Spring Boot Microservices Case Study – ZetaEats

Executive Summary

ZetaEats is an online food delivery platform inspired by services like Swiggy or UberEats. It demonstrates how a monolithic food ordering platform can be decomposed into scalable, independent, and fault-tolerant microservices using Spring Boot and Spring Cloud.

Business Problem

The business requires a scalable, fault-tolerant food delivery system that can handle high traffic, support multiple restaurants, process payments, and deliver notifications in real time.

Microservices Breakdown

Microservice	Responsibility	Tech Stack
User Service	User registration, authentication, profiles	Spring Boot, Spring Security, JWT, MySQL
Restaurant Service	Restaurant details, menus, ratings	Spring Boot, JPA, MySQL
Order Service	Cart, checkout, tracking	Spring Boot, JPA, Kafka, MySQL
Payment Service	Payment gateway integration	Spring Boot, REST, PostgreSQL
Notification Service	Email/SMS notifications	Spring Boot, Kafka, Twilio/SendGrid
API Gateway	Routing, load balancing	Spring Cloud Gateway
Service Registry	Service discovery	Netflix Eureka
Config Server	Centralized configuration	Spring Cloud Config

Implementation Highlights

- Services communicate via REST (synchronous) and Kafka (asynchronous).
- Each microservice has its own database to avoid coupling.
- Security is managed using Spring Security and JWT.
- Monitoring is enabled with Prometheus, Grafana, and Zipkin.

Challenges & Solutions

1. Service Discovery → Solved with Netflix Eureka
2. Distributed Transactions → Saga Pattern with Kafka
3. Centralized Config Management → Spring Cloud Config Server
4. Resilience & Fault Tolerance → Resilience4j

Outcomes

✔ Independent scalability of services
✔ Resilient and fault-tolerant system
✔ CI/CD friendly and maintainable architecture

✔ Easy to extend with new services like Delivery Tracking

Conclusion

The ZetaEats platform showcases how Spring Boot microservices can be applied to design a real-world, scalable, and fault-tolerant online food delivery system. This architecture is well-suited for organizations transitioning from monoliths to microservices.