

Training Project: Product Store – Spring Boot Microservices













Project Overview

This hands-on training project guides learners through building a **cloud-native product store** application using a **microservices architecture** with **Spring Boot**. The system is composed of independently deployable services responsible for product management, order processing, inventory control, and notifications.

It incorporates **modern cloud technologies** such as Kafka, Keycloak, and Kubernetes to create a scalable, resilient, and secure application, suitable for real-world production environments.

Learning Objectives

Participants will gain practical experience in:

-  **Developing RESTful APIs** using Spring Boot
 -  **Persisting data** with Spring Data JPA, MySQL, MongoDB, and Flyway
 -  **Implementing asynchronous, event-driven communication** with Apache Kafka
 -  **Securing microservices** using Spring Security with OAuth2 and Keycloak
 -  **Configuring an API Gateway** using Spring Cloud Gateway
 -  **Adding resiliency** with Resilience4j (circuit breaker, retries, rate limiting)
 -  **Scheduling jobs** using ShedLock for distributed locks
 -  **Inter-service communication** using RestClient and declarative HTTP interfaces
 -  **Creating aggregated Swagger/OpenAPI documentation** via the API Gateway
 -  **Setting up local environments** using Docker, Docker Compose, and Testcontainers
 -  **Writing automated tests** with JUnit 5, RestAssured, Awaitility, Testcontainers, and WireMock
 -  **Monitoring and observability** with Grafana, Prometheus, Loki, and Tempo
 - **Deploying to Kubernetes** via KinD (local) and EKS (AWS)
-

Microservices Overview

Product Service

- Manages product data (CRUD operations)
- Uses MongoDB for storage
- Exposes RESTful endpoints

Order Service

- Handles order creation and management
- Interacts with Product and Inventory services
- Stores data in MySQL
- Publishes order events to Kafka

Inventory Service

- Manages stock levels and availability
- Validates inventory before processing orders
- Uses MySQL and Kafka for event updates

Notification Service

- Listens to Kafka order events
- Sends real-time email/SMS notifications based on order status

API Gateway

- Built using Spring Cloud Gateway MVC
 - Acts as a single entry point to all services
 - Secured via Keycloak to allow only authorized access
-

Technologies Used

Backend Stack:

- Spring Boot (Core framework)
- Spring Cloud Gateway MVC (API Gateway)
- Kafka (Asynchronous communication)
- Keycloak (Authentication and authorization)

Databases:

- MongoDB (Product & Inventory services)
- MySQL (Order service)
- Flyway (Database migration)

Testing Tools:

- JUnit 5, RestAssured, Awaitility
- Testcontainers (for containerized test environments)
- WireMock (for mocking APIs)

Monitoring & Observability:

- **Prometheus** (Metrics)
- **Grafana** (Dashboards)
- **Loki** (Logs)
- **Tempo** (Distributed tracing)

Deployment & Infrastructure:

- Docker, Docker Compose (Local development)
- Kubernetes (Orchestration)
 - KinD (Development cluster)
 - AWS EKS (Production deployment)

