**Exercise 1: Edge Service for Routing and Filtering**

**Task:**  
Build an Edge Service using Spring Cloud Gateway to route and filter requests.

**Steps:**

1. Create a Spring Boot project with:
   * spring-cloud-starter-gateway
2. Configure application.properties:

spring.cloud.gateway.routes[0].id=example\_route

spring.cloud.gateway.routes[0].uri=http://example.org

spring.cloud.gateway.routes[0].predicates[0]=Path=/example/\*\*

1. Add a logging filter:

@Component

public class LoggingFilter implements GlobalFilter {

@Override

public Mono<Void> filter(ServerWebExchange exchange, GatewayFilterChain chain) {

System.out.println("Request: " + exchange.getRequest().getURI());

return chain.filter(exchange);

}

}

**Exercise 2: Load Balancing in API Gateway**

**Task:**  
Enable load balancing in the API Gateway using Spring Cloud LoadBalancer.

**Steps:**

1. Create a Spring Boot project with:
   * spring-cloud-starter-gateway
   * spring-cloud-starter-loadbalancer
2. Configure application.properties:

spring.cloud.gateway.routes[0].id=load\_balanced\_route

spring.cloud.gateway.routes[0].uri=lb://example-service

spring.cloud.gateway.routes[0].predicates[0]=Path=/loadbalanced/\*\*

1. Load balancer configuration:

@Configuration

public class LoadBalancerConfiguration {

@Bean

public ReactorLoadBalancer<ServiceInstance> randomLoadBalancer(

Environment environment,

LoadBalancerClientFactory loadBalancerClientFactory) {

String name = environment.getProperty(LoadBalancerClientFactory.PROPERTY\_NAME);

return new RandomLoadBalancer(

loadBalancerClientFactory.getLazyProvider(name, ServiceInstanceListSupplier.class),

name

);

}

}

**Exercise 3: Resilience Patterns in API Gateway**

**Task:**  
Implement resilience using Resilience4j with Circuit Breaker and Time Limiter.

**Steps:**

1. Add dependencies:
   * spring-cloud-starter-gateway
   * resilience4j-spring-boot2
2. Configure application.properties:

resilience4j.circuitbreaker.instances.exampleCircuitBreaker.registerHealthIndicator=true

resilience4j.circuitbreaker.instances.exampleCircuitBreaker.slidingWindowSize=10

resilience4j.circuitbreaker.instances.exampleCircuitBreaker.failureRateThreshold=50

1. Resilience configuration:

@Configuration

public class ResilienceConfiguration {

@Bean

public Customizer<ReactiveResilience4JCircuitBreakerFactory> defaultCustomizer() {

return factory -> factory.configureDefault(id -> new Resilience4JConfigBuilder(id)

.circuitBreakerConfig(CircuitBreakerConfig.ofDefaults())

.timeLimiterConfig(TimeLimiterConfig.ofDefaults())

.build());

}

}