

cōdearti

Curso

ARQUITECTURA DE MICROSERVICIOS CON SPRING BOOT Y KUBERNETES

Spring Boot, WebFlux, Docker, Open API, Kafka, Prometheus, Elastic Stack, Jaeger, K8s,...

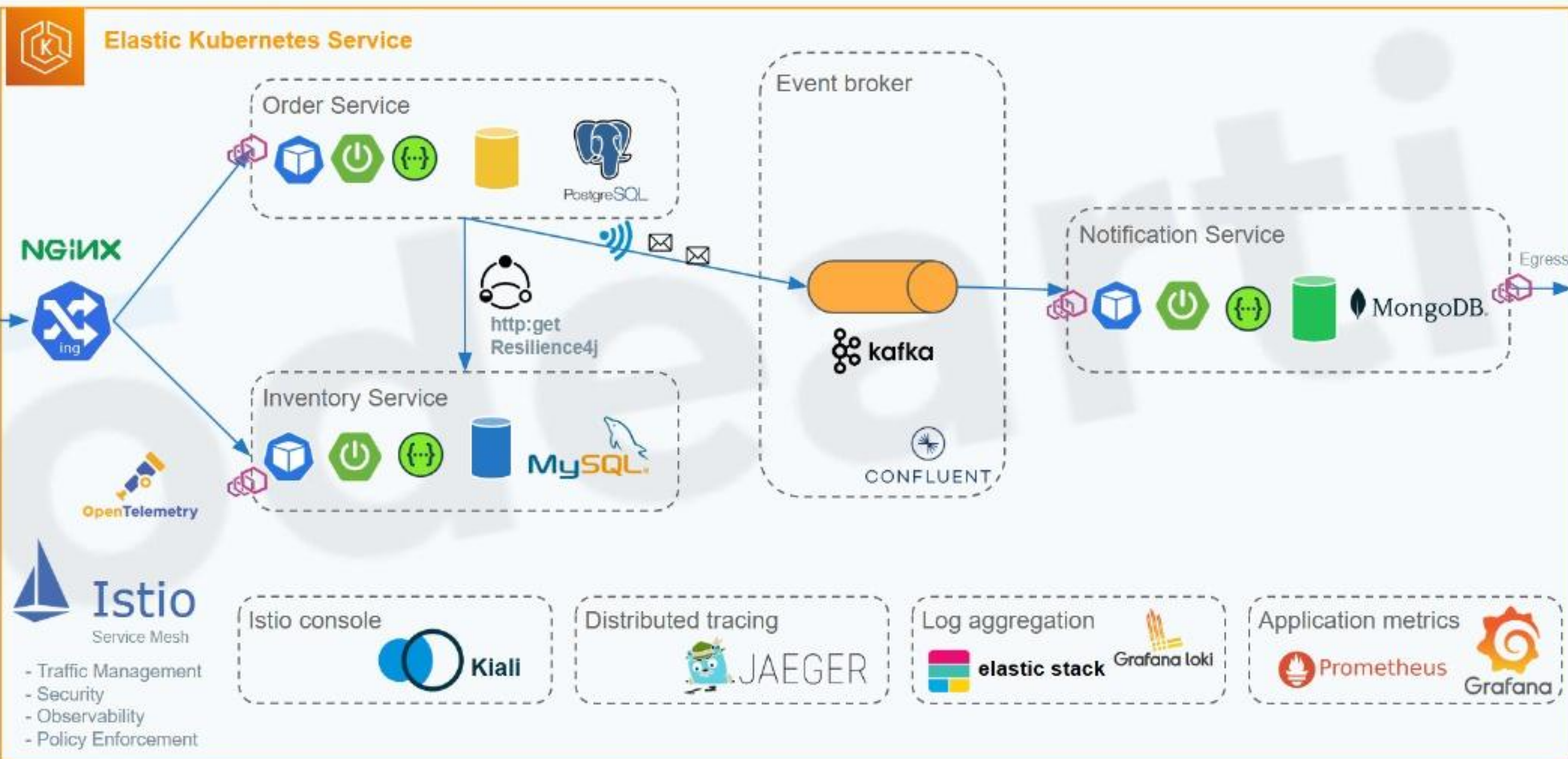
KUBERNETES (K8S)



CURSO DE ARQUITECTURA DE MICROSERVICIOS CON SPRING BOOT Y KUBERNETES



- Deployment
- Service
- Secrets & Config Maps
- Horizontal Pod Autoscaler
- Registry & Discovery
- Load Balancer



¿Qué aprenderemos en esta sesión 2?

Gestión de cambios en base de datos:

- Control de versiones de la base de datos
- Independencia de la base de datos
- Auditoría y seguimiento

Aplicación del Patron de Resiliencia:

- Circuit Breaker
- Retry
- Rate Limiter
- Time Limiter
- Construcción del servicio de ordenes (order-service)
- Despliegue de base de datos sobre docker (PostgreSQL)

Sesión 2

02

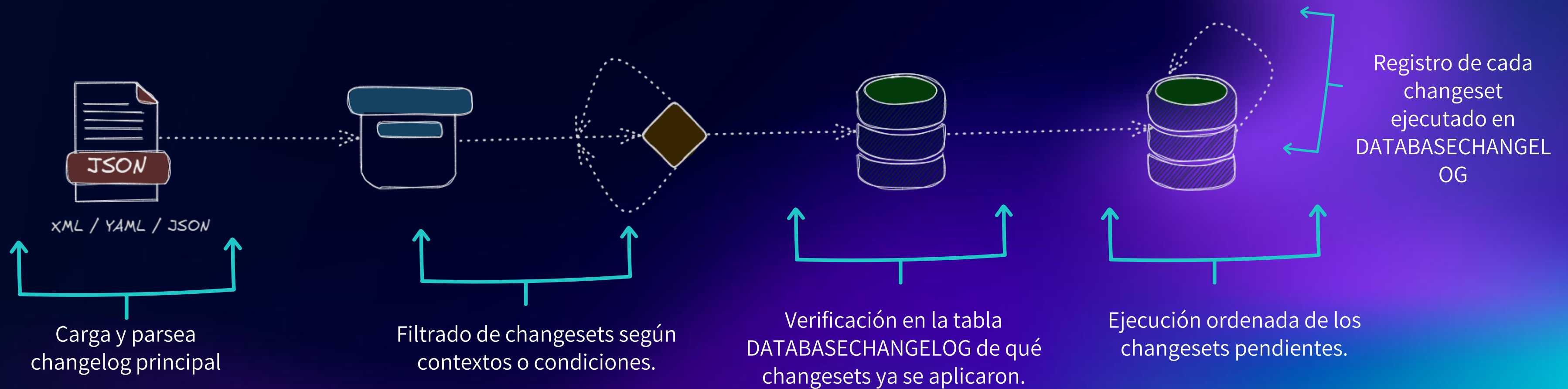
Gestión de Cambios en Bases de Datos

Servicio de E-Commerce

Liquibase



- Control de versiones y la gestión de cambios en bases de datos.
- Liquibase sigue un proceso estructurado para aplicar los scripts (o "changesets") en la base de datos.



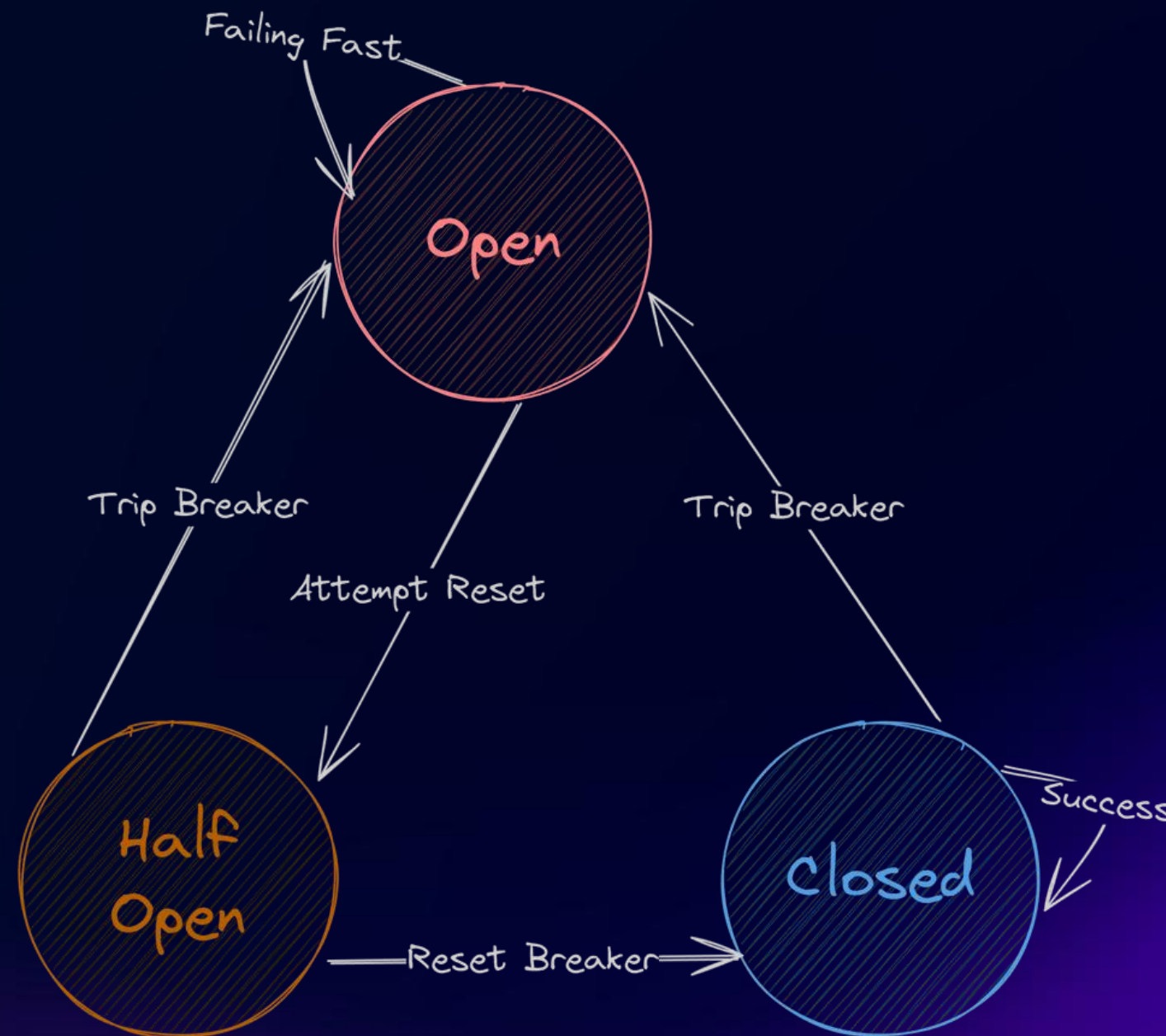
Sesión 2

02

Patrones Resiliencia

Servicio de E-Commerce

Circuit Breaker



- Patrón de diseño para prevenir fallos en cascada en microservicios
- Funciona como un interruptor eléctrico para llamadas entre servicios
- CLOSED (Normal)
 - Permite las llamadas al servicio
 - Monitorea fallos
 - Si fallos > umbral → OPEN
- OPEN (Protección)
 - Bloquea llamadas al servicio
 - Retorna respuesta fallback
 - Después de tiempo → HALF-OPEN
- HALF-OPEN (Prueba)
 - Permite algunas llamadas de prueba
 - Si éxito → CLOSED
 - Si fallo → OPEN

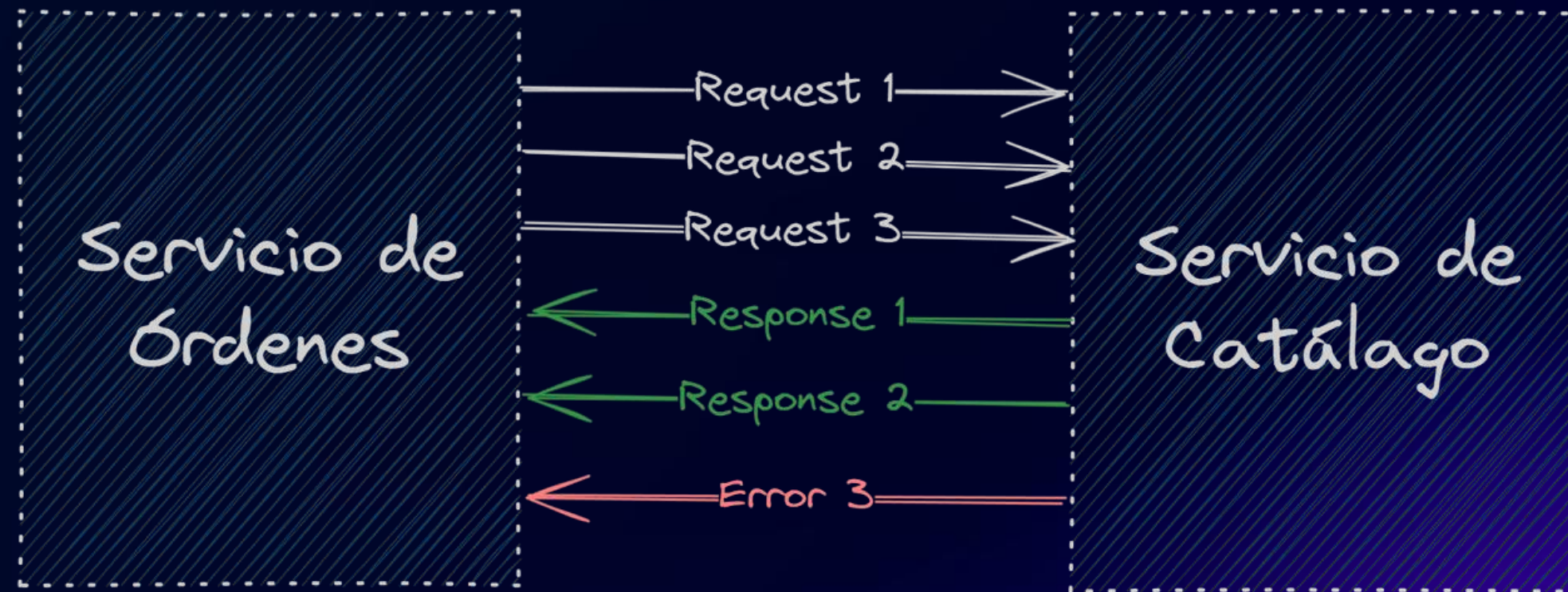
Retry

- Patrón de diseño para manejar fallos temporales en comunicaciones
- Reintenta operaciones fallidas de forma automática y controlada



- Inmediato
 - Reintenta inmediatamente después del fallo
 - Útil para fallos muy breves
- Intervalo Fijo
 - Espera tiempo constante entre reintentos
 - Ejemplo: 2 segundos entre cada intento

Rate Limit



Limita el número de solicitudes que un servicio puede recibir en un período de tiempo específico.

- Proteger servicios contra sobrecarga
- Prevenir ataques DoS
- Gestionar recursos de manera eficiente

Time Limiter



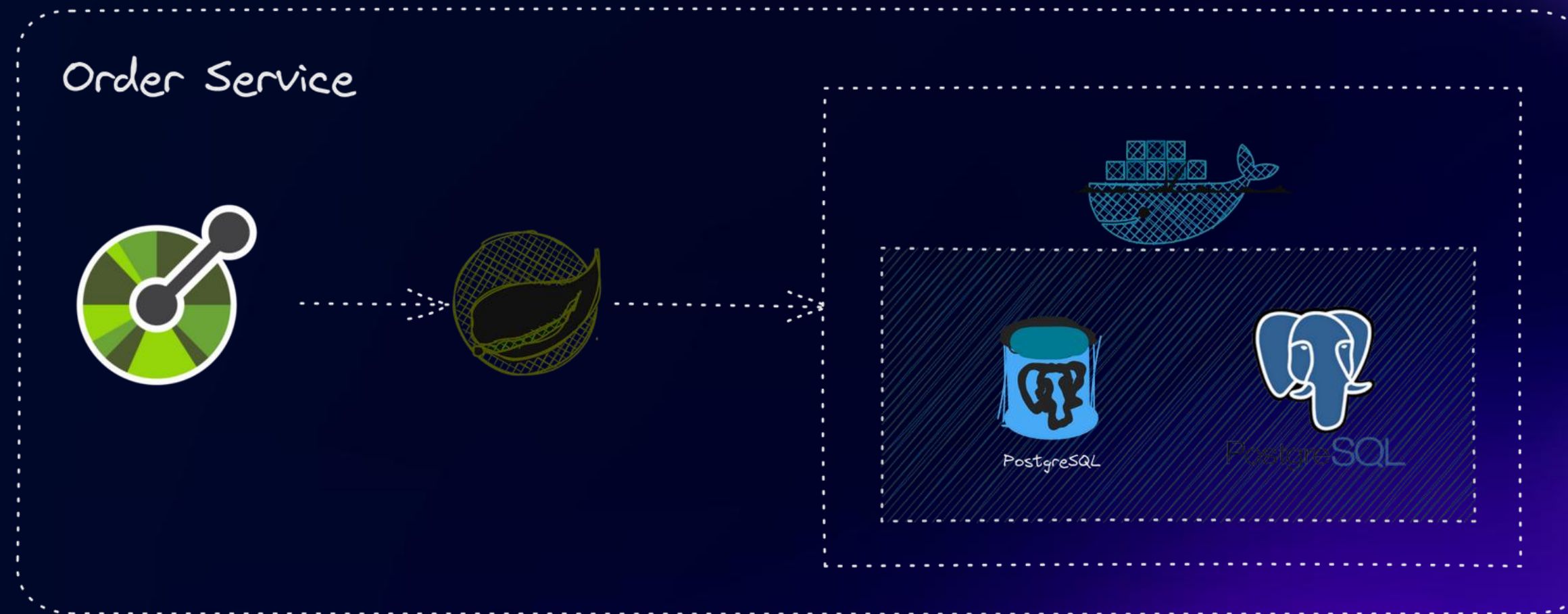
Limita el tiempo de espera de una solicitud



Implementación

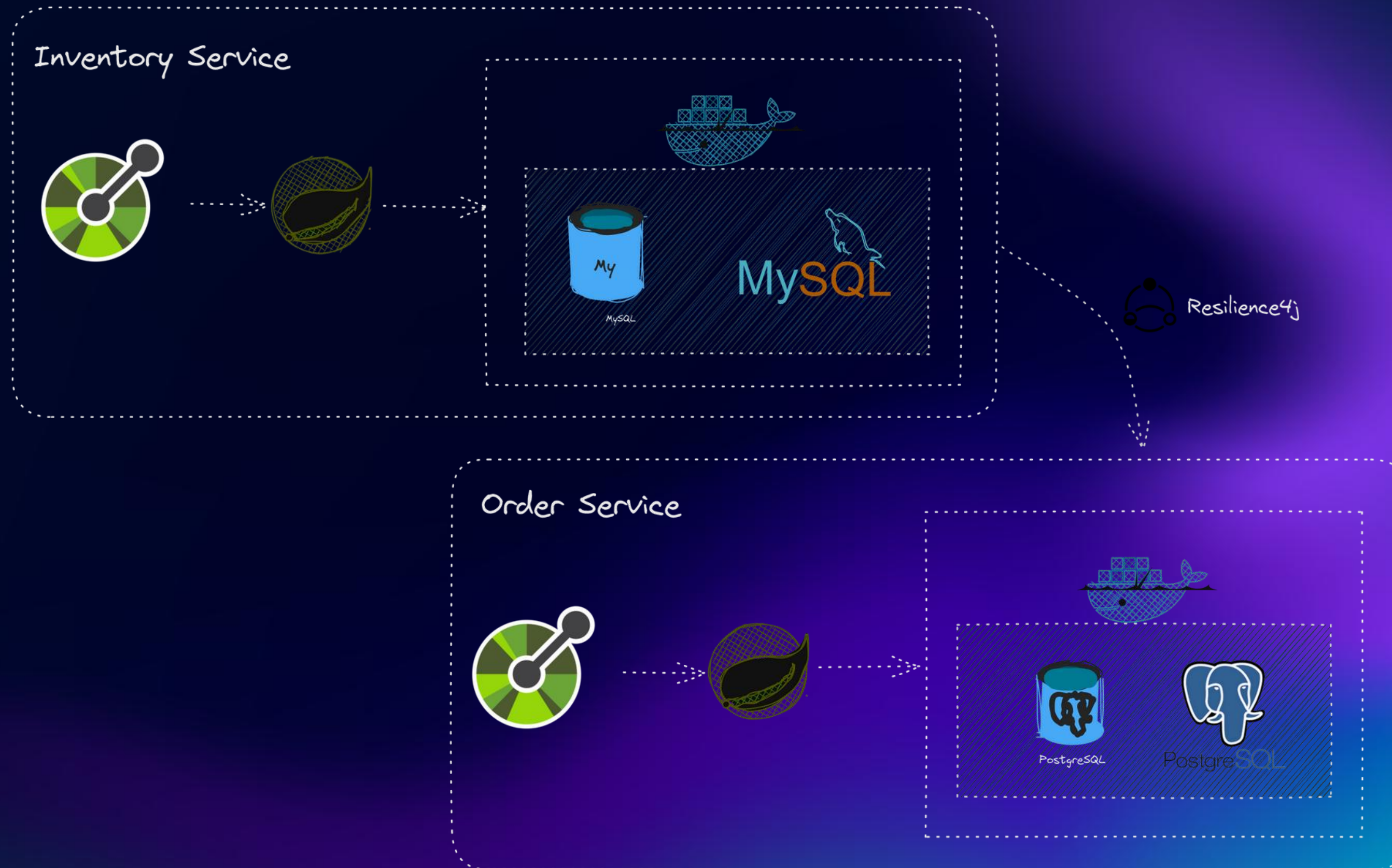
Servicio de E-Commerce

API - Servicio de Ordenes



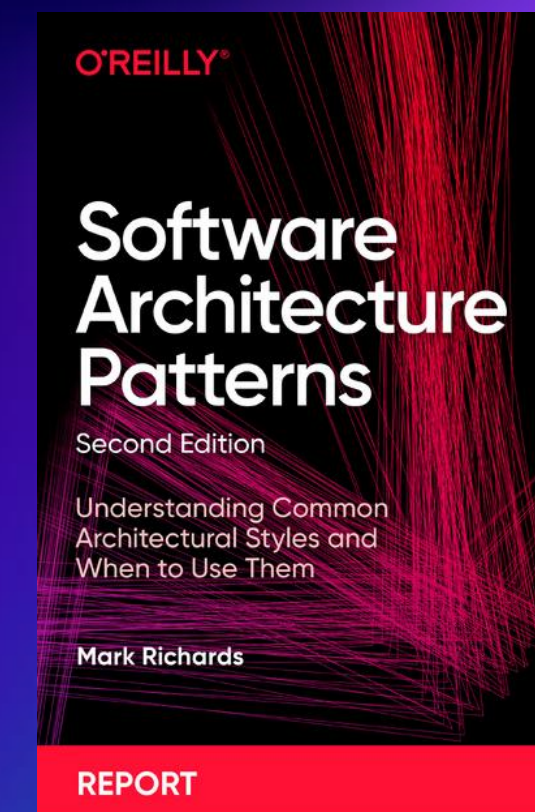
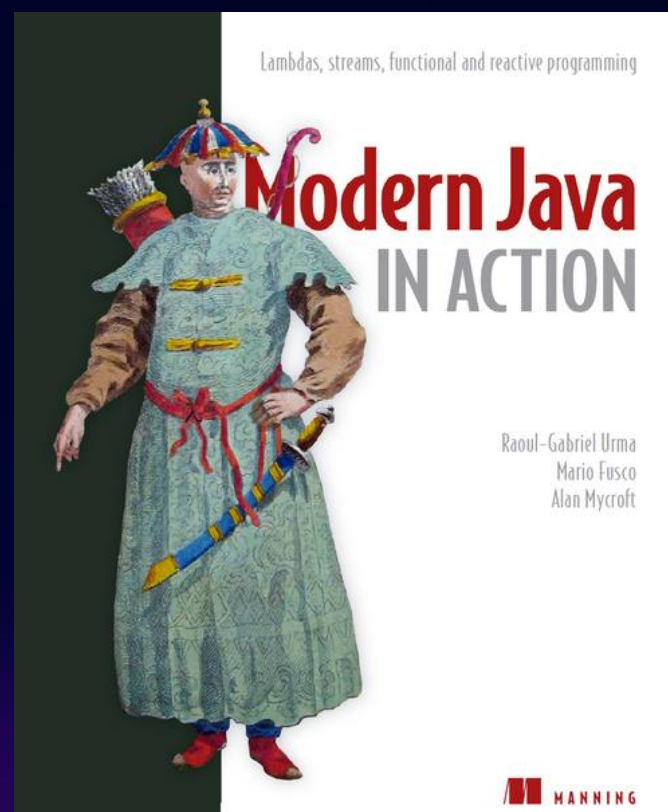
- Open API
- Spring Boot + Spring WebFlux
- Docker
- PostgreSQL

Integración - Servicio de Inventario + Servicio de Ordenes



Referencias

- <https://resilience4j.readme.io/docs/getting-started>
- <https://www.paradigmadigital.com/dev/4-patrones-resiliencia-fundamentales/>
- <https://medium.com/@barbieri.santiago/resilience-patterns-6db662227e82>
- <https://contribute.liquibase.com/extensions-integrations/directory/integration-docs/springboot/>



Gracias!



¿Tienes alguna Pregunta?

contacto@codearti.com

+51 953 888 029

codearti.com