E-commerce System Architecture

Overview

The e-commerce system is designed as a microservices-based architecture utilizing Spring Boot, gRPC, MongoDB, GraphQL, Kafka, and Kubernetes. The system follows the Backend-for-Frontend (BFF) pattern, allowing efficient communication between the frontend and backend services.

Technology Stack

Backend: Spring Boot
API Communication: gRPC
GraphQL Layer: Rejoiner
Database: MongoDB
Event Streaming: Kafka
Containerization: Docker, Kubernetes
Authentication: OAuth2 (Keycloak)
Logging & Monitoring: Prometheus, Grafana, ELK Stack

Architecture Overview

The system consists of several independent microservices communicating through gRPC. The BFF (Backend-for-frontend) acts as a GraphQL layer that aggregates data from different microservices and exposes it to the front end.

High-Level Components

Frontend (Next.js / React) communicates with the BFF via GraphQL.
BFF (GraphQL Gateway) fetches data from microservices via gRPC and converts it into GraphQL Schema using Rejoiner.
Microservices handle domains such as Products, Categories, Users, Orders, Inventory, and Notifications.
Kafka facilitates event-driven communication.
MongoDB serves as the primary database.
Kubernetes manages deployment and scaling

Microservices Breakdown

Product Service

Manages product catalogue (CRUD operations).
Stores data in MongoDB.
Communicates via gRPC.

Category Service

Handles product categorization (hierarchical structure).
Exposes gRPC API for category retrieval.
Stores data in MongoDB.

User Service

Handles authentication and user profiles.
Uses Keycloak for OAuth2 authentication.
Stores user data in MongoDB.

Order Service

Processes orders and payments.
Publishes order events to Kafka.
Uses gRPC for inter-service communication.

Inventory Service

Tracks stock levels.
Listens to Kafka Order Events.
Updates inventory based on order status.

Notification Service

Consumes Kafka Order Events.
Sends real-time notifications to users.
Simple email for now and maybe Twilio

Backend-for-Frontend (BFF) Architecture

The BFF layer communicates with microservices using gRPC.
Rejoiner converts gRPC responses into a GraphQL schema.
Frontend queries the BFF using GraphQL for optimized data fetching.

Event-driven architecture with Kafka

Event Flow

Order Service publishes an OrderCreated event.
Inventory Service consumes the event and updates stock.
Notification Service listens for the event and sends user notifications.

Kafka Topics

order-events
inventory-updates
User-notifications

Security & Authentication

OAuth2 (Keycloak) for user authentication.
JWT tokens are used for securing microservices.
RBAC (Role-Based Access Control) for user permissions.

Deployment & Scaling with Kubernetes

Dockerized microservices deployed on Kubernetes.
K8s Ingress Controller for external access.
Horizontal Pod Autoscaling (HPA) for load management.
ConfigMaps & Secrets for environment management.

Monitoring & Logging

Prometheus & Grafana for monitoring.
ELK Stack (Elasticsearch, Logstash, Kibana) for centralized logging.
Jaeger for distributed tracing of gRPC requests.