GUÍA TÉCNICA DE INTEGRACIÓN

CONFIGURACIÓN APIs PAGO CON DAVIPLATA

EQUIPO OPEN Y CONEXIÓN CON TERCEROS

DAVIVIENDA

ÍNDICE

Uso del DEVELOPER PORTAL de Daviplata

Aclaraciones:

- Las configuraciones descritas corresponden a un ambiente de laboratorio.
- Para poder realizar las configuraciones es necesario estar registrado en el DEVELOPER PORTAL y haber crear la aplicación
- Es necesario descargar Postman y cargar el certificado dispuesto en el API Market (este proceso se describe más adelante en este documento).
- Se comparte anexo con los errores conocidos o posibles errores que pueden presentarse durante este proceso y los códigos asociados a los tipos de documento de identificación para las pruebas de consumo.
- La API Read OTP solo se utiliza en el ambiente de laboratorio.
- Una vez realizadas estas configuraciones se debe notificar al Banco para que se genere la data de pruebas, esta solicitud se debe realizar a través de la Coordinación de Integración al correo electrónico edgar.forero@davivienda.com
- El horario establecido para la ejecución de pruebas y disponibilidad del ambiente de Laboratorio es de lunes a viernes de 8:00 a 17:00.

DATA PARA PRIMEROS CONSUMOS

Tipo de documento	Identificación en el API	Número de documento
CC	01	1134568019
CE	02	786630
TI	04	1389123506

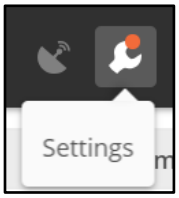
Nota: Se recomienda utilizar valores pequeños para las compras de pruebas mientras se genera la data exclusiva.

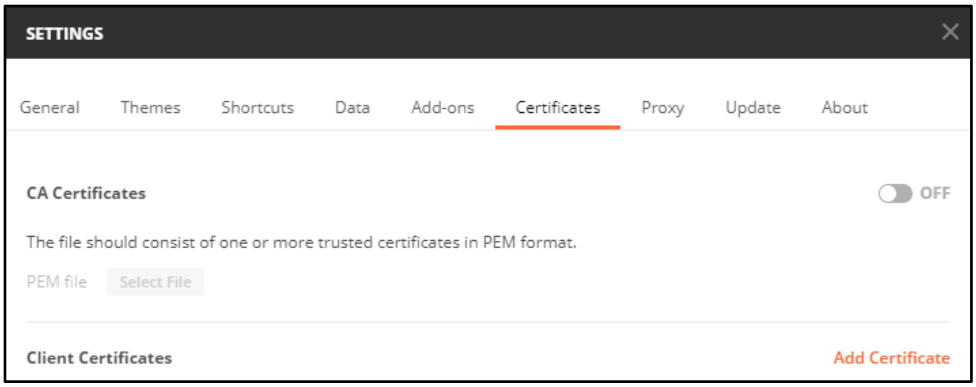
Para poder probar el funcionamiento de nuestras APIs, usaremos herramientas de llamadas a una API. En el caso de esta guía, te mostramos cómo probar una API usando Postman.
Se puede descargar de su página web: https://www.getpostman.com/

Desarrollo en POSTMAN

1) Ingresamos a POSTMAN.

2) Para configurar el certificado ingresamos al icono de llave "Settings", en la pestaña "Certificates" damos clic en la opción "Add Certificate".



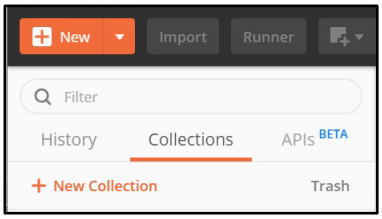


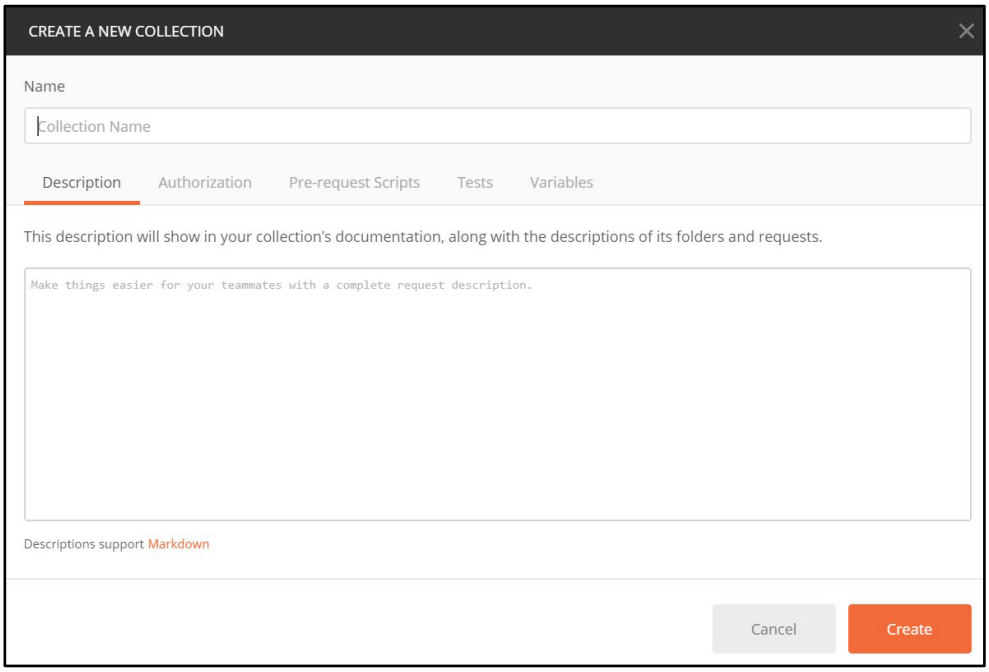
3) Agregamos el Host que es "apislab.daviplata.com"

4) En el espacio donde dice CRT file vamos a seleccionar el archivo con la extensión ".cer".
5) En el espacio donde dice KEY file vamos a seleccionar el archivo con la extensión ".pem".
6) En el espacio donde dice PFX file no debemos cargar nada.
7) Los campos PFX file y Passphrase deben ir en blanco.

Creación de la Colección

8) En el menú vertical del lado izquierdo, dentro de la pestaña "Collections" damos clic al botón "+ New Collection", en la ventana que nos abre le damos un nombre en el campo "Name" y le damos clic al botón "Create".





Agregar API OAUTH SERVER2SERVER

9) Abrimos la colección creada y damos clic a la opción el botón de "...", después de esto damos clic a la opción de "Add Request".
10) En esta ventana asignamos datos en el campo de "Name Request", se sugiere colocar un índice con el número del orden de llamado de la API y el nombre de la API. (1_Server2Server).
11) Dentro de esta documentación tendremos en cuenta la siguiente información a cargar dentro de POSTMAN:
    - El método lo encontramos dentro de la documentación el cual es **POST**.
    - La URL (ENDPOINT) a cargar es: https://apislab.daviplata.com/oauth2Provider/type1/v1/token
    - Los Headers los cuales están cargados en el mismo espacio de la URL en este es necesario cargar el campo de KEY y el campo de VALUE dentro de POSTMAN.
      Cabeceras:

	KEY	VALUE
☑	accept	application/json
☑	content-type	application/x-www-form-urlencoded

- En la sección de Body, elegimos la opción Raw e introducimos el siguiente código:

  grant_type=client_credentials&client_id=REPLACE&client_secret=REPLACE&scope=daviplata

  En este caso es necesario realizar el cambio en esta línea del Client ID(Es el mismo consumer Key encontrado en su aplicación) y del Client Secret(Es el mismo Consumer Secret encontrado en su aplicación) a partir de la información creada por la aplicación, más abajo encontramos la información del SCOPE el cual aparece relacionado y también debe ser cargado dentro de esta línea, en el ejemplo el de "daviplata".

12) Al tener cargada esta información podemos realizar el consumo de la API, para este proceso debemos dar clic en el POSTMAN en el botón "Send".
Esta llamada nos devuelve un response como este:

{
"token_type": "Bearer",
"access_token": "AAIkMmI1Y2U1ZDYtZWI5Zi00NjA5LTkwZjktZDIyMTcxNjk0ZWZi-DFnhfohYrWUnkL7RqZGIoW8YPnk7pZdcokZmcxyKisw6b7n07Ga94Gw464AnVJd3rTE4ztJJLaUJP2iVCG_8p9hIwy2RdKjxUzybfm2X_YxDv0sJLOUiliqjdJc7USF",
"expires_in": 900,
"scope": "daviplata"
}

La API se encargará de dar respuesta del consumo, esto aparecerá en el campo de abajo, en el caso en el que este consumo sea efectivo aparecerá en su Status "200 OK".

El ***access_token*** que nos devuelve el servicio lo usaremos para la invocación a la siguiente API.

Agregar API Compra Daviplata

13) Abrimos la colección creada y damos clic a la opción el botón de "...", después de esto damos clic a la opción de "Add Request".
14) En esta ventana asignamos datos en el campo de "Name Request", se sugiere colocar un índice con el número del orden de llamado de la API y el nombre de la API. (2_CompraDaviplata).
15) Dentro de esta documentación tendremos en cuenta la siguiente información a cargar dentro de POSTMAN:
    - El método lo encontramos dentro de la documentación el cual es **POST**.
    - La URL (ENDPOINT) a cargar es:
      https://apislab.daviplata.com/daviplata/v1/compra
    - Los Headers se deben cargar el campo de KEY y el campo de VALUE dentro de POSTMAN. Dentro de los headers a cargar se hace necesario cargar la información del access token generado en la API anterior OAUTH SERVER2SERVER y el Consumer Key en el header x-ibm-client-id.
      Cabeceras:

	KEY	VALUE
☑	accept	application/json
☑	authorization	Bearer REPLACE
☑	content-type	application/json
☑	x-ibm-client-id	REPLACE

    - En la sección de Body, elegimos la opción Raw e introducimos el siguiente código:

```
{
        "valor":"50","numeroIdentificacion":"REPLACE" (se debe validar que en este campo no queden espacios en blanco y que el número no supere 14 digitos.)

        ,"tipoDocumento":"REPLACE"
}
```

Al dar clic sobre el botón "SEND" nos generará el siguiente Response con el idSessiónToken

```
{
  "idSessionToken": "24515022",
  "fechaExpiracionToken": "2020-05-14T16:30:05.137-05:00"
}
```

Agregar API READ OTP (Únicamente para laboratorio)

16) Abrimos la colección creada y damos clic a la opción el botón de "...", después de esto damos clic a la opción de "Add Request".
17) En esta ventana asignamos datos en el campo de "Name Request", se sugiere colocar un índice con el número del orden de llamado de la API y el nombre de la API. (3_ReadOTP).
18) Dentro de esta documentación tendremos en cuenta la siguiente información a cargar dentro de POSTMAN:
    - El método lo encontramos dentro de la documentación el cual es **POST**.
    - La URL (ENDPOINT) a cargar es:
      https://apislab.daviplata.com/otpSec/v1/read
    - Los Headers los cuales están cargados en el mismo espacio de la URL en este es necesario cargar el campo de KEY y el campo de VALUE dentro de POSTMAN. Dentro de los headers a cargar se hace necesario cargar la información del Consumer Key en el header x-ibm-client-id.
      Cabeceras:

	KEY	VALUE
☑	accept	application/json
☑	content-type	application/json
☑	x-ibm-client-id	REPLACE

    - En la sección de Body, elegimos la opción Raw e introducimos el siguiente código:

```
{
        "typeDocument":"REPLACE"
        ,"numberDocument":"REPLACE",
        "notificationType":"API_DAVIPLATA"
}
```

En este caso es necesario utilizar el numberDocument del cliente sobre el cual estemos trabajando.

Al dar clic sobre el botón "SEND" nos generará el OTP.

```
{
  "otp": "841376"
}
```

Agregar API Confirmardaviplata

19) Abrimos la colección creada y damos clic a la opción el botón de "...", después de esto damos clic a la opción de "Add Request".
20) En esta ventana asignamos datos en el campo de "Name Request", se sugiere colocar un índice con el número del orden de llamado de la API y el nombre de la API. (4_ConfirmarDaviplata).
21) Dentro de esta documentación tendremos en cuenta la siguiente información a cargar dentro de POSTMAN:
    - El método lo encontramos dentro de la documentación el cual es **POST**.
    - La URL (ENDPOINT) a cargar es:
      https://apislab.daviplata.com/daviplata/v1/confirmarCompra
    - Los Headers se deben cargar el campo de KEY y el campo de VALUE dentro de POSTMAN. Dentro de los headers a cargar se hace necesario cargar la información del access token generado en la API anterior OAUTH SERVER2SERVER después del Bearer y el Consumer Key en el header x-ibm-client-id.
      Cabeceras:

	KEY	VALUE
☑	accept	application/json
☑	authorization	Bearer REPLACE
☑	content-type	application/json
☑	x-ibm-client-id	REPLACE

    - En la sección de Body, elegimos la opción Raw e introducimos el siguiente código, en este caso el OTP se genera desde el API ReadOTP, el idSessionToken desde el API Compradaviplata:

```
{
        "otp":"841376"
        ,"idSessionToken":"24515022"
        ,"idComercio":"0010203040"
        ,"idTerminal":"ESB10934"
        ,"idTransaccion":152648
}
```

        - otp: One-Time Password enviado al celular del cliente (en el caso de laboratorio el generado por el API ReadOTP).
        - idSessionToken: Este valor se genera en el API CompraDaviplata.
        - idComercio: Este es el código único del comercio asignado por **Redeban** (Debe ser confirmado con Línea de negocio/ejecutivo)

- idTerminal: Esta es la terminal asignada por **Redeban** para las efectuar las transacciones (Debe ser confirmado con Línea de negocio/ejecutivo).
- idTransaccion: Es un identificador para la transacción por día por terminal, es decir, durante el día se pueden realizar 999999 transacciones por cada terminal asignada. Este debe contener un máximo de 6 caracteres y al ser un campo de tipo numérico, en el mismo no se pueden anteponer ceros debido a que el JSON no lo reconocería como un campo numérico.

Nota: Los valores de idComercio y idTerminal no cambian o son estándar para el ambiente de laboratorio. En el ambiente productivo se deben relacionar los propios **id** del comercio que está realizando el consumo de APIs.

Los valores de idTransacción no se pueden repetir para un mismo día por terminal.

22) Al dar clic sobre el botón "SEND" nos generará la confirmación exitosa de la compra Daviplata. Ejemplo de respuesta:


```
{
    "fechaTransaccion": "2019-04-07T14:15:45",
    "estado": "Aprobado",
    "numAprobacion": "551631",
    "idTransaccionAutorizador": "000000152648"
}
```

ANEXOS

Tipos de documentos a utilizar

NATURALEZA	TIPO DE IDENTIFICACIÓN	DESCRIPCIÓN
Natural	"01"	Cedula de Ciudadania
Natural	"02"	Cédula de Extranjería
Natural	"04"	Tarjeta de identidad

Errores conocidos

API	Mensaje	Código	Causa
CompraDaviplata	codigoError: 500 mensajeError: valor transacción vacío	codigoError: 500	El campo valor se encuentra vacío
CompraDaviplata	codigoError: 500 mensajeError: número de identificación vacío	codigoError: 500	El campo número Identificación se encuentra vacío
CompraDaviplata	codigoError: 500 mensajeError: tipo de documento vacío	codigoError: 500	El campo tipoDocumento se encuentra vacío
CompraDaviplata	codigoError: 500 mensajeError: Saldo insuficiente	codigoError: 500	El saldo del daviplata es menor al solicitado para la compra

CompraDaviplata	{
	"codigoError": "1304",
	"mensajeError": "DAVIPLATA INVÁLIDO"
	}
		"codigoError": "1304",	Se cargo un documento de identidad que no tiene asociado ningún Daviplata

CompraDaviplata	{
	"codigoError": "321",
	"mensajeError": "PRODUCTO NO EXISTE"
	}
		"codigoError": "321"	Se cargo un documento de identidad que no tiene asociado ningún Daviplata

Read OTP	{
	"errorCode": "0",
	"errorMessage": ""
	}
		"errorCode": "0"	El usuario no ha utilizado el mismo número o tipo de documento en el API de Compradaviplata o de Read OTP.

ConfirmarDaviplata	codigoError: 500 mensajeError: otp vacío	codigoError: 500	El campo otp se encuentra vacío
ConfirmarDaviplata	codigoError: 500 mensajeError: idSessionToken vacío	codigoError: 500	El campo idSessionToken se encuentra vacío
ConfirmarDaviplata	codigoError: 401 mensajeError: Autenticacion Denegada	codigoError: 401	El OTP, el Token o la URL no son válidas.

Generico	Internal Server Error	2100 - Internal Server Error	Cuando un servicio no está disponible "Aplicación no disponible"
Generico	Internal Server Error	3200 - Internal Server Error	Cuando un servicio en el bus retorna "Ha ocurrido un error no esperado en el ESB"
Generico	Internal Server Error	2103 - Internal Server Error	Cuando un servicio dio timeout
Generico	Internal Server Error	6842 - Internal Server Error	Cuando se envía mal la estructura SOAP al servicio
Generico	Unauthorized	401 - Invalid client id or secret.	Cuando no se envia ClientId o ClientSecret
Generico	Unauthorized	401- "This server could not verify that you are authorized to access the URL"	Cuando se envía un Access Token no válido
Generico	Unauthorized	401 - "Application is not registered with mutual tls"	Cuando la Aplicación del API Market no tiene un certificado cargado
Generico	Unauthorized	401 - "Invalid client certificate is provided"	Cuando se envía un certificado distinto al cargado
Generico	Invalid	422- "Validate REST: xa35://tmp/temp_02653:1: [JSV0001] Invalid value type 'integer'."	Cuando se envía un campo que es String pero se envía Numérico
Generico	Invalid	422- "Validate REST: xa35://tmp/temp_02653:1: [JSV0001] Invalid value type 'string'."	Cuando se envía un campo adicional en el body que no pertenece de acuerdo a la definición de entrada
Generico	Invalid	422 - "Validate REST: xa35://tmp/temp_02777:1: [JSV0002] Invalid object: the property 'tipoDocumento' is missing."	Cuando no se envia un campo obligatorio
Generico	""	(en blanco)	Llega mensaje de un servicio (backend) con estructura no esperada
Generico	Forbidden	403 - "Internal Server Error"	Cuando se envía un Access Token que no tiene el scope asociado a las APIs

Generico	Autenticación OAUTH	400 - "error": "invalid scope"	Cuando no se envía ó el valor del scope (tipo de alcance) que se está enviando es incorrecto.
Generico	Autenticación OAUTH	400 - "error": "invalid client"	Cuando no se envía o el valor del Client Secret que se está enviando es incorrecto.
Generico	Autenticación OAUTH	400 - "error": "unknown"	Cuando no se envía o el valor del grant_type (tipo de acceso) que se está enviando es incorrecto.