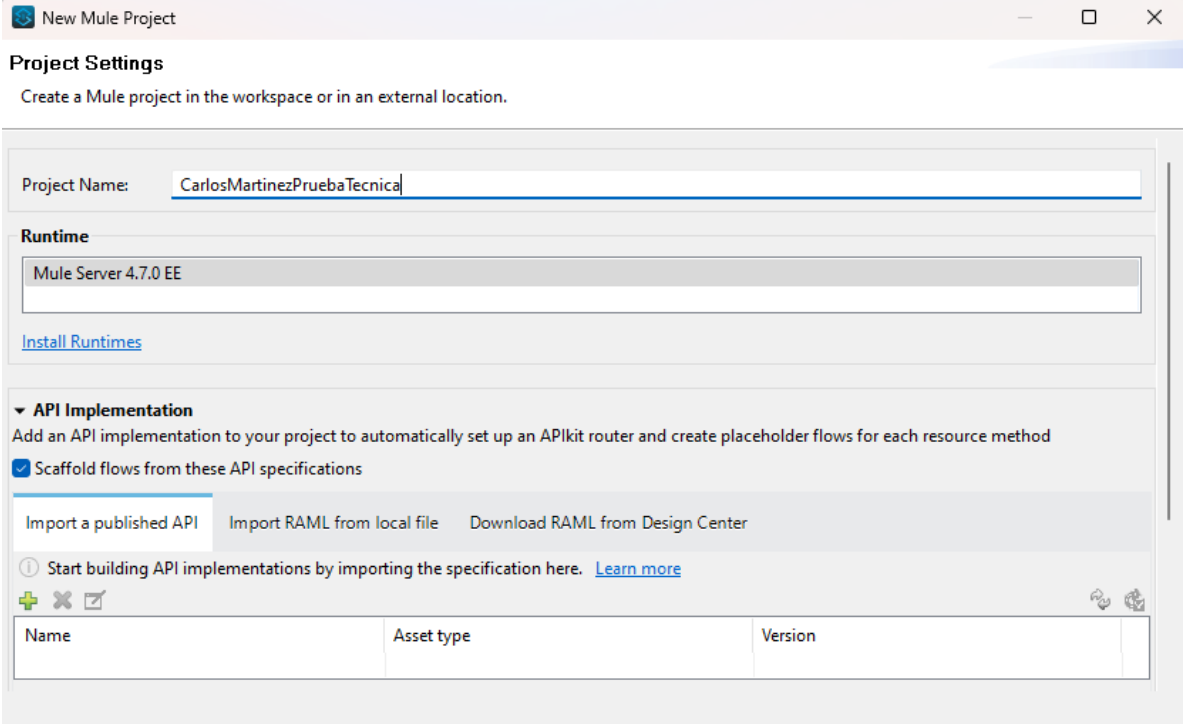


Prueba Técnica

Carlos Martínez

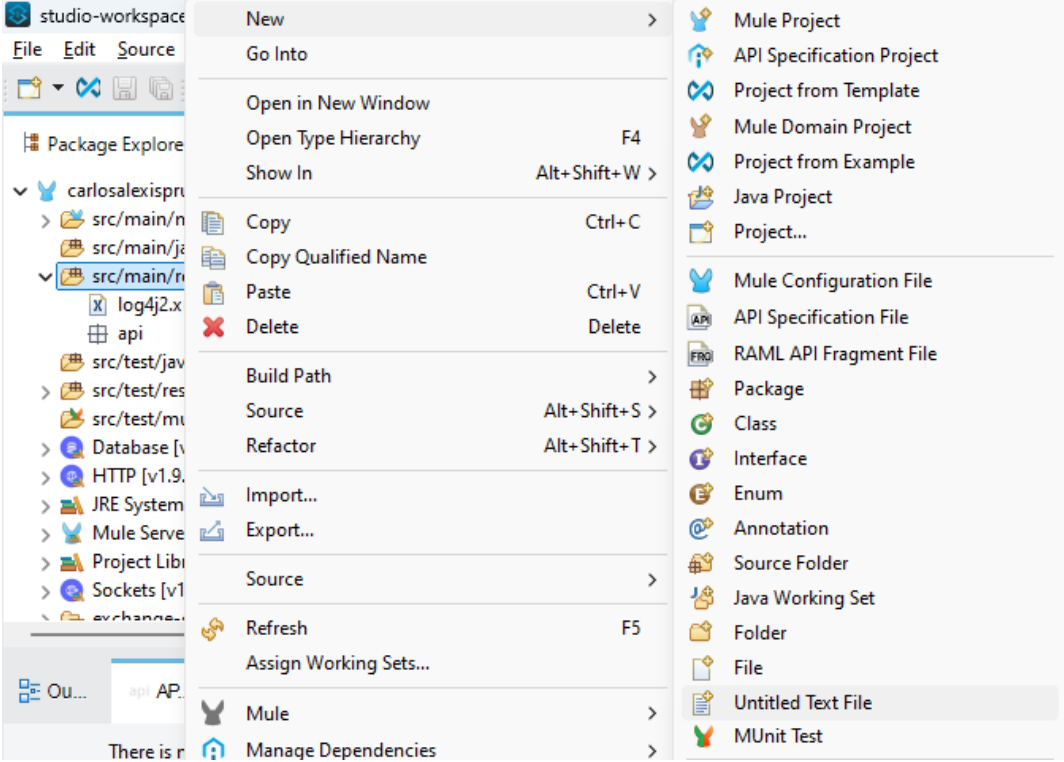
Creación del proyecto

En la esquina superior izquierda hacemos clic en **File > New > Mule Project**

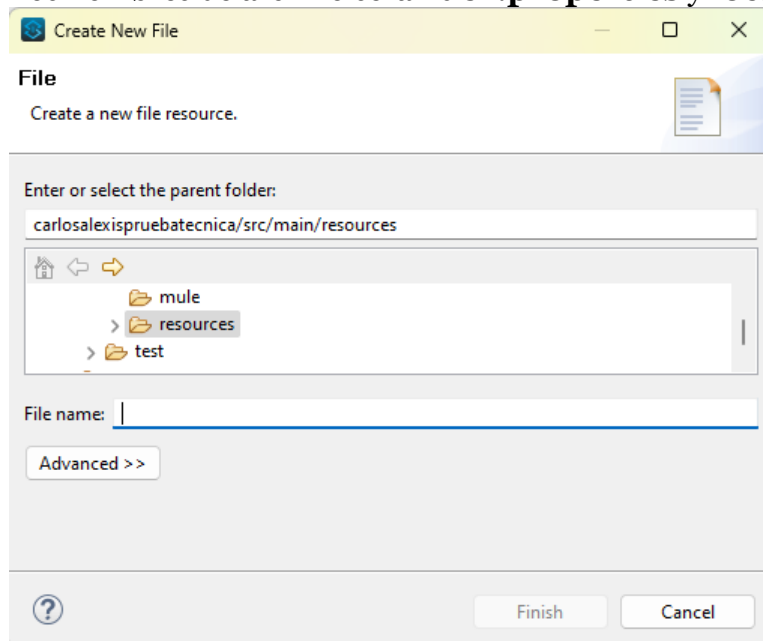


Propiedades no seguras

En la carpeta **src/main/resource** hacemos clic derecho seleccionamos **new > File** y creamos dos archivos para propiedades



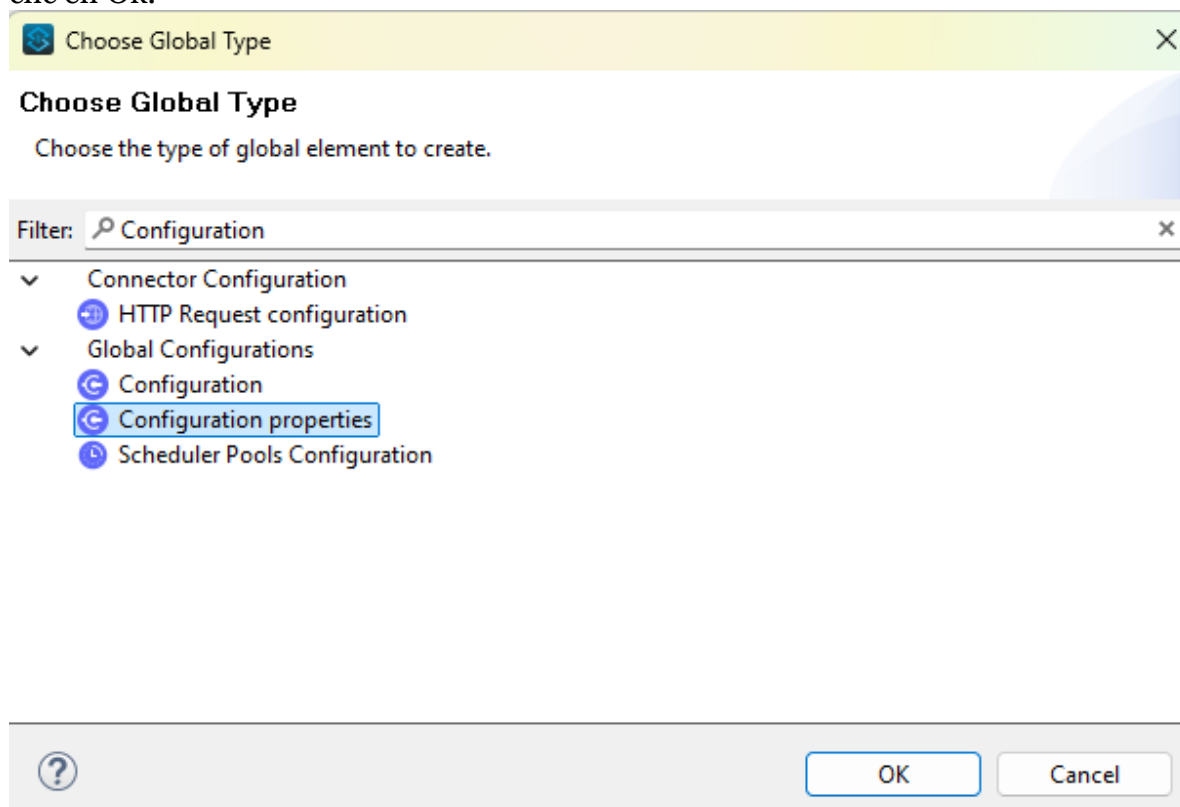
Los nombres de archivo serán **dev.properties** y **local.properties**



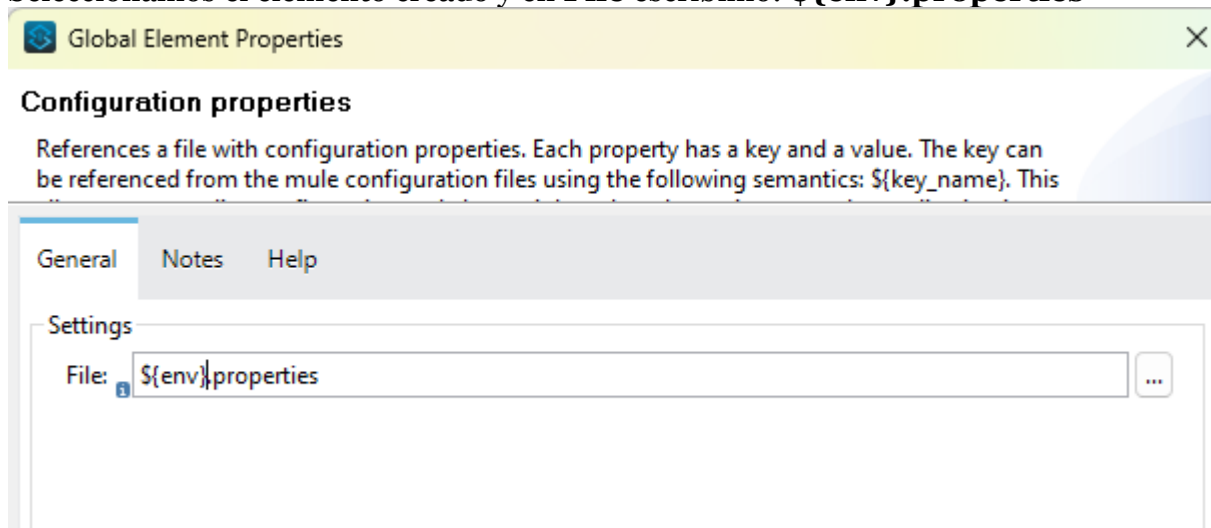
En estos archivos definimos propiedades no seguras, en este caso introducimos en ambos archivos recién creados las siguientes propiedades:

```
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
```

Necesitamos indicar que este será un archivo de propiedades, para ello agregamos un nuevo elemento global con el que se reconocerá nuestro archivo de propiedades, cambiamos la vista a **Global Elements**, hacemos clic en **Create** y buscamos **Configuration Properties** y hacemos clic en Ok.



Seleccionamos el elemento creado y en **File** escribimo: **`${env}.properties`**

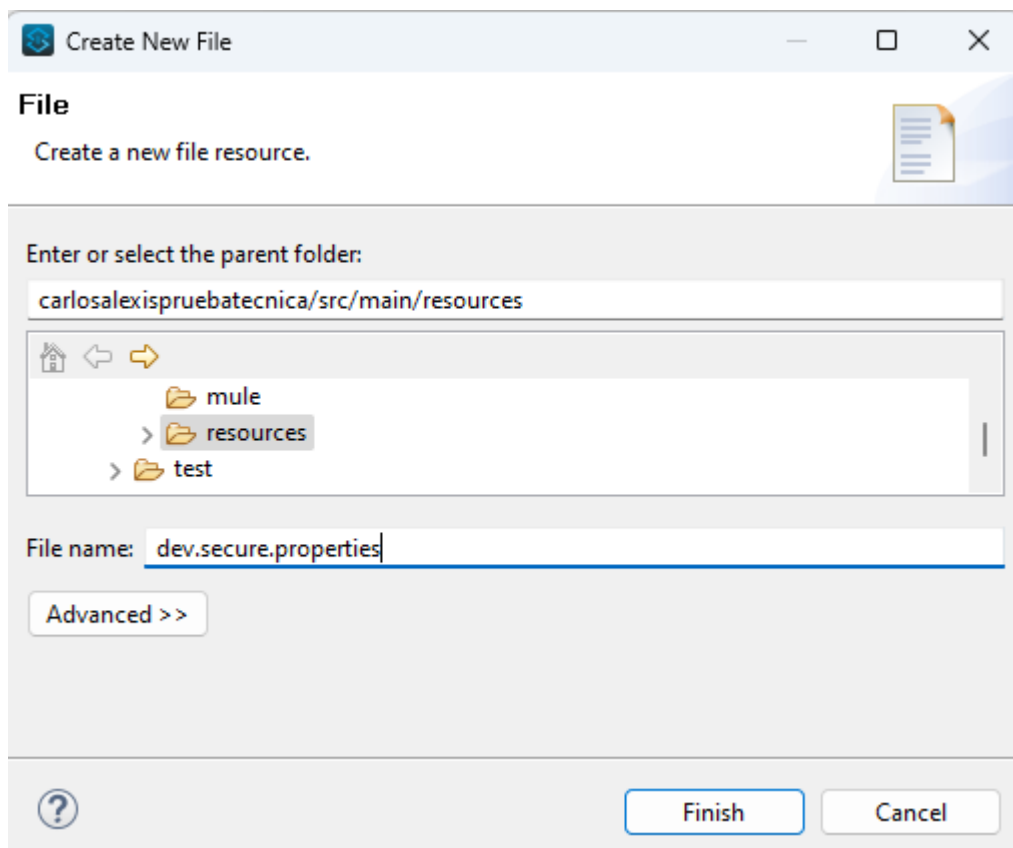


Posteriormente indicaremos el valor de env agregándolo a las variables de entorno en tiempo de ejecución, ahí podremos indicar si utilizamos local o dev, **es una buena práctica separar las propiedades por entorno.**

Propiedades seguras

Otra buena práctica es separar la información sensible en otros archivos y encriptarla, es lo que harémos con las credenciales de la base de datos, para eso creamos dos archivos haciendo clic derecho en **src/main/resource** y seleccionamos **new > File**.

Los nombres de archivo serán **local.secure.properties** y **dev. secure.properties**



Ahora en el **Mule Palette** hacemos clic en **Search in Exchange** para agregar el módulo **Mule Secure Configuration Properties**.

Add Dependencies to Project

Add Dependencies to Project

Search for dependencies in Exchange to add them to the project

Usernamecarlosalexis

Add Account

AsynAPI 2.6 (Beta) requires enablement by an Anypoint Platform organization administrator. [Learn more](#)

Mule Secure Configuration Properties

Available compatible modules

Name	Publisher	Asset type	Latest
Mule Secure Configuration Property	MuleSoft	REST API	1.2.7
AWS Secret Manager Properties Prov	MuleSoft	REST API	1.0.1
Azure Key Vault Connector - Mule 4	MuleSoft	REST API	1.1.4

Add >

< Remove

Selected modules

Name	Asset type	Version
Mule Secure Configur...	Other	1.2.7

Open Exchange Preference Page

?

Finish

Cancel

Cambiamos la vista a **Global Elements**, hacemos clic en **Create** y buscamos **Secure Properties Config** y hacemos clic en **Ok**.

Choose Global Type

Choose Global Type

Choose the type of global element to create.

Filter:Secure Properties Config

Connector Configuration

Secure Properties Config

Esta es la configuración general para el elemento recién creado:

Global Element Properties

Secure Properties Config

General | Advanced | Notes | Help

Name: Secure_Properties_Config

General

File: \${env}.secure.properties

Key: `fx` \${secure.key}

File level encryption: False (Default)

Encoding: `fx` UTF-8

encrypt

Algorithm: `fx` Blowfish

Mode: `fx` CBC (Default)

☐ Use random IVs

OK Cancel

Para encriptar usaremos el archivo debemos abrir la terminal y dirigirnos a la carpeta donde éste se encuentre y utilizar el siguiente comando:

```
PS C:\Users\carlo\Downloads> java -cp secure-properties-tool.jar com.mulesoft.tools.SecurePropertiesTool string encrypt Blowfish CBC PalabraSecreta "mule"
Kq/6VOc/dJQ=
PS C:\Users\carlo\Downloads> |
```

Esto nos regresa la clave mule encriptada (Kq/6VOc/dJQ=).

Abrimos los archivos local y dev.secure.properties y asignamos las credenciales encriptadas, es importante respetar la sintaxis ![“valor encriptado”]

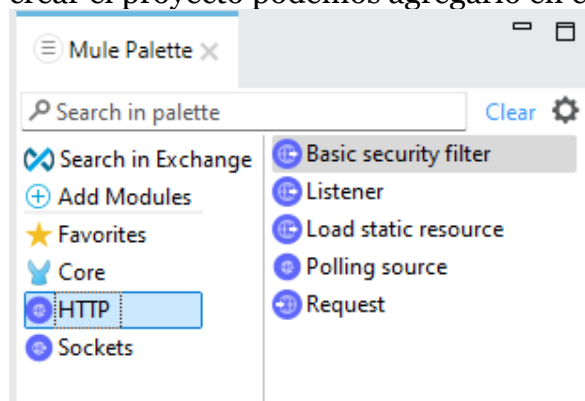
*carlosalexisprueb... | local.secure.prope... | dev.secure.prope... X | local.properties

```
1 database.connection.user = ![Kq/6VOc/dJQ=]
2 database.connection.password = ![Kq/6VOc/dJQ=]
```

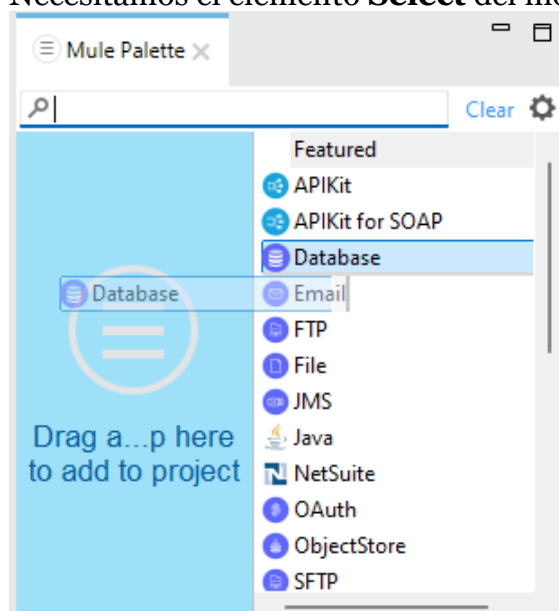
Agregamos \${Secure.key} como variable de entorno en tiempo de ejecución.

Creación del flujo

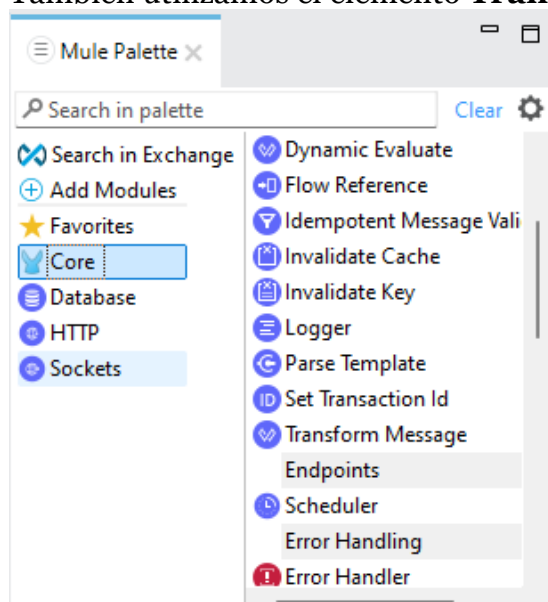
Para crear una api desde anypoint necesitamos partir del **Source** o endpoint que estará escuchando las solicitudes, para ello utilizaremos el elemento Listener del módulo Http, si no se cuenta con él al crear el proyecto podemos agregarlo en el **Mule Palette** haciendo clic en **Add Modules** y buscarlo.



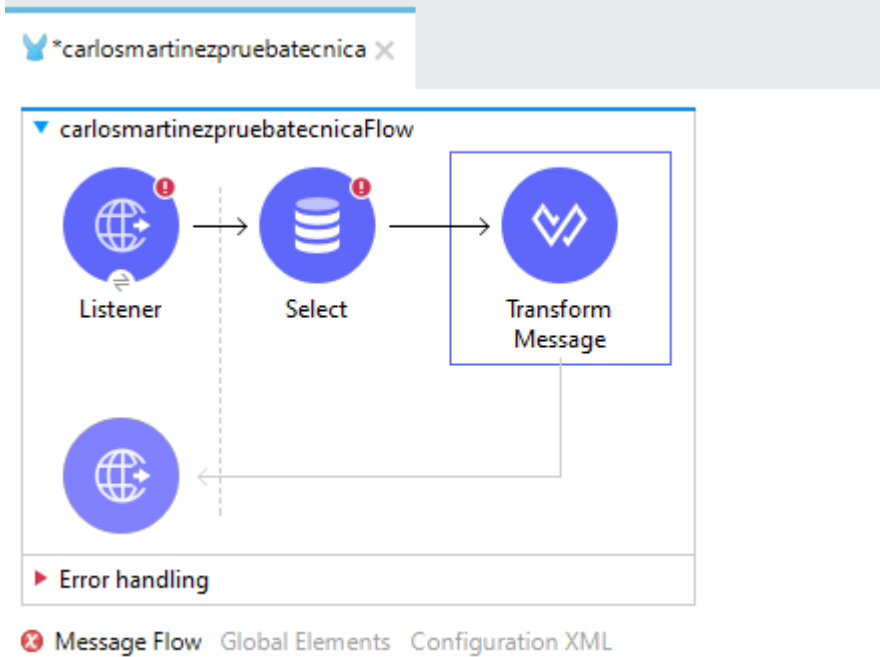
Necesitamos el elemento **Select** del módulo **Database**, lo buscamos y arrastramos a la zona azul



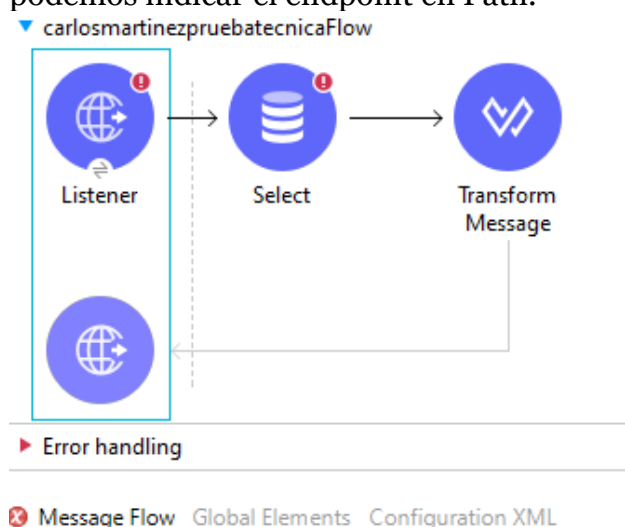
También utilizamos el elemento **Transform Message** disponible en el **Core**



Con los elementos disponibles en el **Mule Palette** los arrastramos a la sección **Canvas**, primero **Listener**, después a la derecha de **Listener** arrastramos el elemento **Select** y a la derecha **Transform Message**.



Para configurar **Listener** hacemos clic sobre este y abajo se mostrarán opciones a modificar, primero necesitamos agregar la configuración de éste elemento haciendo clic en el botón verde y podemos indicar el endpoint en Path.



Listener

Console

Problems

General

MIME Type

Redelivery

Responses

Advanced

Metadata

There are no errors.

Display Name:

Listener

Basic Settings

Connector configuration:

HTTP_Listener_config

+

General

Path:

/api/v1/sps/customers

TÍTULO DEL INFORME

8

Al hacer clic en el botón verde se nos mostrarán campos con información por defecto, la reemplazaremos con las variables de entorno que hemos definido anteriormente.

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

?

Test Connection...

OK

Cancel

Para configurar Select harémops clic sobre este, nos mostrará que podemos agregar una configuración (al hacer clic en el botón verde) y agregar una consulta en el bloque Query.

*carlosalexispru... x

local.secure.prope...

dev.secure.properties

local.properties

dev.properties

▼ carlosalexispruebatecnicaFlow

Listener

Select

Transform Message

Error handling

Message Flow

Global Elements

Configuration XML

Select x

Console

Problems

General

Advanced

Error Mapping

Metadata

Notes

Help

There are no errors.

Display Name:

Select

Basic Settings

Connector configuration:

Database_Config

+

Query

SQL Query Text:

SELECT *
FROM customers

Al hacer clic en el botón verde debemos indicar el tipo Base a la que se conectará en Connection. Necesitamos definir una librería, podemos escoger si la seleccionamos como un archivo que tenemos en la computadora o buscar en el repositorio de Maven (lo que yo hice). Después definiremos las variables con nuestras variables de conexión, tomando en cuenta que creamos un archivo de propiedades seguras.

Global Element Properties

Database Config

Default configuration

General Advanced Notes Help

Name: Database_Config

Connection: MySQL Connection

General Transactions Advanced

Required Libraries

- MySQL JDBC Driver (mysql:mysql-connector-java:8.0.16) Modify...

Connection

Host: fx \${database.connection.host}

Port: fx \${database.connection.port}

User: fx \${secure::database.connection.user}

Password: \${secure::database.connection.password} ☒ Show password

Database: fx i \${database.connection.database}

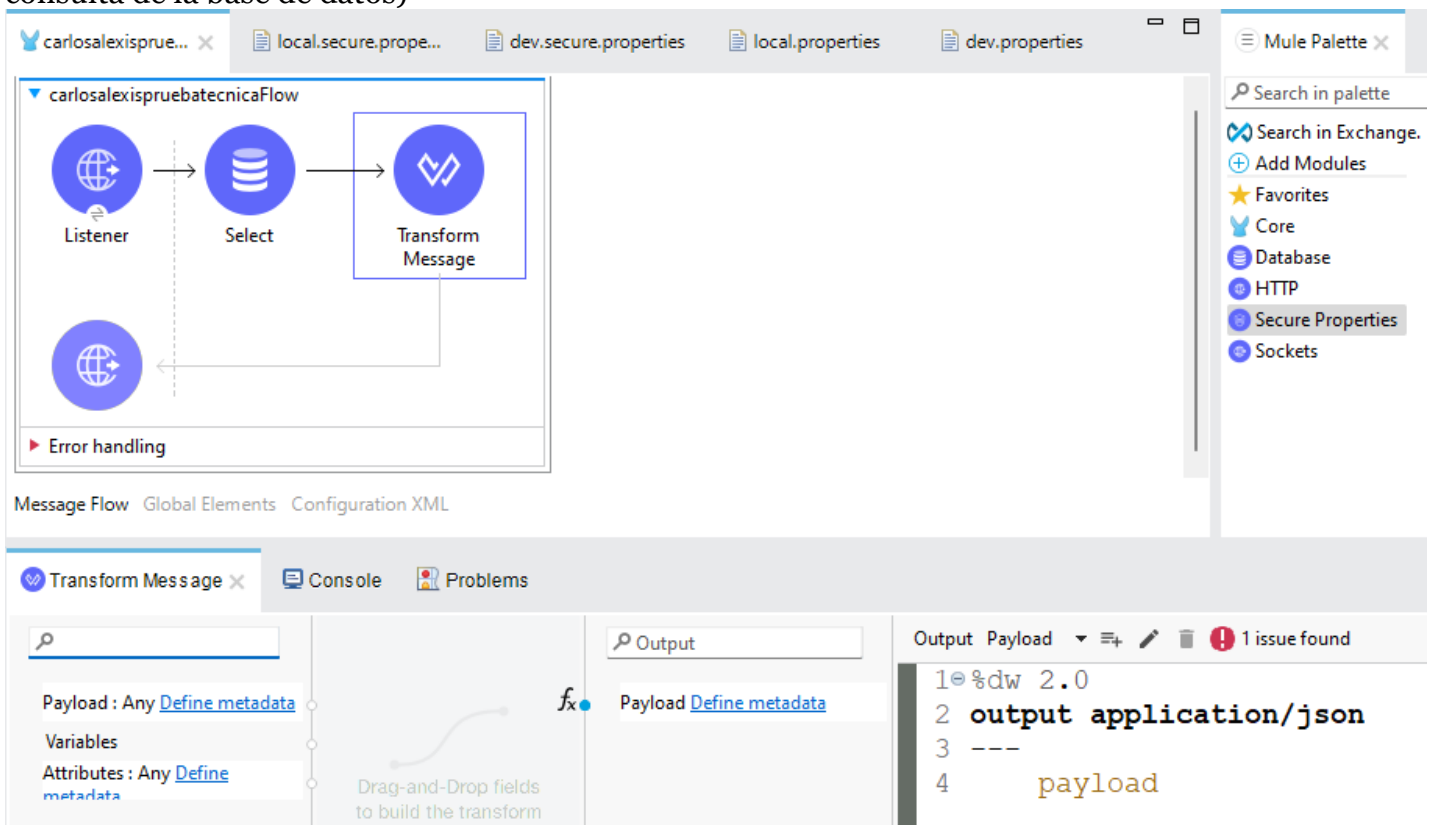
? Test Connection... OK Cancel

La consulta devuelve todos los registros de la tabla **customers**.

```
SELECT *  
FROM customers
```

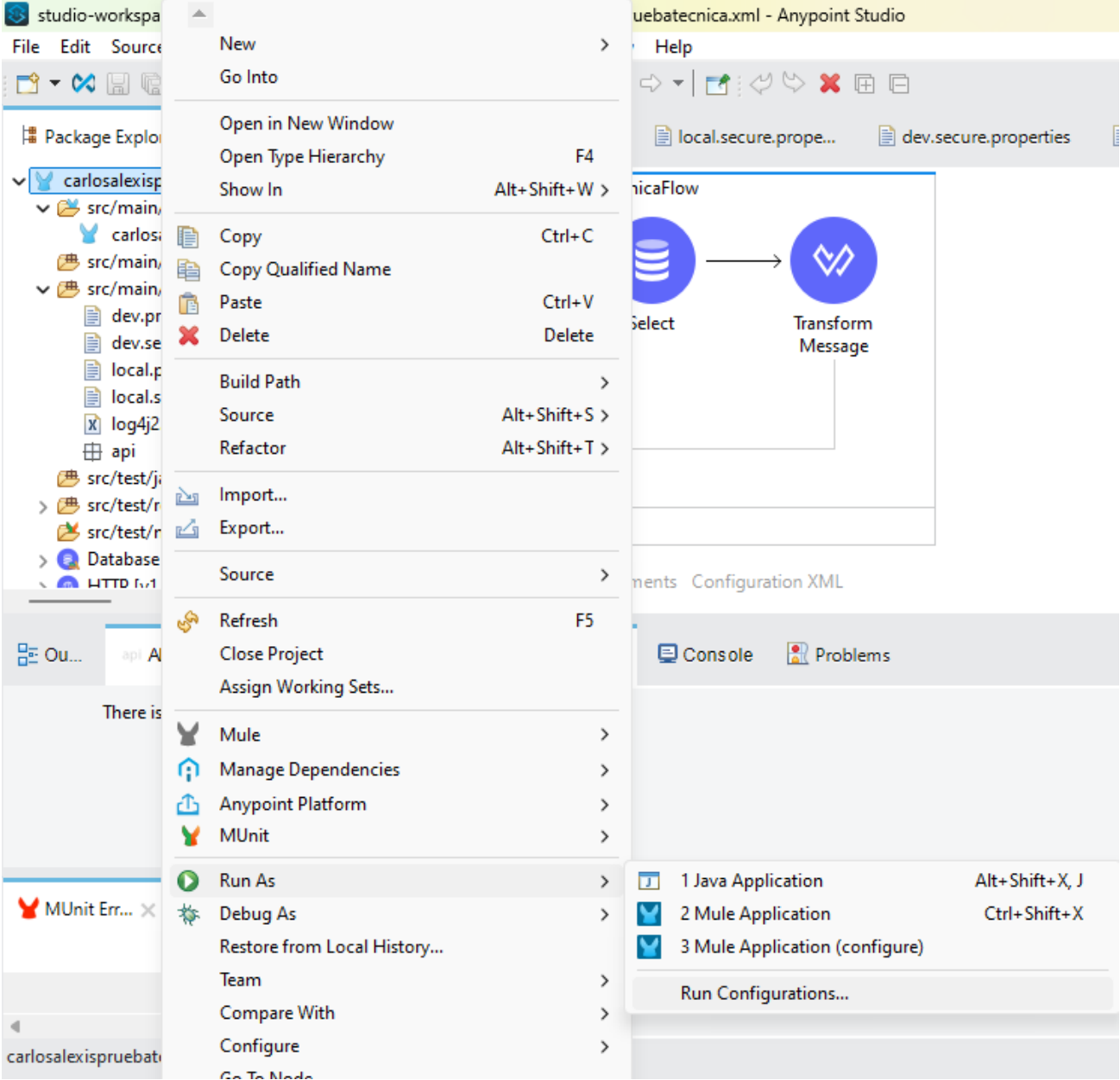
Por último modificamos el elemento Transform Message, esto es necesario para darle formato a la respuesta cuando hacemos la solicitud con un cliente.

Para esto hacemos clic en el elemento Transform Message y en la esquina inferior derecha definimos el tipo de respuesta como **json** y que la respuesta será **payload** (los registros que devuelve la consulta de la base de datos)

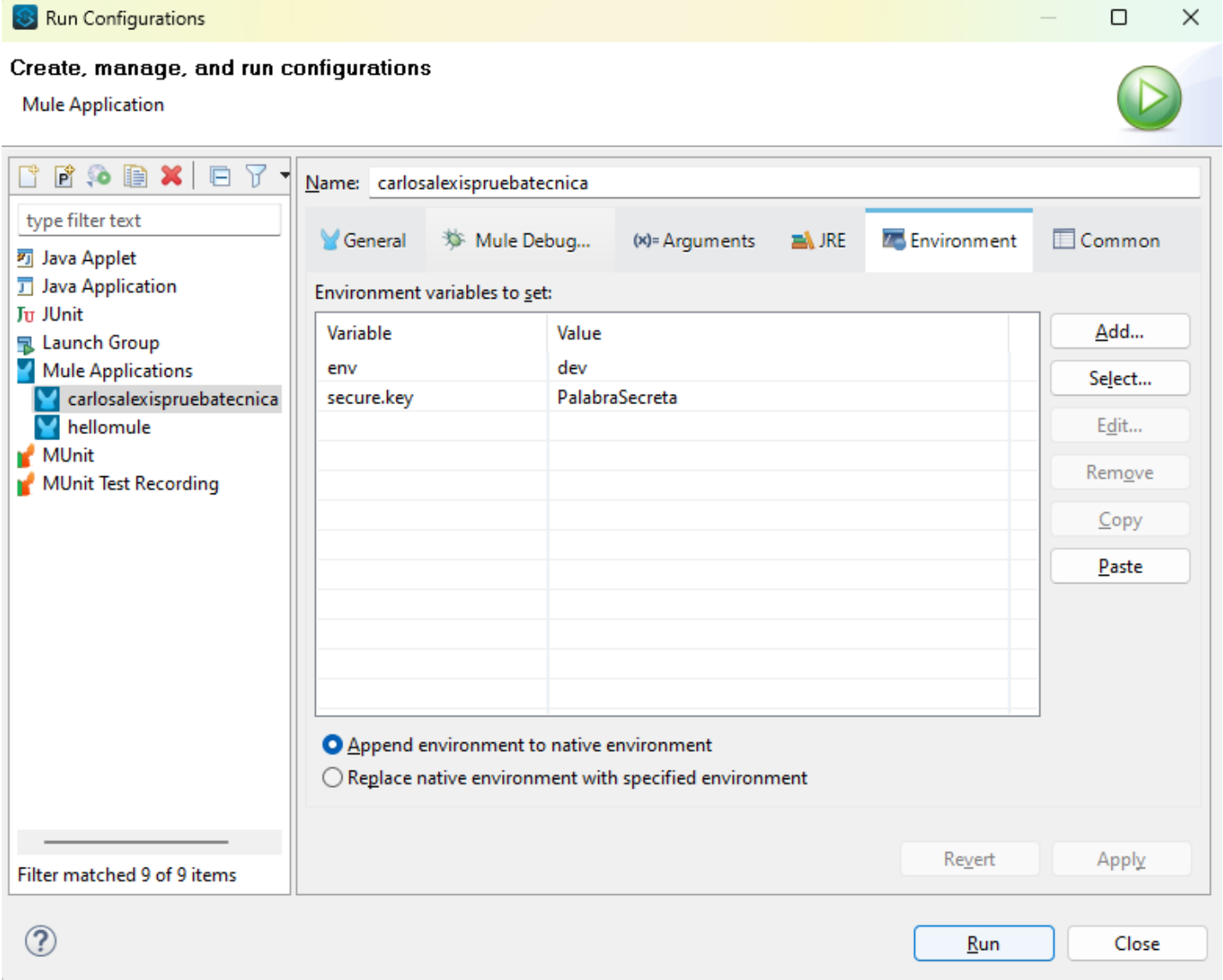


Ejecutar proyecto

Para ejecutar el proyecto debemos definir las variables de entorno que no se definieron en los archivos (env y Secure.key). De forma local debemos ir a la izquierda de la pantalla en el Package Explorer, hacer clic derecho sobre el paquete seleccionar **Run As > Run Configurations**

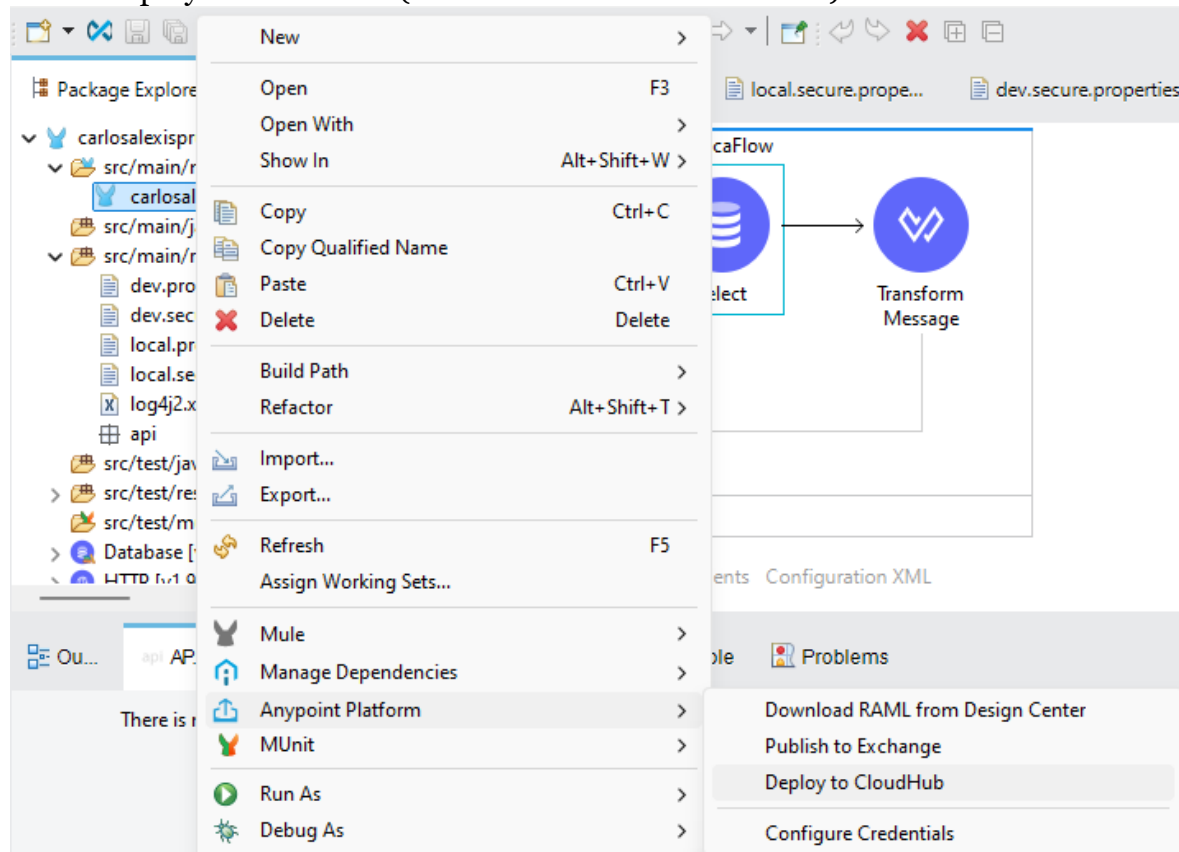


En la sección **Environment** definimos las variables de entorno y ejecutamos:

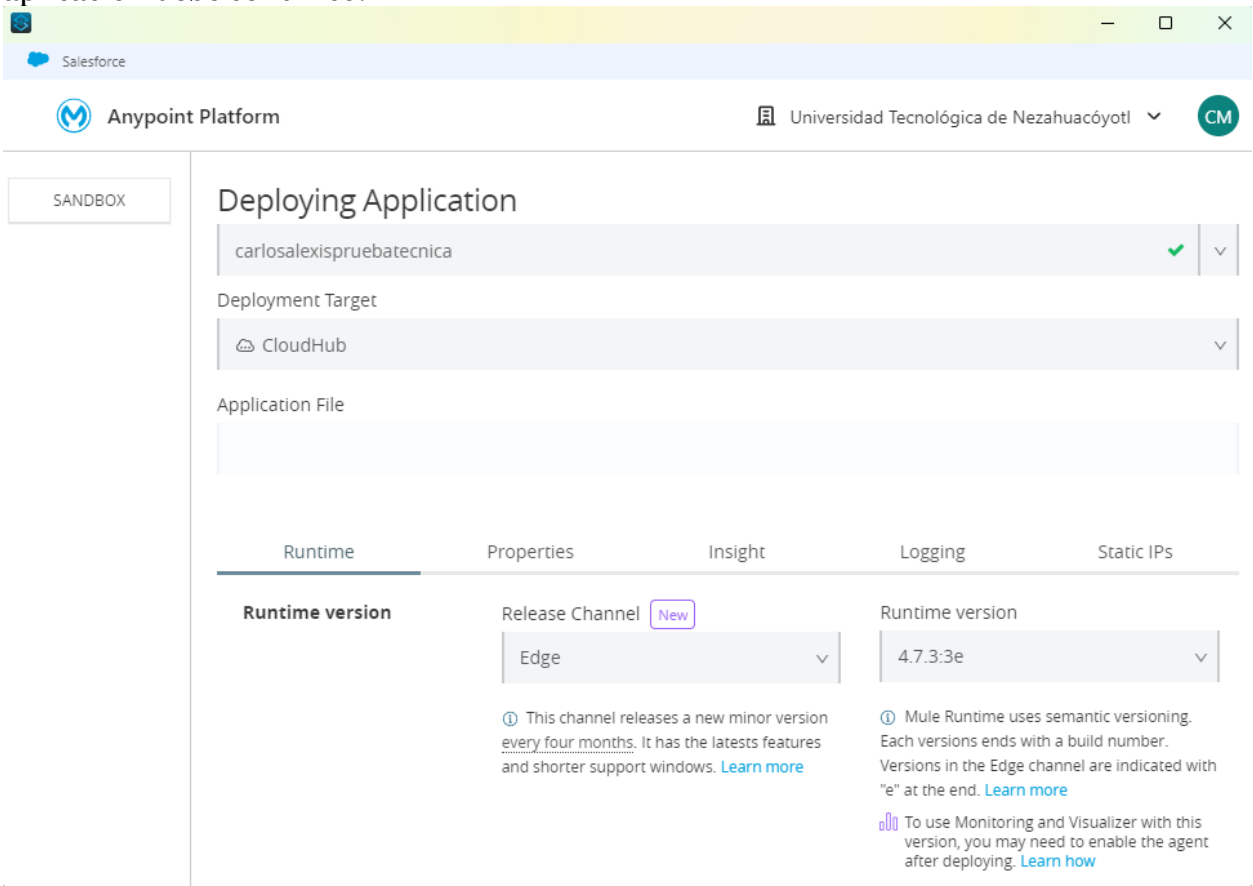


Subir a CloudHub

Debemos hacer clic derecho en nuestro archivo xml que representa el Flow de la API, después hacer clic en Deploy to CloudHub (Debemos de tener una cuenta)



Vamos a slegir **Sandbox** y seleccionar **CloudHub** en **Deployment Target**, el nombre de la aplicación debe ser único.



Podemos dejar la configuración con sus valores por defecto

java version

New

Upgrade to Java 17 based on application compatibility. [Learn more](#)

☒

Java 8

☐

Java 17

Workers

Worker size

0.1 vCores

Workers

1

⚠

Your current subscription allows only one worker per application

Runtime options

☒

Automatically restart application when not responding

☐

Persistent queues

☐

Encrypt persistent queues

☒

Use Object Store v2

Cancel

Deploy Application

Hacemos clic en Properties y definimos las variables, después hacemos clic en **Deploy Application**

Salesforce

Anypoint Platform

Universidad Tecnológica de Nezahuacóyotl

CM

SANDBOX

Deploying Application

carlosalexispruebatecnica

Deployment Target

CloudHub

Application File

Runtime

Properties

Insight

Logging

Static IPs

Table view

Text view

env

dev

secure.key

PalabraSecreta

Cancel

Deploy Application

Para consultar el estado de la aplicación debemos ir a Anypoint Platformer, acceder a nuestra cuenta y dirigirnos a **Runtime Manager**, en **Applications** veremos nuestra app, si todo es correcto sólo debemos esperar a que termine el despliegue.

SANDBOX

Applications

Servers

Flex Gateways NEW

Alerts

VPCs

Private Spaces NEW

Load Balancers

Runtime Manager

Universidad Tecnológica de Nezahuacóyotl

CM

Deploy application

Search Applications

Any Deployment Model

Any Release Channel

Clear filters

Name ^	Target Name	Target Type	Status	Runtime Version	Update Available ⓘ	Date Modified
carlosalexispruebatec...	CloudHub	CloudHub	Started	4.7.3.3e ⓘ	None	2024-10-07 14:57:03
hellomuleapptest	CloudHub	CloudHub	Started	4.7.3.3e ⓘ	None	2024-10-07 08:34:48

Al hacer clic en el nombre de nuestra API podemos ver el nombre del dominio

SANDBOX

Applications

Dashboard

Insight

Logs

Object Store

Queues

Schedules

Settings

Runtime Manager

Universidad Tecnológica de Nezahuacóyotl

CM

Enable Anypoint Monitoring for this application to get a wide array of new metrics and insights. [Learn more](#)

carlosalexispruebatecnica

Domain: carlosalexispruebatecnica.us-e2.cloudhub.io · Last Updated 2024-10-07 2:57:03PM · 1 micro worker, using 4.7-e-java8

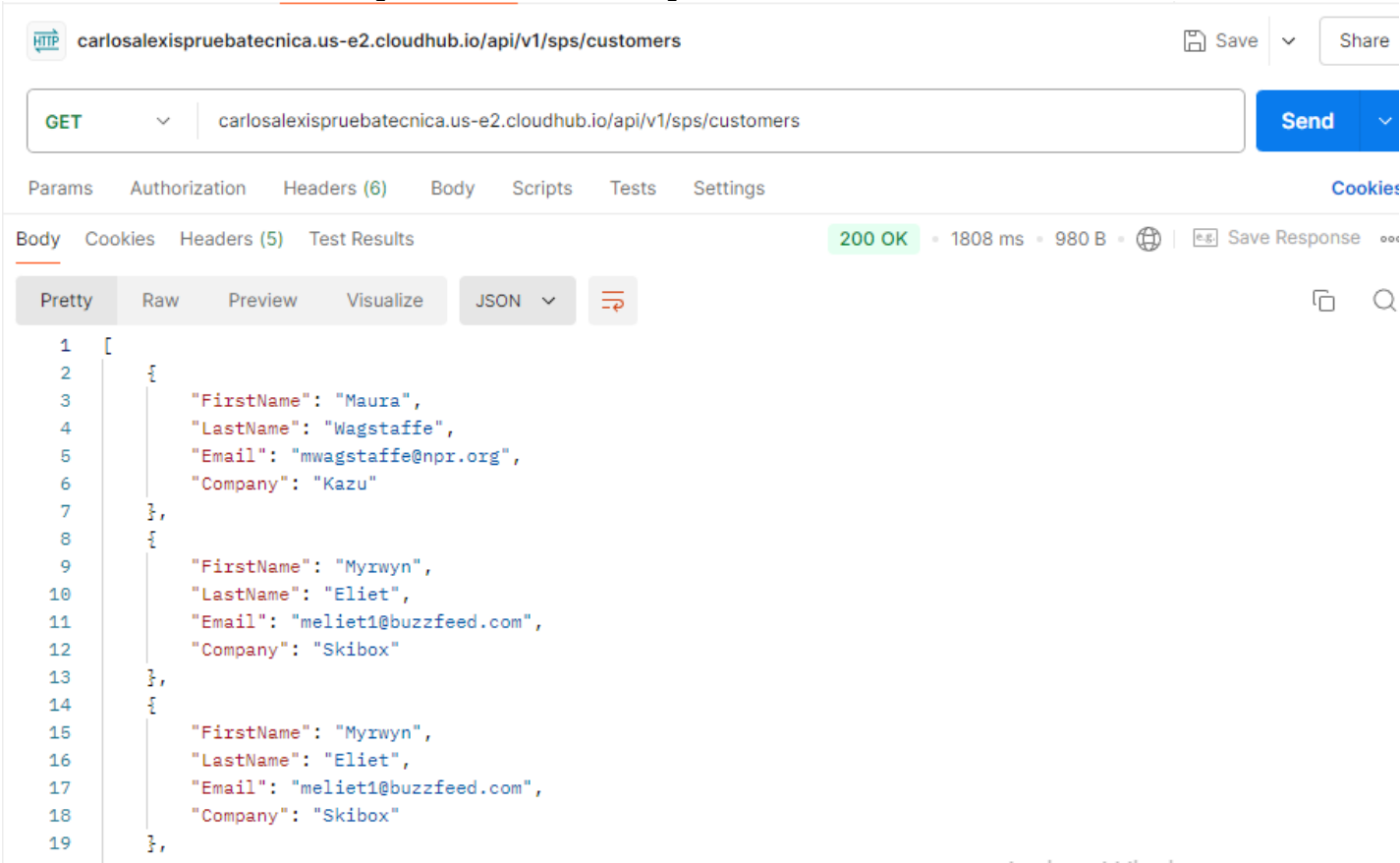
Mule messages

Last hourLast 24hsLast week

Consultar API

Ruta:
carlosalexispruebatecnica.us-e2.cloudhub.io/api/v1/sps/customers

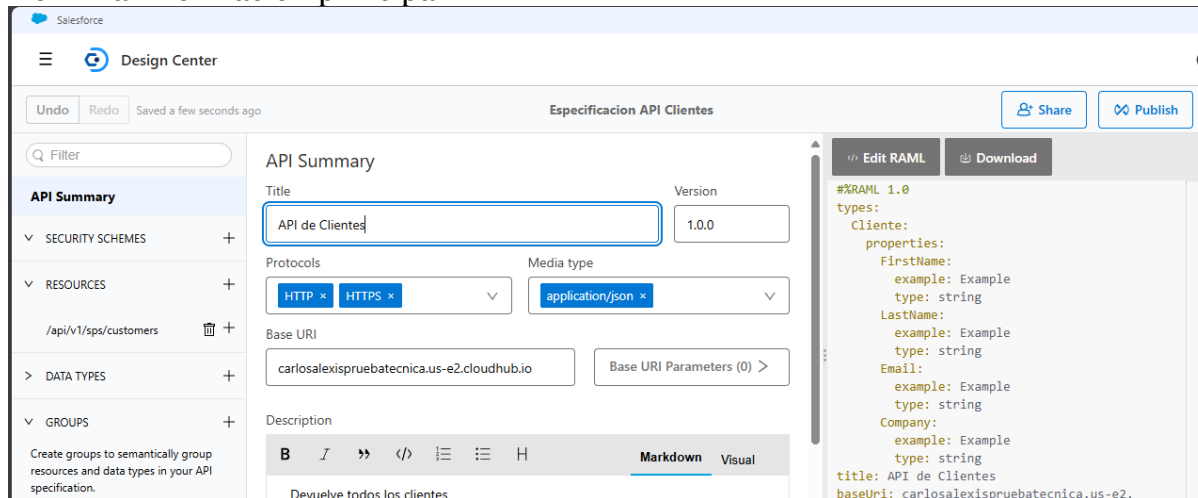
En mi caso utilcé Postman para consultar el endpoint



Especificación de la API

Hay que ingresar a **Anypoint Platformer**, después acceder **Design Center** y hacer clic en **Create**. En mi caso la cree utilizando la ayuda visual.

Definí la información principal



Después de definir el tipo de Dato **Cliente** y el **Resource**, el código de la especificación fue el siguiente:

```
#%RAML 1.0
```

```
types:
```

```
  Cliente:
```

```
    properties:
```

```
      FirstName:
```

```
        example: Example
```

```
        type: string
```

```
      LastName:
```

```
        example: Example
```

```
        type: string
```

```
      Email:
```

```
        example: Example
```

```
        type: string
```

```
      Company:
```

```
        example: Example
```

```
        type: string
```

```
title: API de Clientes
```

```
baseUrl: carlosalexispruebatecnica.us-e2.cloudhub.io
```

```
description: Devuelve todos los clientes
```

```
mediaType:
```

```
  - application/json
```

```
version: 1.0.0
```

```
protocols:
```

```
  - HTTP
```

```
  - HTTPS
```

```
/api/v1/sps/customers:
```

```
  get:
```

```
    displayName: Obtener lista de clientes
```

```
    responses:
```

```
      "200":
```

```
        body:
```

```
          items:
```

```
            type: Cliente
```

Conclusiones

Es la primera vez que utilizo los productos de Mule Soft y en todo fue cómodo, mi primer acercamiento con las integraciones fue usando OIC, pero siendo franco las sentí complejas, “pesadas”. Trabajar con Anypoint Studio ayudó por su similitud a Eclipse.

El alcance de la actividad no define su dificultad, ya que con los mismos tutoriales todo fue quedando muy claro, me gustaría haberla terminado, ya que en más de un momento me pareció una actividad entretenida.

Me gustó y con tiempo me veo dominando el tema.