E-Commerce Application – Spring Boot Microservices

This document describes the requirements and low-level design of a Spring Boot based E-Commerce Microservices application.

1. User Microservice

- Manages customer accounts and roles

- **Tables**: users, roles

- **APIs**: POST /users, GET /users/{id}

- **Database**: MySQL

2. Product Microservice

- Manages products and categories

- **Tables**: products, categories

- **APIs**: POST /products, GET /products/{id}

- **Database**: MySQL

3. Inventory Microservice

- Handles stock validation and updates

- **Table**: inventory

- **APIs**: GET /inventory/{productId}, PUT /inventory/reduce

- Uses Resilience4j

- **Database**: MySQL

4. Cart Microservice

- Manages shopping cart

- **Tables**: cart, cart_items

- **APIs**: POST /cart/add, DELETE /cart/remove

- **Database**: Redis / MySQL

5. Order Microservice

- Core orchestration service

- **Tables**: orders, order_items

- **APIs**: POST /orders, GET /orders/{id}

- Kafka Producer: ORDER_CREATED

- **Database**: MySQL

6. Payment Microservice

- Handles payment processing

- **Table**: payments

- **API**: POST /payments

- Kafka Events: PAYMENT_SUCCESS, PAYMENT_FAILED

- **Database**: MySQL

7. Notification Microservice

- Sends email/SMS notifications

- Kafka Consumer

- **Events**: ORDER_CREATED, PAYMENT_SUCCESS

- No Database

8. Review & Rating Microservice

- Manages product reviews and ratings

- **Table**: reviews

- **APIs**: POST /reviews, GET /reviews/product/{id}

- **Database**: MySQL

INFRASTRUCTURE SERVICES

---

1. Config Server

- Centralized configuration management

- Reads configuration from Git repository

- Provides configs to all microservices

2. Eureka Server

- Service discovery and registration

3. API Gateway

- Single entry point

- Routing, logging, rate limiting

TECHNOLOGY STACK

- Spring Boot

- Spring Cloud Config

- Eureka Server

- Spring Cloud Gateway

- Spring Data JPA

- Kafka

- Resilience4j

- Zipkin & Sleuth