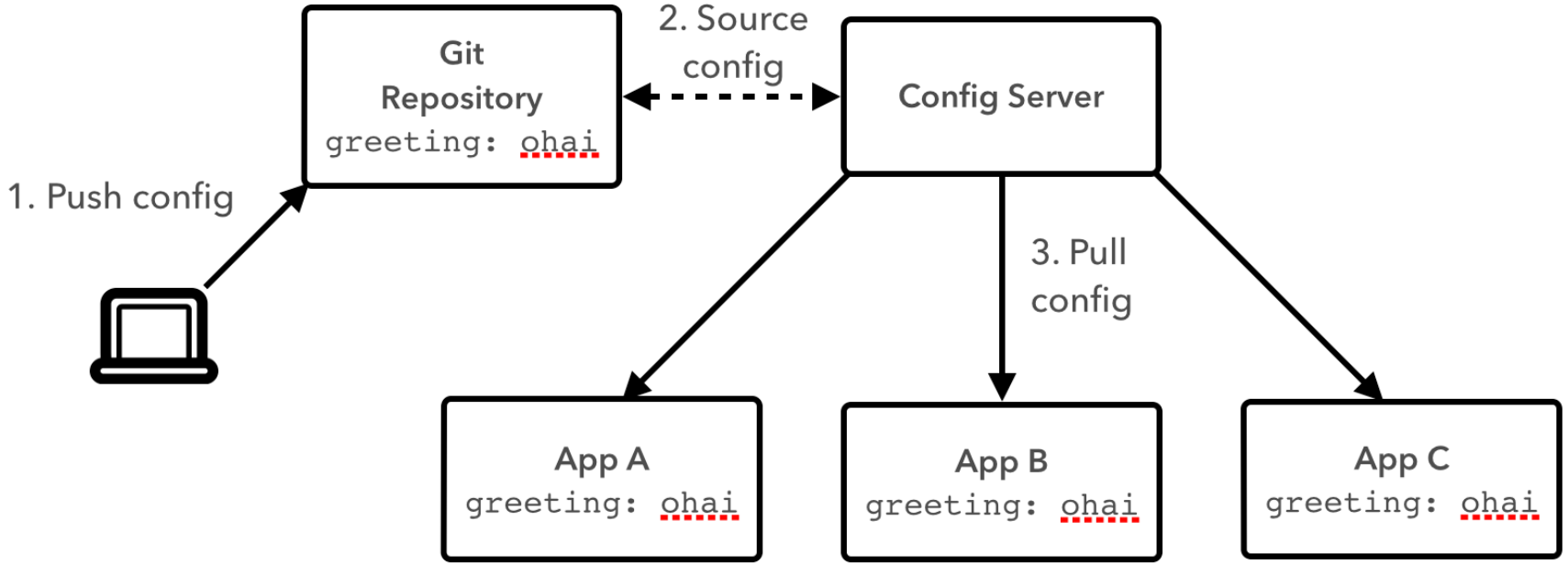
Spring Cloud Netflix

Agenda

- Configuration Management (Config Server/Client)
- Service Discovery (Eureka)
- Client Side Load Balancing (Ribbon)
- Declarative REST Client (Feign)
- Fault Tolerance & Monitoring (Hystrix)
- Router and Filter (ZUUL)

Configuration Management



Config Server

```
<parent>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-parent</artifactId>
<version>1.0.2.RELEASE</version>
<relativePath/>
</parent>
```

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

<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-config-server</artifactId>
<version>1.0.4.RELEASE</version>
</dependency>
```

```
spring:
  cloud:
    config:
      server:
        git:
          uri: [config repo]
          search-paths: [name of folder in case any]
```

```
@SpringBootApplication
@EnableConfigServer
public class ConfigServerApplication {
public static void main(String[] args) {
SpringApplication.run(ConfigServerApplication.class,
args);
}
}
```

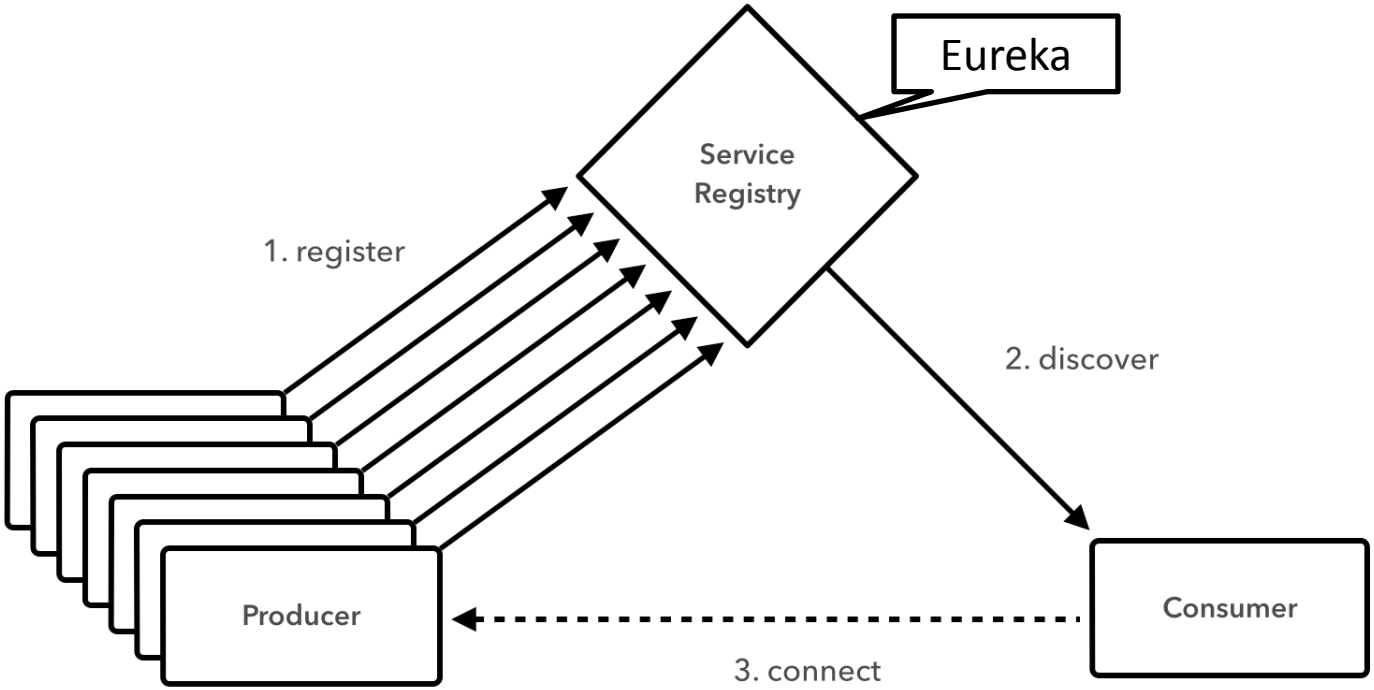
Config Client

```
<parent>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-parent</artifactId>
<version>1.0.2.RELEASE</version>
<relativePath/>
</parent>

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

<dependency>
<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-config-client</artifactId>
</dependency>
```

```
spring:
 cloud:
    config:
     uri: [config server root]
```

```
@SpringBootApplication
@EnableConfigServer
public class ConfigServerApplication {
public static void main(String[] args) {
SpringApplication.run(ConfigServerApplication.class,
args);
}
}
```

Service Discovery



Eureka Server

```
<parent>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-parent</artifactId>
<version>1.0.2.RELEASE</version>
<relativePath/>
</parent>
```

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

<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-eureka-
server</artifactId>
</dependency>
```

```
eureka:
  instance:
    hostname: localhost
  client:
    register-with-eureka: false
    fetch-registry: false
    service-url:
      defaultZone:
http://${eureka.instance.hostname}:${server.port}/eure
ka/
```

```
@SpringBootApplication
@EnableEurekaServer
public class EurekaServerApplication {

public static void main(String[] args) {
SpringApplication.run(EurekaServerApplication.class,
args);
}
}
```

Eureka Client – Provider Service

```
<parent>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-parent</artifactId>
<version>1.0.2.RELEASE</version>
<relativePath/>
</parent>
```

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

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

```
spring:
  application:
    name: exchange-rate-provider
eureka:
  client:
    service-url:
      defaultZone: http://localhost:9090/eureka/
```

```
@SpringBootApplication
@EnableEurekaClient
public class ConversionApplication {

public static void main(String[] args) {
SpringApplication.run(ConversionApplication.class,
args);
}
}
```

Eureka Client – Consumer Service

```
<parent>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-parent</artifactId>
<version>1.0.2.RELEASE</version>
<relativePath/>
</parent>
```

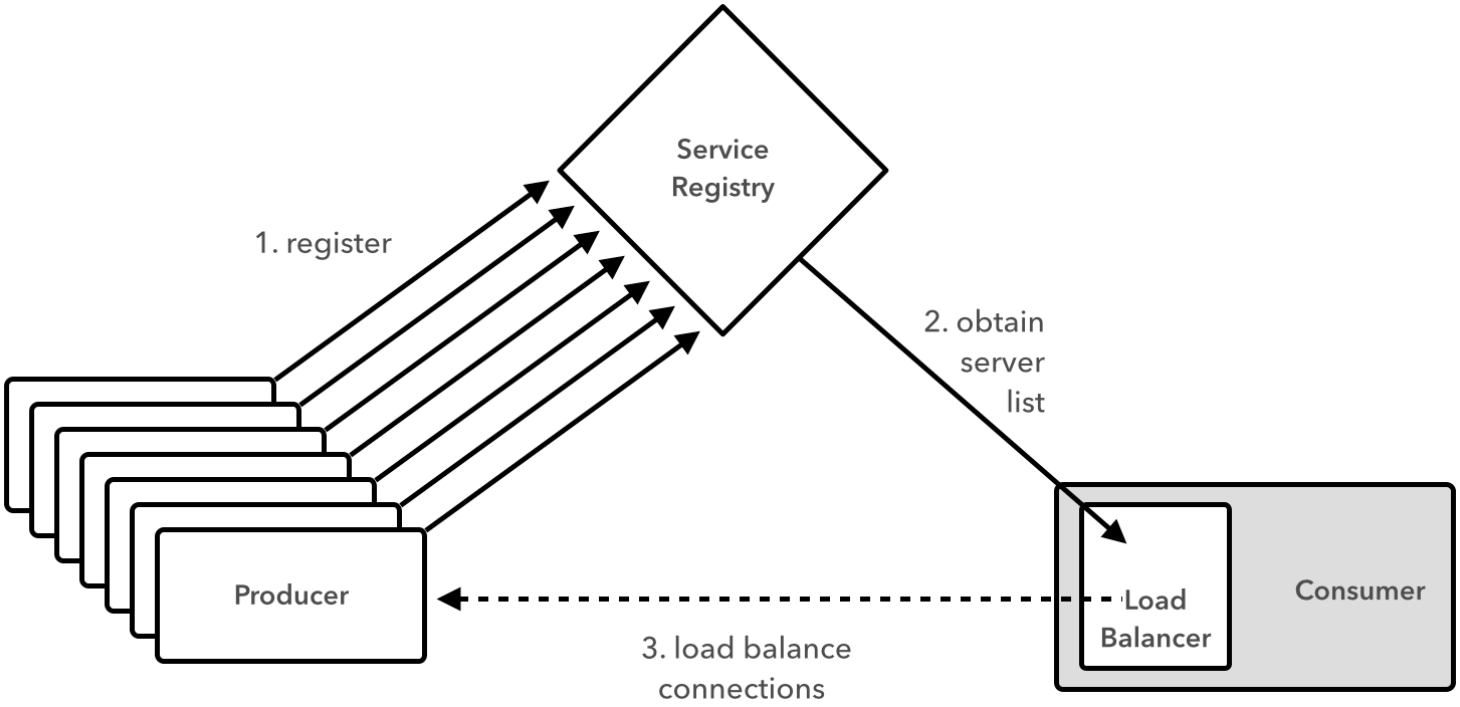
```
<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter</artifactId>
</dependency>

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

```
spring:
  application:
    name: conversion-app
eureka:
  client:
    service-url:
      defaultZone: http://localhost:9090/eureka/
```

```
@SpringBootApplication
@EnableEurekaClient
public class ConversionApplication {

public static void main(String[] args) {
SpringApplication.run(ConversionApplication.class, args);
}
}
```

```
InstanceInfo instance =
discoveryClient.getNextServerFromEureka("exchange-rate-provider",
false);

RestTemplate restTemplate = new RestTemplate();
conversionFactor =
restTemplate.getForObject(instance.getHomePageUrl()+"exchangeRate?cou
ntryCode={countryCode}”, Float.class, countryCode);
```

Load Balance (Ribbon)



Consumer Service

```
<parent>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-
parent</artifactId>
<version>1.0.2.RELEASE</version>
<relativePath/>
</parent>
```

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

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

```
@Autowired
private RestTemplate restTemplate;

@RequestMapping(value="/convert")
public Float convert(@RequestParam("amt") Float amount, @RequestParam("cc")
String countryCode, @RequestParam("sit") Integer serviceInvocationType) {

return conversionFactor = restTemplate.getForObject("http://exchange-rate-
provider/exchangeRate?countryCode={countryCode}”, Float.class,
countryCode);
}
```

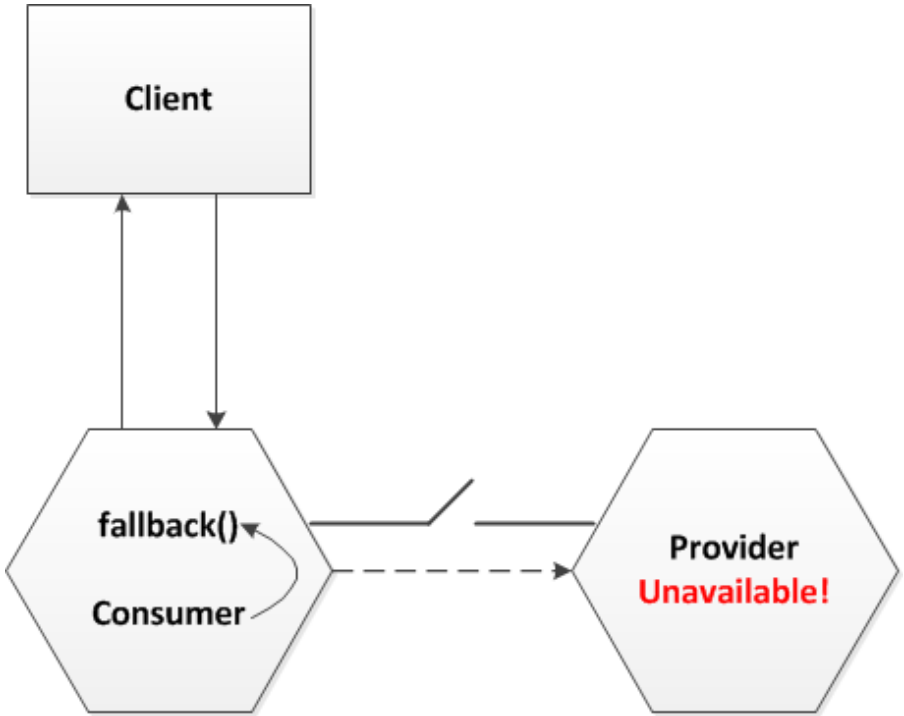
```
@Autowired
private LoadBalancerClient loadBalancerClient;

@RequestMapping(value="/convert")
public Float convert(@RequestParam("amt") Float amount, @RequestParam("cc")
String countryCode, @RequestParam("sit") Integer serviceInvocationType) {

ServiceInstance serviceInstance = loadBalancerClient.choose("exchange-rate-
provider");
RestTemplate restTemplate = new RestTemplate();
return
restTemplate.getForObject(serviceInstance.getUri()+"/exchangeRate?countryCo
de="+countryCode, Float.class);
}
```

Declarative REST Client

```
<parent>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-parent</artifactId>
<version>1.0.2.RELEASE</version>
<relativePath/>
</parent>


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

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

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

```
@SpringBootApplication
@EnableEurekaClient
@EnableFeignClients
public class ConversionApplication {

public static void main(String[] args) {
SpringApplication.run(ConversionApplication.class, args);
}
}


@FeignClient("exchange-rate-provider")
public interface ExchangeRateClient {

@RequestMapping(value="/exchangeRate", method=RequestMethod.GET)
public String getConversionFactor(@RequestParam("countryCode")
String countryCode);
}
```

Fault Tolerance (Circuit Breaker)



Fault Tolerance Using Hystrix

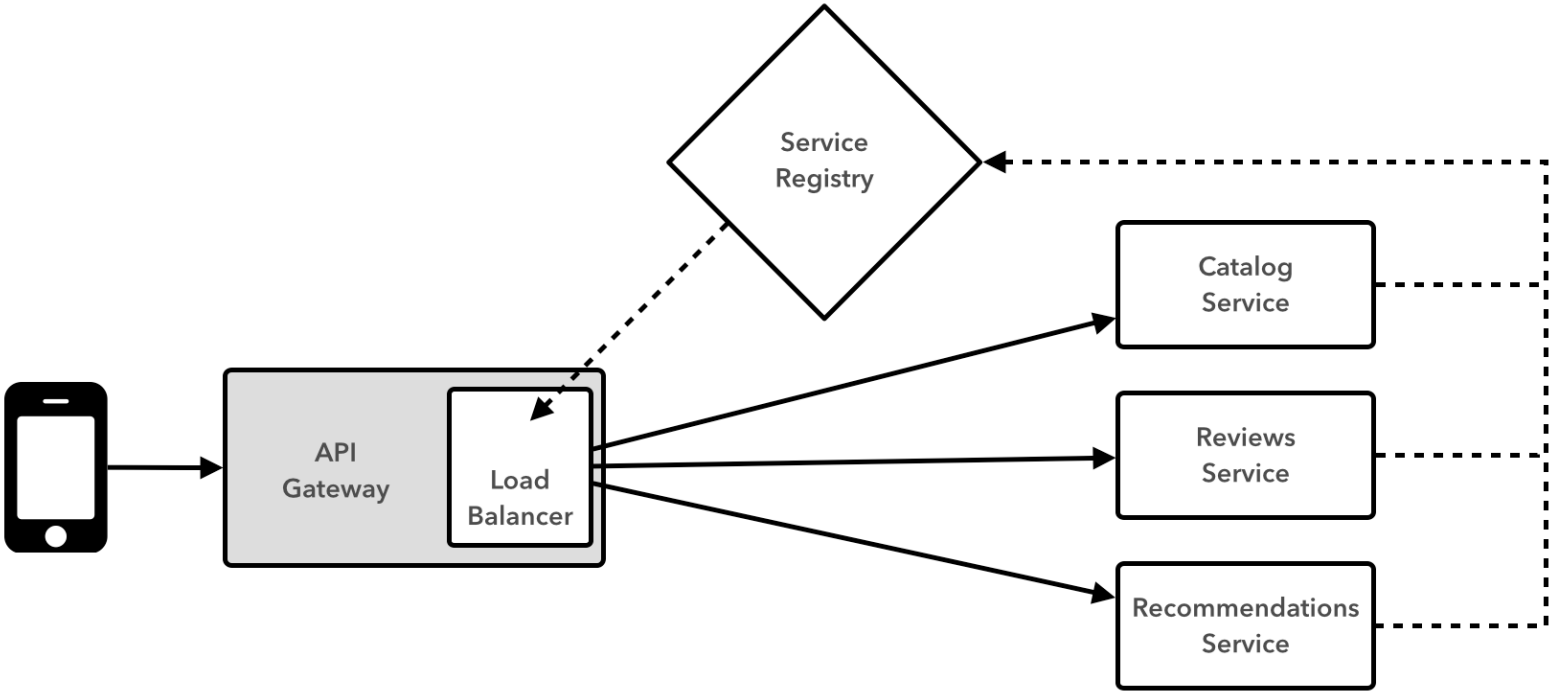
```
<parent>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-parent</artifactId>
<version>1.0.2.RELEASE</version>
<relativePath/>
</parent>
```

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

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

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

```
@SpringBootApplication
@EnableEurekaClient
@EnableCircuitBreaker
public class ConversionApplication {

public static void main(String[] args) {
SpringApplication.run(ConversionApplication.class, args);
}
}
```

```
@Component
public class ExchangeRateHystrixClient {

@Autowired
@LoadBalanced
RestTemplate restTemplate;

@HystrixCommand(fallbackMethod = "getConversionFactorFallback")
public String getConversionFactor(String countryCode) {

return restTemplate.getForObject("http://exchange-rate-
provider/exchangeRate?countryCode="+countryCode, String.class);
}

public String getConversionFactorFallback(String countryCode) {
return "0";
}
}
```

API Gateway Pattern



Reverse Proxy Using ZUUL

```
<parent>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-
parent</artifactId>
<version>1.0.2.RELEASE</version>
<relativePath/>
</parent>
```

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

```
server:
  port: 8000
zuul:
  routes:
    exchange-rate:
      path: /exchangeRate/**
      strip-prefix: false
      url: http://localhost:7070
    exchange-rate-root:
      path: /
      strip-prefix: false
      url: http://localhost:7070
```

```
@SpringBootApplication
@EnableZuulProxy
public class ZuulReverseProxyApplication {

public static void main(String[] args) {
SpringApplication.run(ZuulReverseProxyApplication.class, args);
}
}
```