**GATEWAY, ROUTING & CROSS-CUTTING CONCERNS TRONG MICROSERVICES**

**1. Giới Thiệu**

Trong hệ thống Microservices, các service thường giao tiếp với nhau và với client từ nhiều nguồn khác nhau. Việc quản lý request từ client đến service phụ thuộc vào Gateway và Routing.

Ngoài ra, các vấn đề chung như **Bảo mật, Logging, Authentication, Rate Limiting** cũng cần được quản lý một cách hợp lý, đó là lý do các **Cross-cutting Concerns** ra đời.

**2. API Gateway là gì?**

**API Gateway** là một lớp trung gian quản lý tất cả request từ client đến hệ thống Microservices.

**Chức năng chính của API Gateway:**

* **Routing:** Chuyển hướng request đến service phù hợp.
* **Authentication & Authorization:** Xác thực và phân quyền user.
* **Load Balancing:** Phân phát request đồng đều giữa các instance.
* **Rate Limiting:** Giới hạn request từ client.
* **Logging & Monitoring:** Ghi lại request và phân tích lưu lượng.

**3. Routing trong API Gateway**

**Routing** là quá trình xác định service đích cho request dựa trên đường dẫn URL, Header, Query Params.

**3.1. Sử dụng Spring Cloud Gateway**

Spring Cloud Gateway là một API Gateway được phát triển trên Spring Boot, dùng để xử lý Routing, Security, Logging.

**3.2. Cách triển khai Spring Cloud Gateway**

1. Thêm dependency trong pom.xml:

<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-gateway</artifactId>  
</dependency>

1. Cấu hình application.yml:

spring:  
 cloud:  
 gateway:  
 routes:  
 - id: order-service  
 uri: lb:*//ORDER-SERVICE*  
 predicates:  
 - Path=/orders*/\*\**  
 *- id: user-service*  
 *uri: lb://USER-SERVICE*  
 *predicates:*  
 *- Path=/users/\*\**

1. Cài đặt Load Balancer:

eureka:  
 client:  
 service-url:  
 defaultZone: http:*//localhost:8761/eureka/*

1. Kết nối với Eureka, API Gateway sẽ tự động tìm dịch vụ phù hợp.

**4. Cross-cutting Concerns trong Microservices**

Cross-cutting Concerns là các tính năng định tuyến chung, không phải là một service đối với client, nhưng vẫn có tầm quan trọng.

**4.1. Authentication & Authorization**

* **OAuth2, JWT, Keycloak** dùng để xác thực user.
* API Gateway chỉ chấp nhận request hợp lệ.

**4.2. Logging & Monitoring**

* Sử dụng **ELK Stack (Elasticsearch, Logstash, Kibana)** để thu thập và phân tích logs.
* **Zipkin, Prometheus, Grafana** dùng để theo dõi hiệu suất.

**4.3. Rate Limiting**

* Ngăn chặn request quá tải bằng **Redis Rate Limiting**.
* Cài đặt trong Spring Cloud Gateway:

spring:  
 cloud:  
 gateway:  
 routes:  
 - id: order-service  
 uri: lb:*//ORDER-SERVICE*  
 filters:  
 - name: RequestRateLimiter  
 args:  
 redis-rate-limiter.replenishRate: 10  
 redis-rate-limiter.burstCapacity: 20

**4.4. Circuit Breaker (Ngăn chặn lỗi lan rộng)**

* Sử dụng **Resilience4j** để chống lại lỗi lan truyền khi service bị sắp.
* Cài đặt Circuit Breaker trong Gateway:

spring:  
 cloud:  
 gateway:  
 routes:  
 - id: order-service  
 uri: lb:*//ORDER-SERVICE*  
 filters:  
 - name: CircuitBreaker  
 args:  
 name: orderServiceCircuitBreaker