Exercise 1: Implementing Edge Services for Routing and Filtering

Poxm.xml

<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-gateway</artifactId>  
</dependency>  
 \*\*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/\*\*  
 \*\*Custom 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 an API Gateway

Pom.xml:  
<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-gateway</artifactId>  
</dependency>  
<dependency>  
<groupId>org.springframework.cloud</groupId>  
<artifactId>spring-cloud-starter-loadbalancer</artifactId>  
</dependency>

Application propertyes:

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/\*\*  
 \*\*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);}}