INSTRUCTIVO DE INTEGRACIÓN

DE COMERCIOS A PASARELA

VERSION 1.6.11

Julio 2026

CONTROL DE CAMBIOS		
VERSION	**FECHA**	**RAZÓN DE LA ACTUALIZACIÓN**
1.0	2024/01/05	Integración con página de pagos de pasarela a través de redirección.
1.1	2024/01/30	Adición de método de pago Nequi.
1.2	2024/02/26	Adición de endpoints de consulta de estado de transacciones.
1.2.1	2024/03/04	Adición de ejemplo de firma en Java.
1.3	2024/03/18	Adición de endpoints para creación de pago push Nequi.
1.4	2024/05/14	Adición de endpoints para creación de pago PSE por API.
1.5	2024/05/18	Login transaccional y endpoint de reportería.
1.6	2024/06/12	Login para comercios. Endpoint para transacciones con tarjeta.
1.6.1	2024/07/30	Ajuste en campo de fecha de expiración de tarjeta de crédito y adición de ReGex de validación en campos para creación de transacción.
1.6.2	2024/08/06	Ajuste en respuesta por creación de transacción declinada con tarjeta de crédito.
1.6.3	2024/08/21	Endpoint para creación de enlace de pago.
1.6.4	2024/09/11	Endpoint para creación, lectura y confirmación de pines de pago.
1.6.5	2024/12/02	Webhook para notificación de cambios de estado de transacciones.
1.6.6	2025/04/01	Creación de transacciones con método de pago PIN Cambio en validación de documento de cliente. Numérico pasa a alfanumérico
1.6.7	2025/05/20	Mensaje de error de transacción no encontrada.
1.6.8	2025/07/02	Adición de campo Signature o firma en mensaje webhook.
1.6.9	2026/01/23	Adición Validación de autenticidad del callback (signature RSA)
1.6.10	2026/05/27	Adición de validaciones en la creación de enlace de pago. Adición de uso de enlace de pago.
1.6.11	2026/07/24	Adición de campos extraData y subMerchantIdentification para payloads en creación de transacción.

Contenido

Introducción 4

Flujo de la información en Modo Redirect – Creación del pago 4

API transaccional 18

Login 18

Creación de la transacción: 19

Nequi 20

PSE 20

Tarjeta Crédito 22

PIN 25

Consulta de estado de la transacción: 28

Consulta reporte paginado 31

Creación de enlace de pago 34

Uso de enlace de pago 37

Creación de pin de pago mediante integración REDIRECT 39

Confirmación de pago del PIN 39

Consulta de información del PIN de pago 42

Webhooks 43

Webhook de notificación de cambio de estado de transacciones 43

Introducción

Una página de pagos segura de una pasarela de pagos es una página web donde los compradores introducen la información de pago para realizar transacciones en línea de forma segura. Esta página es parte integrante de una pasarela de pagos, sistema que facilita y procesa transacciones financieras en línea entre compradores y comercios.

Características clave de una página de pagos segura:

- Protocolos de seguridad: Utiliza el protocolo de seguridad HTTPS para asegurar que la información transmitida esté encriptada y protegida de posibles interceptaciones.

- Cumplimiento con estándares de seguridad: Cumple con estándares de seguridad PCI-DSS (Payment Card Industry Data Security Standard) para garantizar que los datos de tarjetas de crédito estén manejados de forma segura.

- Interfaz amigable y confiable: Ofrece una interfaz de usuario intuitiva y fácil de usar para que los clientes ingresen sus detalles de pago de manera clara y segura.

Flujo de la información en Modo Redirect – Creación del pago

Cuando un cliente agrega productos al carrito de compras y decide realizar el pago, el proceso de transacción se inicia. El carrito de compras, que actúa como intermediario entre el cliente y la pasarela de pagos, recopila los detalles de la compra, como los productos seleccionados, la cantidad, el costo total y la información del cliente.

Al proceder al pago, el carrito de compras redirige al cliente a la página de pagos segura, la cual es gestionada por la pasarela de pagos. Esta página de pagos segura es un entorno protegido que utiliza protocolos de seguridad como HTTPS y encriptación para salvaguardar los datos sensibles durante la transferencia.

Una vez en la página de pagos segura, el cliente ingresa los datos de su tarjeta de crédito, débito u otra información de pago. La pasarela de pagos se comunica con la institución financiera del cliente para verificar la autenticidad de los datos y la disponibilidad de fondos. Al finalizar la transacción, la pasarela de pagos envía al cliente de regreso a la página de pagos segura como se muestra en el siguiente diagrama.

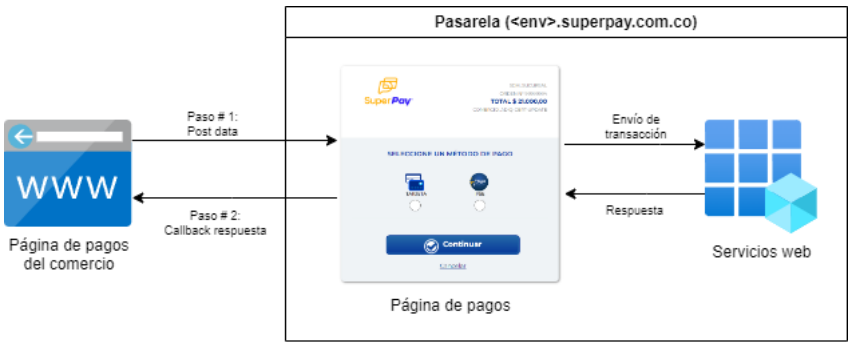


Imagen 1: Flujo de la información entre comercio y pasarela de pagos para invocación de página de pagos

A continuación, se brindan detalles técnicos de los pasos # 1 y #2 del diagrama del flujo de la información:

Paso # 1: Envío de información para el pago

El cliente procede a pagar su compra dando clic al botón de "Pagar" en el carro de compras de la página del comercio. La página del comercio debe hacer una redirección a la página de pagos de la pasarela enviando la información del pago en una request tipo POST y donde el cliente terminará de confirmar el pago de su compra con alguno de los medios de pago puestos a disposición por la pasarela.

- Request POST:

El comercio debe hacer una request a la URL: *https://<prefijo-ambiente>.superpay.com.co/ws/router/payment-page* y los parámetros enviados en la request son los siguientes:

Parámetro (String)	Descripción	Obligatorio para Envío	Incluir en Firma
billingDocument	Número de documento del cliente. Alfanumérico. ReGex: ^[\\dA-Za-z-]+$	Si	Si
billingDocumentType	Tipo de Documento del cliente: - CC: Cédula de ciudadanía - NI: Nit - PASSPORT: Pasaporte - TI: Tarjeta de Identidad - CE: Cédula Extranjería - RCN: Registro Civil de Nacimiento - TI: Tarjeta de Identidad - TE: Tarjeta de Extranjería - Foreign Id: Documento de identificación Extranjero - Foreign commerce Id: Identificación Comercio Extranjero	Si	Si

branchName	Descripción del comercio definida para desplegar en pantalla	Si	Si
cancelCallbackUrl	URL para redirect del callback en caso de transacción cancelada	Si	Si
client	Valor por defecto para ambiente de pruebas: “supergiros”. Para paso a producción este valor cambia.	Si	Si
countryIsoCode	Valor por defecto: Colombia = COL	Si	Si
currency	Id de la moneda de la transacción (COP = 170)	Si	Si
customerAddress	Dirección de cliente que realiza la compra. Se permiten espacios y símbolos: °#,/-	Si	Si
customerEmail	e-mail del cliente que realiza la compra. ReGex: ^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$	Si	Si
customerName	Nombre del cliente que realiza la compra	Si	Si
customerPhone	Número de teléfono del cliente que realiza la compra. ReGex: ^[0-9\\s+]*$	Si	Si
enableNiubizSetting	Valor por defecto = false	Si	Si
errorCallbackUrl	URL para redirect del callback en caso de transacción errónea	Si	Si
externalId	Identificación única generada por el comercio para identificar la transacción. El formato para la generación del UID es el siguiente: • Últimos ocho caracteres del ID del terminal (relleno con 0 a la izquierda si el ID del terminal tiene menos de 8 caracteres separador de guiones / guiones • Valor del contador (relleno con 0 a la izquierda si el valor actual es menor que 99999999999) ***Ejemplo 1:*** ID de terminal: TZXCALJTUXVS Valor contador comercio: 14 UID generado: ALJTUXVS-00000000014 ***Ejemplo 2:*** ID de terminal: 58963 Valor contador comercio: 25637 UID generado: 00058963-00000025637	Si	Si
extraTaxAmount	Importe de IAC de la transacción. Incluir este monto en el importe total.	Si	Si
invoiceNumber	ID Factura o de la transacción asignado por el comercio.	Si	Si

maxInstallmentCount	Cantidad máxima de cuotas que el comercio desea ofrecer al cliente para pagar. Ejemplo: “1”	Si	Si
method	selectOption = se le preguntará al usuario el método de pago card = tarjeta crédito o débito pse = PSE nequi = Nequi pin = Pin - Efectivo	Si	Si
mode	• redirect = Valor por defecto • api = cuando se crea la transacción por API. Diferente a Redirect	Si	Si
origin	URL de origen desde donde se hace la petición	Si	Si
paymentRequiresLogin	Valor por defecto = true		
paymentShippingAddress	Valor por defecto = false		
password	Contraseña de la terminal del comercio	Si	Si
rejectCallbackUrl	URL para redirect del callback en caso de transacción rechazada	Si	Si
successCallbackUrl	URL para redirect del callback en caso de transacción exitosa	Si	Si
taxAmount	Importe de IVA de la transacción. Incluir este monto en el importe total.	Si	Si
terminalId	Id de la terminal del comercio.	Si	Si
totalAmount	Importe total de la transacción. El punto se usa para los decimales. ReGex: ^\\d+(\\.\\d+)?$	Si	Si
userName	Usuario de la terminal del comercio	Si	Si
extraData1	Dato extra. 255 caracteres máximo.	No	No
extraData2	Dato extra. 255 caracteres máximo.	No	No
extraData3	Dato extra. 255 caracteres máximo.	No	No
subMerchantIdentification	Dato extra. Id del sub-comercio.	No	No

- Firma:

Para verificar la autenticidad e integridad de la información enviada, al conjunto de parámetros de la request POST se le debe agregar un campo llamado “signature” (Firma). El valor del campo “signature” es una cadena que contiene una firma generada con pares de claves RSA. Para generar la firma se deben seguir los siguientes pasos:

- Todos los parámetros para enviar en la request POST, exceptuando el campo “signature”, deben ser concatenados como “clave=valor”, separados por el carácter “&” y ordenados alfabéticamente. Ejemplo: “billingDocument=1235140197&billingDocumentType=CC&…”

- Al comercio se le brindará una llave privada RSA con la cual cifra la cadena obtenida del paso anterior y dicha cadena cifrada se envía en el campo “signature”. A continuación, se ejemplifica con lenguaje Python la generación de una firma:

```
from cryptography.hazmat.backends import default_backend
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.asymmetric import padding
from cryptography.hazmat.primitives.serialization import load_pem_private_key

string_to_sign = "billingDocument=123&billingDocumentType=CC&customerEmail=test@test.com"
data_to_sign = string_to_sign.encode("utf-8")

with open('private_key.pem', 'rb') as key_file:
    private_key_pem = key_file.read()

private_key = load_pem_private_key(private_key_pem, password=None,
backend=default_backend())

signature = private_key.sign(
    data_to_sign,
    padding.PKCS1v15(),
    hashes.SHA512()
)

# Firma generada en hexadecimal
print(signature.hex())
```

- Otro ejemplo con lenguaje Java para la generación de la firma:

```
package com.xxxx.xxxx;

import org.bouncycastle.jce.provider.BouncyCastleProvider;
import org.bouncycastle.util.io.pem.PemObject;
import org.bouncycastle.util.io.pem.PemReader;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.core.io.ClassPathResource;

import java.io.FileReader;
import java.io.IOException;
import java.security.KeyFactory;
import java.security.PrivateKey;
import java.security.Security;
import java.security.Signature;
import java.security.spec.PKCS8EncodedKeySpec;

@SpringBootApplication
public class FirmaRsaJavaApplication {

    public static void main(String[] args) throws Exception {
        // Agregar el proveedor Bouncy Castle
        Security.addProvider(new BouncyCastleProvider());

        // Crear la cadena a firmar
        String stringToSign = "billingDocument=80265579&billingDocumentType=UNDEFINED";

        // Cargar la llave privada desde el archivo PEM
        ClassPathResource resource = new ClassPathResource("private_key.pem");
        PrivateKey privateKey = loadPrivateKey(resource.getFile().toPath().toString());

        // Crear un objeto Signature con el algoritmo SHA512withRSA
        Signature signature = Signature.getInstance("SHA512withRSA");

        // Inicializar el objeto Signature con la llave privada
        signature.initSign(privateKey);

        // Firmar una cadena
        signature.update(stringToSign.getBytes());

        // Obtener la firma
        byte[] signatureBytes = signature.sign();

        // Convertir la firma a hexadecimal
        StringBuilder signatureHex = new StringBuilder();
```

```
        for (byte b : signatureBytes) {
            signatureHex.append(String.format("%02x", b));
        }

        // Usar esta firma en el campo signature del request
        System.out.println(signatureHex);
    }

    private static PrivateKey loadPrivateKey(String path) throws Exception {
        // Leer el contenido del archivo PEM
        try (PemReader reader = new PemReader(new FileReader(path))) {
            PemObject pemObject = reader.readPemObject();
            byte[] keyBytes = pemObject.getContent();

            // Crear un objeto PKCS8EncodedKeySpec con los bytes de la llave privada
            PKCS8EncodedKeySpec keySpec = new PKCS8EncodedKeySpec(keyBytes);

            // Crear un objeto KeyFactory con el algoritmo RSA
            KeyFactory keyFactory = KeyFactory.getInstance("RSA");

            // Generar la llave privada
            return keyFactory.generatePrivate(keySpec);
        } catch (IOException e) {
            throw new Exception("Error al leer el archivo PEM", e);
        }
    }
}
```

- El servidor recibe la solicitud, descifra la firma con la clave pública del comercio y valida si la cadena obtenida coincide con los parámetros recibidos.

Post de ejemplo:

```
curl 'https://<ambiente_de_pasarela>.superpay.com.co/ws/router/payment-page' \
  -H 'authority: <ambiente_pasarela>.superpay.com.co' \
  -H 'accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*
/*;q=0.8,application/signed-exchange;v=b3;q=0.7' \
  -H 'accept-language: es-ES,es;q=0.9' \
  -H 'cache-control: max-age=0' \
  -H 'content-type: application/x-www-form-urlencoded' \
  -H 'origin: http://localhost:5000' \
  -H 'referer: http://localhost:5000/' \
  -H 'sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Google Chrome";v="120"' \
  -H 'sec-ch-ua-mobile: ?0' \
  -H 'sec-ch-ua-platform: "Windows"' \
  -H 'sec-fetch-dest: document' \
  -H 'sec-fetch-mode: navigate' \
  -H 'sec-fetch-site: cross-site' \
  -H 'upgrade-insecure-requests: 1' \
  -H 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/120.0.0.0 Safari/537.36' \
  --data-raw
'billingDocument=1235140197&billingDocumentType=CC&branchName=SGM_SUCURSAL&cancelCallbac
kUrl=http%3A%2F%2Flocalhost%3A5000%2Fresponse-from-
gateway%3Fstatus%3DCANCEL&client=supergiros&currency=170&customerEmail=ego999%40yopmail.
com&customerName=harold+castro+*&errorCallbackUrl=http%3A%2F%2Flocalhost%3A5000%2Frespon
se-from-gateway%3Fstatus%3DERROR&externalId=00000003-
0099999994&extraTaxAmount=0&invoiceNumber=99999994&maxInstallmentCount=12&method=selectO
ption&mode=redirect&password=*redeban4321User*&paymentRequiresLogin=true&rejectCallbackU
rl=http%3A%2F%2Flocalhost%3A5000%2Fresponse-from-
gateway%3Fstatus%3DREJECT&successCallbackUrl=http%3A%2F%2Flocalhost%3A5000%2Fresponse-
from-
gateway%3Fstatus%3DSUCCESS&taxAmount=0&terminalId=3&totalAmount=21000&userName=redeban-
user&signature=047de3a0d84de42255fee93bc3f0eabddbce4b771eb69edec2219f90f5dec603f0f256dd6
9d196e34213584589b3e22a0763dfffe3623b911f4d77610d6f3a67065e29c113f16144183ed24f01ff59504
9cda3f6dfd7c20386376996ce816f8ecf0e83dc3a0427a1c81a863bc5568cae99437afe77631ec69ddc02a3e
89161a95dad734b3203a4108715d693c2d813dc0495a920393b0f1dcf04f50a4ab74e0b404feafa81b3217c5
8666e02cbdbd868099e466181a90329c397e53c8a4636389fefddb9c7cc01f98152073be8caf1b042b68d0fb
8baba33643763b2b09deb23c5cd1cf7d913ee4f7e13c79f47ec0a529e3abc92de1d36acfc75ea01ccb51cfa'
\
  --compressed
```

Paso # 2: Respuesta frente a la finalización de la transacción en pasarela

Luego de que el cliente realice la transacción en la página de pagos de la pasarela, la página del comercio obtiene una respuesta en las URL definidas en los parámetros de la request POST hecha en el paso # 1 (successCallbackUrl, cancelCallbackUrl, rejectCallbackUrl) dependiendo el estado final de la transacción. La página del comercio puede descifrar la respuesta con la clave pública de la pasarela de pagos.

Los campos que contiene la request POST del callback son los siguientes:

Parámetro	Descripción
status	Status de la transacción
trxId	Id Transacción (ID único por cada transacción)
branchName	Descripción del comercio definida para desplegar en pantalla
terminalSerialNumber	Serial ID del terminal utilizado

cardHolderName	Nombre del cliente que realiza la compra
totalAmount	Importe total de la transacción
authAmount	Importe autorizado
taxAmount	Importe de IVA de la transacción (incluido en el importe total)
discountAmount	Importe descuento
tipAmount	Importe propina
installmentFirstAmount	Importe primera cuota
installmentNoFirstAmount	Importe siguientes cuotas
installmentCount	Cantidad de cuotas
installmentDeferral	Cuotas diferidas
currency	Moneda de la transacción
gatewayResponseCode	Código de Respuesta
gatewayResponseText	Descripción de la respuesta
trxStatus	Estado de la transacción en la pasarela. Los valores ENUM de este campo se explican en el apartado “Consulta de estado de la transacción”
protocolVersion	Versión Protocolo
trxType	Tipo de Transacción
invoiceNumber	ID Factura/Transacción del comercio
acqRrn	Retrieval Reference Number del Adquirente
acqAuthCode	Código de Respuesta del Adquirente
dtRequest	Fecha del Request
dtRequestBranchTz	Fecha y hora Request en el horario del comercio
dtResponse	Fecha y hora de respuesta en el horario del host
dtResponseBranchTz	Fecha y hora de respuesta en el horario del comercio
successCallbackUrl	*URL* para redirect del callback en caso de transacción exitosa (get SUCCESS Status)
cancelCallbackUrl	*URL* para redirect del callback en caso de transacción cancelada (get CANCEL status)
rejectCallbackUrl	*URL* para redirect del callback en caso de transacción rechazada/negada (get REJECT status)
errorCallbackUrl	*URL* para redirect del callback en caso de transacción con error (get ERROR status)
externalId	Identificación externa generada por el cliente que se utiliza para identificar de forma única la transacción

Paso # 3: Validación de autenticidad del callback (signature RSA)

Cuando la transacción finaliza en la pasarela (modo REDIRECT), el usuario es redirigido a la URL de respuesta del comercio (SUCCESS/CANCEL/REJECTED/ERROR/PENDING) incluyendo en la query string un conjunto de parámetros de la transacción y un parámetro adicional llamado signature.

El objetivo de signature es que el comercio pueda verificar la autenticidad e integridad de los parámetros recibidos y confirmar que el callback proviene realmente de la pasarela, y no de un tercero que haya intentado alterar o suplantar la respuesta.

- Importante: aunque a veces se use el término “descifrar”, técnicamente este proceso corresponde a verificar una firma digital RSA usando la llave pública de la pasarela.

Concepto (quién firma y quién valida)

- **Firma del comercio (en la creación REDIRECT):** el comercio firma el request con su llave privada y la pasarela valida con la llave pública del comercio.
- **Firma de la pasarela (en el callback):** la pasarela firma el callback con su llave privada y el comercio valida con la llave pública de la pasarela (archivo public_key_callback.pem).

Este mecanismo es equivalente al descrito en la sección “Firma” del documento (armado de signedData, orden, concatenación), con la diferencia de que aquí el comercio actúa como validador de la firma generada por la pasarela.

Datos incluidos en la firma del callback (signedData)

Para validar signature, el comercio debe construir una cadena signedData con los parámetros recibidos excluyendo el campo signature, usando exactamente los siguientes campos (siempre que existan en el callback):

- acqAuthCode
- acqRrn
- acqStan
- acquirerId
- branchId
- currency
- externalId
- extraTaxAmount
- invoiceNumber
- mode
- taxAmount
- terminalId
- totalAmount
- trxId
- trxStatus
- trxType

Regla de armado de signedData

1. Tomar los parámetros anteriores tal como llegan en la URL (posterior a URL-decoding normal, sin “inventar” valores).
2. Excluir el parámetro signature.
3. Ordenar los campos alfabéticamente por el nombre del parámetro.
4. Concatenar en el formato:
   clave=valor&clave=valor&…
   usando & como separador, sin espacios, y respetando exactamente mayúsculas/minúsculas de valores (por ejemplo, APPROVED).

Ejemplo de signedData (con valores ilustrativos)

A modo de referencia, usando los valores del ejemplo:

```
acqAuthCode=0005213400&acqRrn=122624726800&acqStan=517735&acquirerId=12189&branchId=21898&currency=170&externalId=00026080-90001896&extraTaxAmount=0&invoiceNumber=12074649&mode=redirect&taxAmount=0&terminalId=26080&totalAmount=100&trxId=1000000001432103&trxStatus=APPROVED&trxType=BANK_TRANSFER
```

- Nota: el orden mostrado arriba ya corresponde al orden alfabético por clave. Si el comercio arma el signedData con otro orden, la validación fallará.

Proceso de validación de signature (paso a paso)

Para validar la firma del callback, el comercio debe:

1. **Recibir y almacenar** todos los parámetros del callback (incluyendo signature).
2. **Construir el signedData** con las reglas descritas (excluyendo signature y ordenando alfabéticamente).
3. **Cargar la llave pública RSA de la pasarela** desde public_key_callback.pem.
4. **Validar la firma**:
    a) Convertir el valor signature al formato esperado por la librería (por ejemplo, bytes si viene en hexadecimal/base64 según el caso).
    b) Ejecutar la verificación RSA usando **el mismo esquema/algoritmo definido en la sección "Firma"** (hash + padding) pero aplicado a:
        1. **Mensaje:** signedData
        2. **Firma:** signatura
        3. **Llave:** pública de la pasarela (public_key_callback.pem)
5. **Decisión**:
    a) Si la validación es **exitosa**, el comercio puede confiar en que el callback es auténtico y no fue alterado.
    b) Si la validación **falla**, el comercio **no debe** confiar en esos datos (recomendado: registrar el evento y manejarlo como intento inválido o estado no verificable).

Consideraciones importantes

- El comercio debe realizar la validación **antes** de marcar una transacción como pagada/aprobada.
- La firma valida la **integridad** de los parámetros firmados. Si un tercero modifica cualquier valor (ej. totalAmount, trxStatus, trxId), la verificación debe fallar.
- Si el callback incluye parámetros adicionales (distintos a los listados), estos **no** deben incorporarse al signedData para este proceso, a menos que se indique explícitamente una actualización del conjunto de campos firmados.
- Es recomendable registrar en logs (sin exponer datos sensibles) cuando la firma falle, junto con terminalId, externalId, trxId y timestamp.

Ejemplos de implementación

A continuación, se presentan ejemplos de validación de firma en los lenguajes más comunes para integraciones:

- Java:

```
package com.superpay.signature_example;

import java.nio.file.Files;

import java.nio.file.Paths;
```

```
import java.security.KeyFactory;

import java.security.PublicKey;

import java.security.Signature;

import java.security.spec.X509EncodedKeySpec;

import java.util.Base64;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;


@SpringBootApplication

public class SignatureExampleApplication {


    public static void main(String[] args) {

        SpringApplication.run(SignatureExampleApplication.class, args);


        String signedData =

            "externalId=00026080-90001962&totalAmount=100&trxStatus=APPROVED";

        String signatureBase64Hex =

            "87a55778f3d4d5cb4312d3cfb0453a0fd3fa1fbbe82a1242b641e31820d9c744016e286ccebc5bc714953ad53bb06c2a6a73b935
2d2ed5140a34916fbf7cb39d65bfa87db6e15b953b0b7225dc0571877757e9bff5c5aa1cee04924dace211dac53b7701d38b860aba02569a72317
8086640ab46f302c5fc20f229b97ec862eb29e134dd6317219e6171bd18b8868d0138eb1c3aef75cc531f52484d7c36136928c9cf5e1743ea10b6
11d2db79c3642f0e1da50168a38f0a09bb772338356e3f0c3b4d5a4bcd29e5ef3ccfdbf45c689de008d2c93fb85a2bfeca18aecebd3c89913d36c
60f66ea024c74b70818fe706078f77df262fd6ef1e7d57efdbdf86337";

        String publicKeyFilePath ="src\\main\\resources\\public_key_gateway_webhook.pem";

        boolean isValid = isSignatureValid(

            signedData,

            signatureBase64Hex,

            publicKeyFilePath

        );

        System.out.println("¿Es válida la firma? " + isValid);

    }

    public static boolean isSignatureValid(

        String signedData,

        String signatureBase64Hex,

        String publicKeyFilePath

    ) {

        try {

            String publicKeyContent = new String(

                Files.readAllBytes(Paths.get(publicKeyFilePath))

            )

                .replace("-----BEGIN PUBLIC KEY-----", "")

                .replace("-----END PUBLIC KEY-----", "")

                .replaceAll("\\s+", "");

            byte[] publicKeyBytes = Base64.getDecoder().decode(

                publicKeyContent

            );
```

```
            X509EncodedKeySpec keySpec = new X509EncodedKeySpec(publicKeyBytes);
            KeyFactory keyFactory = KeyFactory.getInstance("RSA");
            PublicKey publicKey = keyFactory.generatePublic(keySpec);
            Signature signature = Signature.getInstance("SHA512withRSA");
            signature.initVerify(publicKey);
            signature.update(signedData.getBytes("UTF-8"));

            byte[] signatureBytes = hexStringToByteArray(signatureBase64Hex);

            return signature.verify(signatureBytes);
        } catch (Exception e) {
            e.printStackTrace();
            return false;
        }
    }

    public static byte[] hexStringToByteArray(String hex) {
        int len = hex.length();
        byte[] data = new byte[len / 2];
        for (int i = 0; i < len; i += 2) {
            data[i / 2] = (byte) ((Character.digit(hex.charAt(i), 16) << 4) +
                Character.digit(hex.charAt(i + 1), 16));
        }
        return data;
    }
}
```

- Php

//

```
<?php

$signedData =

    "acqAuthCode=0005246122&acqRrn=082418378946&acqStan=847991&acquirerId=12189&branchId=21898&currency=170&externalId=00026080-
90001962&extraTaxAmount=0&invoiceNumber=12074649&mode=redirect&taxAmount=0&terminalId=26080&totalAmount=100&trxId=1000000001489549&trxStatus=APPROVED&trxType=BANK_TRANSFER";



$signatureBase64Hex =

    "31d086f3d1e983e9af33d5775944900a3b6719c2d8ffc01c17f2d38273ab2eaae735f40cf8cbb42a1fe9defb57ccc5c227458483f678785462de9314e9d502ae9536e7c5d265ecb07c08a792582531a28e4dc6b80c9dc9ed7ef7c60b1a87b10125d2880179d7990121d298044cfa636b99685b57d0385add1c5f7f829f276927dc95816bdbb6d1616e472a2cd38251a1da0b32328658baed1708627eed53447c3ec77e17cc1ec3da42ae2a9954dd125540f883f96d378eaa3d8b27aeb4cf87af87e269cd4e3d9bf4b6f3fd4da60f3860cd38f277911eec29548d0c9394257544dc198c48e45475bfa124c4ab734bc85f675183d8f63a783ba89ed35297d3bd4f";



$publicKeyFilePath = "public_key_callback.pem";

$isValid = isSignatureValid(

    $signedData,

    $signatureBase64Hex,
```

```
    $publicKeyFilePath,
);
echo "¿Es válida la firma? " . ($isValid ? "true" : "false") . PHP_EOL;
function isSignatureValid(
    string $signedData,
    string $signatureBase64Hex,
    string $publicKeyFilePath,
): bool {
    try {
        $publicKeyPem = @file_get_contents($publicKeyFilePath);
        if ($publicKeyPem === false) {
            throw new Exception(
                "No se pudo leer el archivo de llave pública: {$publicKeyFilePath}",
            );
        }
        $publicKey = openssl_pkey_get_public($publicKeyPem);
        if ($publicKey === false) {
            $errors = collectOpenSslErrors();
            throw new Exception(
                "No se pudo cargar la llave pública. Errores: " .
                    implode(" | ", $errors),
            );
        }
        $signatureBytes = hex2bin($signatureBase64Hex);
        if ($signatureBytes === false) {
            openssl_free_key($publicKey);
            throw new Exception("La firma no es un HEX válido.");
        }

        $verify = openssl_verify(
            $signedData,
            $signatureBytes,
            $publicKey,
            OPENSSL_ALGO_SHA512,
        );

        openssl_free_key($publicKey);

        if ($verify === 1) {
            return true;
        }
        if ($verify === 0) {
```

```
            return false;
        }
        $errors = collectOpenSslErrors();
        throw new Exception(
            "Error al verificar la firma. Errores: " . implode(" | ", $errors),
        );
    } catch (Throwable $e) {
        fwrite(STDERR, $e->__toString() . PHP_EOL);
        return false;
    }
}
function collectOpenSslErrors(): array
{
    $errs = [];
    while ($msg = openssl_error_string()) {
        $errs[] = $msg;
    }
    return $errs;
}
```

API transaccional

Login

- Login por terminal – Para transacciones:

Antes de crear o consultar una transacción es necesario hacer un login con las credenciales de la terminal asignada al comercio. Este login es necesario para obtener el API token que se usa en las peticiones siguientes.

```
curl --location 'https://<ambiente_de_pasarela>.superpay.com.co/api/router-
bck/terminals/login' \
--header 'x-api-key: X1DGb0eyZh8LznHpSffPS9iY3P59EqmO4Ctcx9eJ' \
--header 'Content-Type: application/json' \
--data '{
    "userName": "<terminal userName enviado>",
    "password": "<terminal password enviado>",
    "terminalId": "<terminalId enviado>"
}'
```

Respuesta:

```
{
    "terminalId": "<terminalId enviado>",
    "terminalSerialNumber": null,
    "terminalType": "E_COMMERCE",
    "dtRequest": "2024-02-13 21:19:25",
    "externalId": null,
    "acqStan": "2626",
    "trxId": "2000000000311064",
    "uniqueId": "128d29a5-ef30-4ed9-8244-5e4c8abc9d74",
    "trxType": "LOGIN",
    "merchantId": "23782",
    "branchId": "21898",
    "trxStatus": "APPROVED",
    "dtResponse": "2024-02-13 21:19:25",
    "dtRequestBranchTz": "2024-02-13 16:19:25",
    "dtResponseBranchTz": "2024-02-13 16:19:25",
    "acqRrn": "348624486925",
    "apiToken": "V7oUtxVRDlHDl5bifYps7MhKzD52jTt5jZ4nSTZhPuIKe7IIae6G1uQw0TjnybcD5LN3",
    "apiTokenExpiration": "2024-03-14 21:19:25",
    "merchantName": "SuperGiros Movil",
    "branchName": "SuperGiros Movil",
    "terminalName": "SuperGiros Movil"
}
```

- Login por comercio – Para reportería, enlaces y pines de pago:

Para generar el token <base64(clientId:clientSecret)> se deben concatenar el clientId y el clientSecret enviados de la siguiente forma clientId:clientSecret y codificar dicha cadena en base 64. También el parámetro <merchantName> se enviará junto con las credenciales.

```
curl --location --request POST 'https://<ambiente_de_pasarela>.superpay.com.co/router-bck/merchants/login/<merchantName>' \
--header 'Authorization: Basic <base64(clientId:clientSecret)>'
```

Respuesta:


```
{
	"accessToken":
"eyJraWQiOiI3NGc1K2taTnNyQWdDcXJ5c2NkTTFieW9Xdmd1MVhZUWxycGg3Q2tnRHYwPSIsImFsZyI6IlJTMjU2In0
.eyJzdWIiOiI0ZjRzMGFsczN0ajVkMmlxZmNsODVobTkyOCIsInRva2VuX3VzZSI6ImFjY2VzcyIsInNjb3BlIjoicmV
zLXNlcnZlci1kZXZcL2NvbXBsZXRlLWFwaSIsImF1dGhfdGltZSI6MTcxODMwNzg0MSwiaXNzIjoiaHR0cHM6XC9cL2N
vZ25pdG8taWRwLnVzLWVhc3QtMi5hbWF6b25hd3MuY29tXC91cy1lYXN0LTJfUjR2RXpMRkI1IiwiZXhwIjoxNzE4MzE
xNDQxLCJpYXQiOjE3MTgzMDc4NDEsInZlcnNpb24iOjIsImp0aSI6ImJiYTkyNmRhLTY3MzItNDM4Ni04NDNhLWFkNGU
wNDQyODFjMyIsImNsaWVudF9pZCI6IjRmNHMwYWxzM3RqNWQyaXFmY2w4NWhtOTI4In0.plHwFRnsIMlmRsp2gEamelw
y9kX6m4TIOuBnItXIR3SDFvpmqDjFMYs2OJ1ff_9TVqmuLRnZ6GZ33RqZyXYf0a96B0guszu4zqlDVizKUdEpu0fqFQ1
3MtsdIFG0xy3f8coP4jbB_oDU6jJ9hA-hAxd9XgXvoIQl3Yd6irExh34BL6gfGXCduwgK0dXhZurcOWl1rU--
kuqnoYR2uW2RdyaTryRq12Y5HQGeEmybgffbVYq3NKwulLeJgwPS25oSQJ7m55tZIfjEUcieEZ-
RWykwWD1SOsTiNgfW7iOlah8fMbm05ODvUqeu8NdJXvM05VLt4BXMyGlbSje1QXrFhw",
	"expiresIn": "3600",
	"tokenType": "Bearer"
}
```


Creación de la transacción:

Endpoint para la creación de transacciones. De acuerdo con el método de pago algunos parámetros son obligatorios, por lo que a continuación, a manera de ejemplo, se listan CURLs que muestran el consumo del endpoint de acuerdo con el método de pago. Cabe resaltar que la función de los campos ya se explicó en el apartado “Flujo de la información en Modo Redirect – Creación del pago” en los pasos “Paso # 1: Envío de información para el pago” y “Paso # 2: Respuesta frente a la finalización de la transacción en pasarela”:

Nequi

```
curl --location 'https://<ambiente_pasarela>/api/router-bck/transactions' \
--header 'Content-Type: application/json' \
--data '{
    "accessToken": "<api-token-login-por-terminal>",
    "client": "supergiros",
    "countryIsoCode": "COL",
    "currency": "170",
    "customerPhone": "3720002109",
    "externalId": "00026080-00010280",
    "invoiceNumber": "1234567",
    "method": "nequi",
    "mode": "api",
    "origin": " https://elcomercio.com.co/carrito-compras",
    "terminalId": "26080",
    "totalAmount": "21000"
}'
```

Respuesta:

```
{
    "id": "6ea321ab-e2f7-48fb-bdcc-f5e8e4d4c9f9",
    "client": "nombre-cliente",
    "countryIsoCode": "COL",
    "currency": "170",
    "customerPhone": "3720002109",
    "externalId": "00026080-00010280",
    "invoiceNumber": "1234567",
    "method": "nequi",
    "mode": "api",
    "origin": "https://elcomercio.com.co/carrito-compras",
    "terminalId": "26080",
    "totalAmount": "21000",
    "selectedPaymentMethod": "nequi",
    "createdAt": "2024-03-19T12:06:07.0347649",
    "trxStatus": "PENDING"
}
```

PSE

```
curl --location 'https://sbx.superpay.com.co/api/router-bck/transactions' \
--header 'Content-Type: application/json' \
--data-raw '{
    "accessToken": ""<api-token-login-por-terminal>",
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "billingCity": "BOGOTA",
    "billingCountryIsoCode": "170",
    "customerAddress": "addressTest",
    "customerAddress2": "addressTest",
    "customerEmail": "test@email.com",
    "customerName": "Mathias Suarez",
    "customerPhone": "3205104892",
    "externalId": "00026080-032876513",
    "extraTaxAmount": "0",
```

```
    "paymentShippingAddress": "addressTest",
    "taxAmount": "0",
    "mode": "api",
    "method": "pse",
    "terminalId": "26080",
    "totalAmount": 21000,
    "bankCode": "1022",
    "branchName": "Tienda 1",
    "cancelCallbackUrl": "http://url:8080/ws/launcher/response-fromgateway?status=CANCEL",
    "enableNiubizSetting": "false",
    "errorCallbackUrl": " http://url:8080/ws/launcher/response-fromgateway?status=ERROR",
    "maxInstallmentCount": "1",
    "paymentRequiresLogin": "true",
    "rejectCallbackUrl": " http://url:8080/ws/launcher/response-fromgateway?status=REJECT",
    "successCallbackUrl": " http://url:8080/ws/launcher/response-fromgateway?status=SUCCESS"
}'
```

Respuesta

```
{
    "accountType": "UNKNOWN",
    "bankCode": "1022",
    "bankUrl":
"https://registro.desarrollo.pse.com.co/PSENF/index.html?enc=47BdZlu0gjUYh3sTWQ0y2jrbGSCUfIw
37ZP9r%2fegWdA%3d",
    "billingCity": "BOGOTA",
    "billingCountryIsoCode": "170",
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "branchName": "BRANCH_ACH",
    "currency": "170",
    "customerAddress": "addressTest",
    "customerAddress2": "addressTest",
    "customerEmail": "test@email.com",
    "customerName": "Mathias Suarez",
    "customerPhone": "3205104892",
    "enableNiubizSetting": "false",
    "errorCallbackUrl": " http://url:8080/ws/launcher/response-fromgateway?status=ERROR",
    "externalId": "00026080-032876513",
    "extraTaxAmount": 0.0,
    "gatewayResponseCode": "APPROVED",
    "gatewayResponseText": "Approved",
    "maxInstallmentCount": "1",
    "method": "pse",
    "mode": "api",
    "paymentRequiresLogin": "true",
    "paymentShippingAddress": "addressTest",
    "rejectCallbackUrl": " http://url:8080/ws/launcher/response-fromgateway?status=REJECT",
    "selectedPaymentMethod": "pse",
    "successCallbackUrl": "http://url:8080/ws/launcher/response-fromgateway?status=SUCCESS",
    "taxAmount": 0.0,
    "terminalId": "26080",
    "totalAmount": 21000.0,
    "trxStatus": "PENDING"
}
```

Consulta Listado de Bancos:

```
curl --location 'https://<ambiente_pasarela>.superpay.com.co/api/glc/metadata/list-banks' \
--data ''
```

Respuesta:


```
[
    {
        "financialInstitutionCode": "0",
        "financialInstitutionName": "A continuación seleccione su banco"
    },
    {
        "financialInstitutionCode": "1022",
        "financialInstitutionName": "BANCO UNION COLOMBIANO"
    },
    {
        "financialInstitutionCode": "1077",
        "financialInstitutionName": "BANKA"
    }
]
```


Tarjeta Crédito

Nótese que los campos “mode=api” y “method=card” indican que la transacción se va a hacer vía API con tarjeta de crédito como método de pago.

```
curl --location 'https://<ambiente_pasarela>.superpay.com.co/api/router-bck/transactions' \
--header 'Content-Type: application/json' \
--data-raw '{
    "accessToken": "<api-token-login-por-terminal>",
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "billingCity": "BOGOTA",
    "billingCountryIsoCode": 170,
    "customerAddress": "addressTest",
    "customerAddress2": "addressTest",
    "customerEmail": "test@email.com",
    "customerName": "Mathias Suarez",
    "customerPhone": "3205104892",
    "externalId": "00026080-032876522",
    "extraTaxAmount": 0.0,
    "paymentShippingAddress": "addressTest",
    "taxAmount": 0.0,
    "mode": "api",
    "method": "card",
    "terminalId": "26080",
    "totalAmount": 21000,
    "cardExpDate": "2019-10",
    "pan": "4761340000000035",
    "installmentCount": 2,
    "cvv2": "060"

}'
```

Respuesta:


```
{
    "accountType": "CREDIT",
    "billingCity": "BOGOTA",
    "billingCountryIsoCode": "170",
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "currency": "170",
```

```
    "customerAddress": "addressTest",
    "customerAddress2": "addressTest",
    "customerEmail": "test@email.com",
    "customerName": "Mathias Suarez",
    "customerPhone": "3205104892",
    "externalId": "00026080-032876522",
    "extraTaxAmount": 0.0,
    "gatewayResponseCode": "APPROVED",
    "gatewayResponseText": "Approved",
    "method": "card",
    "mode": "api",
    "paymentShippingAddress": "addressTest",
    "selectedPaymentMethod": "card",
    "taxAmount": 0.0,
    "terminalId": "26080",
    "totalAmount": 21000.0,
    "trxStatus": "APPROVED"
}
```


Descripción de los Campos:

REQUEST			
Nombre	Descripción	Tipo	Obligatorio
accessToken	Token obtenido al hacer login con las credenciales de la terminal.	String	Si
billingDocument	Número de documento del cliente. Alfanumérico. ReGex: ^[\\dA-Za-z-]+$	String	Si
billingDocumentType	Tipo de documento del cliente. Valores: "CC", "NIT", "PASSPORT", "TI", "CE", "RCN", "TI", "TE"	String	Si
billingCity	Ciudad del cliente.	String	Si
billingCountryIsoCode	País del cliente. “170” es igual a Colombia.	String	Si
customerAddress	Dirección del cliente. Se permiten espacios y símbolos: °#,/-	String	Sí
customerAddress2	Segunda línea de la dirección del cliente (opcional).	String	No
customerEmail	Correo electrónico del cliente. ReGex: ^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$	String	Sí
customerName	Nombre del cliente.	String	Sí
customerPhone	Teléfono del cliente. ReGex: ^[0-9\\s+]*$	String	Sí
externalId	ID externo para la transacción. ReGex: ^\\d{8,11}-\\d{8,11}$	String	Sí
extraTaxAmount	Monto de impuestos adicionales.	Double	No
paymentShippingAddress	Dirección de envío del pago.	String	No
taxAmount	Monto de impuestos.	Double	No
mode	Modo de la transacción (e.g., api).	String	Sí
method	Método de pago (e.g., card).	String	Sí
terminalId	ID de la terminal desde la cual se realiza la transacción. ReGex: ^\\d+$	String	Sí
totalAmount	Monto total de la transacción. El punto se usa para los decimales. ReGex: ^\\d+(\\.\\d+)?$	Double	Sí
cardExpDate	Fecha de expiración de la tarjeta (formato: YYYY-MM). Ejemplo: 2024-07. ReGex: \\d{4}-\\d{2}.	String	Sí
pan	Número de la tarjeta (PAN). ReGex: ^\\d{13,16}$	String	Sí
installmentCount	Número de cuotas. ReGex: ^[1-9]\\d*$	String	Si
cvv2	Clave CVV de la tarjeta de crédito. Números de tres a cuatro digitos. ReGex: ^\\d{3,4}$	String	Si
extraData1	Dato extra. 255 caracteres máximo.	String	No
extraData2	Dato extra. 255 caracteres máximo.	String	No
extraData3	Dato extra. 255 caracteres máximo.	String	No
subMerchantIdentification	Dato extra. Id del sub-comercio.	String	No

RESPUESTA			
Nombre	Descripción	Tipo	Ejemplo
accountType	Tipo de cuenta utilizada en la transacción.	String	CREDIT
billingCity	Ciudad de facturación del cliente.	String	BOGOTA
billingCountryIsoCode	Código ISO del país de facturación (COL = 170).	String	170

billingDocument	Número de documento de facturación del cliente.	String	123456789
billingDocumentType	Tipo de documento de facturación.	String	CC
currency	Código ISO de la moneda utilizada.	String	170
customerAddress	Dirección del cliente.	String	addressTest
customerAddress2	Segunda línea de la dirección del cliente.	String	addressTest
customerEmail	Correo electrónico del cliente.	String	test@email.com
customerName	Nombre del cliente.	String	Mathias Suarez
customerPhone	Teléfono del cliente.	String	3205104892
externalId	ID externo para la transacción.	String	00026080-032876522
extraTaxAmount	Monto de impuestos adicionales.	Double	0.0
gatewayResponseCode	Código de respuesta de la pasarela.	String	APPROVED
gatewayResponseText	Texto de la respuesta de la pasarela.	String	Approved
method	Método de pago utilizado.	String	card
mode	Modo de la transacción.	String	api
paymentShippingAddress	Dirección de envío del pago.	String	addressTest
selectedPaymentMethod	Método de pago seleccionado.	String	card
taxAmount	Monto de impuestos.	Double	0.0
terminalId	ID de la terminal utilizada.	String	26080
totalAmount	Monto total de la transacción.	Double	21000.0
trxStatus	Estado de la transacción.	String	APPROVED
extraData1	Dato extra. 255 caracteres máximo.	String	extra-data-1
extraData2	Dato extra. 255 caracteres máximo.	String	extra-data-2
extraData3	Dato extra. 255 caracteres máximo.	String	extra-data-3
subMerchantIdentification	Dato extra. Id del sub-comercio.	String	123456789

Errores:

- Cuando la transacción sea rechazada recibe en el cuerpo de la respuesta la información de la transacción junto con su estado (TrxStatus) y monto (totalAmount) – Código HTTP 201 Created:


```
{
    "accountType": "CREDIT",
    "billingCity": "BOGOTA",
    "billingCountryIsoCode": "170",
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "createdAt": "2024-08-06T21:38:38.571057153",
    "currency": "170",
    "customerAddress": "Calle 46 No 12 d 20",
    "customerEmail": "test@email.com",
    "customerName": "Mathias Suarez",
    "customerPhone": "3205104892",
    "enableNiubizSetting": "false",
    "externalId": "00025627-90000262",
    "extraTaxAmount": 0.0,
    "gatewayResponseCode": "DECLINED",
    "gatewayResponseText": "Declined",
    "id": "c3efcdfe-d483-4ec6-a32b-625243f7c678",
    "maxInstallmentCount": "10",
    "method": "card",
    "mode": "api",
    "paymentRequiresLogin": "false",
    "paymentShippingAddress": "Calle 46 No 12 d 20",
    "selectedPaymentMethod": "card",
    "taxAmount": 0.0,
    "terminalId": "25627",
    "totalAmount": 100100.0,
    "trxStatus": "DECLINED"
}
```

- Al enviar un número de tarjeta que no sea VISA o MASTERCARD – Código HTTP 400 Bad Request:

```
{
    "code": "GLC_ERROR_004",
    "message": "CARD_ROUTE_NOT_MATCH - Card route not match"
}
```

- Al enviar un campo (Mirar campos que tienen una ReGex en sus descripciones) en un formato incorrecto como, por ejemplo, el número CVV2 con un valor como “060a*c” donde solo permiten solo números con 3 a 4 dígitos el endpoint responderá con la expresión regular del formato con el que espera el valor y el nombre del campo como se muestra a continuación - Código HTTP 400 Bad Request:

```
{
    "code": "ROUTER_ERROR_000",
    "message": "cvv2: The value does not meet the regular expression ^\\d{3,4}$"
}
```

PIN

Nótese que los campos “mode=api” y “method=pin” indican que la transacción se va a hacer vía API con PIN como método de pago. El PIN puede ser enviado desde la request y se valida que este sea único por terminal y/o sub-comercio (subMerchantIdentification). En caso de que no sea enviado, el endpoint generará y retornará en respuesta un PIN único por terminal y/o sub-comercio (subMerchantIdentification)

Los campos para la creación de un PIN son los siguientes:

Parámetro	Descripción	Obligatorio	Tipo
billingCity	Ciudad del cliente (Ej: BOGOTA)		
billingDocument	Número de documento del cliente. Alfanumérico. ReGex: ^[\\dA-Za-z-]+$	Si	String
billingDocumentType	Tipo de Documento del cliente: - CC: Cédula de ciudadanía - NI: Nit - PASSPORT: Pasaporte - TI: Tarjeta de Identidad - CE: Cédula Extranjería - RCN: Registro Civil de Nacimiento - TI: Tarjeta de Identidad - TE: Tarjeta de Extranjería - Foreign Id: Documento de identificación Extranjero - Foreign commerce Id: Identificación Comercio Extranjero	Si	String
billingCountryIsoCode	Valor por defecto: Colombia = COL	Si	String
currency	Id de la moneda de la transacción (COP = 170)	Si	String
customerAddress	Dirección de cliente que realiza la compra. Se permiten espacios y símbolos: °#,/-	Si	String
customerEmail	e-mail del cliente que realiza la compra. ReGex: ^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$	Si	String

customerName	Nombre del cliente que realiza la compra	Si	String
customerPhone	Número de teléfono del cliente que realiza la compra. ReGex: ^[0-9\\s+]*$	Si	String
externalId	Identificación única generada por el comercio para identificar la transacción. El formato para la generación del UID es el siguiente: • Últimos ocho caracteres del ID del terminal (relleno con 0 a la izquierda si el ID del terminal tiene menos de 8 caracteres separador de guiones / guiones • Valor del contador (relleno con 0 a la izquierda si el valor actual es menor que 99999999999) ***Ejemplo 1:*** ID de terminal: TZXCALJTUXVS Valor contador comercio: 14 UID generado: ALJTUXVS-00000000014 ***Ejemplo 2:*** ID de terminal: 58963 Valor contador comercio: 25637 UID generado: 00058963-00000025637	Si	String
extraTaxAmount	Importe de IAC de la transacción. Incluir este monto en el importe total.	Si	Double
invoiceNumber	ID Factura o de la transacción asignado por el comercio.	Si	String
method	pin	Si	String
mode	api = cuando se crea la transacción por API. Diferente a Redirect	Si	String
paymentShippingAddress	Direccion del cliente		String
taxAmount	Importe de IVA de la transacción. Incluir este monto en el importe total.	Si	Double
terminalId	Id de la terminal del comercio.	Si	String
totalAmount	Importe total de la transacción. El punto se usa para los decimales. ReGex: ^\\d+(\\.\\d+)?$	Si	Double
startsAt (UTC+0)	Fecha desde la que estará vigente el pin	Si	String
expiresAt (UTC+0)	Fecha hasta la que estará vigente el pin	Si	String
extraData1	Dato extra. 255 caracteres máximo.	No	String
extraData2	Dato extra. 255 caracteres máximo.	No	String
extraData3	Dato extra. 255 caracteres máximo.	No	String
pin	máximo de caracteres = 25	No	String

Request sin PIN (Sin el campo “pin” en el body de la request):

```
curl --location 'https://<ambiente_pasarela>.superpay.com.co/api/router-bck/transactions' \
--header 'Content-Type: application/json' \
--data-raw '{
    "accessToken": "<api-token-login-por-terminal>",
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "billingCity": "BOGOTA",
    "billingCountryIsoCode": 170,
    "currency": "CL 96A 1 No 14Q - 18",
    "customerAddress": "CL 96A 1 No 14Q - 18",
    "customerAddress2": "Casa 7B",
    "customerEmail": "test@email.com",
    "customerName": "Pepe Perez",
    "customerPhone": "3568365379",
    "externalId": "00026680-90000582",
    "extraTaxAmount": "1000",
    "paymentShippingAddress": "addressTest",
    "taxAmount": "1000",
```

```
    "mode": "api",
    "method": "pin",
    "terminalId": "26680",
    "totalAmount": 1000,
    "branchName": "Tienda 1",
    "invoiceNumber": "test-0000",
    "startsAt": "2025-03-16T00:00:00",
    "expiresAt": "2025-04-29T00:00:00",
    "pin": "ABCD-1231",
    "extraData1": "extra-data-1",
    "extraData2": "extra-data-2",
    "extraData3": "extra-data-3"
}'
```

Request con PIN (Nótese al final el campo “pin” en el body de la request):

```
curl --location 'https://<ambiente_pasarela>.superpay.com.co/api/router-bck/transactions' \
--header 'Content-Type: application/json' \
--data-raw '{
    "accessToken": "<api-token-login-por-terminal>",
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "billingCity": "BOGOTA",
    "billingCountryIsoCode": 170,
    "customerAddress": "CL 96A 1 No 14Q - 18",
    "customerAddress2": "Casa 7B",
    "customerEmail": "test@email.com",
    "customerName": "Pepe Perez",
    "customerPhone": "3568365379",
    "externalId": "00026680-90000582",
    "extraTaxAmount": "1000",
    "paymentShippingAddress": "addressTest",
    "taxAmount": "1000",
    "mode": "api",
    "method": "pin",
    "terminalId": "26680",
    "totalAmount": 1000,
    "branchName": "Tienda 1",
    "invoiceNumber": "test-0000",
    "startsAt": "2025-03-16T00:00:00",
    "expiresAt": "2025-04-29T00:00:00",
    "pin": "ABCD-1231",
    "extraData1": "extra-data-1",
    "extraData2": "extra-data-2",
    "extraData3": "extra-data-3"
}'
```

En ambos casos se retorna como respuesta la información enviada, el campo PIN (Ya sea el mismo enviado desde la request o uno generado por el endpoint si este no se envió en la request):

```
{
    "billingCity": "BOGOTA",
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "branchName": "Tienda 1",
    "createdAt": "2025-04-01T21:08:10.428184268",
    "customerAddress": "CL 96A 1 No 14Q - 18",
    "customerEmail": "test@glic-solutions.com",
    "customerName": "Mathias Suarez",
    "customerPhone": "3568365379",
    "externalId": "00026680-90000582",
    "extraTaxAmount": 1000.0,
    "id": "ba348bbe-0abb-41d3-823c-10381737fd94",
```

```
    "invoiceNumber": "test-0000",
    "method": "pin",
    "mode": "api",
    "paymentShippingAddress": "addressTest",
    "taxAmount": 1000.0,
    "terminalId": "26680",
    "totalAmount": 1000.0,
    "trxStatus": "PENDING",
    "startsAt": "2025-03-16T00:00:00",
    "expiresAt": "2025-04-29T00:00:00",
    "pin": " ABCD-1231",
    "extraData1": "extra-data-1",
    "extraData2": "extra-data-2",
    "extraData3": "extra-data-3"
}
```


Al crear la transacción se realizan ciertas validaciones que pueden devolver las siguientes respuestas:

Fechas de inicio o fin de vigencia inválidos:


```
{
    "code": "ROUTER_ERROR_000",
    "message": " the start date or end date is not valid"
}
```


Si se envía PIN en la request y ya existe un PIN igual con la misma terminal y/o sub-comercio (subMerchantId):


```
{
    "code": "ROUTER_ERROR_000",
    "message": "Pin already exists"
}
```


Si el PIN enviado no cumple con el formato permitido:


```
{
    "code": "ROUTER_ERROR_000",
    "message": "The pin format is incorrect"
}
```


Consulta de estado de la transacción:

Se pone a disposición un endpoint para consultar el estado de una transacción por external ID. Cabe recordar que el external ID es el ID generado por el comercio teniendo en cuenta el patrón explicado en el apartado anterior.

De la respuesta obtenida al hacer el login mediante el API se obtiene el API token, el cual es necesario para enviar la solicitud HTTP POST que consulta el estado de la transacción de la siguiente forma:

```
curl --location 'https://<ambiente_pasarela>.superpay.com.co/api/router
```


```
{
    "terminalId": "<<terminalId enviada>>",
    "terminalType": "E_COMMERCE",
    "dtRequest": "2024-02-13T21:23:43.028556470",
    "externalId": "00026080-00020521715",
    "authAmount": "21000",
    "invoiceNumber": "1234567",
    "trxId": "3394d3f9-3fdf-451c-b1dc-3a984d80398e",
    "uniqueId": "3394d3f9-3fdf-451c-b1dc-3a984d80398e",
    "trxType": "NEQUI_TRANSFER",
    "trxStatus": "PENDING",
    "dtResponse": "2024-02-13T21:23:43.028606892",
    "dtRequestBranchTz": "2024-02-13T21:23:43.028589339",
    "dtResponseBranchTz": "2024-02-13T21:23:43.028598755",
    "successCallbackUrl": "https://url.com/ws/launcher/response-from-
gateway?status=SUCCESS",
    "cancelCallbackUrl": "https://url.com/ws/launcher/response-from-
gateway?status=CANCEL",
    "rejectCallbackUrl": "https://url.com/ws/launcher/response-from-
gateway?status=REJECT",
    "errorCallbackUrl": "https://url.com/ws/launcher/response-from-
gateway?status=ERROR",
    "paymentDescription": ""
}
```


- Respuesta para un pago NEQUI:

Los estados para el campo “trxStatus” son:

- PENDING: La transacción aún no se ha pagado y está pendiente.
- APPROVED: La transacción se pagó.
- DECLINED: La transacción fue rechazada por el cliente desde su Nequi o caducó por quedar pendiente un tiempo mayor al que está permitido por Nequi (10 minutos en ambiente de pruebas y en producción la notificación push puede estar hasta 45 minutos pendiente por el pago).
- ERROR: Transacción de Nequi fallida.

- Respuesta para un pago PSE:


```
{
    "terminalId": "<<terminalId enviada>>",
    "terminalType": "E_COMMERCE",
    "dtRequest": "2024-02-13T21:23:43.028556470",
    "externalId": "00026080-00020521715",
    "authAmount": "21000",
    "invoiceNumber": "1234567",
    "trxId": "3394d3f9-3fdf-451c-b1dc-3a984d80398e",
    "uniqueId": "3394d3f9-3fdf-451c-b1dc-3a984d80398e",
    "trxType": "BANK_TRANSFER",
    "trxStatus": "PENDING",
    "dtResponse": "2024-02-13T21:23:43.028606892",
    "dtRequestBranchTz": "2024-02-13T21:23:43.028589339",
    "dtResponseBranchTz": "2024-02-13T21:23:43.028598755",
    "successCallbackUrl": "https://sbx.superpay.com.co/ws/launcher/response-from-
gateway?status=SUCCESS",
    "cancelCallbackUrl": "https://url.com/ws/launcher/response-from-
gateway?status=CANCEL",
    "rejectCallbackUrl": "https://url.com/ws/launcher/response-from-
gateway?status=REJECT",
    "errorCallbackUrl": "https://url.com/ws/launcher/response-from-
gateway?status=ERROR",
    "paymentDescription": ""
}
```


Los estados para el campo “trxStatus” son:

- PENDING: La transacción aún no se ha pagado y está pendiente.
- APPROVED: La transacción se pagó.
- DECLINED: La transacción fue rechazada.
- ERROR: Transacción presentó un error.

- Respuesta para un pago Tarjeta de Crédito:


```
{
    "terminalId": "26080",
    "terminalType": "E_COMMERCE",
    "dtRequest": "2024-06-13 19:56:26",
    "externalId": "00026080-032876523",
    "acqStan": "25821",
    "cardScheme": "VISA-FOREING",
    "accountType": "CREDIT",
    "authAmount": "21000.00",
    "acqResponseCode": "00",
    "acqResponseText": "Aprobado",
    "acqAuthCode": "498461",
    "acquirerId": "6234",
    "routerId": "REDEBAN",
    "invoiceNumber": null,
    "issuerTransactionIdentifier": null,
    "paymentLinkId": null,
    "mcc": null,
    "trxId": "1000000000428937",
    "followOnId": null,
    "uniqueId": "634feffa-861b-43d0-9026-a015c6c9acf7",
    "trxType": "INQUIRY",
```

```
    "merchantId": "23782",
    "branchId": "21898",
    "departmentId": null,
    "gatewayResponseCode": "APPROVED",
    "gatewayResponseText": "Approved",
    "trxStatus": "APPROVED",
    "dtResponse": "2024-06-13 19:56:27",
    "dtAuthorization": "2024-06-13 19:56:26",
    "dtRequestBranchTz": "2024-06-13 14:56:26",
    "dtResponseBranchTz": "2024-06-13 14:56:27",
    "dtAuthorizationBranchTz": "2024-06-13 14:56:26",
    "acqRrn": "174218524626",
    "processorTrxId": "123456789",
    "dtResponseTime": 42.0,
    "merchantName": "SuperGiros Movil",
}
```


Los estados para el campo “trxStatus” son:

- PENDING: La transacción aún no se ha pagado y está pendiente.
- APPROVED: La transacción se pagó.
- DECLINED: La transacción fue rechazada.
- ERROR: Transacción presentó un error.

Si la transacción no existe en la pasarela, se obtiene el siguiente mensaje de error con código 400 BAD REQUEST:


```
{
        "code": "ROUTER_004",
        "message": "Transaction not found"
}
```


Consulta reporte paginado

En esta sección, puedes consultar los reportes generados de manera paginada. Esto te permite navegar a través de grandes cantidades de datos de manera eficiente, visualizando solo una porción de los datos a la vez. Puedes especificar el número de registros que deseas ver en cada página y navegar a través de las páginas según sea necesario.

Request:

```
curl --location 'https://<ambiente_pasarela>.superpay.com.co/api/data-
harvester/transactions/search?parameters=brandId:VISA,merchantName:SuperGiros+Movil,
totalAmount>20000,dtRequest<2024-04-03T15:07:56&page=0&size=10&sort=trxId,asc' \
--header 'x-api-token: Bearer <api-token-login-por-comercio>' \
--header 'x-api-key: <api-key-comercio>'
```

Por favor, ten en cuenta los siguientes detalles al formatear la URL:

- Los caracteres ':' y '< >' deben mantenerse.
- El carácter ':' se utiliza como un equivalente al signo igual '=' para filtrar la búsqueda.
- Los caracteres '< >' se utilizan para indicar "mayor que" '>' o "menor que" '<' en la búsqueda.

- En el identificador brandId se usa el carácter: para realizar la búsqueda "brandId es igual a VISA"
- En el identificador totalAmount se usan los caracteres < > para realizar la búsqueda de montos "totalAmount mayores que " o "totalAmount menores que"
- También en el identificador dtRequest se utilizan los caracteres < > para realizar búsquedas de tiempo "dtRequest menores que " o "dtRequest mayores que "
- Cabe mencionar que el campo dtRequest muestra el tiempo de creación de la request en UTC sin zona horaria y dtRequestBranchTz incluye las 5 horas de la zona horaria colombiana.

Al final de la URL, encontrarás los siguientes parámetros:

- page=0: indica la página de resultados que se está solicitando.
- size=10: indica el número de resultados por página.
- sort=trxId,asc: indica el campo y el orden en que se deben ordenar los resultados.

Estos valores se emplean para filtrar la búsqueda por página &page=0, tamaño de la búsqueda &size=10 y el elemento que se quiere ordenar &sort=trxId,asc.

Los campos que se pueden usar para los filtros son:

trxId, trxIdStrValue, externalId, totalAmount, brandId, trxStatus, dtRequest, dtRequestBranchTz, invoiceNumber, maskedPan, billingAddress, billingCity, billingCountryIsoCode, billingPhone, billingZipCode, billingDocument, billingState, customerName, cardHolderName

Ejemplo respuesta:


```
{
            "content": [
                {
                    "id": "88ff17c8-0a75-4a9d-b295-3fa4010c627c",
                    "trxId": 2000000000363956,
                    "trxIdStrValue": "2000000000363956",
                    "externalId": "00026080-00013242",
                    "merchantId": 23782,
                    "merchantName": "SuperGiros Movil",
                    "branchId": 21898,
                    "branchName": "SuperGiros Movil",
                    "terminalId": 26080,
                    "terminalName": "SuperGiros Movil",
                    "terminalSerialNumber": null,
                    "acquirerMerchantId": "12345678",
                    "acquirerId": 6234,
                    "totalAmount": 21000.0,
                    "currency": "0",
                    "trxStatus": null,
                    "trxType": null,
                    "dtRequest": "2024-04-03T15:07:56",
                    "dtRequestBranchTz": "2024-04-03T10:07:56",
                    "brand": null,
                    "invoiceNumber": "1234567",
                    "sourceIp": "181.49.251.81",
                    "keyReference": "cardEncrypt/v3",
                    "nodeId": 2,
                    "rotateKeyVersion": 3,
```

```
                "maskedPan": "411111******1111",
                "gatewayResponseCode": null,
                "gatewayResponseText": "Approved",
                "acqStan": "13243",
                "acqRrn": "934999336756",
                "productId": null,
                "installmentType": null,
                "installmentCount": 1,
                "salesChannel": null,
                "trxConfirmationStatus": null,
                "clearingStatus": null,
                "acqResponseCode": "00",
                "acqResponseText": "Aprobado",
                "uniqueId": "e55b266f-87cb-4221-b149-edd9e8cbda34",
                "localCurrencyIsoCodeAlpha": "COP",
                "localCurrencyIsoCodeDesc": "Peso colombiano",
                "routerId": "REDEBAN",
                "creditDay": 0,
                "replicable": false,
                "documentType": null,
                "document": "9000847779",
                "techType": null,
                "billingAddress": "NA",
                "billingCity": "Cali",
                "billingCountryIsoCode": 170,
                "billingPhone": "1",
                "billingZipCode": "1",
                "billingDocument": "80265579",
                "billingState": "VAC",
                "customerName": "Mathias Suarez",
                "ecommerceIndicator": null,
                "commissionFixValue": 0.0,
                "commissionPercentage": 0.0,
                "commissionBonification": 0.0,
                "origTrxType": null,
                "settlementType": null,
                "singleMessage": false,
                "acquirerTerminalId": "ESB19065",
                "cardExpDate": null,
                "cardHolderName": "diana rizo",
                "accountType": null,
                "authAmount": 21000.0,
                "posEntryModePANEntry": null,
                "posEntryModePINEntry": null,
                "totalAmountForMixedTrx": 0.0
            }
        ],
        "pageable": {
            "pageNumber": 0,
            "pageSize": 10,
            "sort": {
                "unsorted": false,
                "sorted": true,
                "empty": false
            },
            "offset": 0,
            "paged": true,
            "unpaged": false
        },
        "totalPages": 1,
        "totalElements": 1,
        "last": true,
        "numberOfElements": 1,
        "size": 10,
```

```
            "number": 0,
            "sort": {
                "unsorted": false,
                "sorted": true,
                "empty": false
            },
            "first": true,
            "empty": false
        }
```


Creación de enlace de pago

Para la creación de enlace de pago se envía una estructura de datos similar a la de creación de transacciones. Cabe resaltar que por cada compra se debe generar un enlace de pago.

Parámetro	Descripción	Obligatorio	Tipo
billingDocument	Número de documento del cliente. Alfanumérico. ReGex: ^[\\dA-Za-z-]+$	Si	String
billingDocumentType	Tipo de Documento del cliente: - CC: Cédula de ciudadanía - NI: Nit - PASSPORT: Pasaporte - TI: Tarjeta de Identidad - CE: Cédula Extranjería - RCN: Registro Civil de Nacimiento - TI: Tarjeta de Identidad - TE: Tarjeta de Extranjería - Foreign Id: Documento de identificación Extranjero - Foreign commerce Id: Identificación Comercio Extranjero	Si	String
branchName	Descripción del comercio definida para desplegar en pantalla		String
cancelCallbackUrl	URL para redirect del callback en caso de transacción cancelada	Si	String
client	Valor por defecto para ambiente de pruebas: “supergiros”. Para paso a producción este valor cambia.	Si	String
countryIsoCode	Valor por defecto: Colombia = COL	Si	String
currency	Id de la moneda de la transacción (COP = 170)	Si	String
customerAddress	Dirección de cliente que realiza la compra. Se permiten espacios y símbolos: °#,/-	Si	String
customerEmail	e-mail del cliente que realiza la compra. ReGex: ^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$	Si	String
customerName	Nombre del cliente que realiza la compra	Si	String
customerPhone	Número de teléfono del cliente que realiza la compra. ReGex: ^[0-9\\s+]*$	Si	String
enableNiubizSetting	Valor por defecto = false	Si	String
errorCallbackUrl	URL para redirect del callback en caso de transacción errónea	Si	String
externalId	Identificación única generada por el comercio para identificar la transacción. El formato para la generación del UID es el siguiente: • Últimos ocho caracteres del ID del terminal (relleno con 0 a la izquierda si el ID del terminal tiene menos de 8 caracteres separador de guiones / guiones • Valor del contador (relleno con 0 a la izquierda si el valor actual es menor que 99999999999) ***Ejemplo 1:*** ID de terminal: TZXCALJTUXVS Valor contador comercio: 14	Si	String

	UID generado: ALJTUXVS-00000000014 *Ejemplo 2:* ID de terminal: 58963 Valor contador comercio: 25637 UID generado: 00058963-00000025637		
extraTaxAmount	Importe de IAC de la transacción. Incluir este monto en el importe total.	Si	Double
invoiceNumber	ID Factura o de la transacción asignado por el comercio.	Si	String
maxInstallmentCount	Cantidad máxima de cuotas que el comercio desea ofrecer al cliente para pagar. Ejemplo: "1"	Si	String
method	selectOption = se le preguntará al usuario el método de pago card = tarjeta crédito o débito pse = PSE nequi = Nequi	Si	String
mode	• redirect = Valor por defecto • api = cuando se crea la transacción por API. Diferente a Redirect	Si	String
origin	URL de origen desde donde se hace la petición	Si	String
paymentRequiresLogin	Valor por defecto = true		String
paymentShippingAddress	Valor por defecto = false		String
password	Contraseña de la terminal del comercio	Si	String
rejectCallbackUrl	URL para redirect del callback en caso de transacción rechazada	Si	String
successCallbackUrl	URL para redirect del callback en caso de transacción exitosa	Si	String
taxAmount	Importe de IVA de la transacción. Incluir este monto en el importe total.	Si	Double
terminalId	Id de la terminal del comercio.	Si	String
totalAmount	Importe total de la transacción. El punto se usa para los decimales. ReGex: ^\\d+(\\.\\d+)?$	Si	Double
userName	Usuario de la terminal del comercio	Si	String
startsAt (UTC+0)	Fecha desde la que estará vigente el enlace de pago	Si	String
expiresAt (UTC+0)	Fecha hasta la que estará vigente el enlace de pago	No	String
extraData1	Dato extra. 255 caracteres máximo.	No	String
extraData2	Dato extra. 255 caracteres máximo.	No	String
extraData3	Dato extra. 255 caracteres máximo.	No	String

A continuación, se muestra un CURL con los datos a enviar.

```
curl --location 'https://<ambiente_pasarela>.superpay.com.co/api/payments/payment-links' \
--header 'x-api-token: Bearer
        eyJraWQiOiI3NGc1K2taTnNyQWdDcXJ5c2NkTTFieW9Xdmd1MVhZUWxycGg3Q2tnRHYwPSIsImFsZyI6IlJT
        MjU2In0.eyJzdWIiOiI2cHFmbW1jMnRiNHE2YnAxZHRkNmlxN2R2MyIsInRva2VuX3VzZSI6ImFjY2VzcyIs
        InNjb3BlIjoibWVyY2hhbnQtYXBpXC9iZXRwbGF5IiwiYXV0aF90aW1lIjoxNzIzNjY3MjU1LCJpc3MiOiJo
        dHRwczpcL1wvY29nbml0by1pZHAudXMtZWFzdC0yLmFtYXpvbmF3cy5jb21cL3VzLWVhc3QtMl9SNHZFekxG
        QjUiLCJleHAiOjE3MjM2NzA4NTUsImlhdCI6MTcyMzY2NzI1NSwidmVyc2lvbiI6MiwianRpIjoiNDAyZmFh
        YTEtMDZjMi00OTQyLTgzZDktZTQxMzQyNGY4MDZmIiwiY2xpZW50X2lkIjoiNnBxZm1tYzJ0YjRxNmJwMWR0
        ZDZpcTdkdjMifQ.oZVwGQ9hMvKdgoBzkfem2mPZTo5AL9dgSmtDL8Jl8PXD0xj9E9lxX7GqjtHQImaG00Dm0
        vu06mxFnwvPfsZbHYohvuoZJcQPcdtOW-aXJ92EXANoqYgR5uwCFSflibO-
        uJDHunQDXD9AOWkQgsaY5NfeTJiTIqLCNzRk1I8AbxmCiCPiS2pzQfUirr0AVGIESHzQR5ibl2Hms-
        PQSvRr22ptF5xXFRMr5EtdQuReAs0on2i3gXnAhg_mwM4lUViXXx_ahZmaje4iLv6xOdsbJpzAaGp_QMAN3u
        auGhGXA2y_oYowh_R6D2DfAEE1Iu6VwlXadizDY3dqI8m-ZHMuPQ' \
--header 'Content-Type: application/json' \
--data-raw '{
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "cancelCallbackUrl": "https://sbx.superpay.com.co/ws/launcher/response-
fromgateway?status=CANCEL",
    "client": "supergiros",
    "countryIsoCode": "COL",
    "currency": "170",
    "customerAddress": "CL 96A 1 No 14Q - 18",
```

```
    "customerEmail": "test@glic-solutions.com",
    "customerName": "Mathias Suarez",
    "customerPhone": "3568365379",
    "enableNiubizSetting": "false",
    "errorCallbackUrl": "https://sbx.superpay.com.co/ws/launcher/response-
fromgateway?status=ERROR",
    "externalId": "00026080-90000314",
    "extraTaxAmount": 1000.0,
    "origin": "https://sbx.superpay.com.co/ws/launcher/payment-form",
    "paymentShippingAddress": "addressTest",
    "taxAmount": 1000.0,
    "mode": "redirect",
    "method": "pse",
    "terminalId": "26080",
    "totalAmount": 1000.0,
    "branchName": "Tienda 1",
    "maxInstallmentCount": 1,
    "paymentRequiresLogin": "true",
    "rejectCallbackUrl": "https://sbx.superpay.com.co/ws/launcher/response-
fromgateway?status=REJECT",
    "successCallbackUrl": "https://sbx.superpay.com.co/ws/launcher/response-
fromgateway?status=SUCCESS",
    "invoiceNumber": "test-0000",
    "startsAt": "2024-08-20T16:15:17.733611",
    "expiresAt": "2024-08-25T16:15:17.733611",
    "billingCity": "BOGOTA D.C.",
    "extraData1": "dato extra 1",
    "extraData2": "dato extra 2",
    "extraData3": "dato extra 3"
}'
```


Ejemplo respuesta:


```
{
    "id": "cf865433-45f2-4a3f-8068-2e171656170b",
    "billingDocument": "123456789",
    "billingDocumentType": "CC",
    "branchName": "Tienda 1",
    "cancelCallbackUrl": "https://sbx.superpay.com.co/ws/launcher/response-
fromgateway?status=CANCEL",
    "client": "supergiros",
    "code": "fb07dc8d96714d5ea820f8132b597838",
    "countryIsoCode": "COL",
    "currency": "170",
    "customerAddress": "CL 96A 1 No 14Q - 18",
    "customerEmail": "test@glic-solutions.com",
    "customerName": "Mathias Suarez",
    "customerPhone": "3568365379",
    "createdAt": "2024-08-21T14:08:23.2336958",
    "enableNiubizSetting": "false",
    "errorCallbackUrl": "https://sbx.superpay.com.co/ws/launcher/response-
fromgateway?status=ERROR",
    "externalId": "00026080-90000312",
    "extraTaxAmount": "1000",
    "invoiceNumber": "test-0000",
    "maxInstallmentCount": "1",
    "method": "pse",
    "mode": "redirect",
    "origin": "https://sbx.superpay.com.co/ws/launcher/payment-form",
    "paymentRequiresLogin": "true",
    "paymentShippingAddress": "addressTest",
    "rejectCallbackUrl": "https://sbx.superpay.com.co/ws/launcher/response-
fromgateway?status=REJECT",
```

```
    "successCallbackUrl": "https://sbx.superpay.com.co/ws/launcher/response-
fromgateway?status=SUCCESS",
    "taxAmount": "1000",
    "terminalId": "26080",
    "totalAmount": "1000",
    "link": "https:// <ambiente_pasarela>.superpay.com.co/ws/router/payment-
link/528f63792840455da92da6cf3c42209c",
    "startsAt": "2024-08-20T16:15:17.733611",
    "expiresAt": "2024-08-25T16:15:17.733611",
    "billingCity": "BOGOTA D.C.",
    "extraData1": "dato extra 1",
    "extraData2": "dato extra 2",
    "extraData3": "dato extra 3"
}
```


- En caso de que el enlace de pago haya expirado, se obtiene un error con código Http 400 Bad Request con el siguiente body:


```
"code": "PAYMENTS_ERROR_000",
"message": "Payment link(s) data is not valid"
```

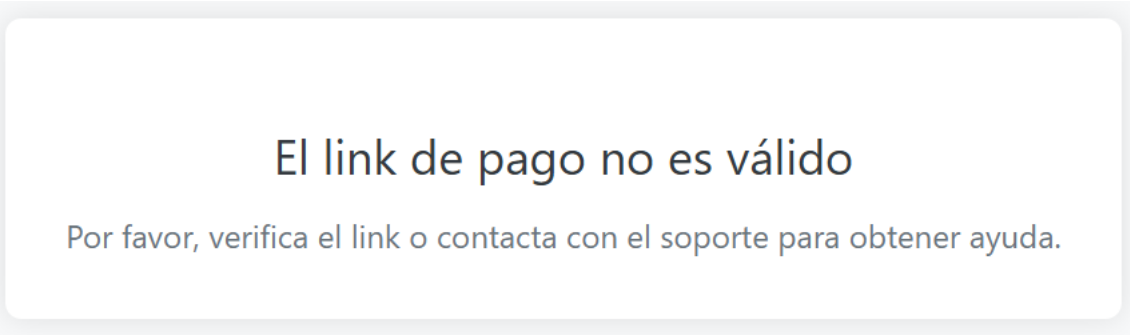

- En caso de que el externalId ya exista se obtiene un error con código Http 400 Bad Request con el siguiente body:


```
"code": "PAYMENTS_ERROR_000",
"message": " Payment link with similar data or external id already exists"
```


Uso de enlace de pago

En el body de respuesta en el apartado “Creación del enlace de pago” se debe tomar el Link generado para continuar con el proceso de pago. Luego se debe abrir en el navegador donde será redirigido a la Payment Page de acuerdo con el método de pago enviado.

En caso de que el enlace de pago se use en una fecha posterior al vencimiento, el usuario pagador verá el siguiente mensaje.



Si el enlace de pago se usa durante su vigencia, el usuario pagador visualizará la Payment Page, con el método de pago enviado en la creación de la transacción y los datos del usuario.

Método de pago PSE Método de pago Tarjeta Método de pago Nequi

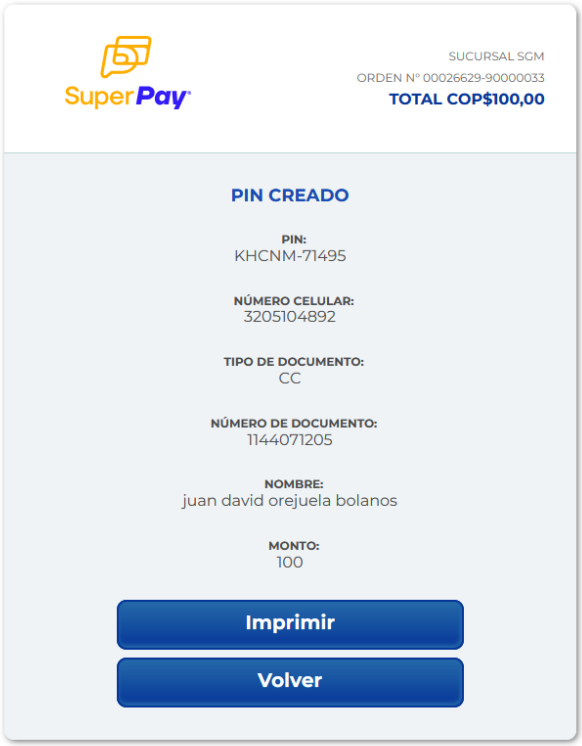


En caso de que se envíe en el method el valor selectOption, el usuario pagador podrá seleccionar el método de pago en la Payment Page.



Creación de pin de pago mediante integración REDIRECT

Para crear un pago mediante PIN el comercio debe integrarse a la pasarela siguiendo los pasos del apartado “Flujo de la información en Modo Redirect – Creación del pago”. Al crear un pago con método de pago igual a “pin” el cliente verá en la interfaz la siguiente información:



Confirmación de pago del PIN

Para la confirmación del pago se debe tener el token de autorización que se explica en el apartado “Login” – “Login por comercio – Para reportería, enlaces y pines de pago”. A continuación, se muestra el CURL para la confirmación del pago:

```
curl --location 'https://dev.superpay.com.co/api/payments/payment-pins/confirm-pin' \
--header 'x-api-token: Bearer
eyJraWQiOiI3NGc1K2taTnNyQWdDcXJ5c2NkTTFieW9Xdmd1MVhZUWxycGg3Q2tnRHYwPSIsImFsZyI6IlJTMjU2In0.
eyJzdWIiOiI1cjgxN3IyYXNjdTBqazFqZGltaG9jczB2NCIsInRva2VuX3VzZSI6ImFjY2VzcyIsInNjb3BlIjoibWVy
Y2hhbnQtYXBpXC9ibHVlbGluayIsImF1dGhfdGltZSI6MTcyNjAwNTcxNSwiaXNzIjoiaHR0cHM6XC9cL2NvZ25pdG8t
aWRwLnVzLWVhc3QtMi5hbWF6b25hd3MuY29tXC91cy1lYXN0LTJfUjR2RXpMRkI1IiwiZXhwIjoxNzI2MDA5MzE1LCJp
YXQiOjE3MjYwMDU3MTUsInZlcnNpb24iOjIsImp0aSI6IjkzNTU5OTU1LThiNWUtNDJiMy1hZDM5LTc2MDFkNTNkZWY1
OSIsImNsaWVudF9pZCI6IjVyODE3cjJhc2N1MGprMWpkaW1ob2NzMHY0In0.O6fQIzGlso9eEtx8-
GQd0Mb0mQIxDwjae3dZF5_G3aQsV830f0TrF1JS6W-
```

```
lphzijTWFDNYT_hm2KSyyd5PCAZ4zzNvknA5x3SfWuzAtt12H3VUSoQ1GuxwJPeUdxa3cQoU_bO4odNgnt353wtw-
89mrlvKc3NgwWDRkjNRUnRKpaYNIDb8ieucCdeheuKNGSI2xp6AnyuD_fyeFeS_7ql6JOskQrCFVWQav7L_43ylXxOu9
-DMC0lJqlDceF98XuMHu-mKOZF_ceLU7cAzgt1Yi0I44zytEZHA2ZMUy4G1_K_6ivBoVIBwklO5KpnqGZ-
JaKn3uhUaDapYzlYwDzA' \
--header 'Content-Type: application/json' \
--data '{
    "pin": "KHCNM-71495",
    "totalAmount": 100.0,
    "billingDocumentType": "cc",
    "billingDocument": "1144071205"
}'
```

Los campos se explican a continuación:

Parámetro	Descripción	Obligatorio	Tipo
billingDocument	Número de documento del cliente. Alfanumérico. ReGex: ^[\\dA-Za-z-]+$	Si	String
billingDocumentType	Tipo de Documento del cliente: - cc: Cédula de ciudadanía - ni: Nit - passport: Pasaporte - ti: Tarjeta de Identidad - ce: Cédula Extranjería - rcn: Registro Civil de Nacimiento - ti: Tarjeta de Identidad - te: Tarjeta de Extranjería - foreign_id: Documento de identificación Extranjero - foreign_commerce_id: Identificación Comercio Extranjero	Si	String
pin	Código del pin generado.	Si	String
totalAmount	Total del monto pagado, que debe corresponder con el monto total de la transacción original	Si	Double

En la respuesta se recibe la misma información de la transacción adicional, adicionando que el campo “trxStatus” indica el estado de pago del PIN como se puede ver a continuación:

```
{
    "billingCity": null,
    "billingDocument": "1144071205",
    "billingDocumentType": "CC",
    "branchId": null,
    "branchName": "Sucursal SGM",
    "cancelCallbackUrl": "https://supergirosmovil.com.co/",
    "client": "supergiros",
    "countryIsoCode": "COL",
    "createdAt": "2024-09-12T19:16:58.572343",
    "currency": "170",
    "customerAddress": "Calle 123 # 456 - 789",
    "customerEmail": "juan.orejuela@hotmail.com",
    "customerName": "juan david orejuela bolanos",
    "customerPhone": "3205104892",
    "enableNiubizSetting": "false",
    "errorCallbackUrl": "https://supergirosmovil.com.co/",
    "expiresAt": null,
    "externalId": "00026629-90000033",
    "extraData1": "extra-data-1",
    "extraData2": "extra-data-2",
```

```
    "extraData3": "extra-data-3",
    "extraTaxAmount": "0",
    "id": "f70956b5-09c4-476d-9896-a98ed8555185",
    "invoiceNumber": "12074649",
    "isValid": true,
    "maxInstallmentCount": "12",
    "method": "pin",
    "mode": "redirect",
    "origin": "https://supergirosmovil.com.co/",
    "paymentRequiresLogin": "false",
    "paymentShippingAddress": "Calle 123 # 456 - 789",
    "pin": "KHCNM-71495",
    "reasons": [],
    "rejectCallbackUrl": "https://supergirosmovil.com.co/",
    "startsAt": null,
    "successCallbackUrl": "https://supergirosmovil.com.co/",
    "taxAmount": "0",
    "terminalId": "26629",
    "totalAmount": "100",
    "trxStatus": "APPROVED"
}
```


Los estados para el campo “trxStatus” son:

- PENDING: La transacción del PIN aún no se ha pagado y está pendiente.
- APPROVED: La transacción del PIN se pagó.
- DECLINED: La transacción del PIN quedó declinada por validaciones.
- ERROR: La transacción del PIN presentó alguna falla.
- EXPIRED: La transacción del PIN expiró sin pagarse.

Para el caso de los estados DECLINED o ERROR si el error es en la confirmación del pago del PIN por los valores enviados se obtiene una respuesta con código 400 Bad Request y el mensaje de la validación efectuada. Los mensajes de error que puede obtener son:

- El tipo o número de documento no coinciden con los de la transacción original:


```
"code": "PAYMENTS_ERROR_002",
"message": "El documento de identidad de la confirmación del pin de pago no coincide con el de la transacción original"
```


- El monto no coincide:


```
"code": "PAYMENTS_ERROR_002",
"message": "El monto total de la confirmación del pin de pago no coincide con el de la transacción original"
```


- El pin no existe:


```
"code": "PAYMENTS_ERROR_000",
"message": "Payment pin not found"
```


- El pin está expirado:


```
"code": "PAYMENTS_ERROR_000",
"message": "Payment pin expired"
```

Consulta de información del PIN de pago

El PIN de pago está relacionado con una transacción con external id, en caso tal que se quiera consultar el estado de la transacción con el external id se pueden seguir los pasos explicados en el apartado “Consulta de estado de la transacción” o en su defecto se puede hacer uso del CURL que se expone a continuación (Adicionando el token obtenido haciendo el login por merchant explicado en “Login” – “Login por comercio – Para reportería, enlaces y pines de pago”). En la URL se debe agregar el código del PIN de pago como se muestra a continuación:

```
curl --location 'https://dev.superpay.com.co/api/payments/payment-pins/get-by/pin/KHCNM-
71495' \
--header 'x-api-token: Bearer
eyJraWQiOiI3NGc1K2taTnNyQWdDcXJ5c2NkTTFieW9Xdmd1MVhZUWxycGg3Q2tnRHYwPSIsImFsZyI6IlJTMjU2In0.
eyJzdWIiOiI1cjgxN3IyYXNjdTBqazFqZGltaG9jczB2NCIsInRva2VuX3VzZSI6ImFjY2VzcyIsInNjb3BlIjoibWVy
Y2hhbnQtYXBpXC9ibHVlbGluayIsImF1dGhfdGltZSI6MTcyNjAwNTcxNSwiaXNzIjoiaHR0cHM6XC9cL2NvZ25pdG8t
aWRwLnVzLWVhc3QtMi5hbWF6b25hd3MuY29tXC91cy1lYXN0LTJfUjR2RXpMRkI1IiwiZXhwIjoxNzI2MDA5MzE1LCJp
YXQiOjE3MjYwMDU3MTUsInZlcnNpb24iOjIsImp0aSI6IjkzNTU5OTU1LThiNWUtNDJiMy1hZDM5LTc2MDFkNTNkZWY1
OSIsImNsaWVudF9pZCI6IjVyODE3cjJhc2N1MGprMWpkaW1ob2NzMHY0In0.O6fQIzGlso9eEtx8-
GQd0Mb0mQIxDwjae3dZF5_G3aQsV830f0TrF1JS6W-
lphzijTWFDNYT_hm2KSyyd5PCAZ4zzNvknA5x3SfWuzAtt12H3VUSoQ1GuxwJPeUdxa3cQoU_bO4odNgnt353wtw-
89mrlvKc3NgwWDRkjNRUnRKpaYNIDb8ieucCdeheuKNGSI2xp6AnyuD_fyeFeS_7ql6JOskQrCFVWQav7L_43ylXxOu9
-DMC0lJqlDceF98XuMHu-mKOZF_ceLU7cAzgt1Yi0I44zytEZHA2ZMUy4G1_K_6ivBoVIBwklO5KpnqGZ-
JaKn3uhUaDapYzlYwDzA' \
--data ''
```

La respuesta es la información de la transacción junto con el campo “trxStatus” del cual en el anterior apartado ya se explicaron sus valores:


```
{
    "billingCity": null,
    "billingDocument": "1144071205",
    "billingDocumentType": "CC",
    "branchId": null,
    "branchName": "Sucursal SGM",
    "cancelCallbackUrl": "https://supergirosmovil.com.co/",
    "client": "supergiros",
    "countryIsoCode": "COL",
    "createdAt": "2024-09-12T19:16:58.572343",
    "currency": "170",
    "customerAddress": "Calle 123 # 456 - 789",
    "customerEmail": "juan.orejuela@hotmail.com",
    "customerName": "juan david orejuela bolanos",
    "customerPhone": "3205104892",
    "enableNiubizSetting": "false",
    "errorCallbackUrl": "https://supergirosmovil.com.co/",
    "expiresAt": null,
    "externalId": "00026629-90000033",
    "extraData1": "extra-data-1",
    "extraData2": "extra-data-2",
    "extraData3": "extra-data-3",
    "extraTaxAmount": "0",
    "id": "f70956b5-09c4-476d-9896-a98ed8555185",
    "invoiceNumber": "12074649",
    "isValid": true,
    "maxInstallmentCount": "12",
    "method": "pin",
    "mode": "redirect",
```

```
    "origin": "https://supergirosmovil.com.co/",
    "paymentRequiresLogin": "false",
    "paymentShippingAddress": "Calle 123 # 456 - 789",
    "pin": "KHCNM-71495",
    "reasons": [],
    "rejectCallbackUrl": "https://supergirosmovil.com.co/",
    "startsAt": null,
    "successCallbackUrl": "https://supergirosmovil.com.co/",
    "taxAmount": "0",
    "terminalId": "26629",
    "totalAmount": "100",
    "trxStatus": "APPROVED"
}
```


- En caso de que el PIN no exista se obtiene un error con código Http 400 Bad Request con el siguiente body:


```
"code": "PAYMENTS_ERROR_000",
"message": "Payment pin not found"
```


Webhooks

Webhook de notificación de cambio de estado de transacciones

Este webhook se encarga de enviar una solicitud POST a una URL proporcionada por el cliente para notificar cambios en el estado de una transacción. La notificación incluye algunos detalles de la transacción y el estado actualizado. El payload se envía en formato JSON e incluye la siguiente estructura:

```
curl --location 'https://url-comercio' \
--header 'Content-Type: application/json' \
--data '{
    "externalId": "00026624-80000031",
    "invoiceNumber": "12074649",
    "totalAmount": 100.0,
    "trxId": "6696a030-a2e9-439c-b8de-d6987a46f364",
    "trxStatus": "APPROVED",
    "trxType": "SALE",
    "bankInfo": "1022 - BANCO UNION COLOMBIANO",
    "selectedPaymentMethod": "pse",
    "signature": "8d3d0a41acf460a0d685"
}'
```

Descripción de campos:

- **externalId:** Consecutivo asignado por el comercio para identificar la transacción desde su plataforma como también desde el lado de la pasarela.
- **invoiceNumber:** Número de factura asignado por el comercio.
- **totalAmount:** Monto total de la transacción
- **trxId:** Identificador de la transacción en pasarela.
- **trxStatus:** Estado de pago de la transacción.

- **trxType:** Tipo de transacción.
- **selectedPaymentMethod:** Indica el método de pago seleccionado para la transacción. Valores: pse, card, nequi.
- **bankInfo:** Si el método de pago seleccionado fue PSE este campo muestra el banco con el cual el usuario realizó el pago.
- **signature:** Campo en el cual se muestra la firma para que el comercio pueda asegurarse de que el mensaje proviene de la pasarela. La explicación de cómo descifrar este campo se amplía en el siguiente punto “Seguridad”.

Respuesta Esperada del Cliente

El cliente debe responder con un código HTTP para confirmar la recepción del webhook:

- **200 OK / 201 Created**: La notificación se recibió o se creó correctamente.
- **4xx/5xx**: Si el servidor del cliente responde con un error, nuestro sistema intentará reenviar la notificación nuevamente.

Seguridad

- **HTTPS**: Requerimos que la URL del cliente utilice HTTPS para proteger la información transmitida.
- **Firma (Campo signature en mensaje webhook):** A continuación, se anexa un ejemplo de un valor de este campo:

- 8d3d0a41acf460a0d685191b6c1bda9c7337921fb64031bb3130d8a83d0742d7274f98e46d cccee3a5b7d6ac1ac74d90ac181a5f00e3fad8261a9c7a3903ed7e207844fdcbbcbb6fa51899 d284d497f6de2237c279e81476cd0bd1c2faaa10fbf9b717ce71c14346003dcbbc52851de48 ff99c0b46b87e663560b2d6e42a37c33f668a5e03f4179452c00c7867b777f257d6957b90f4 ea84261894763ff757c13a0a810e375599c359b0f0a183105272d3f02411866dcb06c96a399 f9dbb4790f542af71c678527db40dc1c7687b1d70a99484a03f45d8e754b13f23bae12082ec 930e49d2b4976a427a163a964fdf712e9978cf68506864ccec7f401514aab7

Este es una firma generada por una llave privada RSA que posee la pasarela, es decir que para que el comercio pueda descrifrarlo y verificar que el webhook proviene de la pasarela el comercio debe poseer la llave pública de la pasarela que se brindará por alguno de los medios de contacto acordados y la cadena que viene firmada. A continuación, se anexa un ejemplo en JAVA para descifrar la firma:

```
import java.nio.file.Files;
import java.nio.file.Paths;
import java.security.KeyFactory;
import java.security.PublicKey;
import java.security.Signature;
import java.security.spec.X509EncodedKeySpec;
import java.util.Base64;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
@SpringBootApplication
public class SignatureExampleApplication {

        public static void main(String[] args) {
        SpringApplication.run(SignatureExampleApplication.class, args);

        String signedData = "externalId=00026080-90000002&totalAmount=100.0&trxStatus=APPROVED";
        String signatureBase64Hex = "signature_received";
        String publicKeyFilePath = "public_key_gateway.pem";

        boolean isValid = isSignatureValid(signedData, signatureBase64Hex, publicKeyFilePath);
        System.out.println("¿Es válida la firma? " + isValid);
    }

    public static boolean isSignatureValid(String signedData, String signatureBase64Hex, String
publicKeyFilePath) {
        try {
            String publicKeyContent = new String(Files.readAllBytes(Paths.get(publicKeyFilePath)))
                    .replace("-----BEGIN PUBLIC KEY-----", "")
                    .replace("-----END PUBLIC KEY-----", "")
                    .replaceAll("\\s+", "");

            byte[] publicKeyBytes = Base64.getDecoder().decode(publicKeyContent);
            X509EncodedKeySpec keySpec = new X509EncodedKeySpec(publicKeyBytes);
            KeyFactory keyFactory = KeyFactory.getInstance("RSA");
            PublicKey publicKey = keyFactory.generatePublic(keySpec);

            Signature signature = Signature.getInstance("SHA512withRSA");
            signature.initVerify(publicKey);
            signature.update(signedData.getBytes("UTF-8"));

            byte[] signatureBytes = hexStringToByteArray(signatureBase64Hex);
            return signature.verify(signatureBytes);
        } catch (Exception e) {
            e.printStackTrace();
            return false;
        }
    }

    public static byte[] hexStringToByteArray(String hex) {
        int len = hex.length();
        byte[] data = new byte[len / 2];
        for (int i = 0; i < len; i += 2) {
            data[i / 2] = (byte) ((Character.digit(hex.charAt(i), 16) << 4)
                                 + Character.digit(hex.charAt(i + 1), 16));
        }
        return data;
    }
}
```

Nótese que el campo signedData indica la cadena que se firma que para este ejemplo es “externalId=00026080-90000002&totalAmount=100.0&trxStatus=APPROVED”. El orden de los campos es alfabetico, en formato clave=valor y concatenados por el signo “&”. Los campos externalId, totalAmount y trxStatus son los únicos campos que estarán incluidos en la firma ya que son los campos que, al momento de generar esta versión del documento, se consideran como importantes para el comercio en cuanto a la actualización del estado de la transacción.