Spring Cloud API Gateway

Zuul API Gateway

- Zuul, Netflix OSS tarafından geliştirilmiş JVM tabanlı bir **router** ve **server side load balancer**'dır
- Routing ve load balancing işlerinin dışında **güvenlik**, **statik response üretme** gibi kabiliyetler de sunar

pom.xml

```
<dependencies>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-netflix-zuul</artifactId>
    </dependency>
</dependencies>
```

Zuul API Gateway

```
@EnableZuulProxy
@SpringBootApplication
public class GatewayServiceApplication {

...
}
```

Reverse proxy konfigürasyonu yapar böylece UI client'lar backend servislere gateway üzerinden erişebilirler

Proxy backend servislere erişmek için Ribbon loader balancer'ı kullanmaktadır

Zuul starter kendisi service discovery client dahil etmediği için, Eureka gibi bir service Discovery client'ın ekli olması gerekir

Zuul API Gateway

- Default olarak **Discovery client** tarafından tespit edilen bütün servisler api gateway tarafından proxy'lenir
- Bütün servis URI'larına **ortak bir prefix** eklenebilir

application.properties

```
eureka.client.service-url.defaultZone=http://localhost:8761/eureka

zuul.prefix=/api
```

Explicit Servis Tanımı

- İstenirse otomatik proxy'lenen servisler devre dışı bırakılıp **explicit servis tanımı** yapılabilir

application.properties


```
zuul.ignored-services=*

zuul.routes.cs.path=/cs/**
zuul.routes.cs.service-id=counterservice

zuul.routes.rs.path=/rs/**
zuul.routes.rs.service-id=readingservice
```

Route ve Filter Detayları

- Spring Boot Actuator aktif ise **/routes/details** ve **/filters** endpoint'lerinden api gateway'deki routing ve filter ayarları görüntülenebilir

pom.xml

```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-actuator</artifactId>
    </dependency>
</dependencies>
```

application.properties

```
management.endpoints.web.exposure.include=routes,filters
```

Cookie ve Header Değerleri

- **Request header** değerlerinin aynı sistem içerisindeki servislerde paylaşılması, fakat **harici servislere** erişilirken **ignore** edilmesi istenebilir
- Api gateway ardından erişilen bütün servislerin cookie'leri aynı origin'e sahip olacağı için api gateway ile **cookie kullanımı** çok da sağlıklı olmayacaktır

Cookie ve Header Değerleri

- Ayrıca global olarak bütün routing'lerde bir takım **request ve response header**'larının **ignore** edilmesi sağlanabilir
- Default durumda Spring Security classpath'de ise bunlar bir takım **security header**'larıdır

Cookie ve Header Değerleri

sensitiveHeaders'ın default değeri de böyledir

```
zuul.routes.cs.path=/cs/**
zuul.routes.cs.url=http://localhost:8080
zuul.routes.cs.sensitive-headers=Cookie,Set-Cookie,Autorization

zuul.ignored-headers=
zuul.ignore-security-headers=false
```

Route Fallback

- Routing sırasında meydana gelen hatalı durumlarda **fallback response** üretmek için **FallbackProvider** arayüzü implement edilmeli ve **bir bean olarak** tanıtılmalıdır
- Ayrıca api gateway üzerinde **Hystrix**'de aktif olmalıdır

```
public interface FallbackProvider {

    String getRoute();

    ClientHttpResponse fallbackResponse(
                            String route, Throwable cause);
}
```

service id dönülmelidir,
Bütün servisler için default
bir fallback tanımlamak istenirse
* veya null dönülebilir

Location Header Rewrite

- Eğer zuul bir web uygulaması önünde yer alıyorsa, bu web uygulamasından dönecek 3xx statü kodlu response içindeki **Location header'ının rewrite edilmesi** gerekir

```
@Bean
public LocationRewriteFilter locationRewriteFilter() {
    return new LocationRewriteFilter();
}
```

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com