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

**pom.xml**

<dependencies>

<!-- Spring Cloud Gateway -->

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-gateway</artifactId>

</dependency>

<!-- Resilience4j Spring Boot -->

<dependency>

<groupId>io.github.resilience4j</groupId>

<artifactId>resilience4j-spring-boot3</artifactId>

</dependency>

<!-- For Reactive Circuit Breaker -->

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-circuitbreaker-reactor-resilience4j</artifactId>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>2023.0.1</version> <!-- adjust based on your setup -->

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>2023.0.1</version> <!-- adjust based on your setup -->

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

**Configure application.properties:**

# Route config with circuit breaker

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

spring.cloud.gateway.routes[0].uri=http://httpbin.org/status/500

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

spring.cloud.gateway.routes[0].filters[0]=CircuitBreaker=name=exampleCircuitBreaker,fallbackUri=forward:/fallback

# Resilience4j settings

resilience4j.circuitbreaker.instances.exampleCircuitBreaker.registerHealthIndicator=true

resilience4j.circuitbreaker.instances.exampleCircuitBreaker.slidingWindowSize=5

resilience4j.circuitbreaker.instances.exampleCircuitBreaker.failureRateThreshold=50

resilience4j.circuitbreaker.instances.exampleCircuitBreaker.minimumNumberOfCalls=3

**ResilienceConfiguration.java**

import io.github.resilience4j.circuitbreaker.CircuitBreakerConfig;

import io.github.resilience4j.timelimiter.TimeLimiterConfig;

import org.springframework.cloud.circuitbreaker.resilience4j.ReactiveResilience4JCircuitBreakerFactory;

import org.springframework.cloud.circuitbreaker.resilience4j.Resilience4JConfigBuilder;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.boot.context.properties.EnableConfigurationProperties;

import org.springframework.cloud.circuitbreaker.resilience4j.ReactiveResilience4JCircuitBreakerFactory;

import org.springframework.context.annotation.Bean;

import java.time.Duration;

import java.util.function.Function;

@Configuration

public class ResilienceConfiguration {

@Bean

public Customizer<ReactiveResilience4JCircuitBreakerFactory> defaultCustomizer() {

return factory -> factory.configureDefault(id ->

new Resilience4JConfigBuilder(id)

.circuitBreakerConfig(

CircuitBreakerConfig.custom()

.failureRateThreshold(50)

.waitDurationInOpenState(Duration.ofSeconds(5))

.slidingWindowSize(5)

.build()

)

.timeLimiterConfig(

TimeLimiterConfig.custom()

.timeoutDuration(Duration.ofSeconds(2))

.build()

)

.build()

);

}

}

**FallbackController.java:**

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class FallbackController {

@GetMapping("/fallback")

public String fallback() {

return " Service is temporarily unavailable. Please try again later.";

}

}