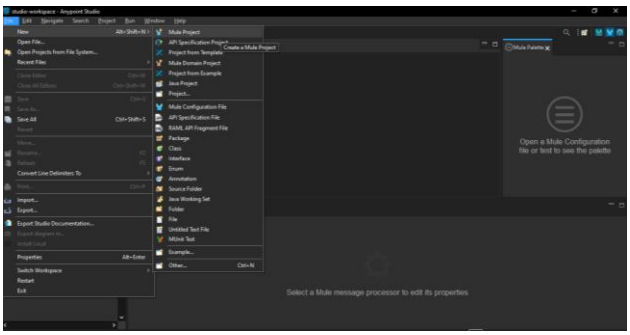
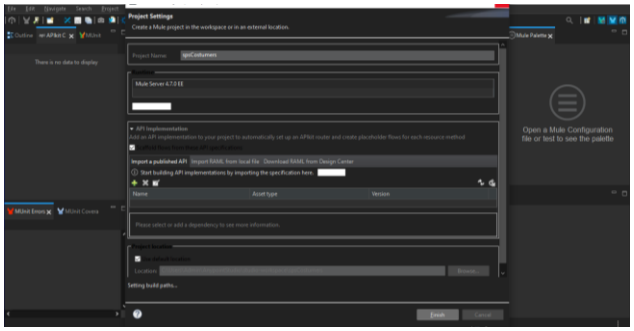
APLICACIÓN EN MULESOFT

Creación de la aplicación

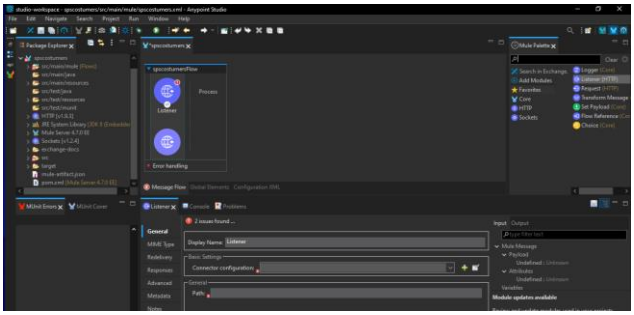
Una vez entrando al IDE de Anypoint Studio del lado superior izquierdo encontraremos la pestaña de File en el cual le daremos click en New y posteriormente en Mule Project, tal como se muestra en la imagen siguiente.



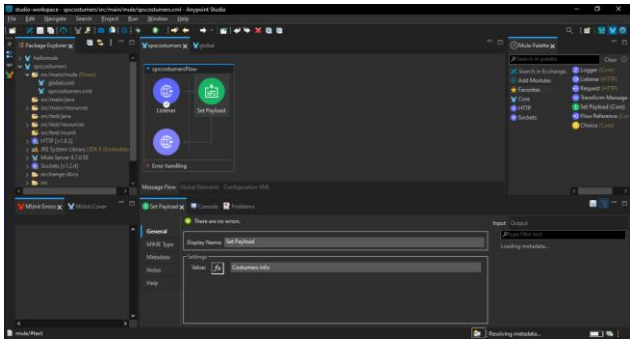
Se nos mostrara una nueva ventana en la cual escribiremos el nombre de nuestro proyecto, en este caso lo llamamos spsCostumers, finalmente daremos click en el botón Finish.



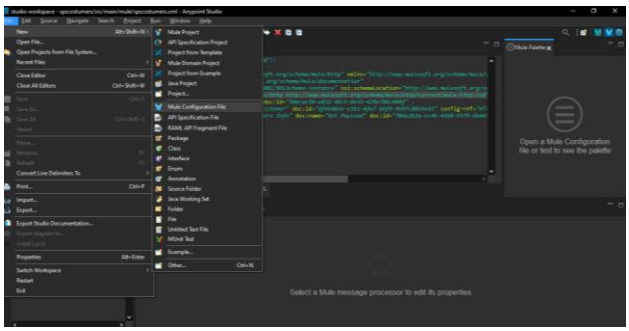
Una vez creado el proyecto se nos mostrará las diferentes ventanas dentro del IDE, tales como el explorador del paquete, el canvas y la paleta donde se muestran los módulos de mule, para iniciar arrastramos un módulo “Listener(http)” de la paleta al canvas, la configuración se deja por default ya que se verá más a profundidad en los siguientes pasos, en la pestaña Path, que se encuentra en la parte inferior se coloca la dirección que se quiere utilizar en este caso usamos la siguiente” /api/v1/sps/customers”.



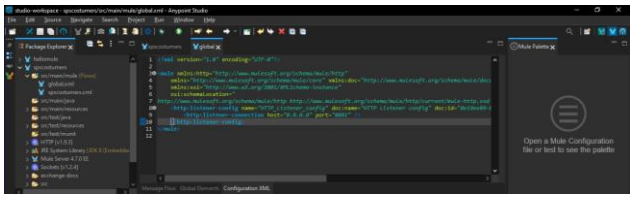
Para continuar agregamos un módulo “Set Layout” de la misma forma que lo hicimos con el Listener, con este módulo la única configuración que se realiza es la de selección el botón de la parte inferior en la sección de value, y designar el mensaje a mostrar, en este caso se optó por “Costumer Info”.



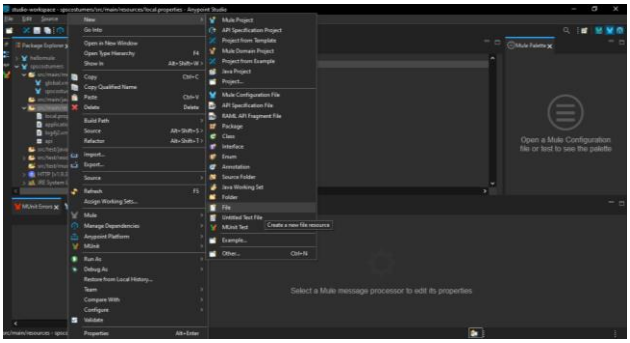
Para cumplir con las buenas practicas se crea un archivo de configuraciones globales, esto se realiza dando click derecho en el nombre del proyecto, que se encuentra en el lado derecho en la ventana del explorador, después de esto se da click en New y finalmente en Mule Configuration File.



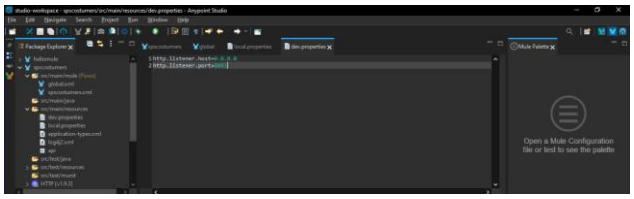
Una vez creado el archivo global pasamos el código de configuración del listener que se encuentra en el archivo spscostumer.xml al archivo global.xml y guardamos para evitar errores de duplicación.



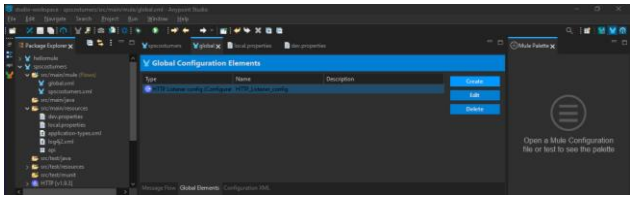
Para continuar con las buenas prácticas creamos dos archivos de propiedades uno para uso local y otro para uso en la nube, para realizar esto damos click derecho en la carpeta con el nombre “src/main/resources”, luego en New y finalmente en File, los nuevos archivos los llamaremos local.properties y dev.properties



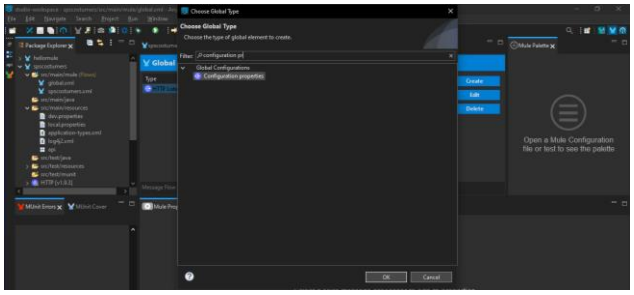
Con los archivos creados, guardaremos en ellos la configuración del puerto y el Host que va a usar nuestra aplicación, los cuales se llaman http.listener.host y http.listener.port con los valores 0.0.0.0 y 8081 correspondientemente.



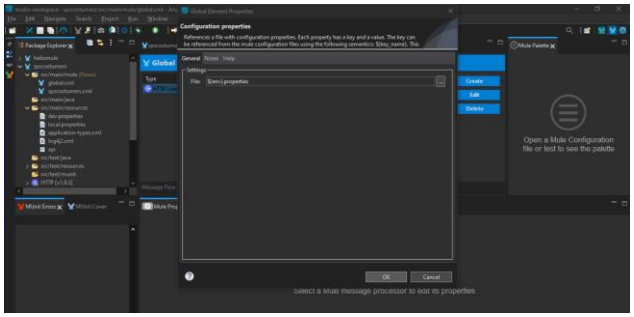
Terminando de crear y guardar nuestros archivos de propiedades, nos dirigiremos al archivo global para hacer las configuraciones correspondientes, editaremos el HTTP listener config en la parte del Host escribiremos lo siguiente “${http.listener.host}” y en la parte del puerto “${http.listener.port}” (el signo de dinero y los corchetes indican que es una variable).



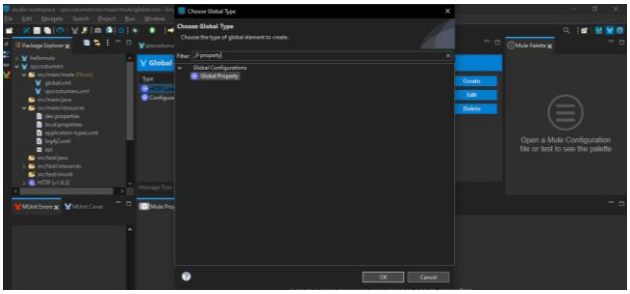
Continuaremos creando un elemento de configuración global, dando click izquierdo en el botón azul llamad CREATE, se nos abrirá una nueva ventana en la cual buscaremos Configuration properties, le daremos click y finalmente en el botón OK.



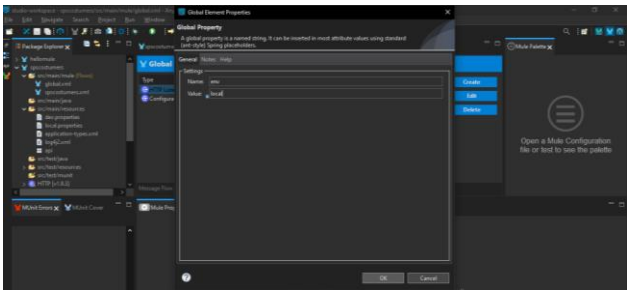
En la siguiente ventana que se nos despliega anotaremos el nombre del archivo de propiedades al que será referenciado, en este caso como son dos archivos se ocupará la siguiente expresión “${env}.properties”.



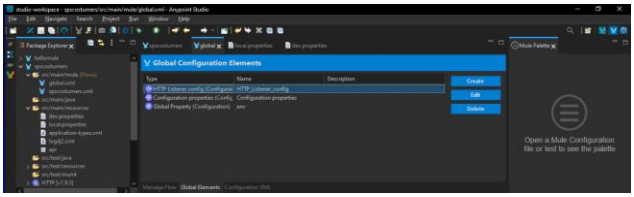
Se continuara con la configuración, agregando un nuevo elemento de configuración global llamado Global property.



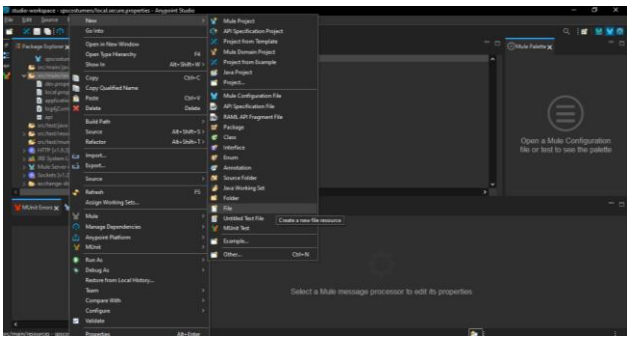
En este caso solo de configura el nombre de la propiedad la cual será “env” con el valor “local” y se dará click en OK.



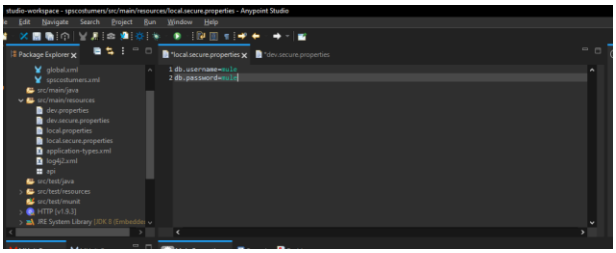
Finalmente en la ventana canvas debemos tener tres configuraciones globales, tal como se muestra en la imagen siguiente



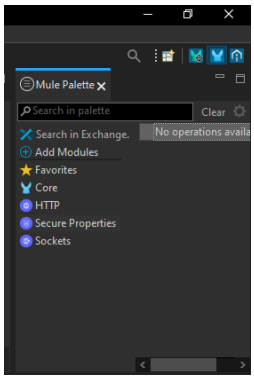
Como sabemos que hay configuraciones u información la cual no debe ser mostrada directamente en el código por seguridad, se crearan dos archivos nuevos de propiedades protegidas, esto se realiza de la misma forma los archivos de propiedades anteriores, con la diferencia de que sus nombres serán los siguientes “dev.secure.properties” y “local.secure.properties”.



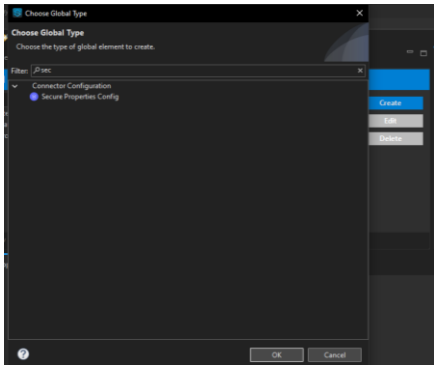
En nuestro caso, una vez creados los archivos, guardaremos en ellos el usuario y contraseña de nuestra base de datos.



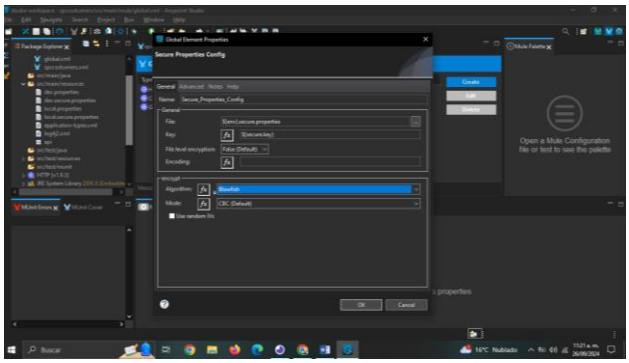
Para proteger nuestros datos es necesario agregar el módulo “Secure Properties” en la paleta de módulos, esto se realiza dando click en el botón de add modules y buscando por nombre el modulo deseado, finalmente se da click en add y finish, de esta manera el modulo agregado se mostrara en la ventana de la paleta.



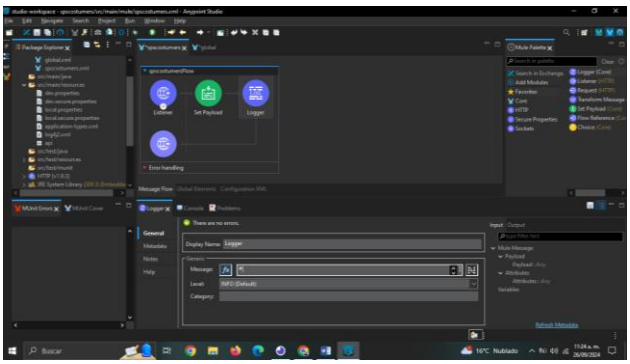
Una vez agregado el modulo iremos al canvas, en el archivo global agregaremos una nueva propiedad llamada “Secure Propeerties Config”.



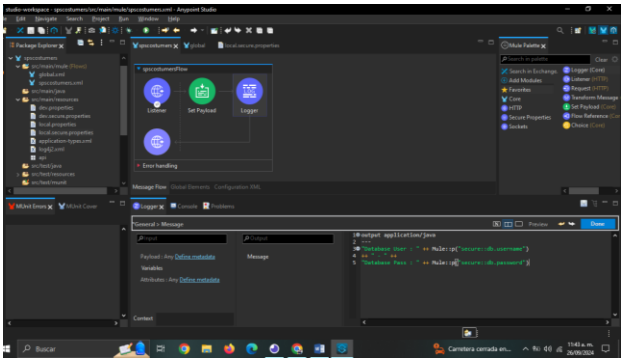
Para configurar eesta propiedad le indicaremos que archivo buscara con la siguiente expresión “${env}.secure.properties”, de la misma manera la llave “${secure.key}” y finalmente el algoritmo de encriptación el cual será “Blowish” finalmente damos click en OK.



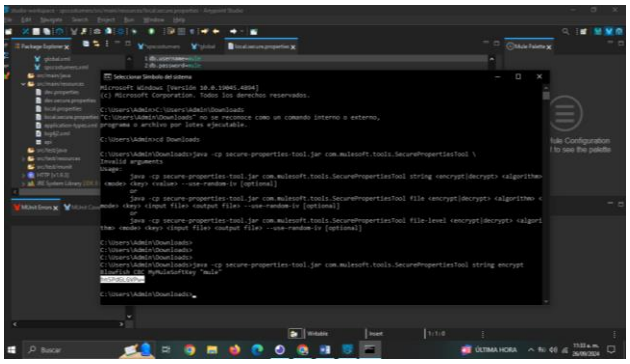
Agregaremos un módulo Logger para verificar la contraseña y el usuario dado por los archivos de propiedades, una vez agregado el modulo le daremos click en el botón que nos da la vista gráfica, el cual se encuentra en la parte inferior en frente de la caja de texto del mensaje.



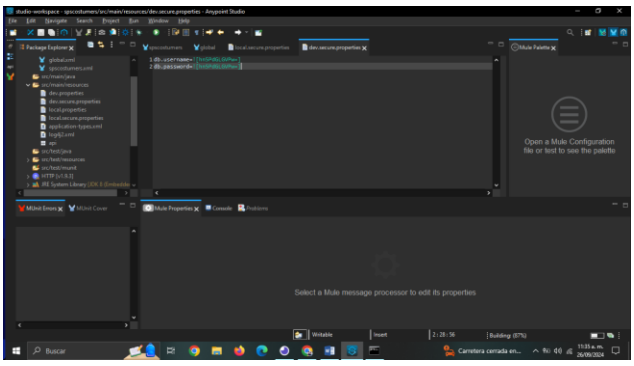
En la ventana que se nos despliega escribiremos el código en datawave que nos mostrara la contraseña y el usuario proporcionados por los archivos de propiedades, tal como se muestra a continuación.



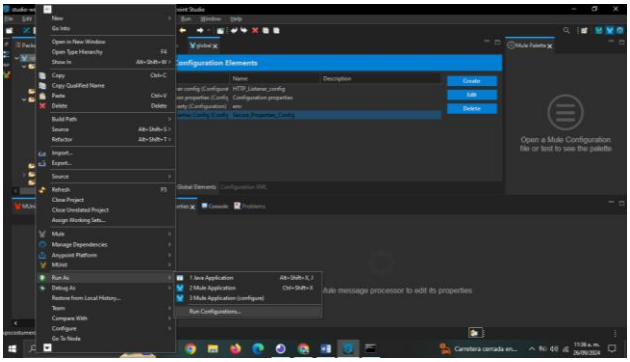
Lo siguiente en hacer es codificar nuetro usuario y contraseña, usando el archivo jar proporcionado en la documentación de mulesoft, esto se logra en el CMD, colocándonos en el directorio donde se encuentra nuestro archivo jar y ejecutando los comandos mostrados a continuación, seguido del string a codificar.



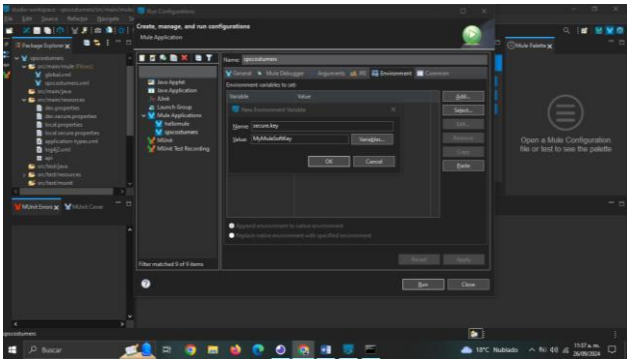
Ya que contamos con la contraseña y el usuario codificado, lo actualizamos en nuestros documentos de propiedades, dado de que los datos están codificados, se usa una expresión nueva para estos datos la cual es la siguiente, ¡[codificado], finalmente guardamos la modificaciones como en a continuación.



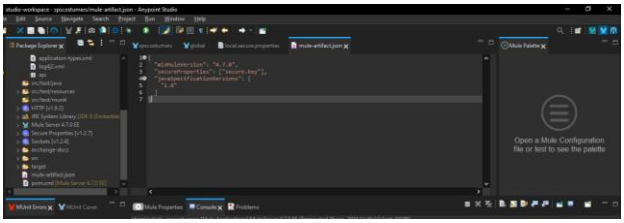
Para continuar con el proceso de codificación, tenemos que relacionar nuestra variable ${secure.key} con un valor, lo cual haremos dando click derecho en nuestro proyecto, luego iremos a “run as” y luego a “run configuration”.



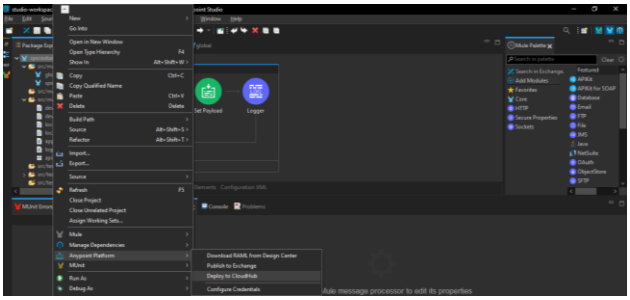
En la ventana que se despliega iremos a la pestaña de environment, ahí agregaremos una nueva variable llamada secure.key con el valor MyMuleSofftKey, damos click en Ok, luego en Apply y listo.



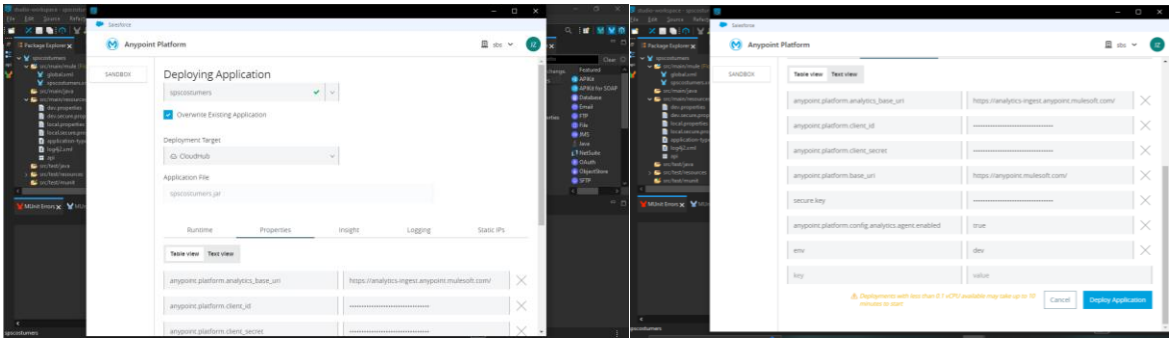
Para hacer que nuestras variables no sean visibles a la hora de hacer deploy en la nube es necesario hacer una modificaciones en el archivo mule-artifact.json, las cuales son agregar la variables que deseamos esconder con la sentencia secureProperties, tal como se muestra en la imagen siguiente.



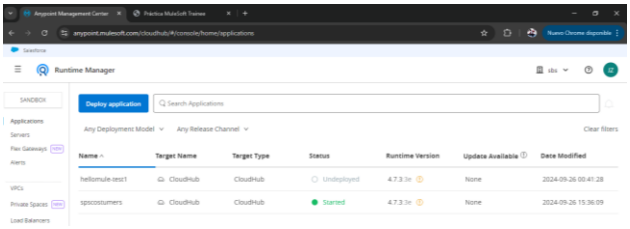
Para hacer deploy en la nube, lo único que realizamos es dar click izquierdo en nuestro proyecto, luego anypoint platform y deploy to cloudhub.



A continuación se nos abrirá una ventana en la cual le daremos nombre a nuestro proyecto y dejaremos las configuraciones por default, también podemos observar que nuestras variables están escondidas en la pestaña propiedades, finalmente damos click en Deploy Application.

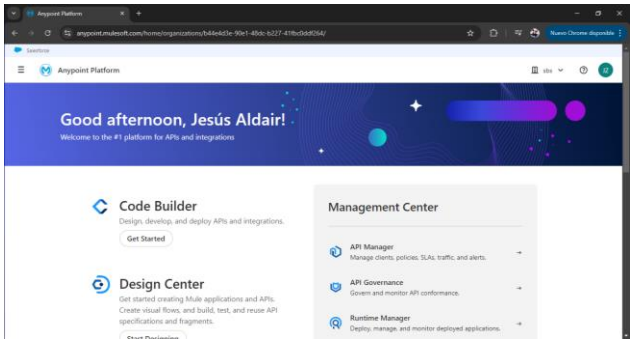


Sabremos que nuestra aplicación funciono por la plataforma de anypoint en la sección de Runtime Manager, veremos el nombre de nuestra aplicación con un foco verde y la palabra Started.

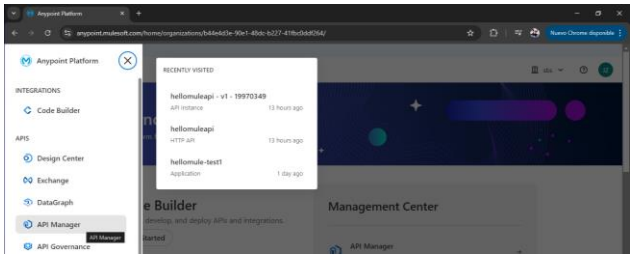


Conectar nuestra aplicación con API Discobery.

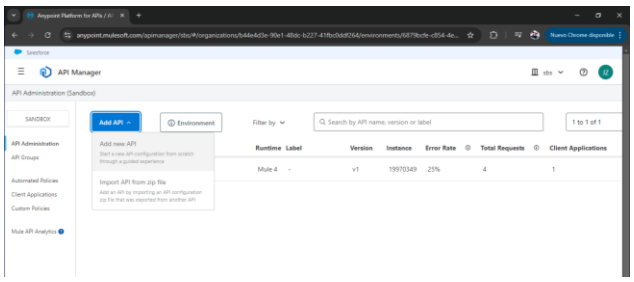
Para empezar entraremos a la plataforma de anypoint.



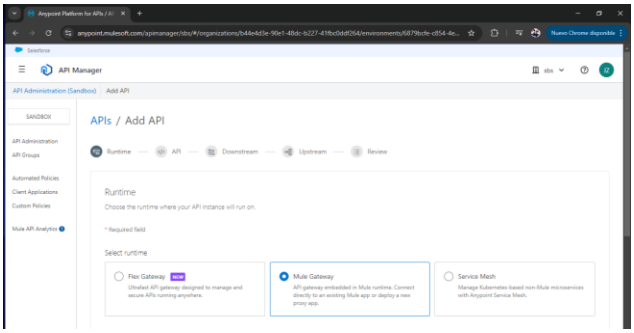
En el menú desplegable del lado derecho iremos al apartado de API Manager.



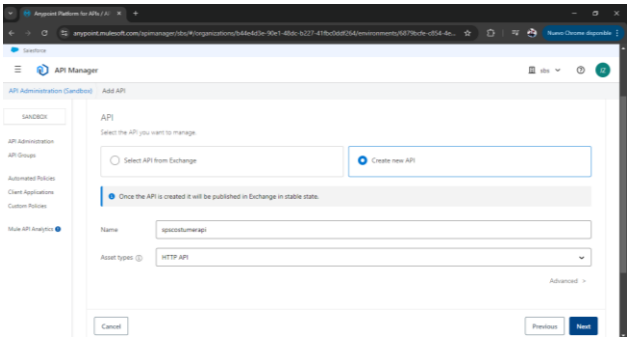
Crearemos una nueva API dando clic en el botón add API Add new API.



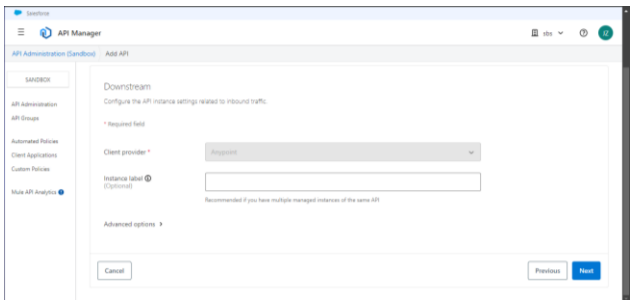
En la siguiente ventana seleccionamos Mule Gateway como runtime y dejamos los valores por defecto, damos click en el botón Next.

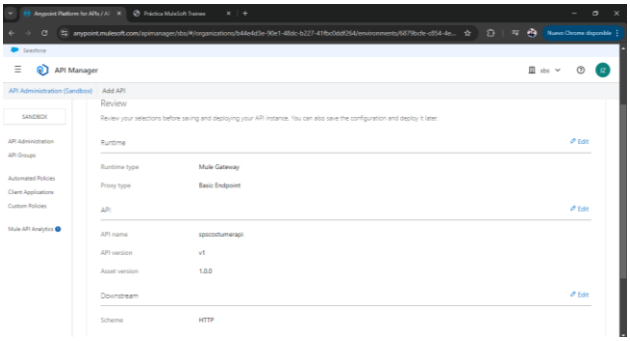


En la siguiente ventana seleccionamos Create new API, le damos nombre a nuestra api en este caso es spscostumerapi, seleccionamos el tipo de API, para este caso es una HTTP API y damos click en Next.

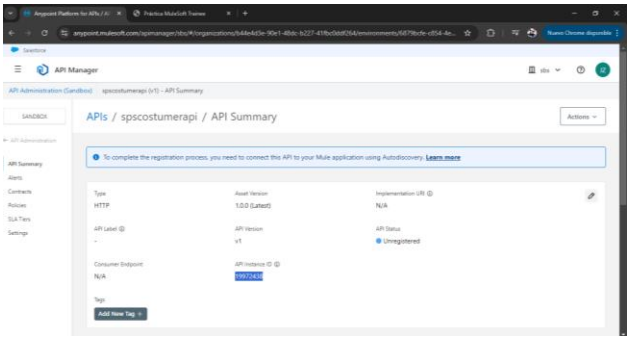


En las siguientes ventanas dejamos los valores predeterminados y continuamos dando click en next.

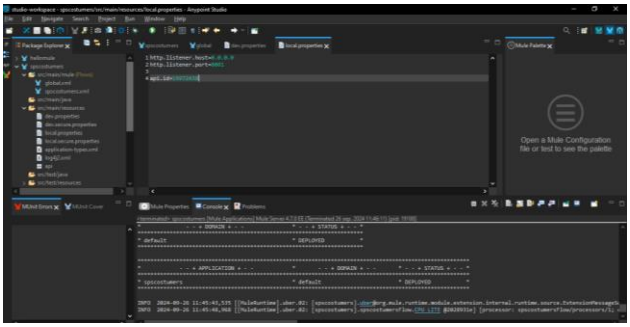




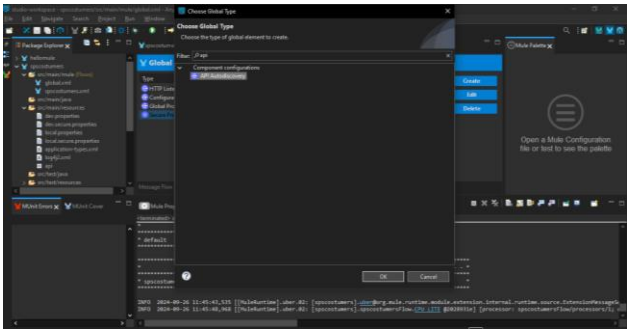
Para finalizar damos click en Save y guardamos el ID de la api quecreamos para relacionarla con nuestra aplicación.



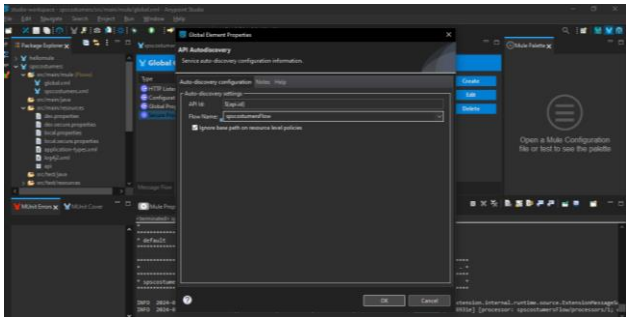
Ya que contamos con el id de nuestra API, lo guardamos en nuestros archivos de propiedades con el nombre api.id, tal como se muestra a continuación.



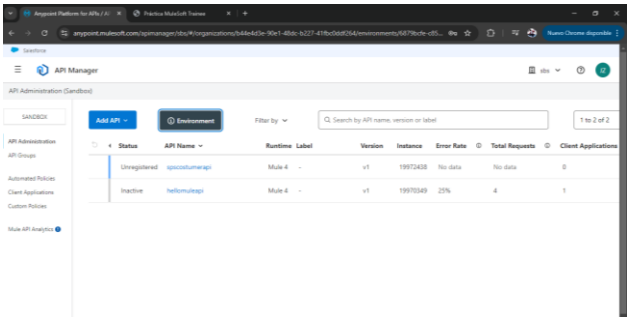
Continuamos agregando una propiedad global llamada API Autodiscobery en nuestro archivo globlal.



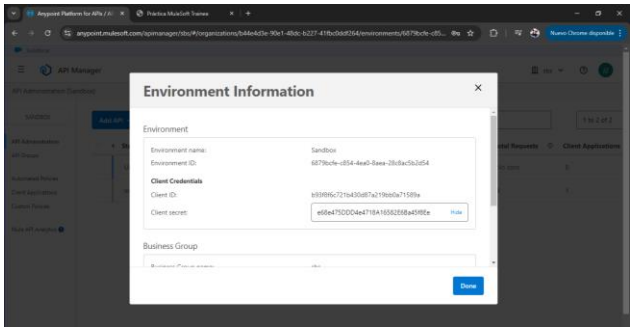
Para la configuración de esta propiedad, en el API ID colocamos ${api.id} y en el flow name seleccionamos nuestro único Flow, el cual es spscostumerFlow.



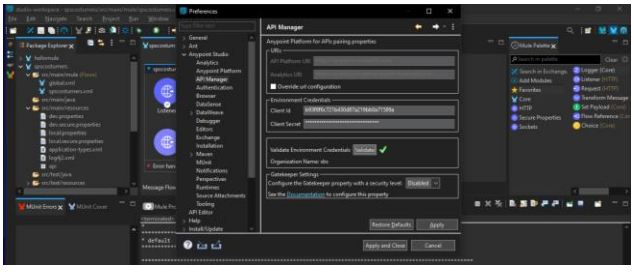
Finalmente veremos en el apartado de API Manager, nuestra api creada.



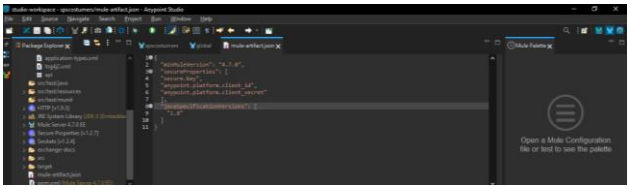
Para conectar nuestra aplicación con nuestra API también es necesario copiar nuestras credenciales de cliente, las cuales se encuentran dando click en el botón de environment, en el apartado de API manager.



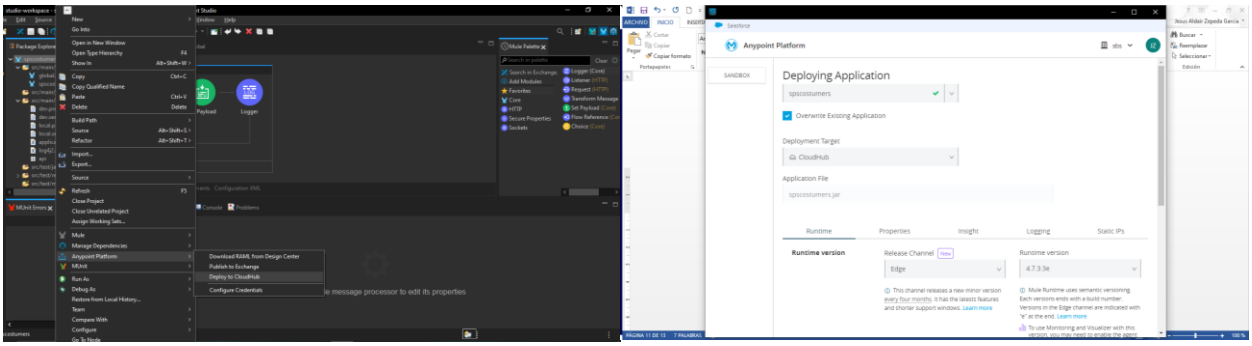
Las credenciales anteriores las pegamos en el apartado de Preferences, API Manager, en el IDE de Anypoint Studio, damos click en validar y si está bien conectada nos mostrará una paloma de color verde.

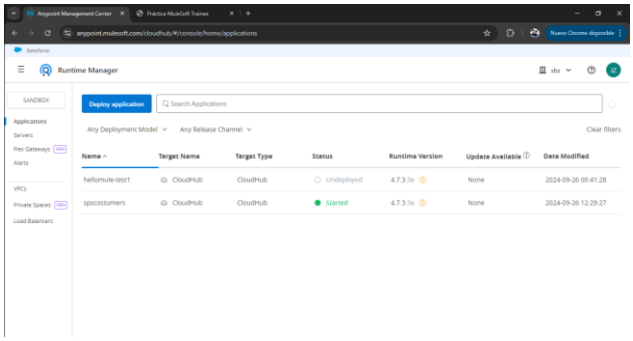


Continuando con la buenas practicas agregamos las credenciales que agregamos al archivo mule-artiact.json para esconderlas y que no se muestren en la nube.

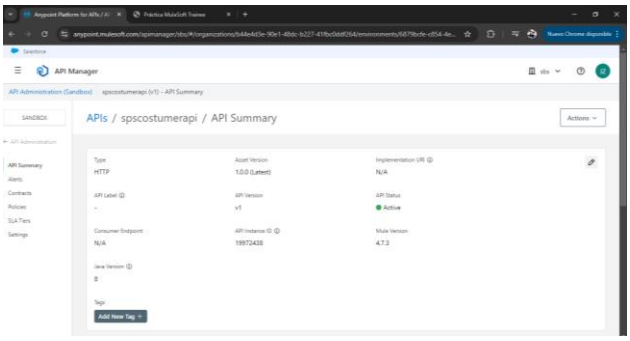


Finalmente hacemos deploy de nuestra aplicación en la nube de la misma manera que se mostró anteriormente.



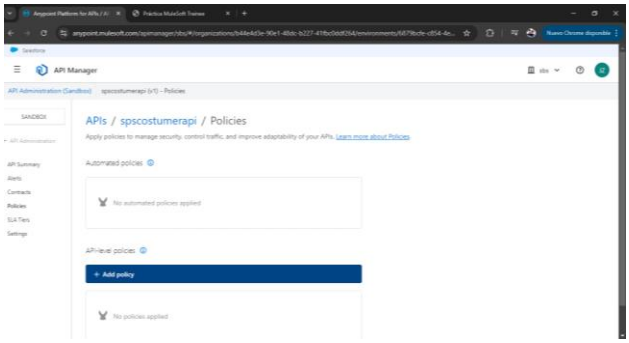


De la misma manera se mostrará en el apartado de API Manager si nuestra api está funcionando con un foco verde y la leyenda active.

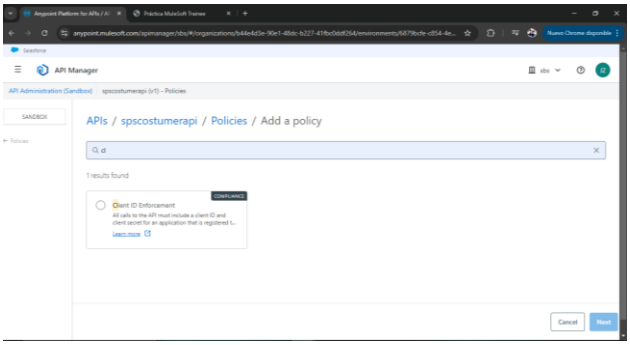


Agregar política Client id Enforcement

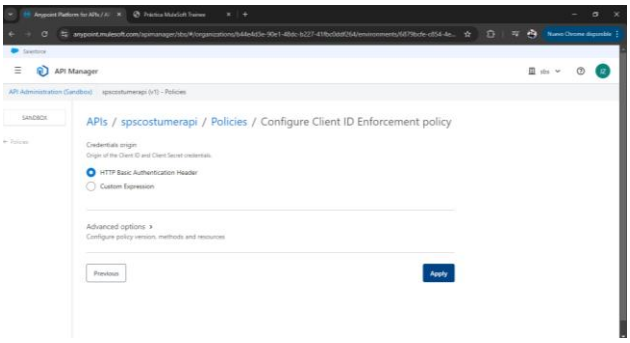
Para agregar políticas solo hay que entrar a nuestra api desde el API manager y dar click en la pestaña policies, a continuación daremos click en add policy.



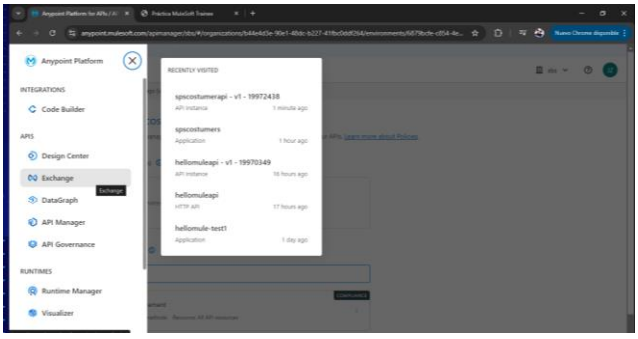
Buscaremos Client ID Enforcement, la seleccionamos y damos click en next.



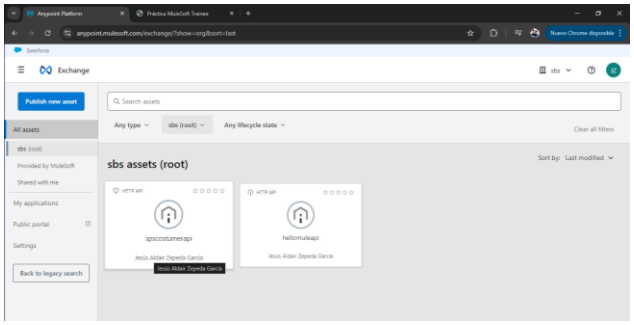
Seleccionaremos un HTTP Basic Authentication Header y daremos click en apply.



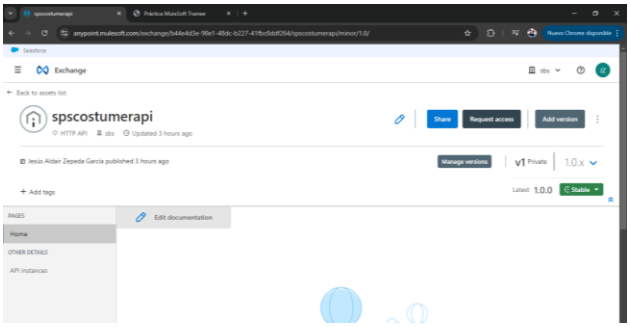
Nos dirigiremos al apartado Exchange de la platafoema de anypoint.



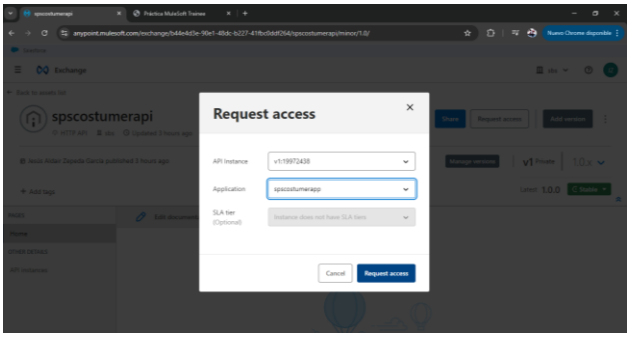
En esta ventana encontraremos nuestra API, la seleccionaremos y nos dirigirá a otra ventana.



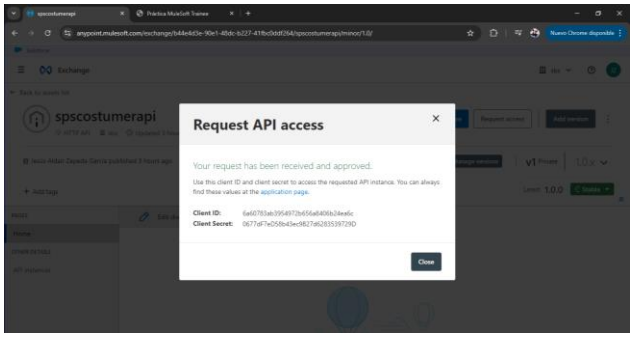
En esta ventana seleccionaremos el botón Request Acces.



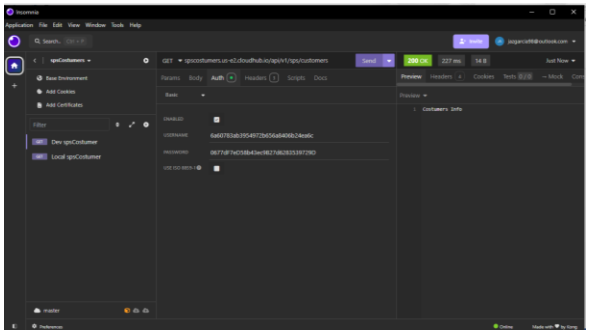
Realizaremos la siguiente configuración, tal como se muestra en la imagen siguiente daremos click en request access.



Finalmente nos generará nuestro id de cliente y nuestra contraseña las cuales copiaremos para hacer las pruebas.



Para hacer las prueba usaremos insomnia como probador de APIs, para la prueba en la nube ocuparemos la url que nos proporciona el Runtime Manager de nuestra aplicación, en conjunto con la dirección que escogimos para nuestro listener tal como se muestra en la imagen siguiente, finalmente enviaremos nuestro user y password generados anteriormente, en el header de nuestra Request GET y la enviaremos, teniendo como retorno nuestro mensaje Customer info.



Para las pruebas locales se realiza lo mismo que en la prueba anterior con la única diferencia que la url es la siguiente “localhost:8081/api/v1/sps/customers”, y el resultado es el mismo, tal como se muestra a continuación.