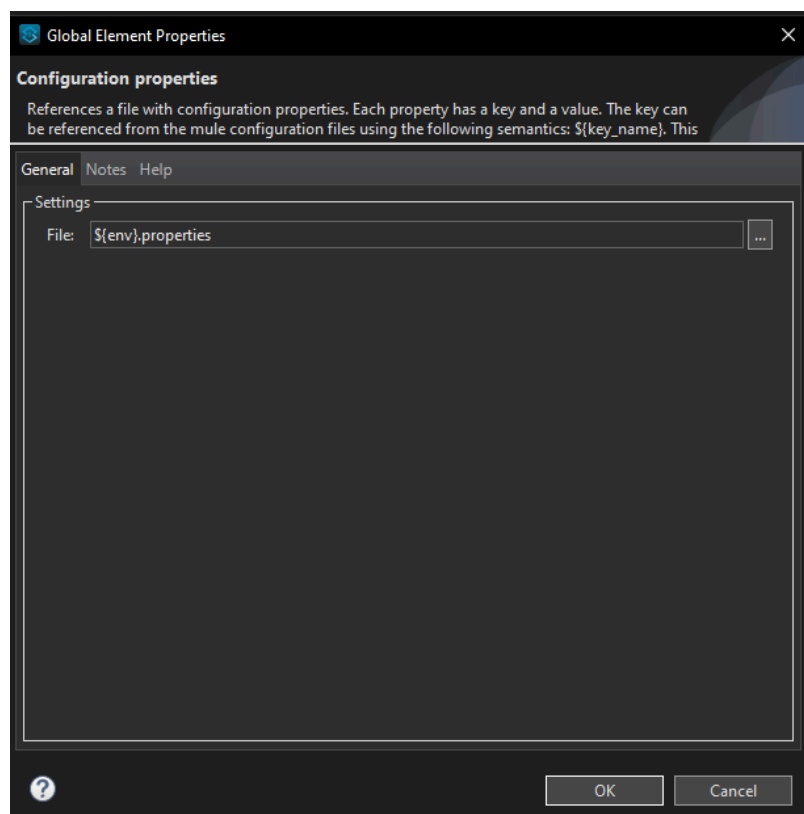
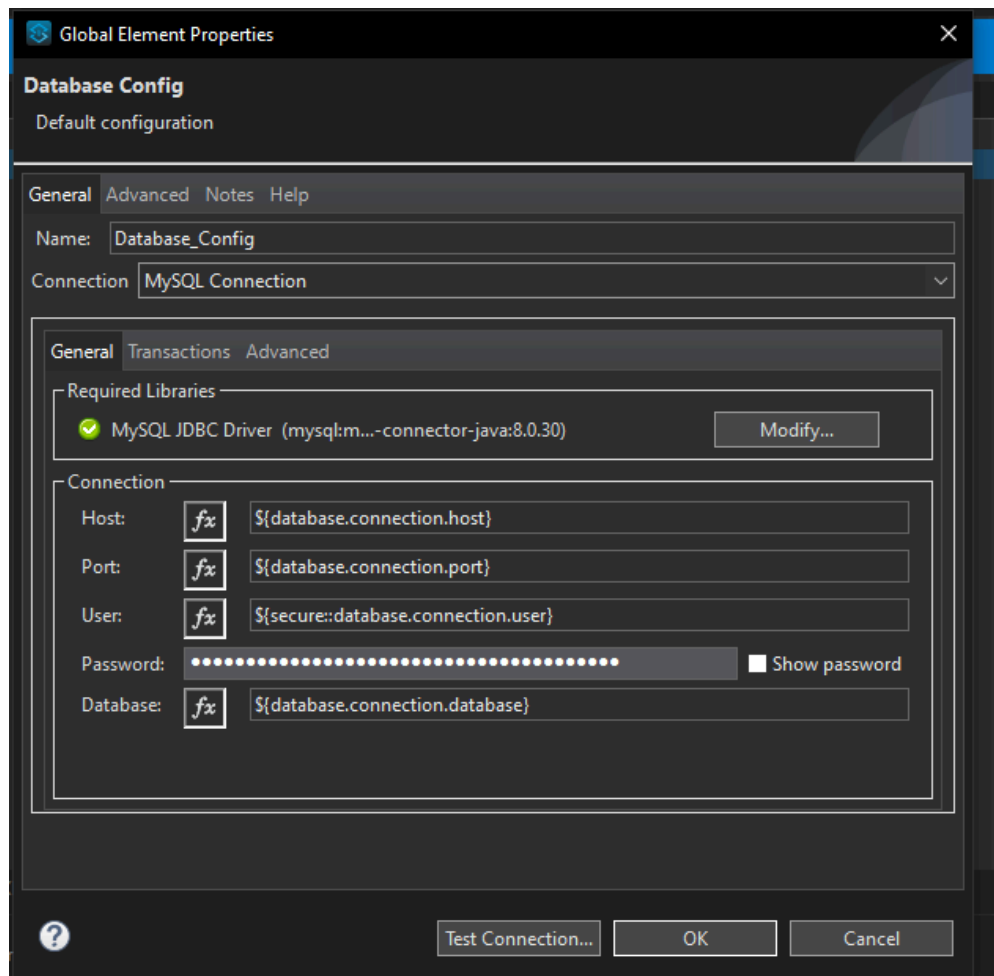
While "Listener" and "Transform Message" are in the default options in Mule Palette, "Select" is a module found inside "Database" just search for it in "Add Modules"

Now we will add a new configuration xml file that will serve as our global configuration file. with the name "global.xml"

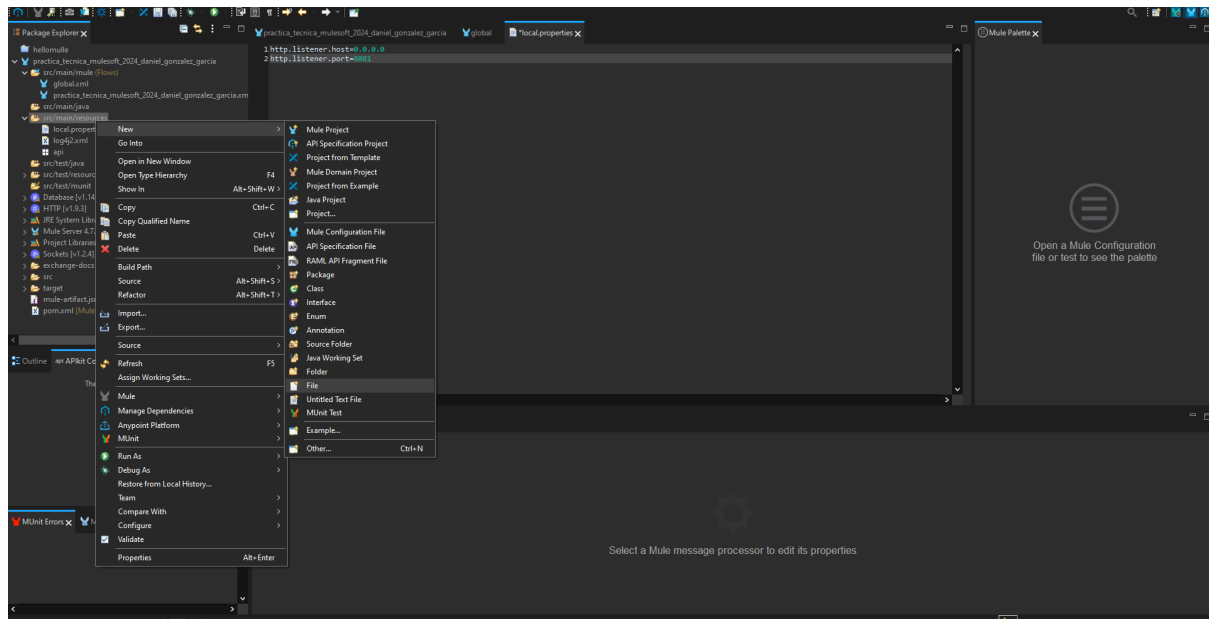


Here we will add three types of elements: "Http Listener config" "DataBase config" "Configuration Properties" and "GlobalProperty" with the following settings





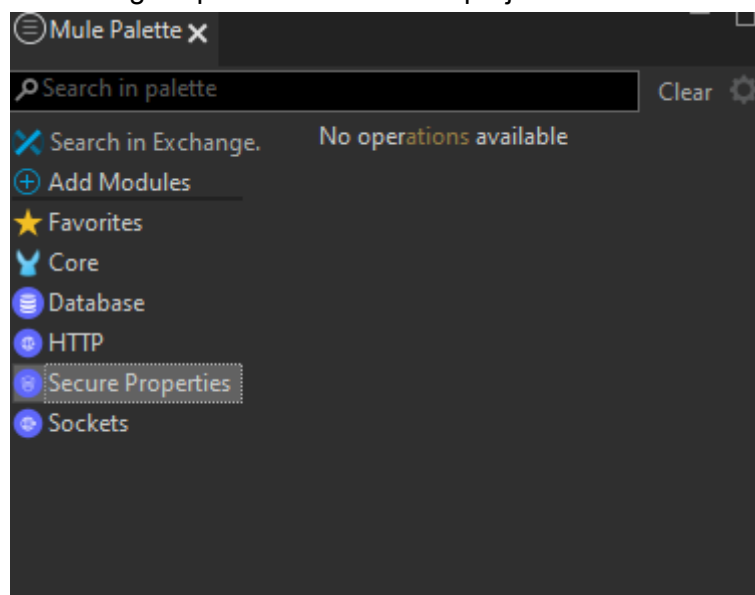
To store the configuration variables we will create the files "local.properties" and "dev.properties"

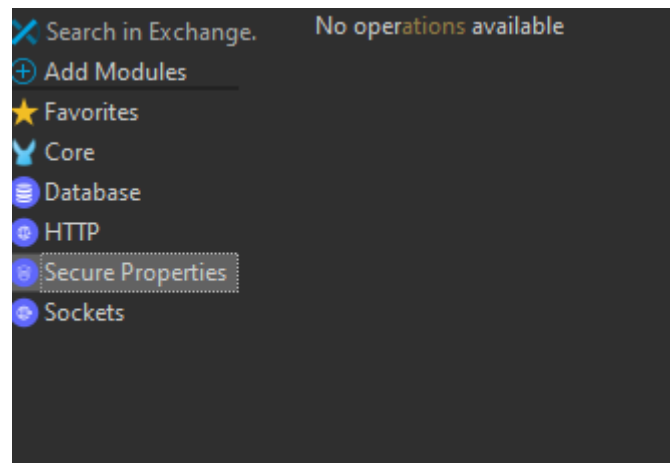
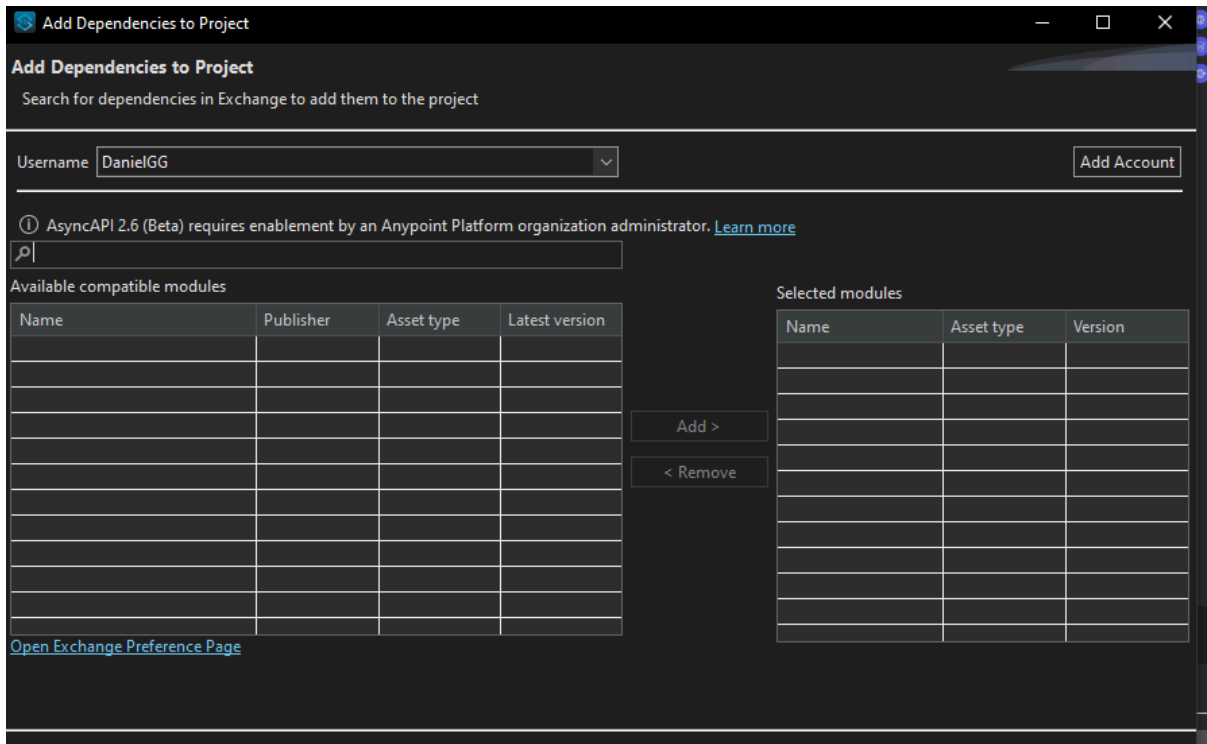


Where we will add the following data:

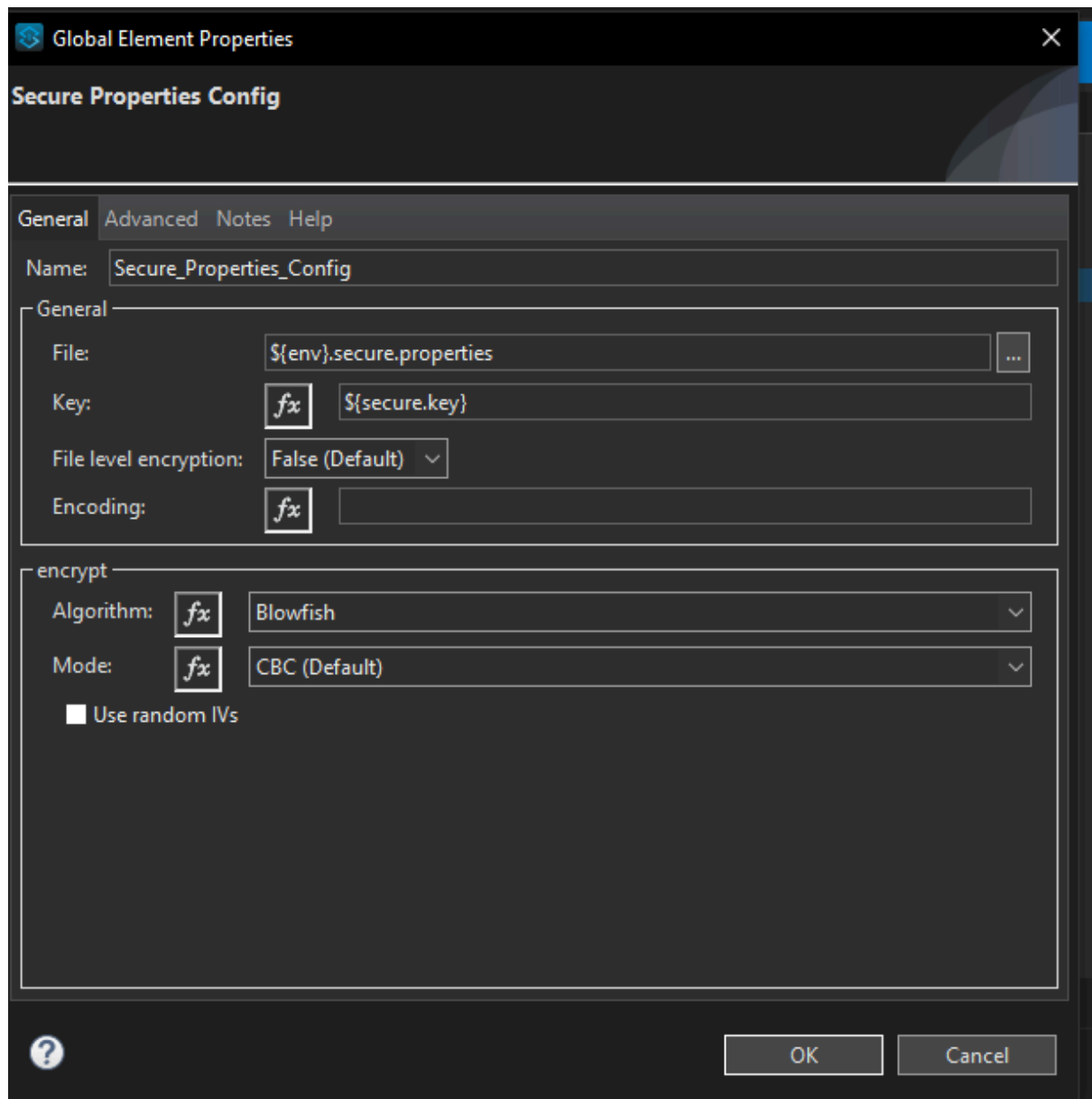
```
http.listener.host=0.0.0.0
http.listener.port=8081
database.connection.host=mudb.learn.mulesoft.com
database.connection.port=3306
database.connection.database=training
```

We can modify this data via the .properties files, but we need secure files to contain sensitive data, so we will use "Secure Properties" to encrypt and decrypt them, so we will use the "Search in exchange" option to add it to our project.

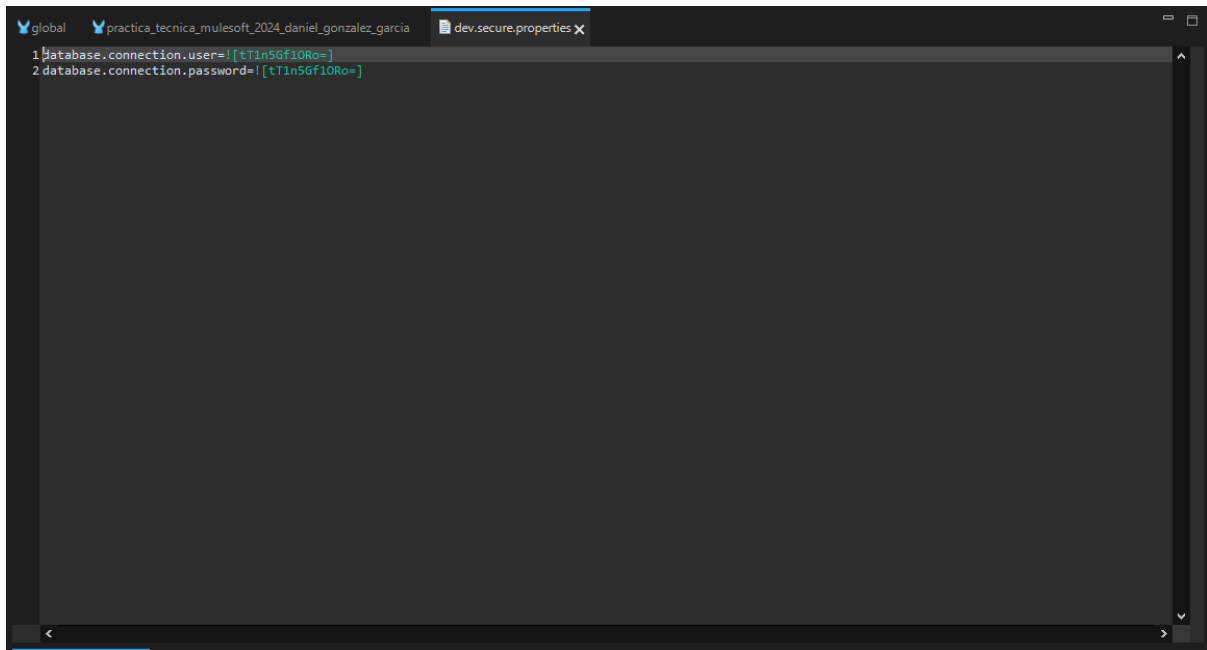




Now we will have available in our global document a new configuration element "Secure properties Config", we add it with the following configuration



And just like before we will now create two new files, "dev.secure.properties" and "local.secure.properties" with the following data



This data must be encrypted, and is obtained using the "secure-properties-tool.jar" file by opening a terminal in the path where the file is located and using the following command

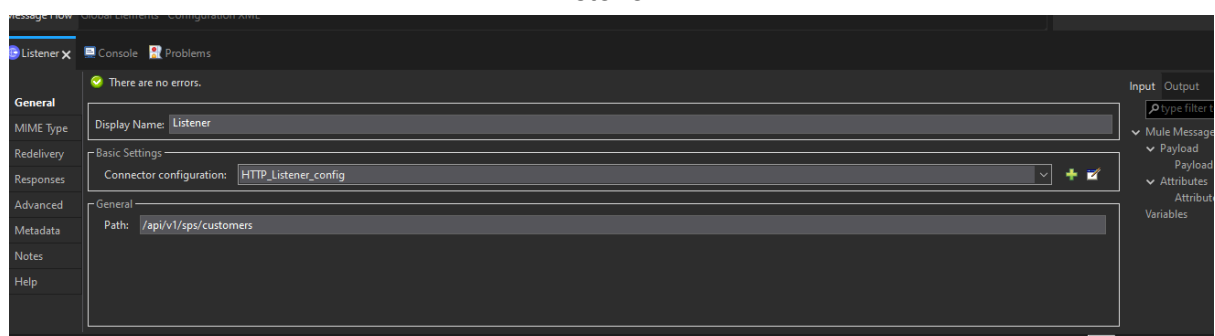
```
C:\Users\Kbylan\Downloads>java -cp secure-properties-tool.jar com.mulesoft.tools.SecurePropertiesTool string encrypt Blo
wfish CBC PruebaTecnica2024 "mule"
tT1n5Gf10Ro=
```

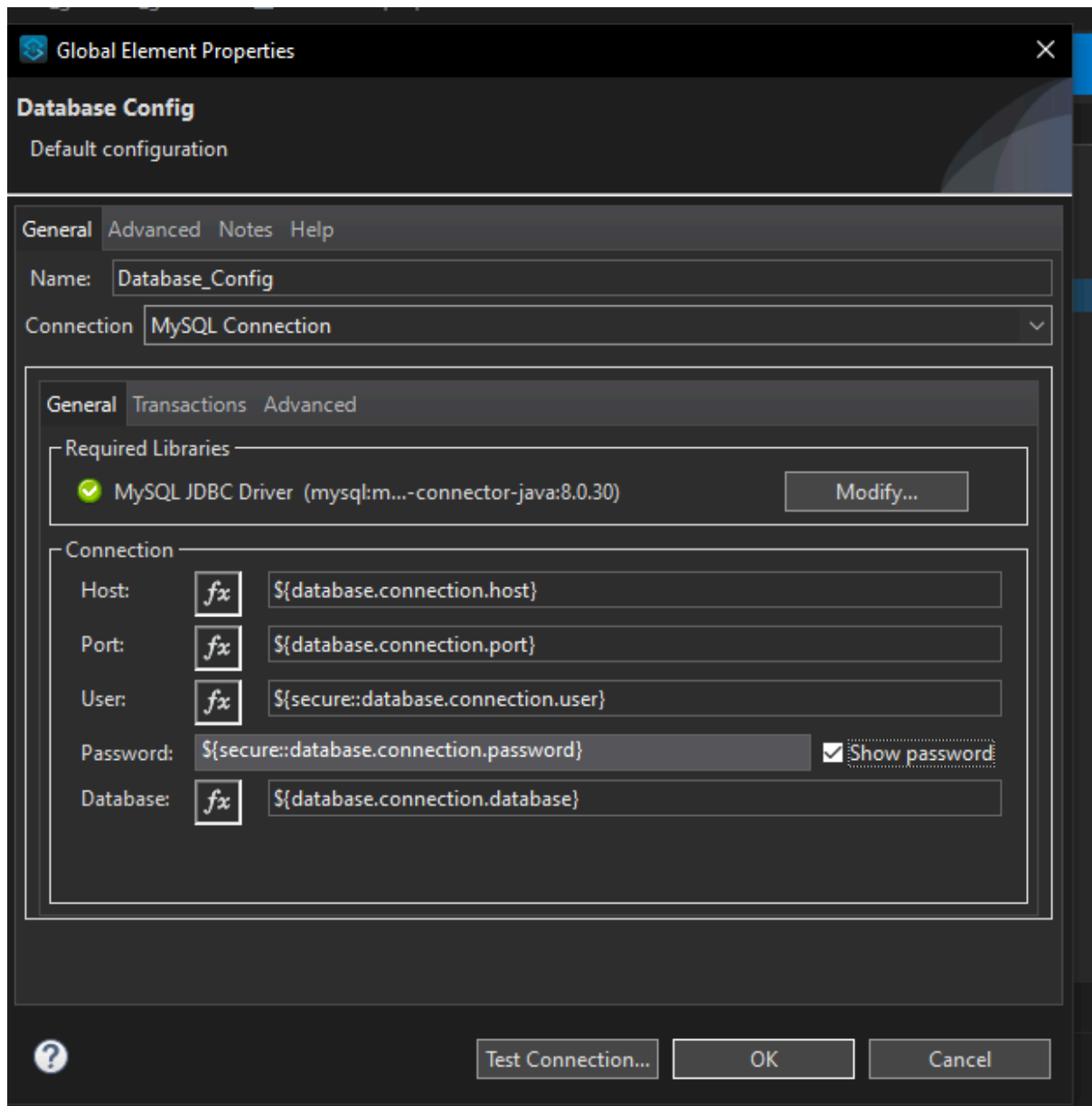
It should be noted that: "PruebaTecnica2024" is the key to encrypt and decrypt and that "mule" is the value to be encrypted.

Let us also remember that for encrypted data it is necessary to add them with a symbol "!" and between "[]"

We can now modify the "Mule properties" of our modules from the beginning of this documentation

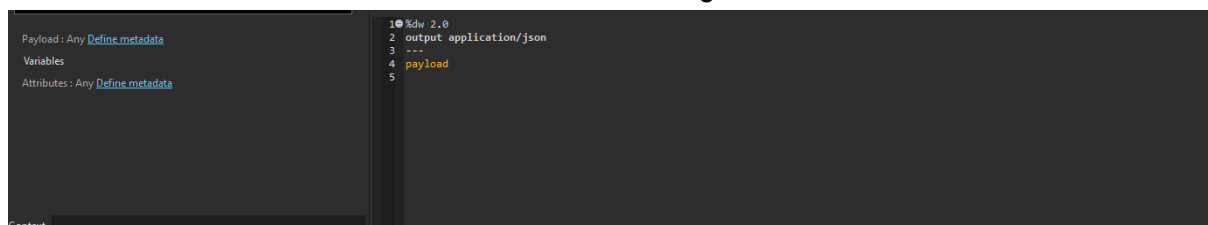
Listener:



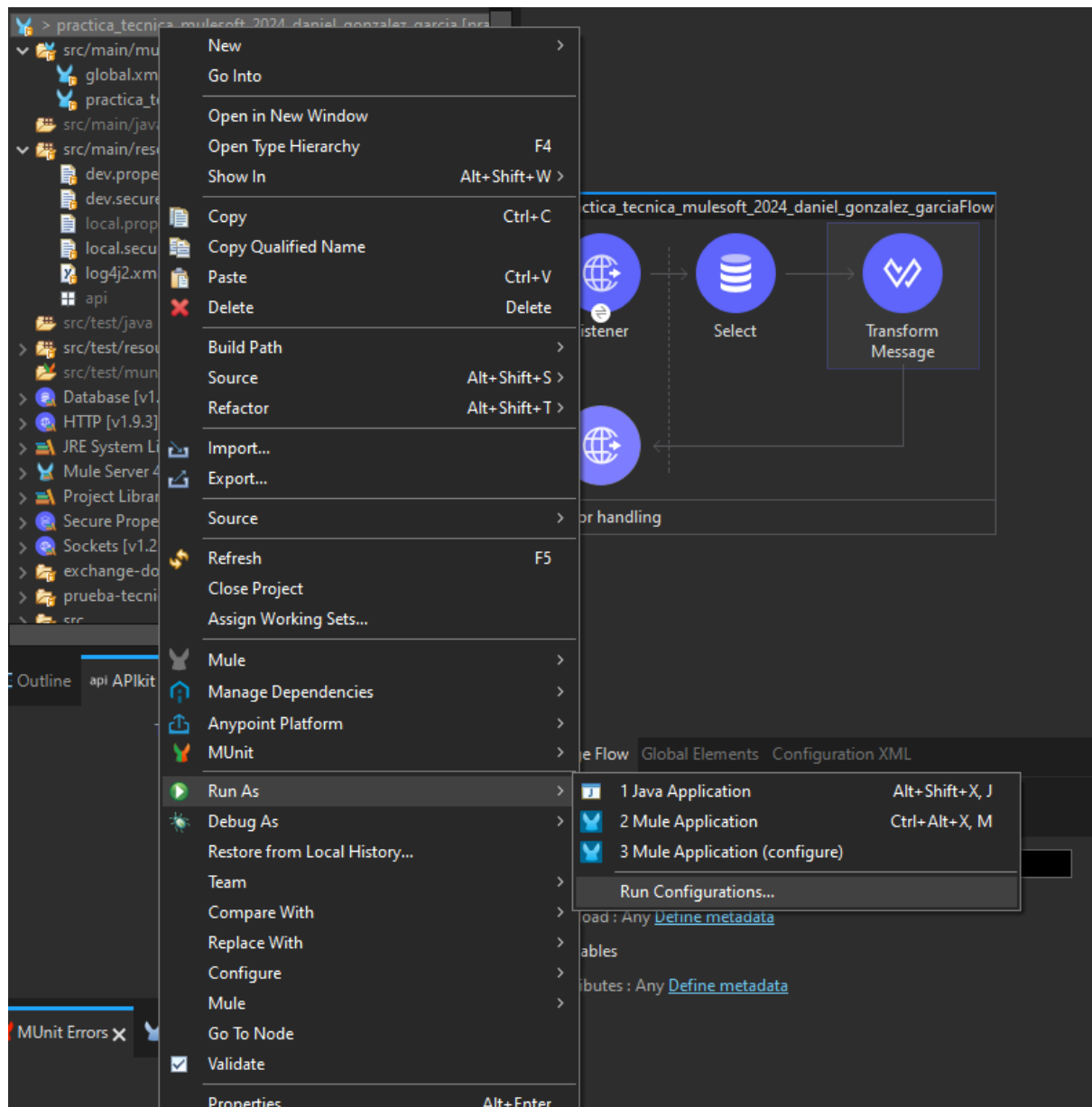


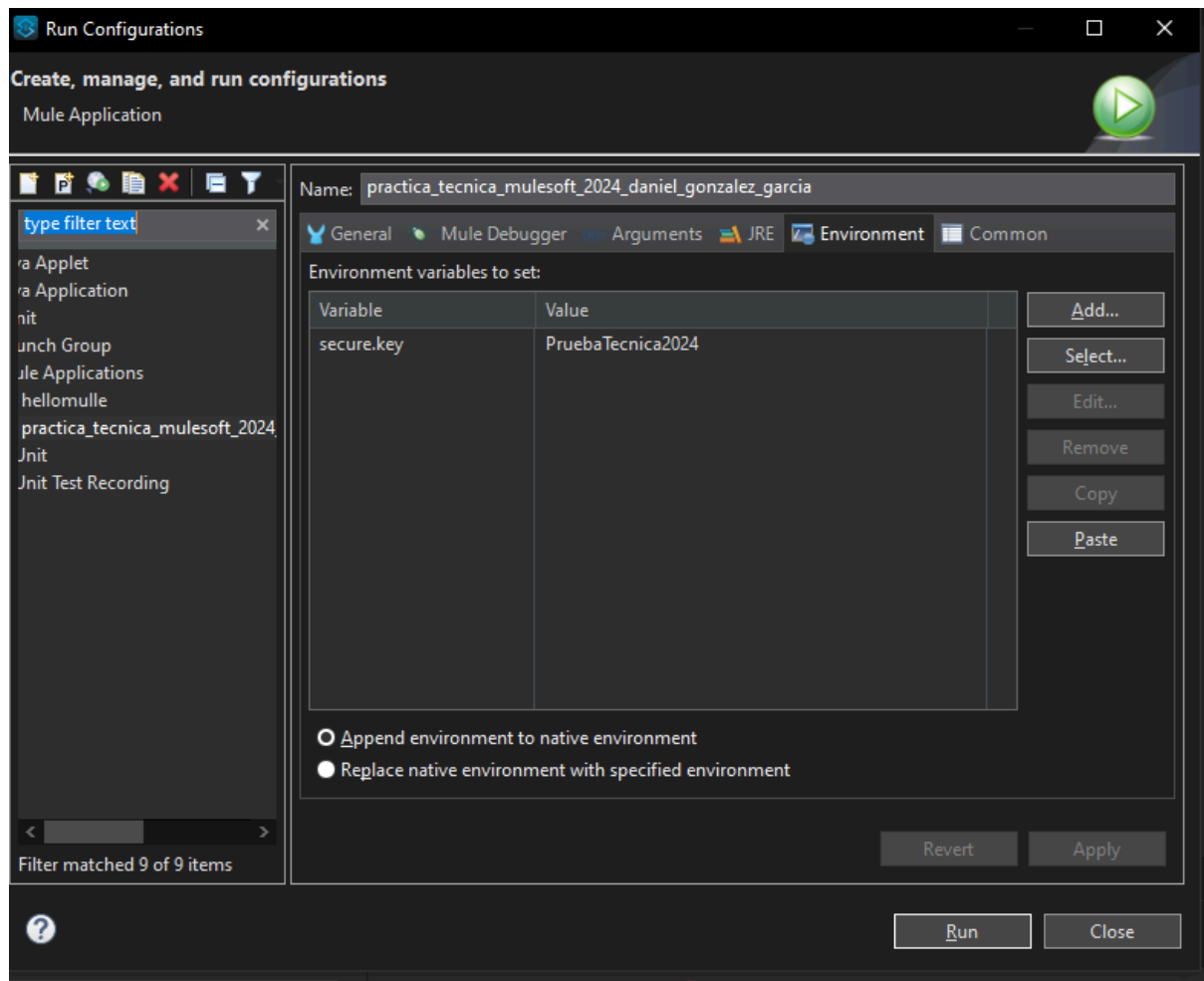
It is important to note that when we use a "Secure Property" we have to add "secure::" as in the case of "user" and "password"

Transform Message:



Finally we have to add the last variable that we are missing "secure.key" but we must add it not in the code but as an environment variable when running the program

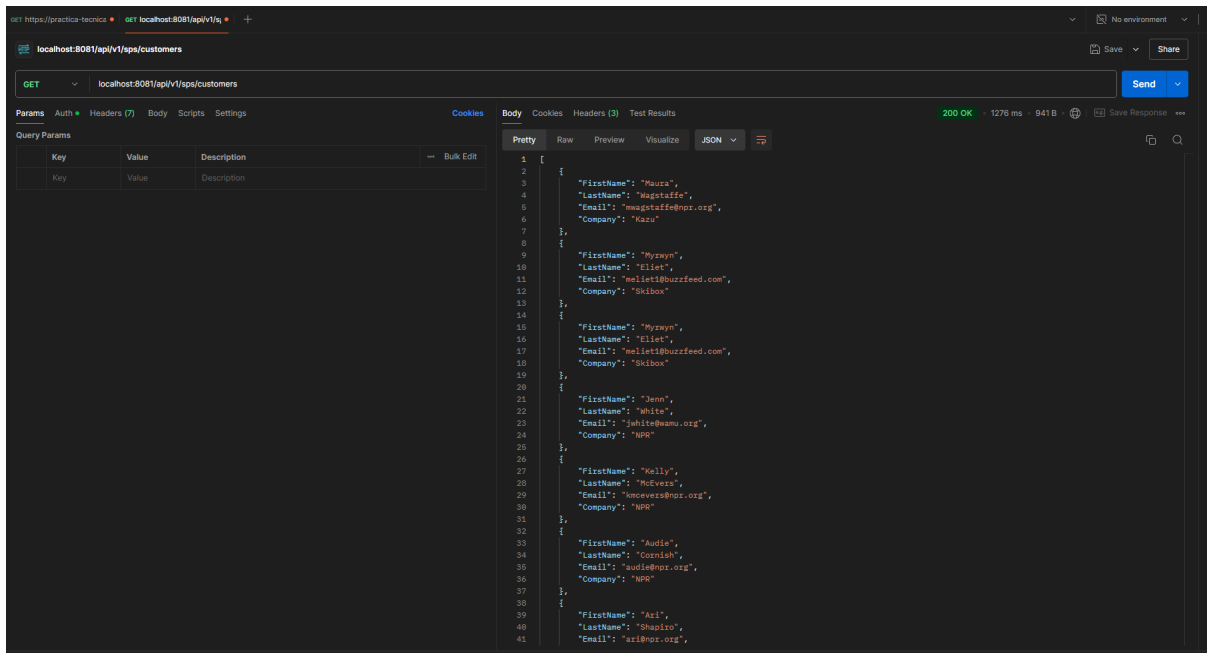




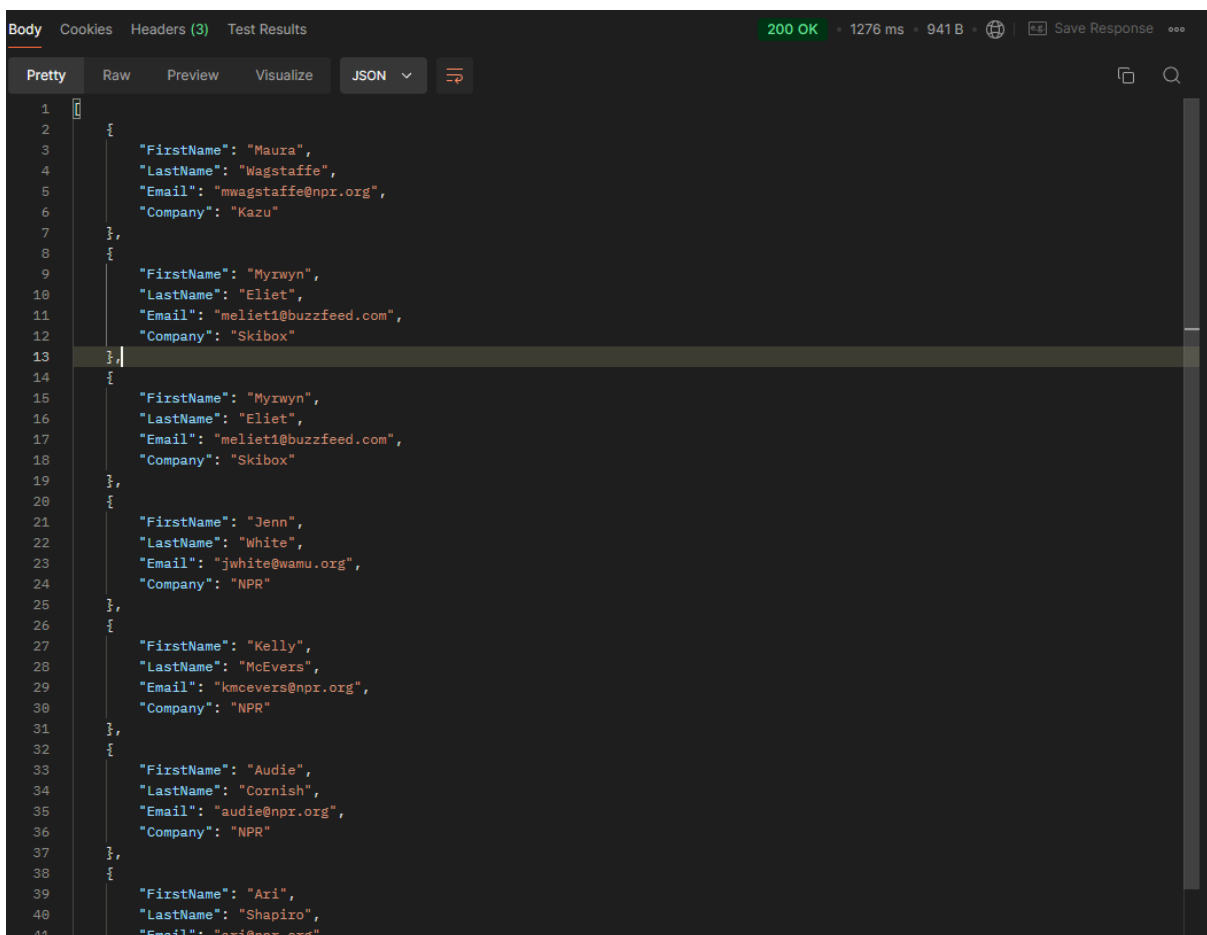
Remember that the value of the variable must be the key that we use to encrypt our values.

Once done, we run the program and we will verify that everything is correct with the help of "Postman"

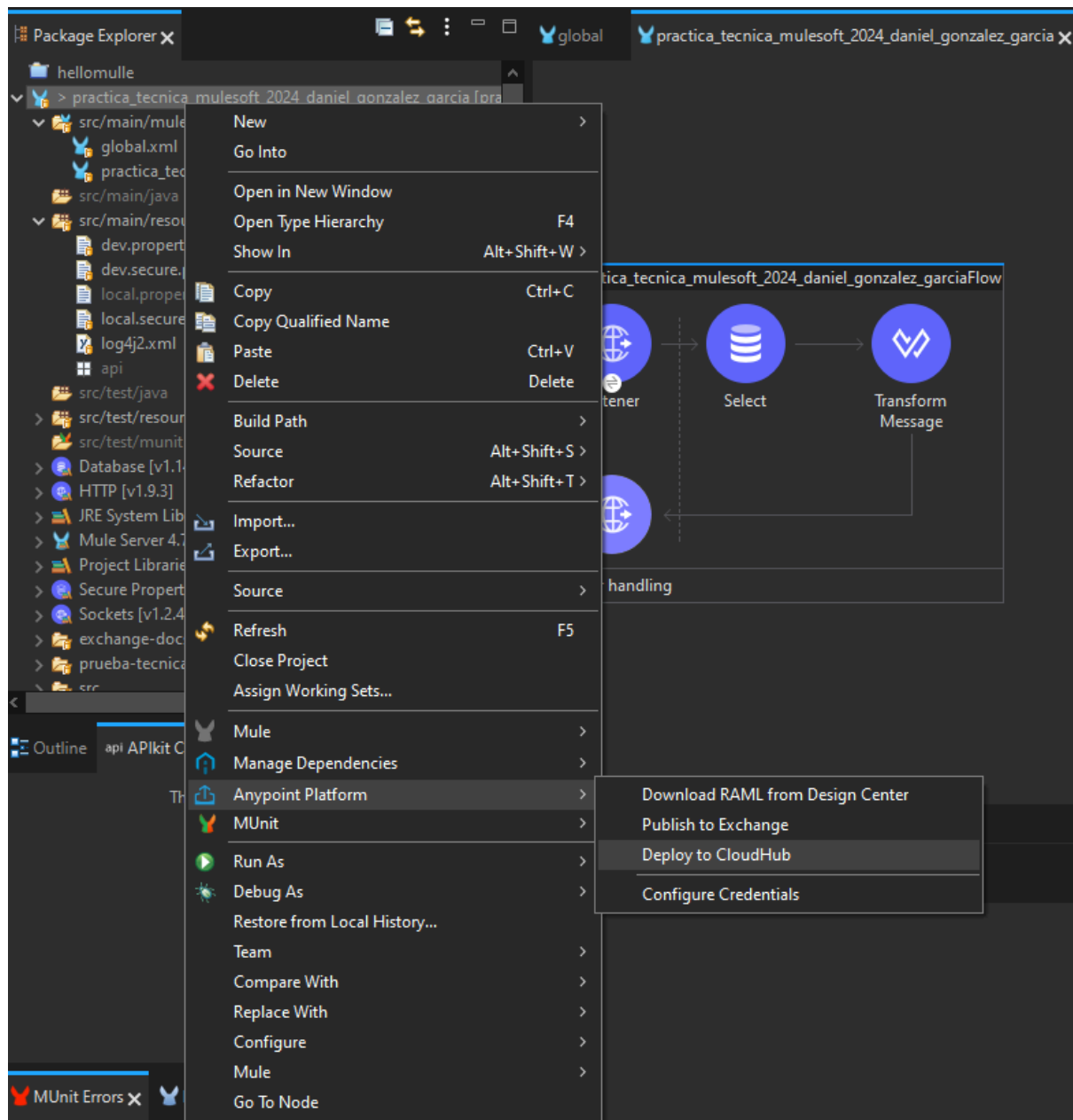
Now we just need to make a get request to the endpoint that we configured in our project with our listener



As a response we received the code "200" and a json file with the data extracted from the database



Now we just need to deploy it in "cloudhub" for this we only need a valid account and click on the following option



And before deploying we must add the following environment variables in the properties tab

SANDBOX

Apply Changes

Application File

Deployment Target

Ingress

Properties

Logging

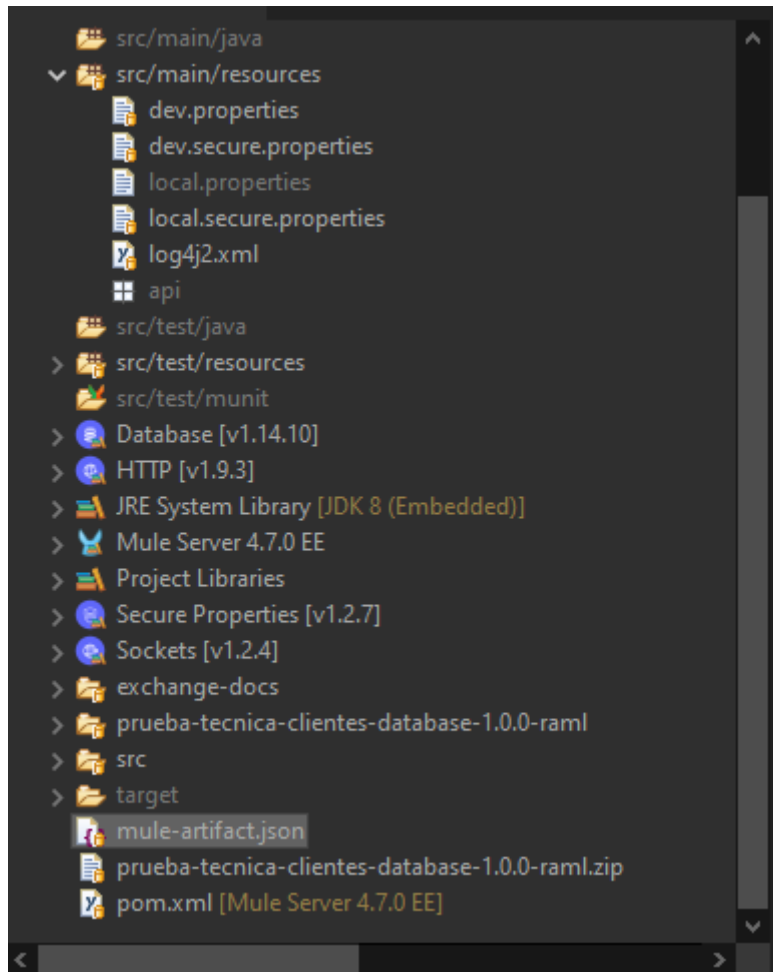
Table view

Text view

Use properties to change the way your app behaves. [Learn more](#)

secure.key	PruebaTecnica2024	Protect	...
env	dev	Protect	...
New Key	New Value		

If you want to hide the property values in "AnyPoint Platform" you just have to modify the following file

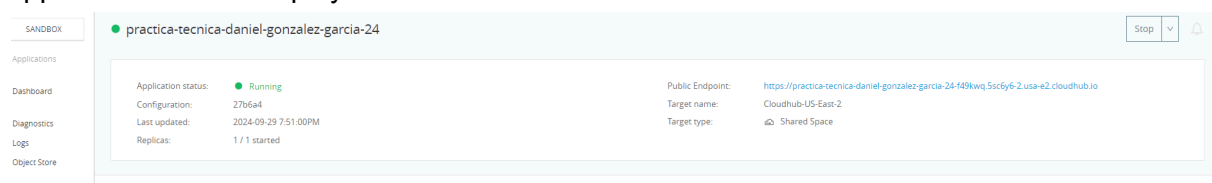


Adding the following line

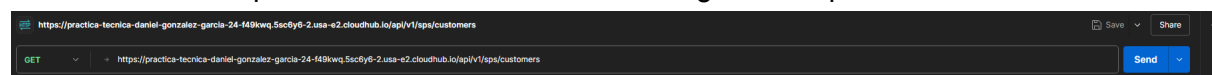
```
"secureProperties": ["secure.key"]
```

Now we can test the deployed version with "Postman"

In "Runtime Manager" of "AnyPoint Management" we must obtain this data from the application that we deploy



And we make a request to that address with the configured endpoint



As a response we will also get the same result as with the local version

GET

https://practica-tecnica-daniel-gonzalez-garcia-24-f49kwq5sc6y8-2.usa-e2.cloudhub.io/api/v1/sps/customers

Send

ParamsAuthorizationHeaders (7)BodyScriptsSettings

Cookies

BodyCookiesHeaders (6)Test Results

200 OK · 616 ms · 1.06 KB · Save Response

Query Params

Key	Value	Description
Key	Value	Description

PrettyRawPreviewVisualizeJSON

```
1 [
2   {
3     "FirstName": "Mauro",
4     "LastName": "Magstaffa",
5     "Email": "magstaffa@not.org",
6     "Company": "Kazu"
7   },
8   {
9     "FirstName": "Myrwyn",
10    "LastName": "Ellet",
11    "Email": "weliott@ourzfeed.com",
12    "Company": "Skibox"
13  },
14  {
15    "FirstName": "Myrwyn",
16    "LastName": "Ellet",
17    "Email": "weliott@ourzfeed.com",
18    "Company": "Skibox"
19  },
20  {
21    "FirstName": "Jenn",
22    "LastName": "White",
23    "Email": "jwhite@nuu.org",
24    "Company": "Wpu"
25  },
26  {
27    "FirstName": "Kelly",
28    "LastName": "McEvers",
29    "Email": "kncevers@not.org",
30    "Company": "Wpu"
31  },
32  {
33    "FirstName": "Audie",
34    "LastName": "Cornish",
35    "Email": "audie@nuu.org",
36    "Company": "Wpu"
37  },
38  {
39    "FirstName": "Arl",
40    "LastName": "Shapiro",
41    "Email": "arls@not.org",
```

Creating API in "API Manager"

Once logged into AnyPoint Platform, we will enter API Manager and click on "Add API" and "Add New API"

The screenshot shows the 'API Administration (Sandbox)' interface. The 'Add API' button is highlighted in blue. Below it, the 'Runtime' tab is selected in the breadcrumb navigation. The 'Runtime' section contains the following fields:

- Select runtime:** Three radio button options:
 - ☐ 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 *:** Two radio button options:
 - ☒ 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 *:** Two radio button options:
 - ☒ Mule 4 (recommended)
 - ☐ Mule 3 or below

Buttons: 'Cancel' and 'Next'.

The screenshot shows the 'API' tab selected in the breadcrumb navigation. The 'API' section contains the following fields:

- Select the API you want to manage:** Two radio button options:
 - ☐ Select API from Exchange
 - ☒ Create new API
- Instructions:** A blue information icon followed by the text: "Once the API is created it will be published in Exchange in stable state."
- Name:** A text input field with the placeholder "Enter asset name".
- Asset types (i):** A dropdown menu with the placeholder "Select a type".
- Footer:** "Templates and Examples can be published from Anypoint Studio." and an "Advanced >" link.

Buttons: 'Cancel', 'Previous', and 'Next'.

API Administration (Sandbox) Add API

SANDBOX

API Administration
API Groups
Automated Policies
Client Applications
Custom Policies
Mule API Analytics

Runtime API Downstream Upstream Review

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

API Administration (Sandbox) Add API

SANDBOX

API Administration
API Groups
Automated Policies
Client Applications
Custom Policies
Mule API Analytics

Runtime API Downstream Upstream Review

Review

Review your selections before saving and deploying your API instance. You can also save the configuration and deploy it later.

Runtime [Edit](#)

Runtime type Mule Gateway

Proxy type Basic Endpoint

API [Edit](#)

API name example

API version v1

Asset version 1.0.0

Downstream [Edit](#)

Scheme HTTP

Upstream [Edit](#)

Previous Next

Which leaves us with the following screen

API Administration (Sandbox) PruebaTecnicaAPI (v1) - API Summary

SANDBOX

API Administration
API Summary
Alerts
Contracts
Policies
SLA Tiers
Settings

APIs / PruebaTecnicaAPI / API Summary

Actions

Type	Asset Version	Implementation URI	API Label	API Version
HTTP	1.0.0 (Latest)	N/A	-	v1
API Status	Consumer Endpoint	API Instance ID	Mule Version	Java Version
Active	N/A	19978120	4.7.3	8

Tags
[Add New Tag](#)

Key Metrics

The Key Metrics feature will undergo significant enhancements. The Anypoint Monitoring Agent is required in order to ensure uninterrupted access to Key Metrics. If you have not already done so, you must install the Anypoint Monitoring Agent by October 31st, 2024.

Requests Top Apps Latency

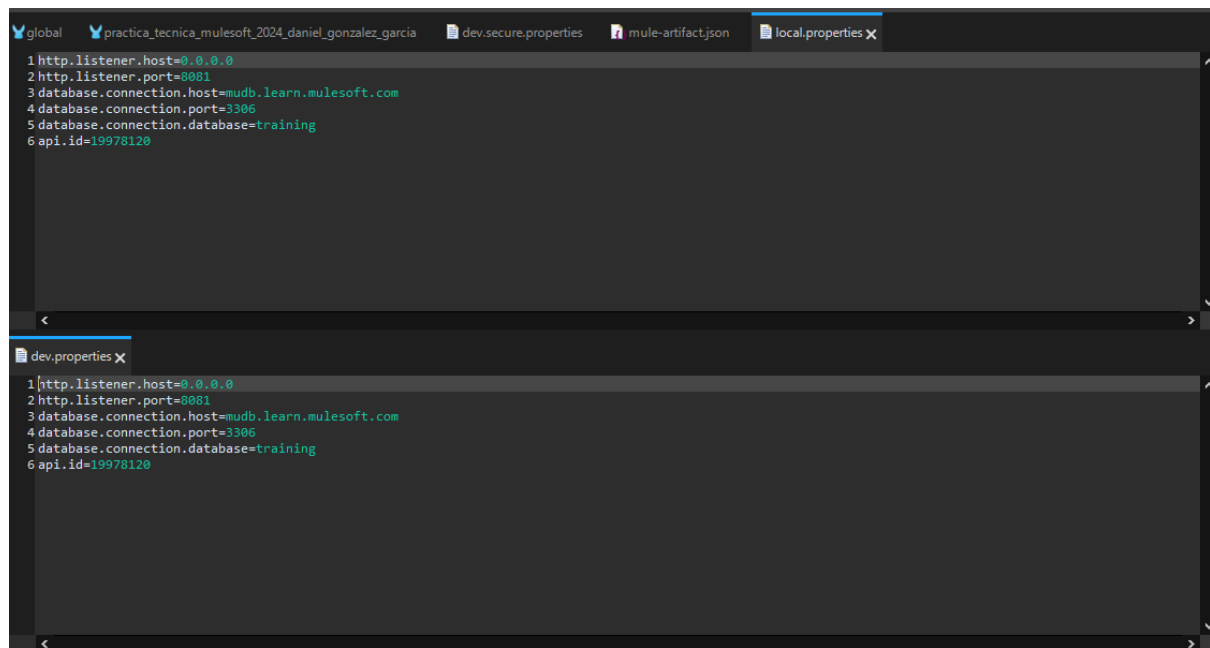
☐ Include Policy Violations

Download CSV

1h 3h 1d 7d 1m 3m 1y


The data we need for now is the "API instance ID"

Now in our "dev.properties" and "local.properties" files we must add this id and they would look like this:



```
1 http.listener.host=0.0.0.0
2 http.listener.port=8081
3 database.connection.host=mudb.learn.mulesoft.com
4 database.connection.port=3306
5 database.connection.database=training
6 api.id=19978120
```

In our global file we will add a new configuration element "API Autodiscovery"

 Global Element Properties

API Autodiscovery
 Service auto-discovery configuration information.


Auto-discovery configuration
 Notes
 Help

Auto-discovery settings

API Id:

Flow Name:

☒ Ignore base path on resource level policies



OK
 Cancel

Now on the API Manager selection screen we must enter the following window

SANDBOX

Add API

Environment

Filter by

1 to 1 of 1

⏪

⏩

⏴

⏵

API Administration

API Groups

Automated Policies

Client Applications

Custom Policies

Mule API Analytics

Status	API Name	Runtime	Label	Version	Instance	Error Rate	Total Requests	Client Applications	Creation Date	Actions
Active	PruebaTecnicaAPI	Mule 4	-	v1	19978120	25%	4	1	09-29-2024 19:26	

And get the data: "Client ID" and "Client Secret" from the "Sandbox" environment

Environment Information



Environment

Environment name: Sandbox
Environment ID: b4ed8faf-2a68-4104-b1f6-51b61bc3891b

Client Credentials

Client ID: 8f97df26e5314e6bb97beb1d83724282

Client secret:

.....

Show

Business Group

Business Group name: SPS Solutions
Business Group ID: 959c14a8-5dee-4588-b750-49cf33f0ad21

Client Credentials

Client ID: e3075f74fa1a4bd580a17aff768864d8

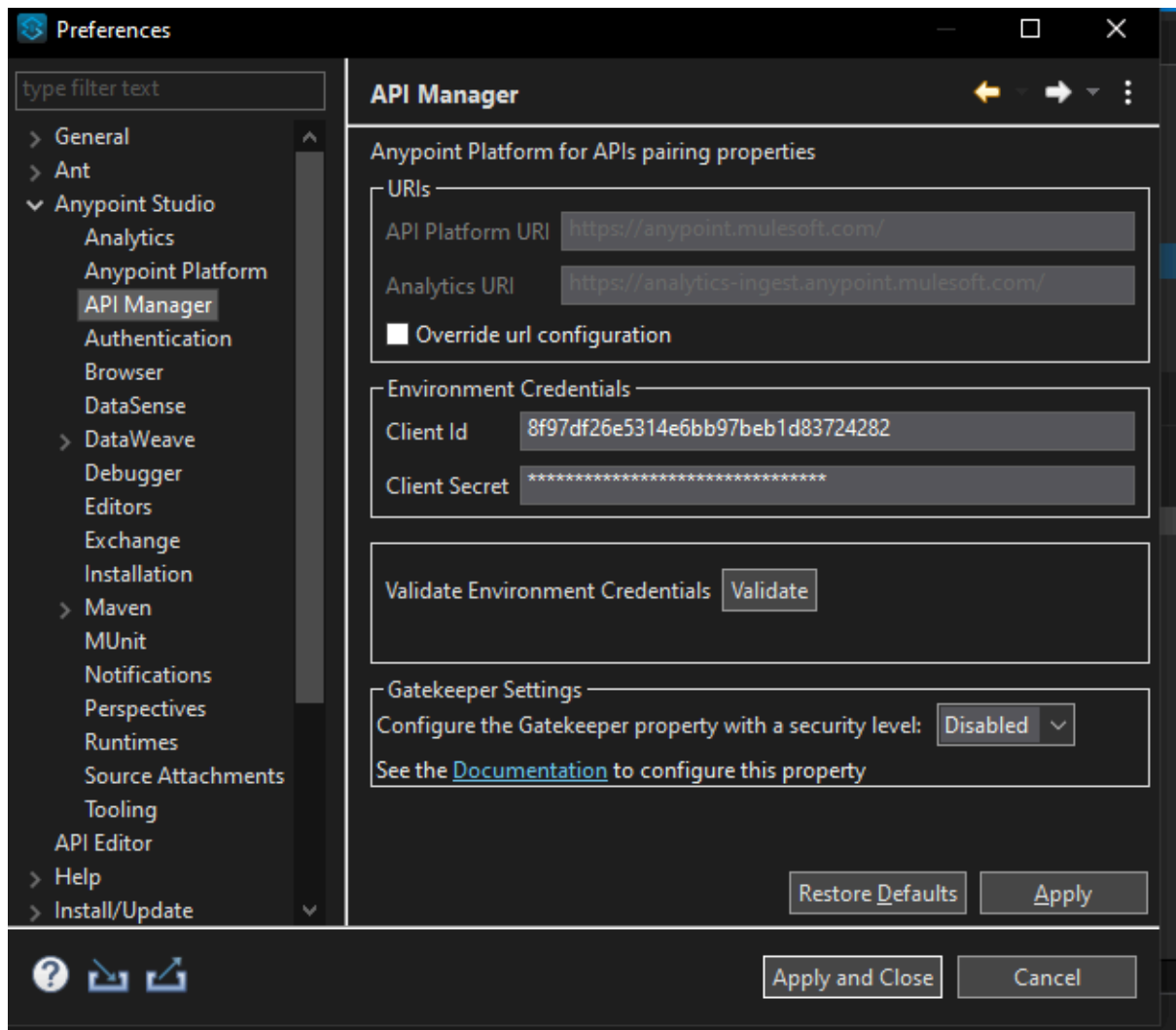
Client secret:

.....

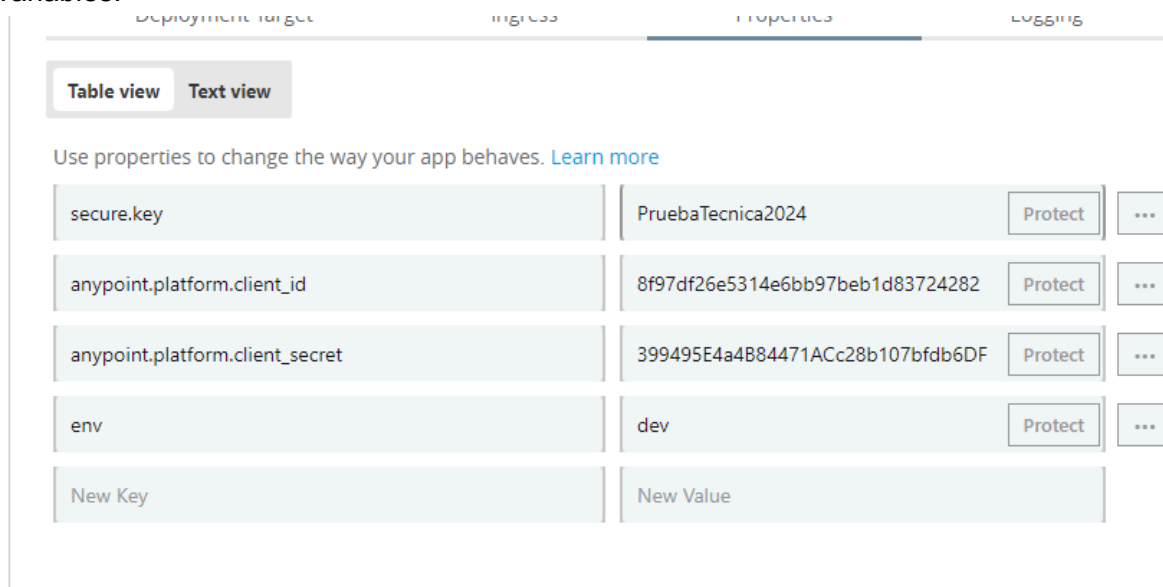
Show

Done

and add them to AnyPoint Studio preferences

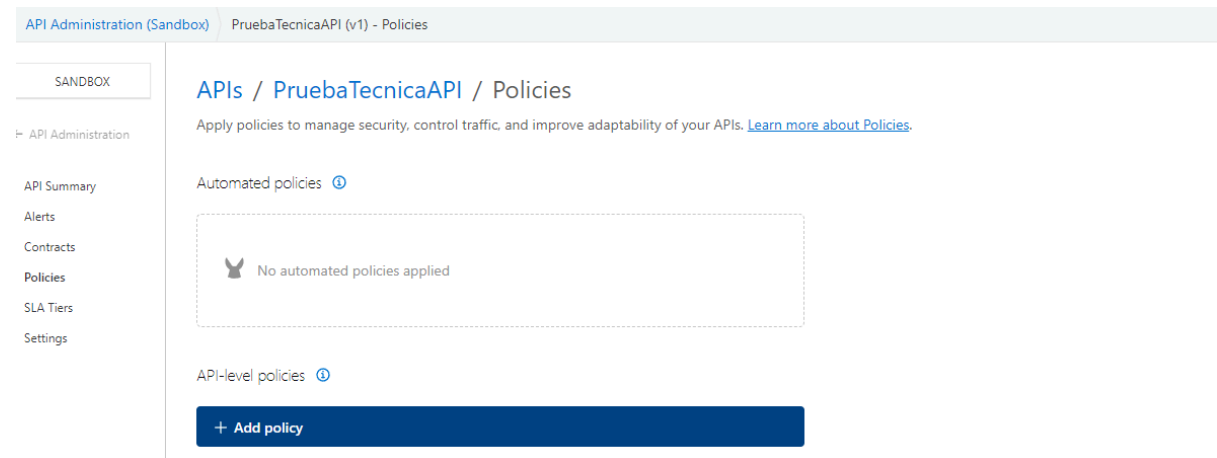


As with other variables, we must add them to the mule-artifact.json file and to the CloudHub variables.

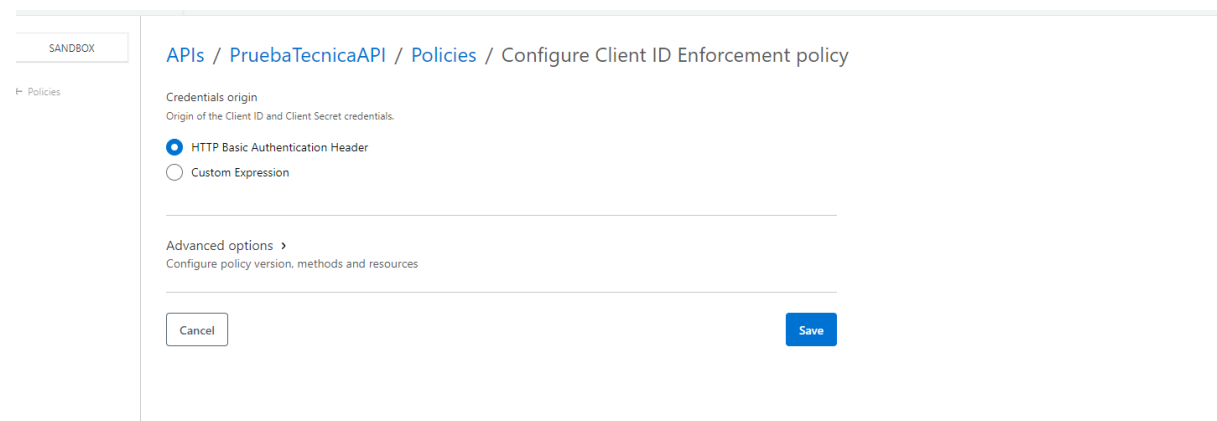
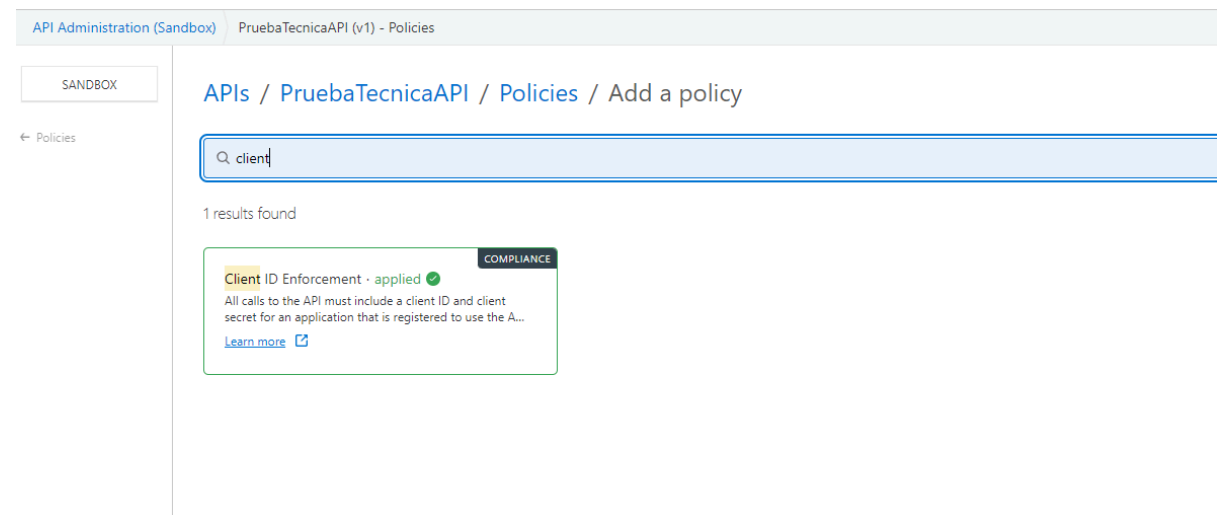


Applying Client ID enforcement policy in API Manager

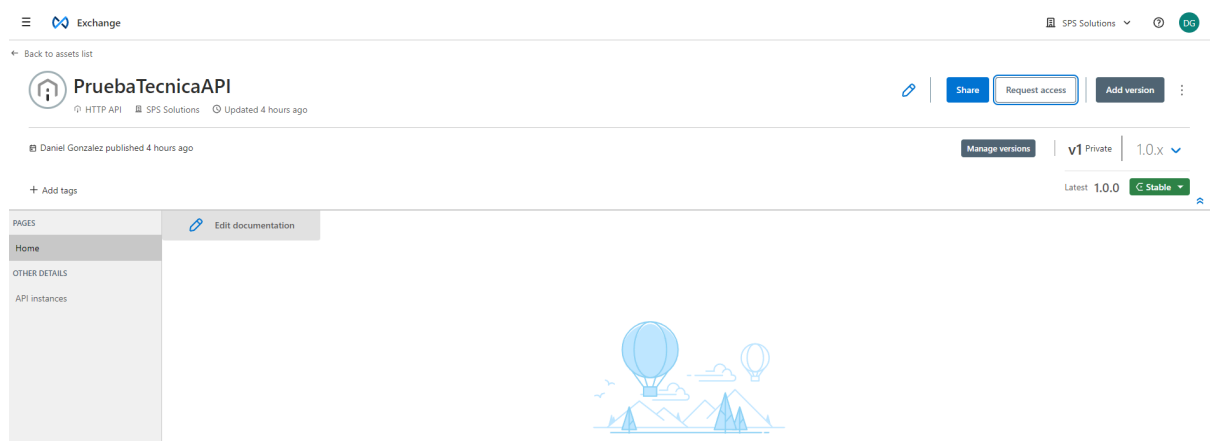
In the newly created API in the "Policies" module we select the following option



We look for "Client" and choose the option that appears.



Now we will go to "Exchange" of AnyPoint Platform and select the newly created one, and click on "Request Access"



We select the "API Instance" and create a new application with a name that tells us who it belongs to.

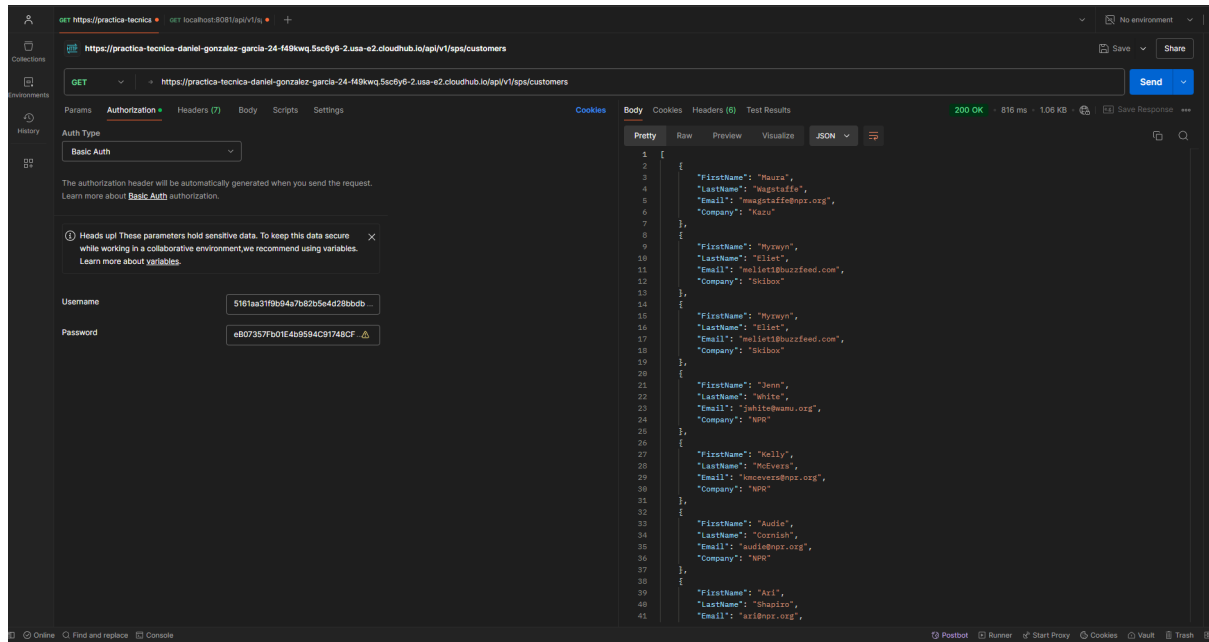
After this it will give us two very important pieces of data: ClientID and ClientSecret

For this specific application these are the data:

Client ID: 5161aa31f9b94a7b82b5e4d28bbdbd65

Client Secret: eB07357Fb01E4b9594C91748CFe0F062

With this the authentication is ready, now when making a request we must add this data as follows:



This must be done both locally and in the deployed version.