

API Legacy: Integración Microfrontend

El API Legacy se debe utilizar para implementar una arquitectura de microfrontend, donde cada aplicación frontend se integra con el API para autorizar el acceso a las funcionalidades que se migran. Esta estrategia facilita el desarrollo y mantenimiento de las aplicaciones frontend, ya que cada una puede ser desarrollada y actualizada de forma independiente.

El flujo para autorizar los accesos desde aplicaciones legacy es:

Paso 1

Las aplicaciones legacy consumirán la url del microfrontend y enviarán un parámetro llamado `legacyToken` de tipo string.

Ejemplo consumo:

```
http://10.1.140.21:8092/?legacyToken=674157584668526572786e5371774d4552512b3741513d3d@1aba4e1f294858f0e063658c010a6fec
```

Paso 2

El parámetro `legacyToken` es un string con el siguiente formato **contextClient@uuid** el cual al ser separados por el simbolo @ se obtendran 2 fragmentos.

- **contextClient:** Código del cliente.
- **uuid:** token de autenticación.
- **separador:** Caracter de separación de los fragmentos del token legacy. Por defecto se define @

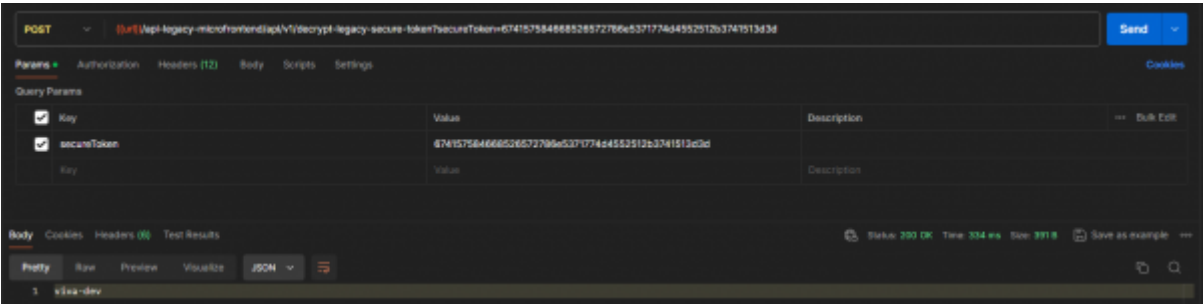
Ejemplo:

```
legacyToken=674157584668526572786e5371774d4552512b3741513d3d@1aba4e1f294858f0e063658c010a6fec
```

- **contextClient:** 674157584668526572786e5371774d4552512b3741513d3d
- **uuid:** 1aba4e1f294858f0e063658c010a6fec
- **separador:** @

Paso 3

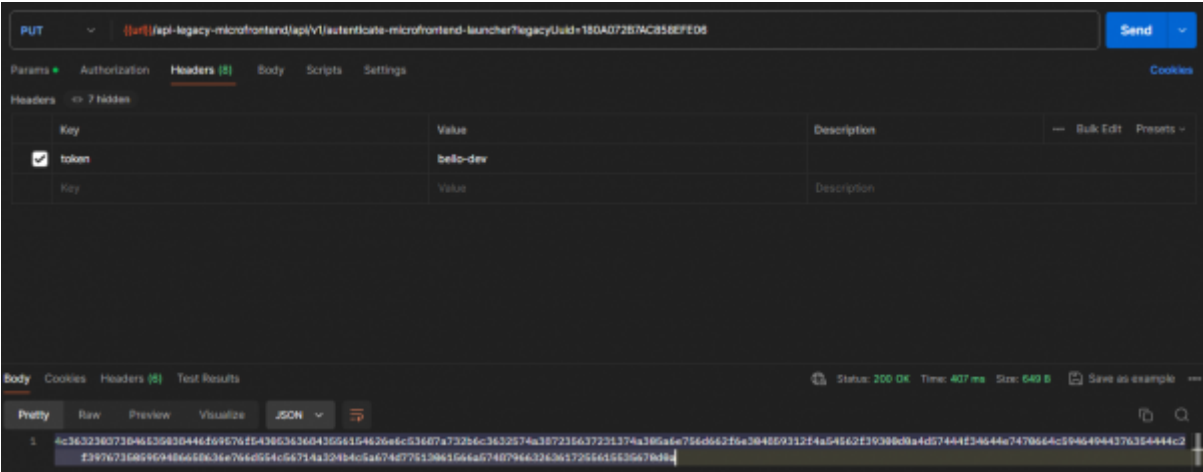
Se debe tomar el `contextClient` y descriptarlo para identificar el cliente. Ejemplo:



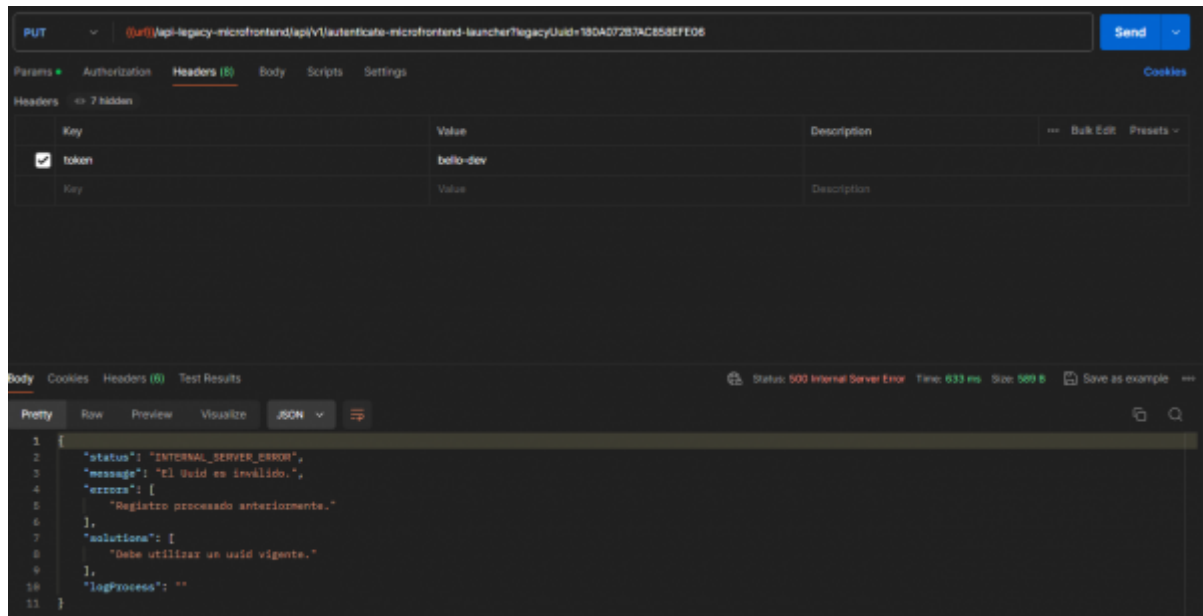
Paso 3

Se debe consumir el método del api **authenticate-microfrontend-launcher** el cual puede ser consultado en la documentación de la [Capa de lógica de negocio](#) y pasar el contextClient como token en el header y el uuid como parametro.

Si el uuid está vigente se devolverán los parametros de sesión encriptados con el código 200. Lo que permitirá el acceso a la funcionalidad.



Si el uuid está vencido o es inválido se devolverá una excepcion con el código 500. Lo que evitará el acceso al microfrontend.



Paso 4

Al realizar la autenticación de forma correcta se obtendrá un hash que contendrá los parametros de sesión pero estarán encriptados por lo tanto se debe consumir el servicio **decrypt-legacy-secure-token** el cual recibirá 2 argumentos uno en el header llamado token el cual recibirá el contextClient y un parámetro secureToken donde se le enviará el hash obtenido del proceso de autenticación como se ve en la imagen.



El servicio devuelve un objeto json con la siguiente estructura.

```

{
  "idOption": 1, //Identificador del Path del microfrontend
  "tituloVentana": "Maestro Terceros", //Titulo de la opción en la aplicación
  legacy
  "nitEmpresa": 800167494, //Nit de la empresa del cliente
  "nombreEmpresa": "Uniemprsa de prueba", //Nombre de la empresa del cliente
  "codigoMempresa": 9999999999, //Código de la empresa del cliente
  "codigoUsuario": 1, //Código del usuario en la aplicación legacy
  "nombreUsuario": "Pepito Perez", //Nombre del usuario en la aplicación
  legacy
  "login": "123456789", //login del usuario en la aplicación legacy
  "fechaSistemaModulo": "06/06/24", //Fecha del sistema en la aplicación
  legacy
  "codigoDependencia": 66, // (Opcional) Código de la dependencia, solo aplica
  para la aplicación de presupuesto
}

```

Consideraciones

- El uuid es de un solo uso, de esta manera aseguramos la integridad de las integraciones y consumos de los nuevos componentes.
- Los parámetros descriptados son la base para la inicialización del microfrontend.
- Salvo los parámetros opcionales los demás son requeridos por lo tanto se aconseja aplicar validaciones de esas propiedades al descriptarlas.
- Todo consumo microfrontend es almacenado para efectos de auditoria.

[←Regresar](#)

From:
<http://wiki.adacsc.co/> - Wiki

Permanent link:
<http://wiki.adacsc.co/doku.php?id=ada:howto:sicoferp:factory:new-migracion-sicoferp:apilegacy:microfrontend&rev=1718291255>

Last update: 2024/06/13 15:07

