

Manual t cnico â PagosDigitales

1. Prop sito

Este manual t cnico describe la operaci n interna del servicio PagosDigitales, su arquitectura, configuraci n, integraci n con proveedores y gu as de despliegue y operaci n.

2. Alcance

-    Aplicaci n Spring Boot 3.x con Apache Camel.
-    API REST para pagos, bancos y notificaciones.
-    Confirmaciones SafetyPay con firma SHA-256.
-    Configuraci n por proveedor en 'application.yaml'.

3. Arquitectura

Estilo: monolito en capas (API, Application, Domain, Infrastructure) con integraci n din mica por proveedor.

Componentes clave:

-    **Controllers:** expone endpoints REST.
-    **Mappers:** transforman requests/responses hacia el formato de cada proveedor.
-    **Servicios de dominio:** firma y resoluci n de proveedor.
-    **Camel Routes:** enrutan solicitudes hacia proveedores externos.

4. Estructura de paquetes

-    'com.femsa.gpf.pagosdigitales.api': controladores y DTOs.
-    'com.femsa.gpf.pagosdigitales.application': orquestaci n y mapeos.
-    'com.femsa.gpf.pagosdigitales.domain': l gica de negocio.
-    'com.femsa.gpf.pagosdigitales.infrastructure': rutas Camel y configuraci n.

5. Endpoints

M todo | Ruta | Descripci n | Content-Type | Respuesta

POST | '/api/v1/banks' | Consulta bancos por proveedor o todos | JSON | JSON

GET | '/api/v1/payments' | Consulta pagos por 'operation_id' | JSON (body) | JSON

POST | '/api/v1/direct-online-payment-requests' | Solicitud de pago en l nea | JSON | JSON

POST | '/api/v1/payments/notifications/merchant-events' | Notificaciones de eventos del comercio | JSON | JSON

POST | '/api/v1/safetypay/confirmation' | Webhook SafetyPay (CSV firmado) | 'application/x-www-form-urlencoded' | 'text/plain'

GET | '/api/v1/pagos/test' | Health check |    | text/plain

6. Flujo de integraci n

6.1 Pagos en línea

1. Controller valida proveedor y solicitud.
2. Mapper crea payload del proveedor según mapping.
3. Camel Route arma URL y headers dinámicamente.
4. Respuesta normalizada hacia DTO interno.

6.2 Consulta de pagos

1. Controller valida 'payment_provider_code' y 'operation_id'.
2. Camel Route agrega parámetros de consulta.
3. Respuesta normalizada a DTO interno.

6.3 Consulta de bancos

1. Controller consulta proveedor específico o todos.
2. Camel Route construye URL con parámetros.
3. Respuesta normalizada por mapping.

6.4 Merchant Events

1. Controller valida proveedor.
2. Mapper transforma payload.
3. Camel Route envía notificación.

6.5 SafetyPay Confirmation

1. Controller mapea form-urlencoded a DTO.
2. Service valida API Key, firma e IP permitida.
3. Aplica idempotencia y registra notificación.
4. Responde CSV firmado.

7. Configuración

Ubicación: 'src/main/resources/application.yaml'

7.1 Base

```
â€ ‘server.port’  
â€ ‘spring.datasource.*’  
â€ ‘spring.jpa.*’
```

7.2 Proveedores por endpoint

```
â€ ‘direct-online-payment-requests.providers.*’  
â€ ‘merchant-events.providers.*’  
â€ ‘payments.providers.*’  
â€ ‘getbanks.providers.*’
```

7.3 Mappings

```
â€ ‘direct-online-payment-requests.mapping.*’  
â€ ‘merchant-events.mapping.*’  
â€ ‘payments.mapping.*’  
â€ ‘getbanks.mapping.*’
```

7.4 Códigos y errores

• 'providers-pay.codes'

• 'error-mapping.providers.*'

7.5 SafetyPay

• 'safetypay.confirmation.enabled'

• 'safetypay.confirmation.providers.*'

8. Seguridad

• Validación de firma SHA-256 y API Key en SafetyPay.

• Lista opcional de IPs permitidas para confirmaciones.

• Headers 'X-API-Key' y 'X-Version' por proveedor configurados en YAML.

9. Despliegue

9.1 Requisitos

• Java 17

• Maven

• Oracle JDBC accesible

9.2 Build

```
mvn clean package
```

9.3 Docker (referencial)

```
FROM eclipse-temurin:17-jre
WORKDIR /app
COPY target/pagosdigitales-0.0.1-SNAPSHOT.jar app.jar
EXPOSE 8080
ENTRYPOINT ["java", "-jar", "app.jar"]
```

10. Operación y monitoreo

• Revisar logs de Camel para trazabilidad de requests/response.

• Validar métricas de latencia por endpoint.

• Ajustar timeouts por proveedor según SLA.

11. Directorio de términos

• **API**: Interfaz de programación que expone los endpoints del servicio.

• **Camel Route**: Ruta de Apache Camel que orquesta llamadas a proveedores externos.

• **DTO**: Objeto de transferencia de datos usado en requests/responses.

• **Idempotencia**: Capacidad de procesar la misma notificación sin duplicar efectos.

• **Provider**: Proveedor externo de pagos configurado en 'application.yaml'.

• **SafetyPay Confirmation**: Webhook firmado que confirma el estado de un pago.

• **Signature**: Firma criptográfica SHA-256 usada para validar autenticidad.

12. Ejemplos de uso

11.1 SafetyPay Confirmation

```
curl -X POST http://localhost:8080/api/v1/safetypay/confirmation \
-H "Content-Type: application/x-www-form-urlencoded" \
--data-urlencode "ApiKey=MI_API_KEY" \
--data-urlencode "payment_provider_code=235689" \
--data-urlencode "RequestDateTime=2020-10-20T12:27:27" \
--data-urlencode "MerchantSalesID=Pruebal" \
--data-urlencode "ReferenceNo=Ref1" \
--data-urlencode "CreationDateTime=2020-10-20T12:27:27" \
--data-urlencode "Amount=50.00" \
--data-urlencode "CurrencyID=USD" \
--data-urlencode "PaymentReferenceNo=606973" \
--data-urlencode "Status=102" \
--data-urlencode "Signature=FIRMA_CALCULADA"
```

11.2 Direct Online Payment Requests

```
curl -X POST http://localhost:8080/api/v1/direct-online-payment-requests \
-H "Content-Type: application/json" \
-d '{
  "chain": 1,
  "store": 148,
  "store_name": "FYBECA AMAZONAS",
  "pos": 90,
  "payment_provider_code": 235689,
  "sales_amount": { "value": 50.00, "currency_code": "USD" },
  "country_code": "EC",
  "bank_id": "0123",
  "merchant_set_pay_amount": true,
  "expiration_time_minutes": 30,
  "language_code": "ES",
  "custom_merchant_name": "TIENDA",
  "merchant_sales_id": "Pruebal",
  "requested_payment_type": "paycash",
  "transaction_email": "cliente@correo.com",
  "send_email_shopper": true
}'
```