

1. Servicios y responsabilidades

1.1a Payments API

- **Responsabilidad:**
 - Entrada pública para crear pagos.
 - Válida request
 - Forward del idempotent_key al service
 - Devuelve la información indispensable que necesita la app
- **Dependencias:** Payment Service
- **Patrón:** Puerta de entrada HTTP

1.1b Refunds API

- **Responsabilidad:**
 - Entrada pública para crear reembolsos.
 - Válida request
 - Forward del idempotent_key al service
 - Devuelve la información indispensable que necesita la app
- **Dependencias:** Refunds Service
- **Patrón:** Puerta de entrada HTTP

1.2 Payment Service

- **Responsabilidad:**
 - Dueño de la entidad Payment
 - Asegura idempotencia
 - Genera identificador único de pago.
 - Gestiona estados (PENDING, COMPLETED, FAILED).
 - Genera sus métricas.
- **Produce Eventos:** `payment.created`, `payment.completed`, `payment.failed`
- **Consume Eventos:** `orchestrator.payment - update_status - completed/failed`
- **Dependencias:** Event Bus, Metrics Service.
- **Patrón:** Event-driven, publica eventos de estado hacia Orchestrator y Metrics Service.

1.3 Refund Service

- **Responsabilidad:**
 - Dueño de entidad Refund
 - Asegura idempotencia
 - Genera identificador único de reembolso.

- Gestiona sus estados (PENDING, COMPLETED, FAILED).
 - Genera sus métricas.
- **Produce Eventos:** `refund.created`, `refund.completed`, `refund.failed`
- **Consume Eventos:** `orchestrator.refund - update_status - completed/failed`
-
- **Dependencias:** Event Bus, Metrics Service.
- **Patrón:** Event-driven, orquestado por Orchestrator.

1.4 Wallet Service

- **Responsabilidad:**
 - Maneja fondos de usuario.
 - Opera Freeze Funds (bloqueo temporal),
 - Debit Funds (cobro definitivo)
 - Release Funds (rollback).
 - Genera métricas en caso de error.
- **Produce Eventos:**
 - `wallet`
 - `.hold_funds`
 - `.debit`
 - `.credit`
 - `.release_funds`
- **Consume eventos:**
 - `orchestrator.wallet`
 - `hold_funds`
 - `release_funds`
 - `debit`
 - `credit`
- **Dependencias:** Event Bus, Metrics Service.
- **Patrón:** Event-driven, responde a comandos del Orchestrator.

1.5 Orchestrator Service

- **Responsabilidad:**
 - Coordina el flujo de pago y reembolso
 - Manejo errores y compensaciones.
 - No produce métricas de negocio.
- **Consume Eventos:**
 - `payment.created/completed/failed`
 - `refund.created/completed/failed`
 - `gateway.authorized/refund`
 - `wallet.hold_funds/credit_funds/debit_funds/released_funds` y los `_failures`

- **Produce eventos:**
 - SAGA OPS Wallet topic/event
 - orchestrator.wallet - hold_funds
 - orchestrator.wallet - release_funds
 - orchestrator.wallet - debit
 - orchestrator.wallet - credit
 - SAGA OPS Payment topic/event
 - orchestrator.payment - update_status - completed/failed
 - SAGA OPS Refund topic/event
 - orchestrator.refund - update_status - completed/failed
 - SAGA OPS Payment Gateway topic/event
 - orchestrator.gateway - authorized/unauthorize/refund
 - Notification to final users topic/event
 - orchestrator.notify - payment_success/payment_failure
 - orchestrator.notify - refund_failure/refund_success
- **Dependencias:** Todos los servicios
- **Patrón:** Event-driven / saga pattern.

1.6 Payment Gateway

- **Responsabilidad:**
 - Se comunica con los proveedores de pago externos
 - Recibe respuesta de los proveedores de manera asincrónica y produce el evento
- **Produce Eventos:** `payment_gateway.response` (authorized/unauthorized/refund)
- **Consume Eventos:** `orchestrator.gateway - authorized/unauthorize/refund`
- **Dependencias:** Orchestrator.
- **Patrón:** Callback asincrónico con los proveedores de pago, Event Bus para notificar al Orchestrator el estado de las autorizaciones de pago.

1.7 Notification Service

- **Responsabilidad:**
 - Notifica estados finales al usuario tanto de pagos como reembolsos (OK/KO).
 - Decide su propia estrategia de comunicación, push, email o web socket.
- **Consume Eventos:** `orchestrator.notify - payment_success/payment_failure/refund_failure/refund_success`
- **Dependencias:** Event Bus.
- **Patrón:** Event-driven.

1.8 Metrics Service

- **Responsabilidad:**
 - Consumir eventos de los servicios
 - Solicitudes ok
 - Solicitudes tasa de errores
 - Latencia
 - Encargado de integrar los datos con el proveedor implementado.
- **Dependencias:** Event Bus.
- **Consume eventos:** `metrics.${domain}.${use_case}`
- **Patrón:** Event-driven.