Práctica MuleSoft Trainee

Brandon Josué Guerrero Pérez

4 de octubre de 2024

Resumen

Esta práctica tiene como objetivo identificar las habilidades de autoaprendizaje y solución de problemas a través de la exploración de una tecnología relativamente nueva llamada ”MuleSoft”.

En SPS el equipo de marketing tiene la necesidad de tener disponible la información de nuestros clientes con la intención de poder generar campañas personalizadas que permitan mejorar la experiencia y la satisfacción del cliente, esta información se encuentra almacenada en una Base de Datos.

Para la realización de esta práctica usé el REST Cliente ”Postmanza que estoy más familiarizado con este, además de tener que descargar Anypoint Studio. En la Figura 1 se muestra la interfaz de Anypoint Studio una vez descargado el software.

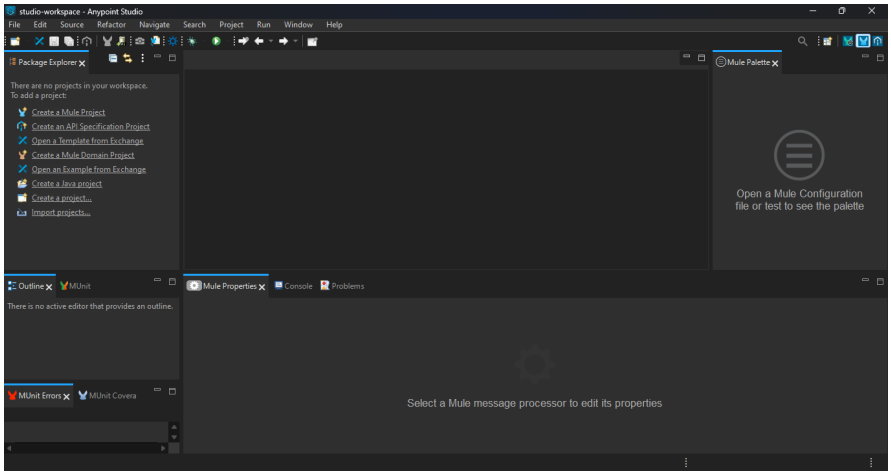
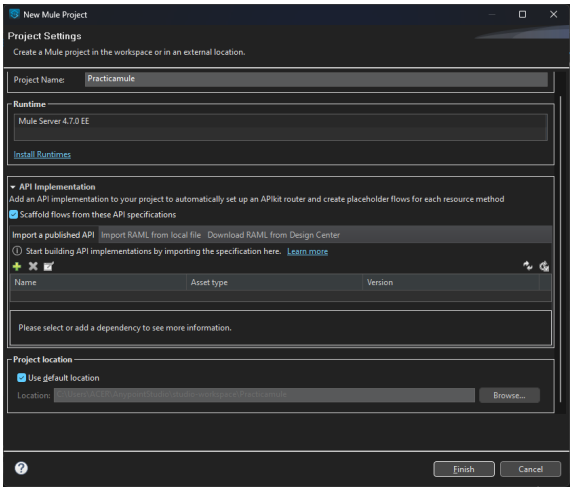


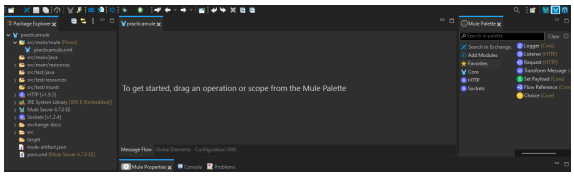
Figura 1: Interfaz de Anypoint Studio.

0.1. Configuración del proyecto en Anypoint Studio

- **Creación del proyecto:** Lo primero que se hará es crear un nuevo proyecto Mule (en este caso lo llamaremos Practicamule), esto nos arrojará una pantalla como se muestra en la Figura 2a. Una vez creado el proyecto nos aparecerán herramientas para poder construir nuestro proyecto como se aprecia en la Figura 2b.



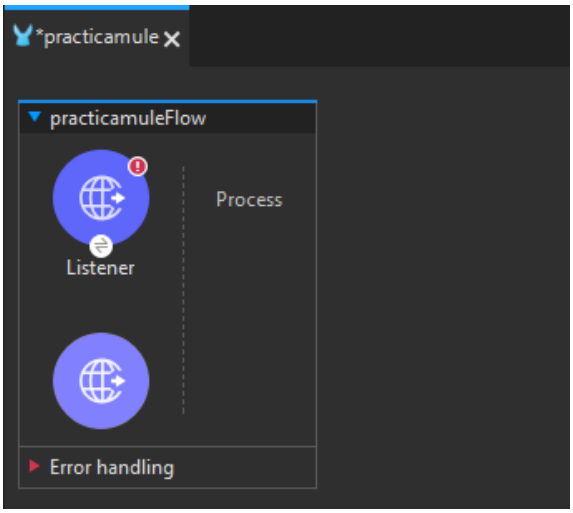
(a) Creación de nuevo proyecto.



(b) Vistas y herramientas.

Figura 2: Proceso de creación y herramientas en Anypoint Studio.

- **Añadir y configurar el conector de escucha HTTP:** Para añadir el conector de escucha HTTP debemos arrastrar el Listener (HTTP) que se encuentra en Mule Palette al canvas y el resultado será como se muestra en la Figura 3a. Al dar click en el listener nos aparecerá una vista para configurar el conector escucha (Para la configuración del Host en este caso será 0.0.0.0 y el puerto será el 8081) y como el conector va a representar el endpoint que ejecutará su flujo cuando se realice una solicitud HTTP a su escucha HTTP, por lo que debemos configurarlo para que la información de los clientes esté expuesta en la ruta: /api/v1/sps/customers, como se muestra en la Figura 3b.



(a) Conector de escucha HTTP.

(b) Configuración de conector de escucha HTTP.

Figura 3: Configuración del conector de escucha HTTP en Anypoint Studio.

Ahora para dar respuesta al cliente utilizaremos ”Set Payload”, arrastrándolo de la sección de favoritos en Mule Palette al flujo dentro de la sección proceso y posteriormente agregaremos lo que se desea que el cliente reciba como respuesta como se muestra en la Figura 4.

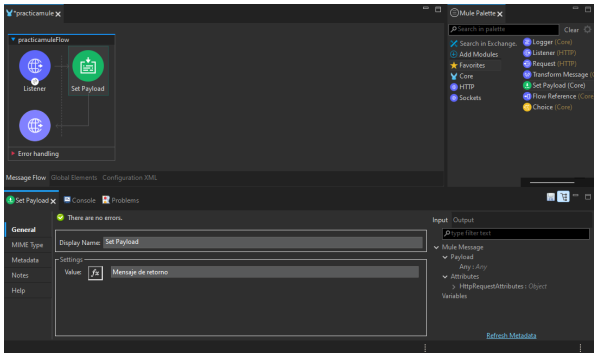


Figura 4: Configuración de payload.

0.2. Configuración de archivos de propiedades

- **Creación de los archivos de propiedades:** Estos archivos deben ser creados dentro de la carpeta src/main/resources, una vez en esa carpeta debemos seleccionar New ->File (Véase en la Figura 5).

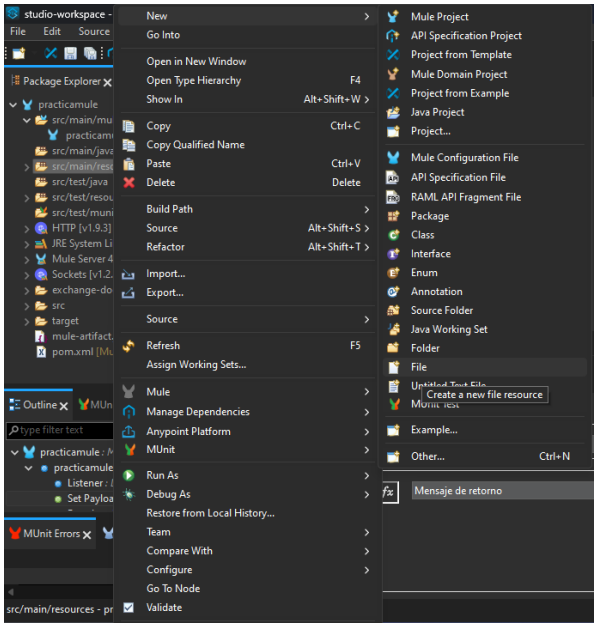


Figura 5: Creación de archivos de propiedades.

Una vez seleccionada la opción de crear un nuevo archivo, nombraremos uno como `local.properties` y el otro como `dev.properties`, como se muestra en la Figura 6. Esta práctica evita tener valores codificados directamente en el código, favoreciendo la externalización en archivos de propiedades.

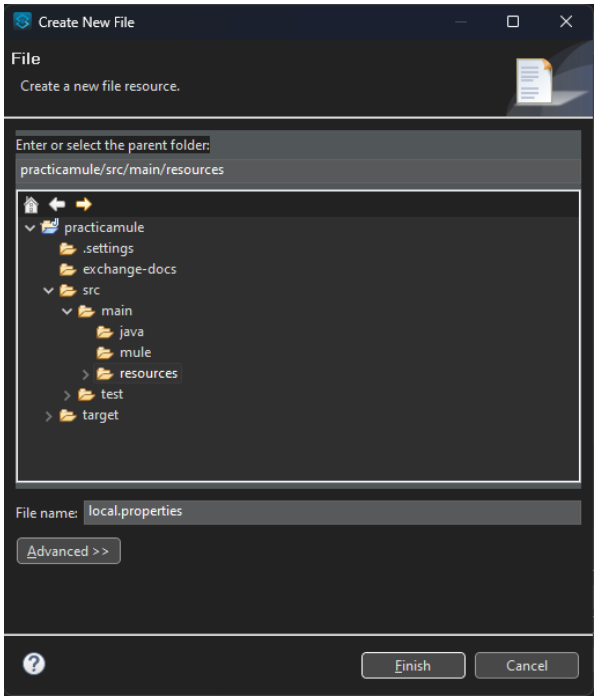


Figura 6: Creación de archivos de propiedades.

Con los archivos ya creados, procederemos a definir las propiedades que contendrá cada uno, como se ilustra en la Figura 7. Esto facilita la gestión diferenciada de configuraciones por entorno: `local.properties` para desarrollo local y `dev.properties` para despliegues en CloudHub.

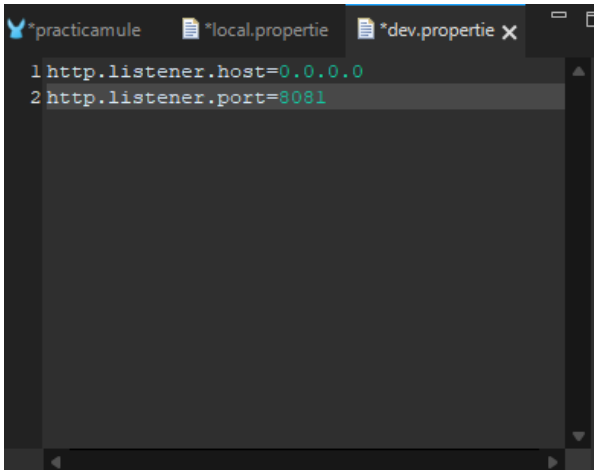


Figura 7: Edición de archivos de propiedades.

0.3. Creación de archivos de configuración global

Para la creación de archivos de configuración Mule, hacemos clic derecho en la carpeta `src/main/mule`, seleccionamos *New* y *Mule Configuration File* como se muestra en la Figura 8. Nombraremos el archivo como `global.xml` y finalizaremos el proceso como se indica en la pantalla.

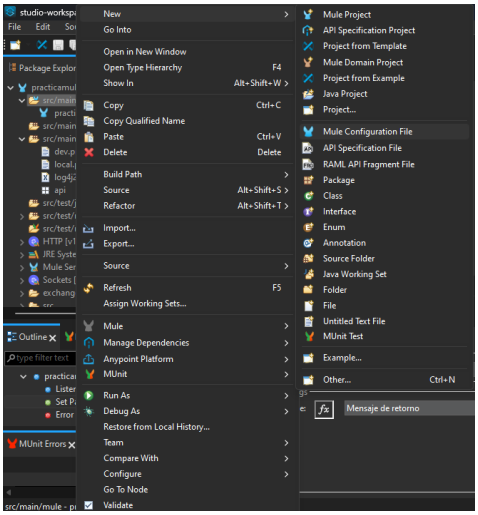


Figura 8: Creación de archivo de configuración global.

Después de crear el archivo de configuración global, nos dirigimos al archivo de configuración de la implementación, cortamos el código `http:listener-config` y lo pegamos en el archivo global, como se observa en la Figura 9.

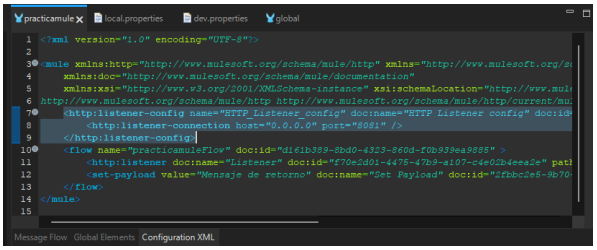


Figura 9: Movimiento de configuración de escucha HTTP.

Al mover el `http:listener-config` al archivo global, el archivo `global.xml` quedará configurado como se ilustra en la Figura 10.

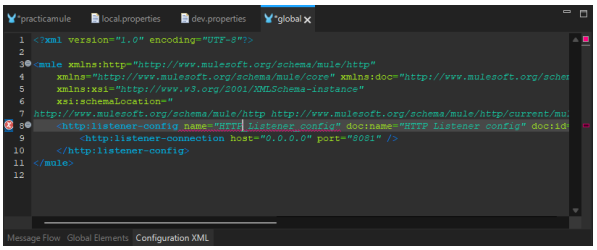
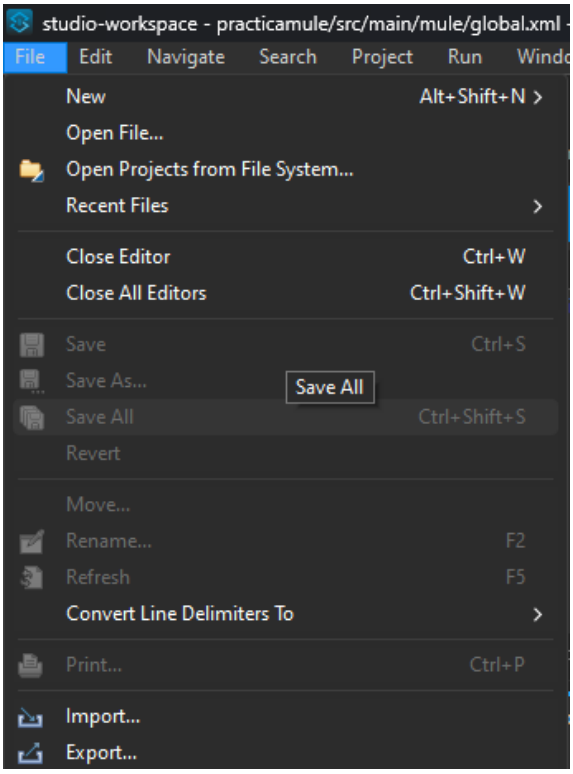
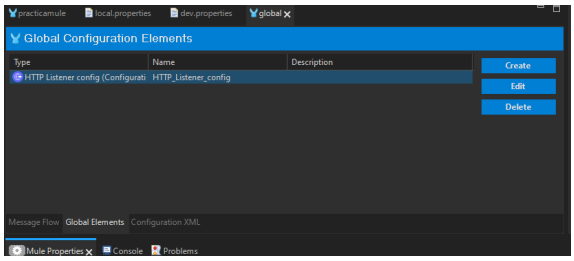


Figura 10: Código http:listener-config en global.xml.

Una vez realizados los cambios y guardados los archivos 11a, la configuración del HTTP Listener ya no se encontrará en el archivo `practicamule.xml` sino que se habrá trasladado al archivo `global.xml`, como se muestra en la Figura 11b.



(a) Guardar cambios en Anypoint Studio.



(b) Cambio de ubicación de HTTP Listener config.

Figura 11: Cambios en HTTP Listener

0.4. Configurar el elemento global de propiedades de configuración

Agregar el archivo de propiedades no es suficiente para que la aplicación Mule sepa dónde buscarlos. Si intenta ejecutar la aplicación sin esta configuración, arrojará un error indicando que no pudo encontrar sus propiedades. En nuestro archivo `global.xml`, iremos a la vista de Global Elements y crearemos un nuevo elemento global seleccionando *Configuration properties* como se muestra en la Figura 12.

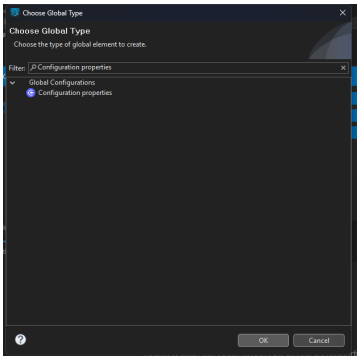


Figura 12: Crear un nuevo elemento global de propiedades.

Como alternamos entre archivos de propiedades según el entorno, utilizaremos la sintaxis $ para referenciar propiedades. Configuraremos el archivo de propiedades como $env.properties, ilustrado en

la Figura 13.

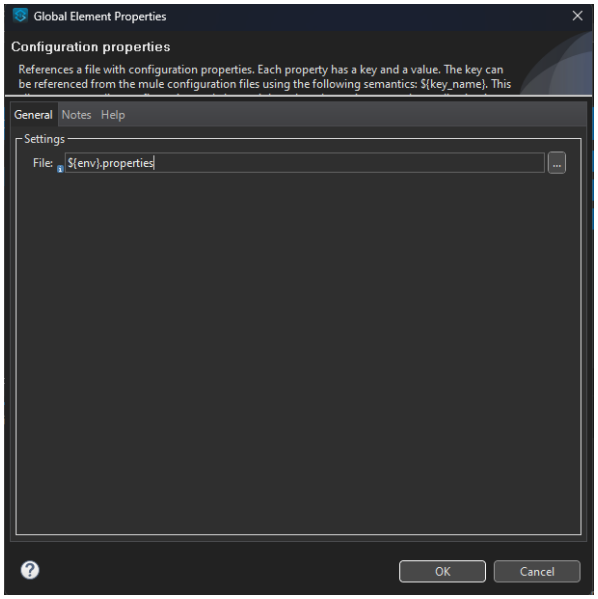


Figura 13: Referenciando archivos de propiedades dinámicos.

Aún necesitamos agregar la propiedad `env` que acabamos de referenciar. Como no es una propiedad segura, la añadiremos a los elementos de configuración global creando una propiedad global con nombre `env` y valor `local` como se muestra en la Figura 14.

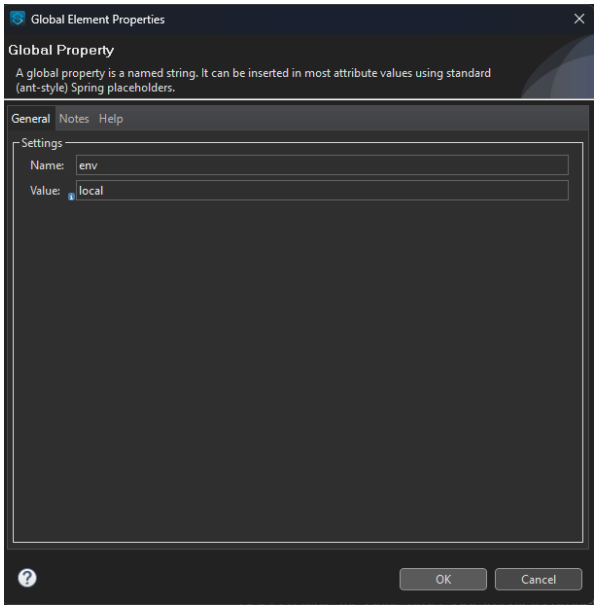


Figura 14: Crear una propiedad global `env`.

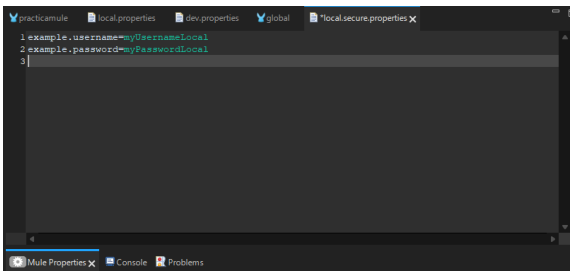
0.5. Seguridad de usuarios y contraseñas con secure properties

- **Cree los archivos de propiedades seguros de sus entornos:** Para mayor seguridad es una buena práctica separar las propiedades seguras en archivos por entorno. Para crear un archivo local.secure.properties, hay que hacer click derecho en la carpeta src/main/resources y dar click en New – File y llamar al archivo local.secure.properties (Vease el ejemplo de la Figura 6).

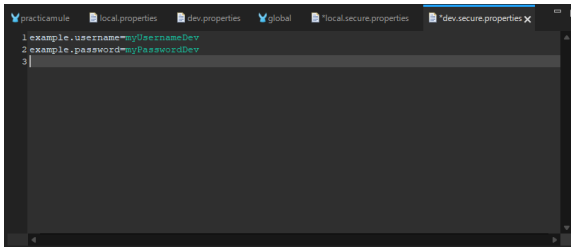
Una vez creado el archivo damos doble click para añadir una nueva entrada y añadimos las propiedades:

example.username=myUsernameLocal
example.password=myPasswordLocal

Después de ingresar las credenciales privadas en el archivo local.secure.properties, se deben repetir los mismos pasos para el archivo dev.secure.properties como se muestra en la Figura 15a.



(a) Credenciales en local.secure.properties



(b) Credenciales en dev.secure.properties

Figura 15: Añadir propiedades.

El siguiente paso es bucar en `Exchange`, que se encuentra en Mule Palette, el módulo *Mule Secure Configuration Properties*. Y una vez que se encuentre daremos click en `Add >` como en la Figura 16.

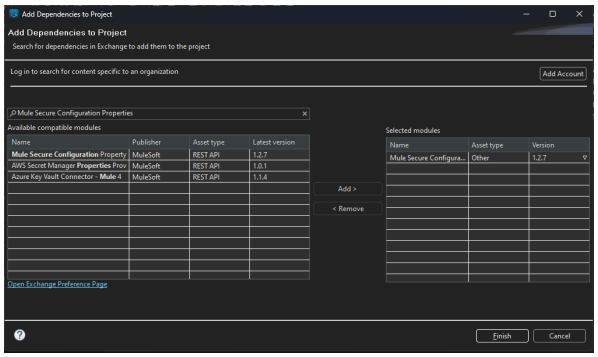


Figura 16: Busqueda en Exchange.

Después de importar el módulo de propiedades de propiedades de seguridad, hay que ir a la vista de elementos globales. Dar click en el botón `Create` y crear *Secure Properties Config* y llenarlo como se muestra en la Figura 17.

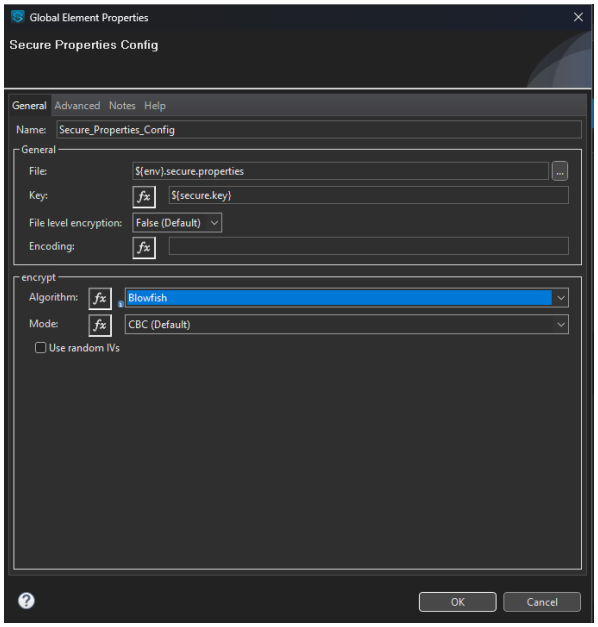
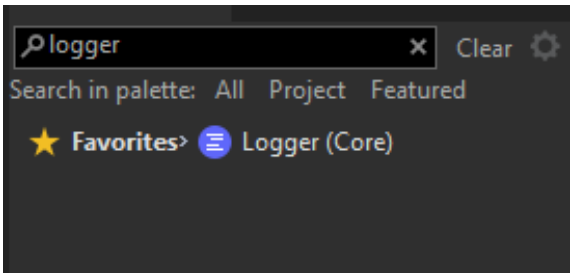
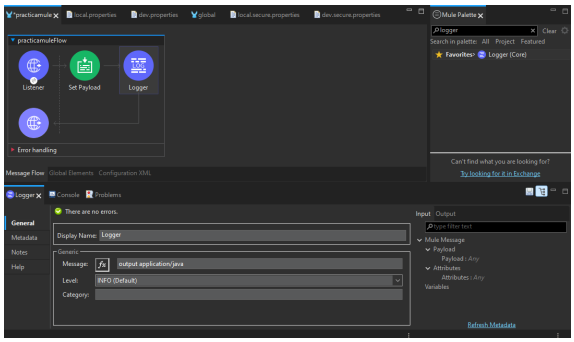


Figura 17: Llenado de Secure Properties Config .

- **Configurar un logger para leer credenciales seguras:** Para configurar el logger debemos de ir al archivo praticamule.xml y agregar un componente logger desde Mule Palette como se ve en la Figura 18a



(a) Ubicación del componente logger



(b) Componente logger agregado

Figura 18: Colocar el componente logger.

Después de agregar el componente logger, entramos a la configuración y damos click en botón fx, después de eso en la derecha de la barra donde esta fx hay un botón para vista de gráfica al hacer click abrirá una pantalla donde se podrá ingresar el siguiente código (Vease la Figura 19):

output application/java

—

Üsername: -+ Mule::p(”secure::example.username”)

++ -+

”Password: -+ Mule::p(”secure::example.password”)

Y posteriormente dar click en ”done”.

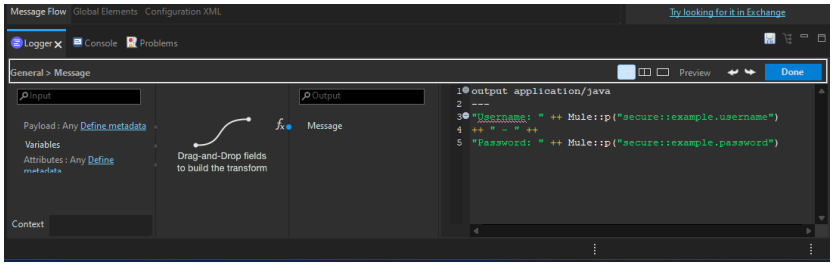


Figura 19: Ingresar el código para el logger.

- **Encriptar los valores de los archivos de propiedades:** Para este paso necesitaremos la `Secure Properties Tool Jar` de la documentación oficial de MuleSoft. Deberemos abrir una terminal en la dirección donde se encuentra nuestro archivo jar y compilar lo siguiente (como se muestra en la Figura 20): "java -cp secure-properties-tool.jar com.mulesoft.tools.SecurePropertiesTool string encrypt Blowfish CBC MyMuleSoftKey "myUsernameLocal

  Esto devolverá el valor cifrado de `myUsernameLocal`, que es la propiedad que estamos usando como ejemplo. Tenga en cuenta que la clave que utilizamos para cifrar la propiedad es `MyMuleSoftKey`. Usaremos este valor como propiedad Secure.key para descifrar los valores en tiempo de ejecución.

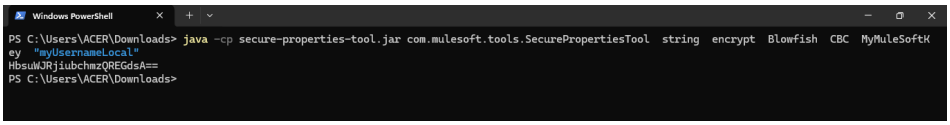


Figura 20: Terminal con el valor de myUsernameLocal.

Después de obtener el valor encriptado, se necesitará agregar la siguiente sintaxis en los archivos de propiedades: ¡[ValorEncriptadoObtenido] y además se necesitará obtener el valor de los usuarios y contraseñas tanto de local.secure.properties como de dev.secure.properties. Una vez obtenidos todos los valores encriptados se deberan poner en los archivos como se muestra en la Figura 21

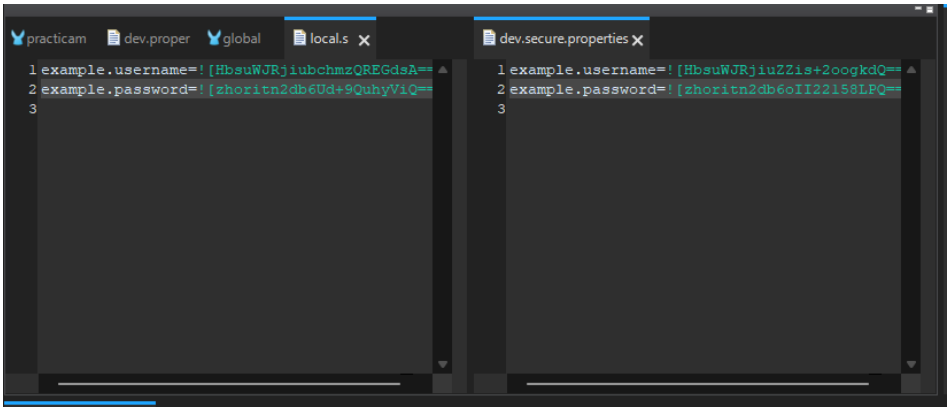


Figura 21: Archivos de propiedad con sus valores encriptados.

- **Probar la aplicación en la computadora local:** Para probar la aplicación debemos de dar click derecho en el proyecto Mule y seleccionar `Run As -- Run Configurations` como se muestra en la Figura 22

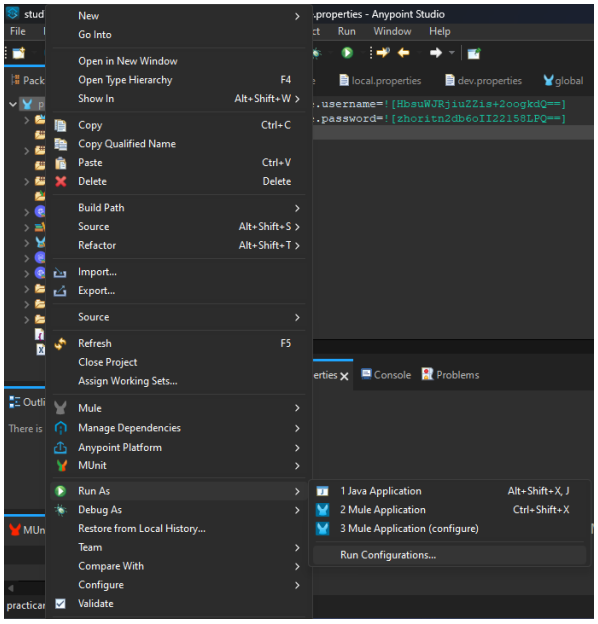


Figura 22: Configuración de compilación.

Una vez abierta la configuración daremos click en `Mule Application` nos iremos a la vista de `Environment`, daremos click en `Add...` y en `name` escribiremos *secure.key* y en `value` escribiremos *MyMuleSoftKey* (Vease la Figura 23).

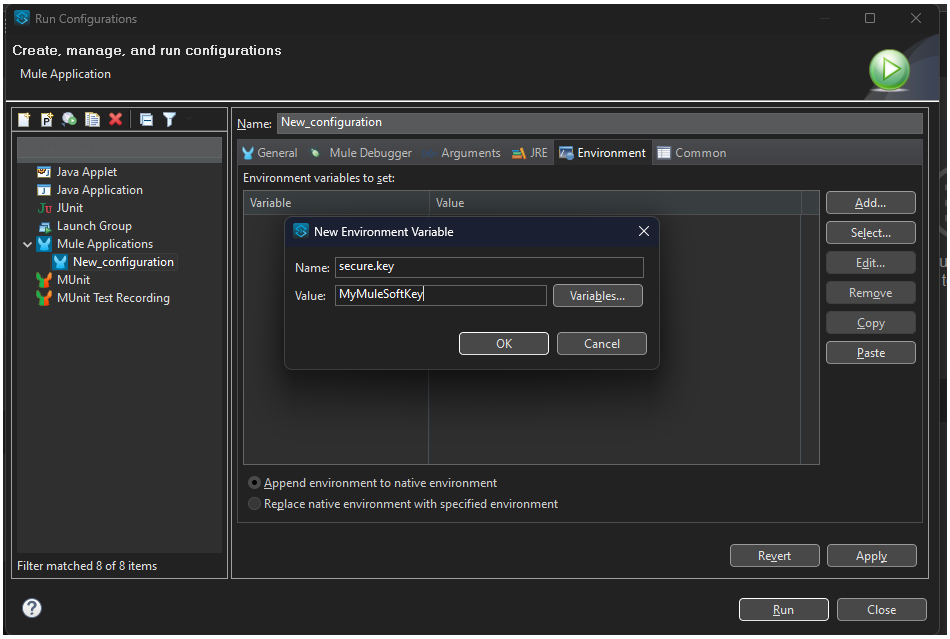
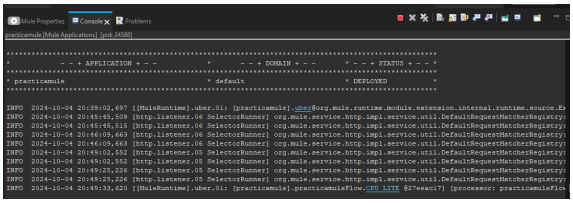
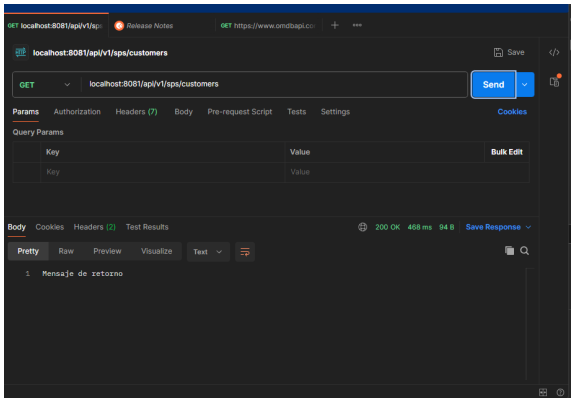


Figura 23: Configuración de compilación.

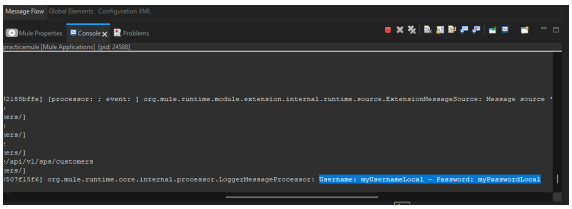
Después de hacer la configuración, se debe correr la aplicación y se realizará la respectiva petición HTTP desde postman como se mostrará en la Figura 24b. Y posteriormente enviar una petición a localhost:8081/practicamule, y los valores que se muestran son los valores desencriptados en la consola como se muestra en la Figura 24c.



(a) Proyecto compilado



(b) Petición HTTP desde postman



(c) Resultado obtenido

Figura 24: Resultados.

- **Implementar en CloudHub con las credenciales seguras:** Para implementar el proyecto en CloudHub damos click derecho en el proyecto y vamos a `anypoint Platform` y después seleccionamos `Deploy to ClubHub`. Una vez que demos click sino tenemos una cuenta se nos pedira que nos registremos como muestra la Figura 25.

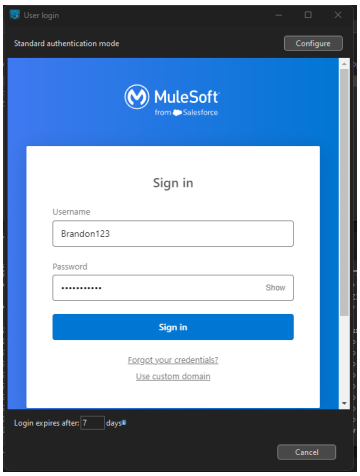


Figura 25: Ingreso a CloudHub.

Ya una vez registrados nos saldra la siguiente pantalla (Vease la Figura 26) para escoger el entorno con el que trabajaremos y escogeremos `sandbox`.

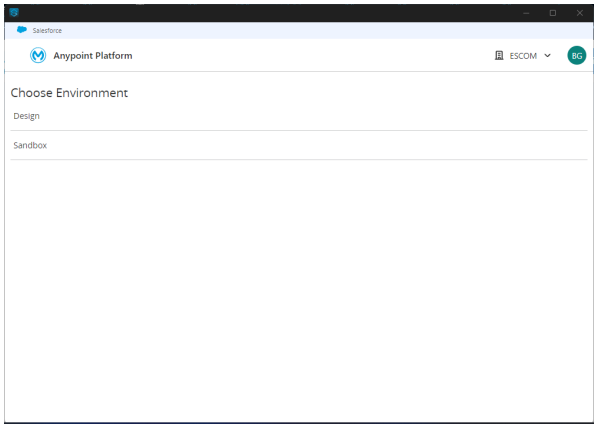


Figura 26: Ingreso a CloudHub.

Una vez ingresado el tipo de entorno con el que se trabajará iremos al apartado de `properties` y añadiremos en `New Key` los valores *env* y *secure.key* y en `New Value` los valores *dev* y *MyMuleSoftKey* respectivamente como lo mostrado en la Figura 27.

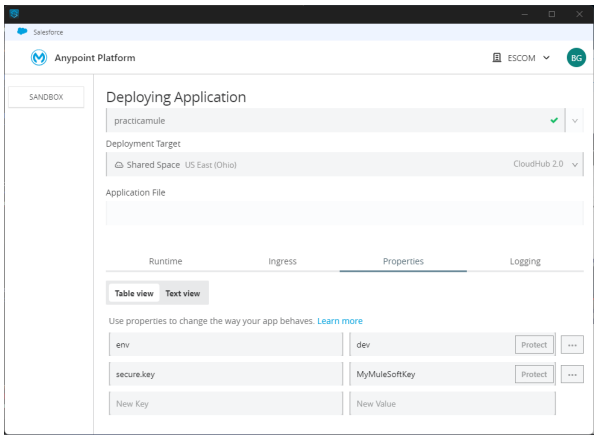


Figura 27: Añadir propiedades a CloudHub.

Finalmente la aplicación esta lista para ser implementada en CloudHub como se puede ver en la Figura 28.

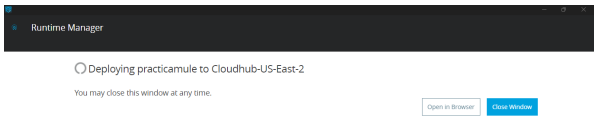


Figura 28: Aplicación implementada en CloudHub.