hiberus

Prueba Técnica Java - Spring Boot

Octubre 2025

hiberus

Índice

1. Prueba Técnica Java

1.1. Requerimiento

La entidad bancaria de la localidad norte de la ciudad está en un proceso de migración de varios de sus servicios legados SOAP hacia servicios REST, los cuales estarán alineados de banca (BIAN). Este proyecto para la empresa es de vital importancia, además de ser crítico dentro del proceso de modernización, porque estos servicios SOAP manejan muchos procesos core en áreas importante dentro de la entidad bancaria. Por lo cual se requiere velocidad en el proceso de migración, sin perder la calidad y eficiencia de los servicios migrados, y que no afecten a los consumidores dentro del ecosistema del área de negocio; de esta manera, se busca incorporar el uso de Inteligencia Artificial como asistente en el desarrollo de los nuevos servicios REST. Con esta definición el proceso y requerimientos en la migración se definen de la siguiente manera:

- Se tomará como punto de partida el servicio SOAP legado de órdenes de pago
- Analizar el WSDL para comprender capas funcionales, campos y estados.
- Obtener un contrato REST (OpenAPI 3.0) alineado al Service Domain BIAN Payment Initiation (BQ: PaymentOrder).
- Implementar un microservicio en Spring Boot 3+ (Java 17+), con enfoque contract-first y arquitectura hexagonal.
- Implementar pruebas unitarias y de integración, métricas de calidad, Docker y evidencias de uso de IA (prompts y paso a paso de generaciones).
- El WSDL se usa solo para análisis.

Funcionalidades REST esperadas

- POST /payment-initiation/payment-orders — Initiate una orden de pago.
- GET /payment-initiation/payment-orders/{id} — Retrieve la orden de pago.
- GET /payment-initiation/payment-orders/{id}/status — Retrieve Status de la orden.

2. Consideraciones

Para la implementación del ejercicio planteado, considerar los siguientes puntos mandatorios y opcionales:

Mandatorios:

- Java 17 o superior

- Spring Boot 3+ (Stack principal)
- Contract-first con OpenAPI 3.0 (usa openapi-generator para generar interfaces/DTOs)
- Arquitectura hexagonal (dominio, aplicación, puertos, adaptadores, config)
- Pruebas: JUnit 5, AssertJ, Mockito, y pruebas de integración con WebTestClient o RestAssured
- Calidad: JaCoCo (≥80% líneas), Checkstyle, SpotBugs
- Docker: Dockerfile (multi-stage) y docker-compose mínimo
- Uso asistido de Inteligencia Artificial
    - Registro de prompts y respuestas (resumen) usados para:
        - resumir el WSDL y proponer mapping a BIAN,
        - generar borrador de openapi.yaml,
        - esqueleto de capas (hexagonal),
        - generación de tests.
    - Explicar qué correcciones manuales se hicieron y por qué (validación humana).

Opcionales (Nice to have)

- Spring WebFlux (reactivo) en lugar de MVC
- Persistencia reactiva con R2DBC (Postgres) y Testcontainers
- RFC 7807 (application/problem+json) para manejo de errores
- Observabilidad (Micrometer/Actuator), idempotencia, validaciones robustas

Expectativas de diseño

- **Alineación BIAN:** nomenclatura de recursos, BQs y estados conforme al SD elegido (Payment Initiation / PaymentOrder).
- **Modelo de dominio limpio**: entidades, invariantes, mapeo claro request/response ↔ dominio.
- **Contract-first real**: controladores implementan interfaces generadas desde openapi.yaml.

3. Entregables

Los entregables esperados para este ejercicio son los siguientes:

- URL del repositorio o repositorios
- Documentación en el archivo README con:
    - Contexto y decisiones en el proceso de migración desde la etap inicial
    - Pasos para la ejecución del servicio de manera local y con Docker
    - Sección para indicar el uso de la IA que incluya prompts, respuestas (un resumen de las mismas), fragmentos generados y correcciones.
- Open API – archivo yml con el contrato.
- Pruebas unitarias (dominio, validaciones, mappers), Integración E2E (End-to-end), reporte de cobertura de código >= 80% de cobertura de código.
- Calidad
    - Checkstyle y Spotbugs sin fallos validados con el comano *mvn verify*

- Docker:
  - Docker file
  - Docker-compose file
- Evidencia de IA
  - Carpeta *ai/* con prompts.md, *generations/* decisions.md

IMPORTANTE:

- **Se envía archivo WSDL, ejemplos de Request y Response, además de un archivo json para ser validado en Postman o herramientas similares.**
- **Tiempo estimado del desarrollo del ejercicio es de 3 días.**

hiberus