netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server

Techefx

Microservices with Spring Boot & Spring Cloud With Live Example & Source Code !

In this session, we explore spring boot and spring cloud microservice architecture. I assume you already have basic knowledge of RESTful API development.

We would write & configure following spring cloud components:

1. Spring Cloud Config Server
2. Netflix Eureka Naming and Discovery server
3. Spring Cloud API Gateway Server
4. Netflix Ribbon Load Balancer
6. Zuul API Gateway Proxy
7. Open Feign Api to Api Communication
8. Zipkin Distributed Tracking with Sleuth and RabbitMQ

To Demonstrate all the above component's functionality, we would write primarily our own microservices mentioned below:

1. Property Access Service
2. Product Stock Service
3. Product Enquiry Service

Please find the complete source code on github URL:
http://www.github.com/techefx

Feel free to download and use the source code for your practice !

Feel free to reach out to me for any queries or need any clarifications, please write to :
toamarkumar@gmail.com

Thanks for watching and please don't forget to subscribe my channel !!

#microservicesArchitecture #springcloudmicroservices #techefx

Tags

what is microservices
microservices architecture
what is spring boot microservices

what is spring cloud microservices
what is spring cloud config server
microservices with spring cloud config server
microservices configuration using spring cloud config server

spring cloud api gateway
microservices using spring cloud api gateway

netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server

microservices with spring cloud api gateway
Microservices with API gateway server
Microservices with eureka naming and discovery server
Microservices naming and discovery server with eureka

netflix ribbon load balancer
load balancer with ribbon
microservices load balancer with rioon

Microservcies with zipkin distributed tracing
Microservices distributed tracing with zipkin
Microservices distributed tracing with zipkin and rabbitMQ

microservices with sleuth
spring boot sleuth

spring boot framework
microservices with actuator
microservices health api monitoring
microservices monitoring
microservices monitoring using spring actuator

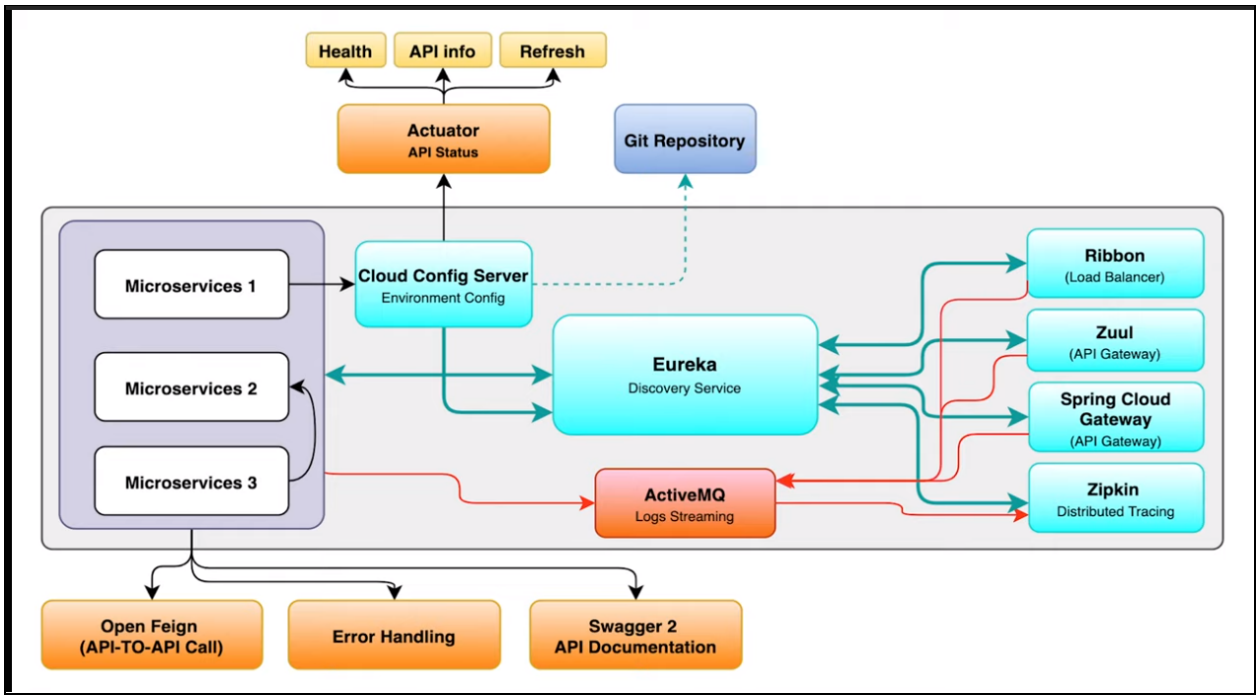
microservices with ELK

microservices using open feign
microservices using restTemplate
microservices api call using open feign
microservices open feign integration for api call
Open-feign

microservices api to api communication
microservices api to api call

learn microservices
learn spring boot microservices
mastering microservices
mastering spring boot microservices
mastering spring cloud microservices
microservice architecture explained

netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server



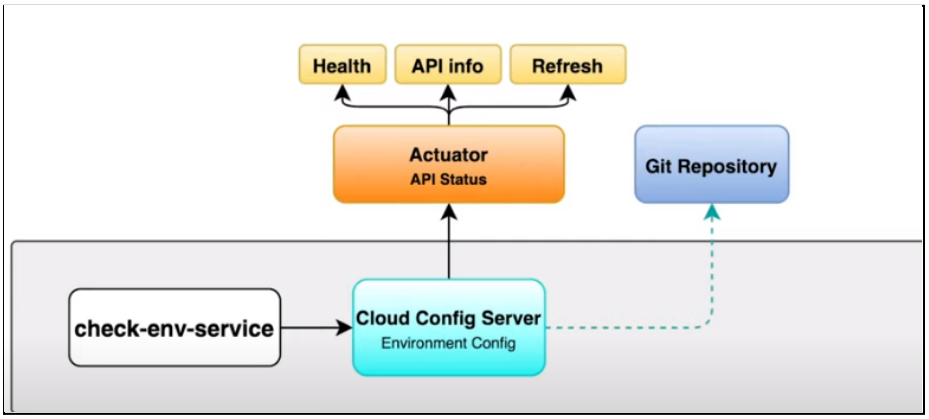
Components	Ports		
Spring Cloud Config Server	8888		
Eureka Naming and Discovery Server	8761		
Zuul API Gateway	8762		
Spring API Gateway	8763		
Zipkin Distributed Tracing	9411		
Microservices			
techefx-property-access-service	8100		
techefx-product-stock-service	8200	8201	8202
techefx-product-enquiry-service	8300	8301	8302

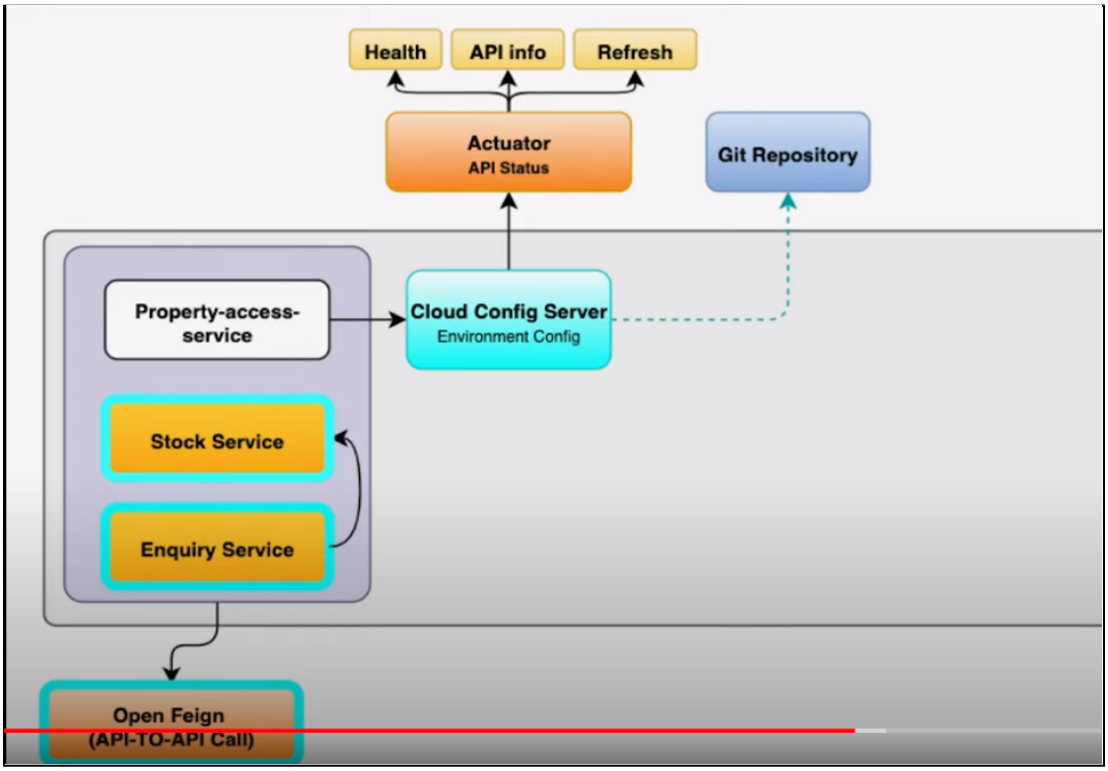
The port for Zuul is 8765!

netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server

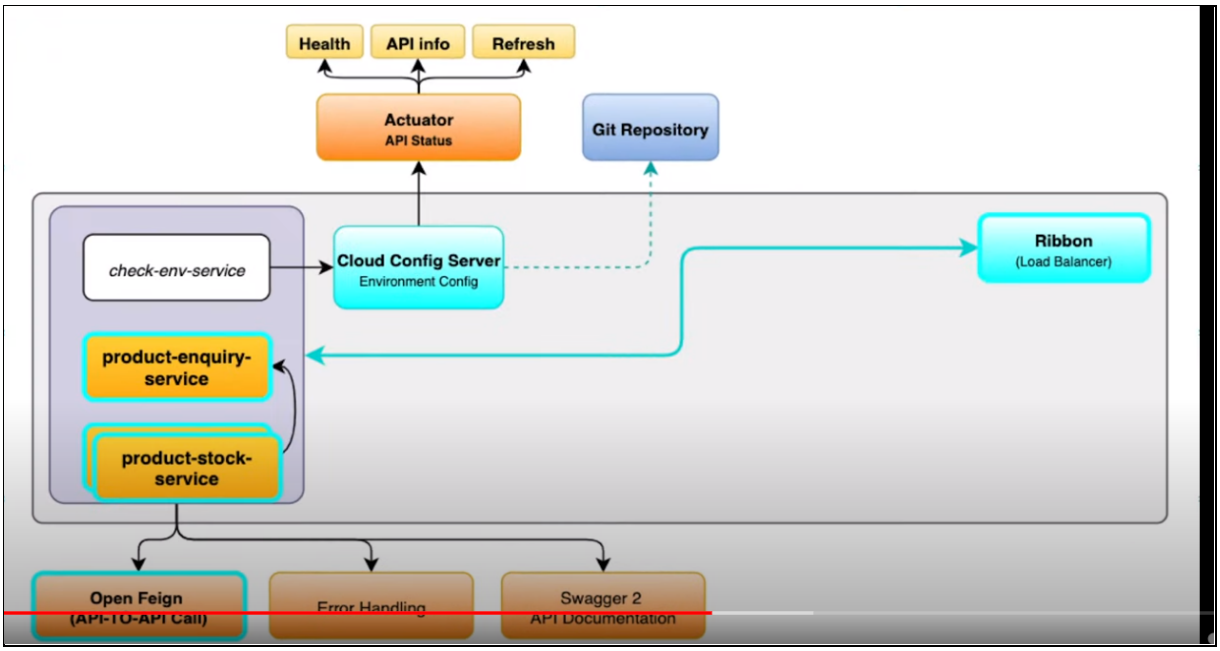
Tools	Version
Eclipse, Sprint Tool Suite, IntelliJ Idea, etc	
OpenJDK	14
Spring Boot	2.3.4 RELEASE
Postman API Testing Tool	Any

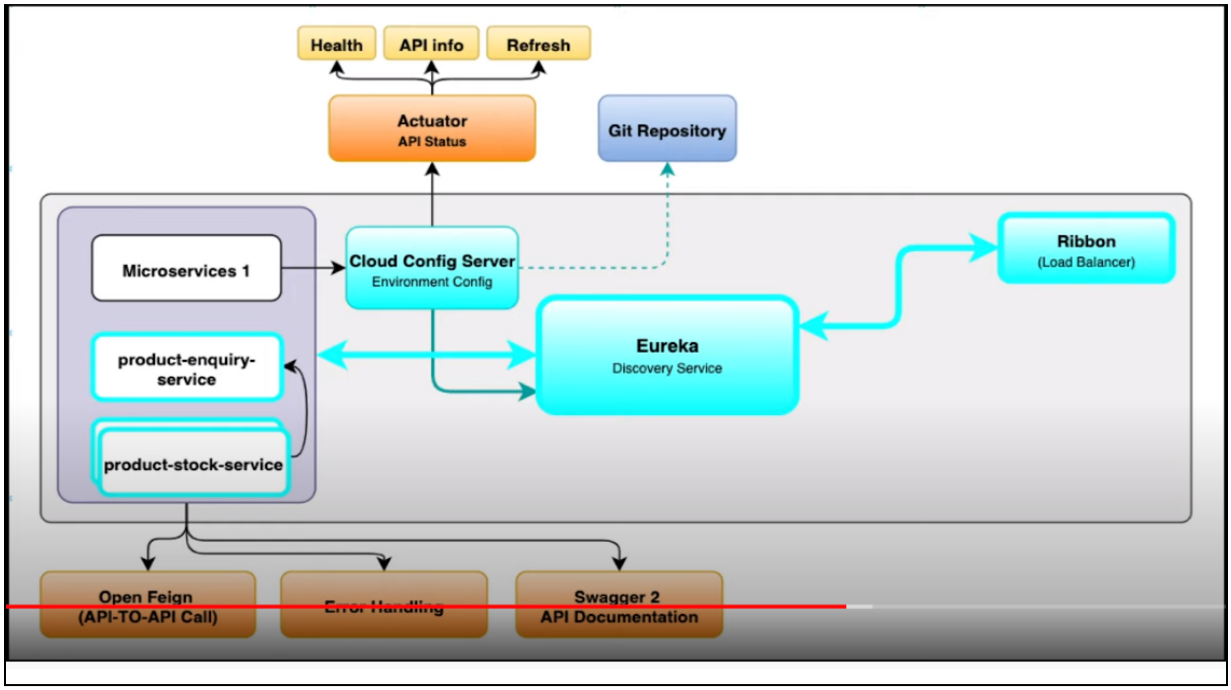
1.



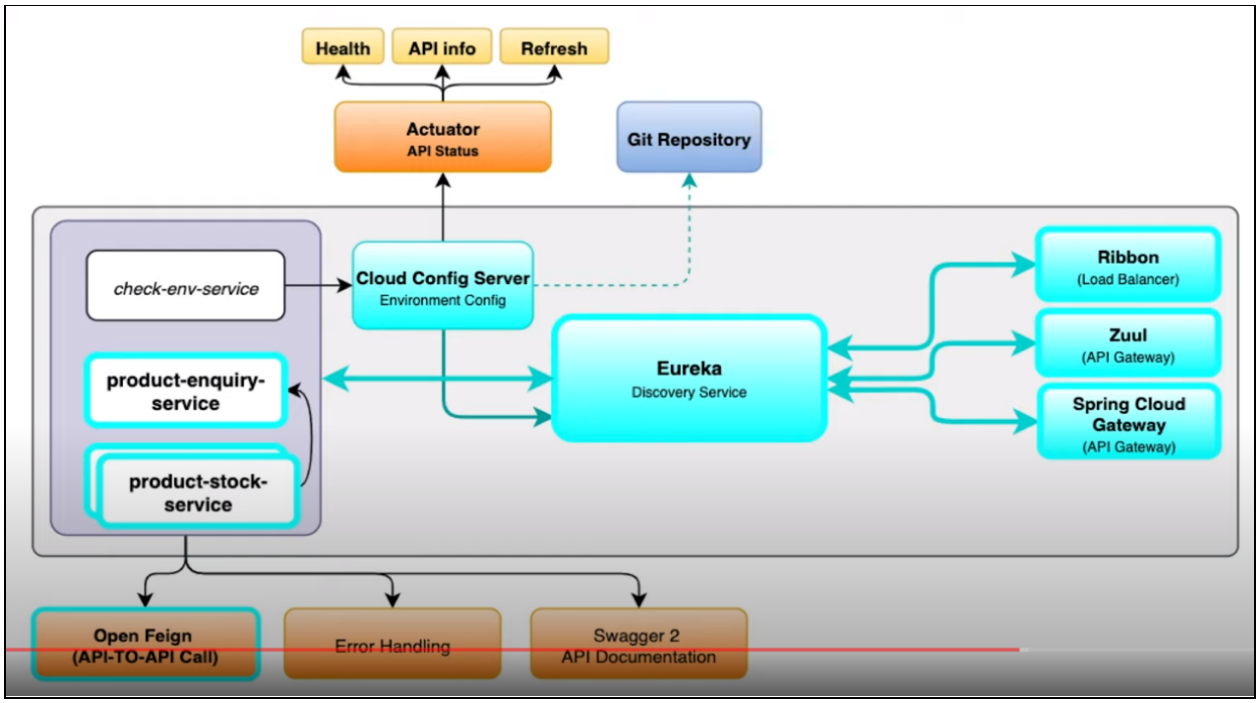


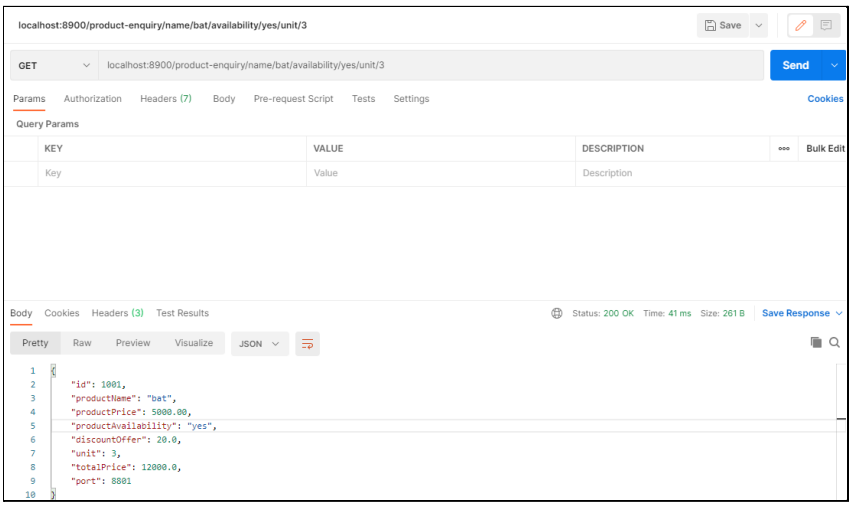
netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server





netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server





GET localhost:8900/product-enquiry/name/bat/availability/yes/unit/3


```
{
    "id": 1001,
    "productName": "bat",
    "productPrice": 5000.00,
    "productAvailability": "yes",
    "discountOffer": 20.0,
    "unit": 3,
    "totalPrice": 12000.0,
    "port": 8801
}
```

netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server

Or:

```
{
    "id": 1001,
    "productName": "bat",
    "productPrice": 5000.00,
    "productAvailability": "yes",
    "discountOffer": 20.0,
    "unit": 3,
    "totalPrice": 12000.0,
    "port": 8800
}
```

YAML - Files

Yaml files - techefx-spring-cloud-config-server

```yaml
spring:
 application:
   name: techefx-spring-cloud-config-server
 cloud:
   config:
     server:
       default-label: main
       git:
         uri: https://github.com/techefx/environment-variable-repo.git
server:
 port: ${port:8888}
```

Yaml - techefx-eureka-naming-server

```yaml
spring:
 application:
   name: techefx-eureka-naming-server
server:
 port: ${port:8761}
```

Yaml - techefx-spring-cloud-api-gateway-service

```yaml
spring:
 application:
   name: techefx-spring-cloud-api-gateway-service
 cloud:
   gateway:
     routes:
      - id: stock-enquiry
        uri: http://localhost:8700/
        predicates:
          - Path=/product-enquiry/**
```

netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server

Yaml - techefx-product-stock-service

```yaml
server:
 port: ${port:8900}
spring:
 application:
   name: techefx-product-stock-service
server:
 port: ${port:8800}
```

Yaml - techefx-**product-enquiry**-service

```yaml
spring:
 application:
   name: techefx-product-enquiry-service
server:
 port: ${port:8700}
```

Yaml - techefx-**product-enquiry**-service

```yaml
spring:
 application:
   name: techefx-product-enquiry-service
server:
 port: ${port:8700}
```

Bootstrap.yml - techefx-**zuul-api-gateway**-server

```yaml
spring:
 application:
   name: techefx-zuul-api-gateway-server

server:
 port: ${port:8765}

zuul:
 routes:
   product-enquiry: /product-enquiry/**
   url: http://localhost:8700/
```

2. Spring Cloud Gateway - programmatically config

```java
@SpringBootApplication
public class TechefxSpringCloudApiGatewayServiceApplication {

  public static void main(String[] args) {
```

netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server

```
SpringApplication.run(TechefxSpringCloudApiGatewayServiceApplication.class,
args);
  }
  @Bean
  public RouteLocator gatewayRoutes(RouteLocatorBuilder builder) {

     return builder.routes()
           .route(r -> r.path("/product-enquiry/**")
                 .uri("http://localhost:8700/")
                 .id("stock-enquiry")
           ).build();
  }
}
```

In application.yml comment config:

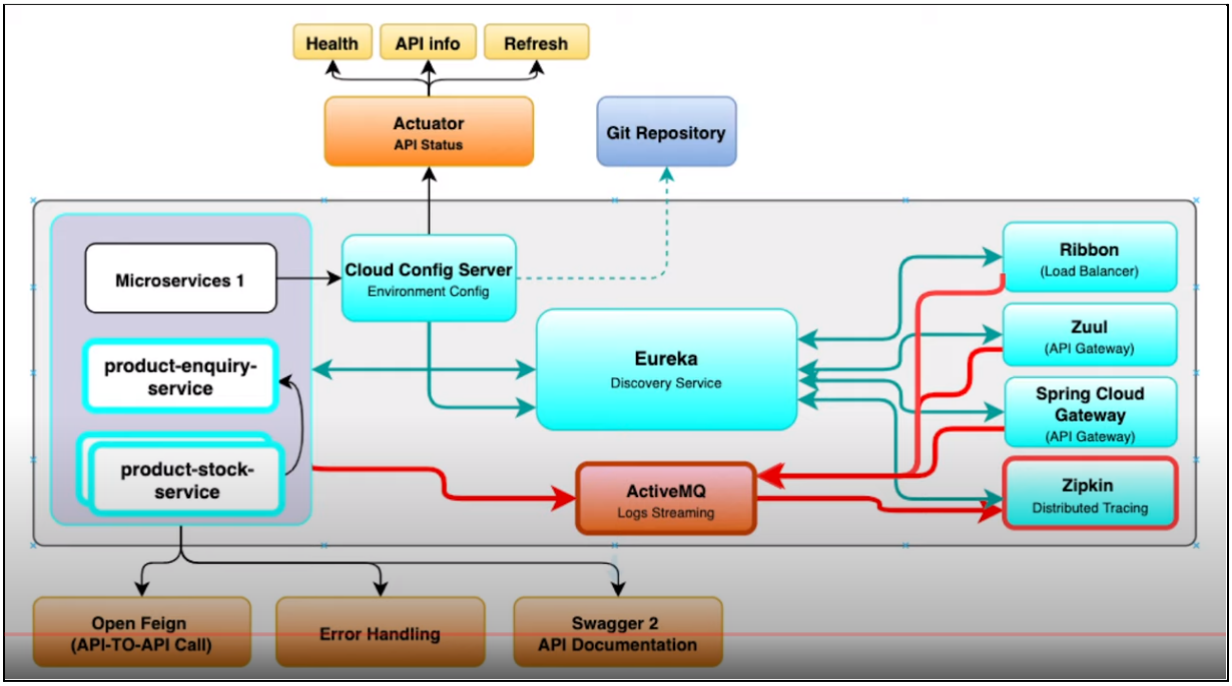
```
spring:
 application:
   name: techefx-spring-cloud-api-gateway-service

#  cloud:
#    gateway:
#      routes:
#       - id: stock-enquiry
#         uri: http://localhost:8700/
#         predicates:
#           - Path=/product-enquiry/**

server:
 port: ${port:8900}
```

localhost:8900/product-enquiry/name/bat/availability/yes/unit/3

There was a 404 response when @Bean method was in a separate configuration class!

netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server

Zipkin



1. https://www.erlang.org/downloads

2. https://www.rabbitmq.com/install-windows.html#installer

Runs as a service.

3. **https://zipkin.io/pages/quickstart**

Java

If you have Java 8 or higher installed, the quickest way to get started is to fetch the latest release as a self-contained executable jar:

```
curl -sSL https://zipkin.io/quickstart.sh | bash -s
java -jar zipkin.jar
```

Jaca@DESKTOP-G102BV9 MINGW64 ~
$ java -jar zipkin.jar

```
        oo
       oooo
      oooooo
     oooooooo
    oooooooooo
```

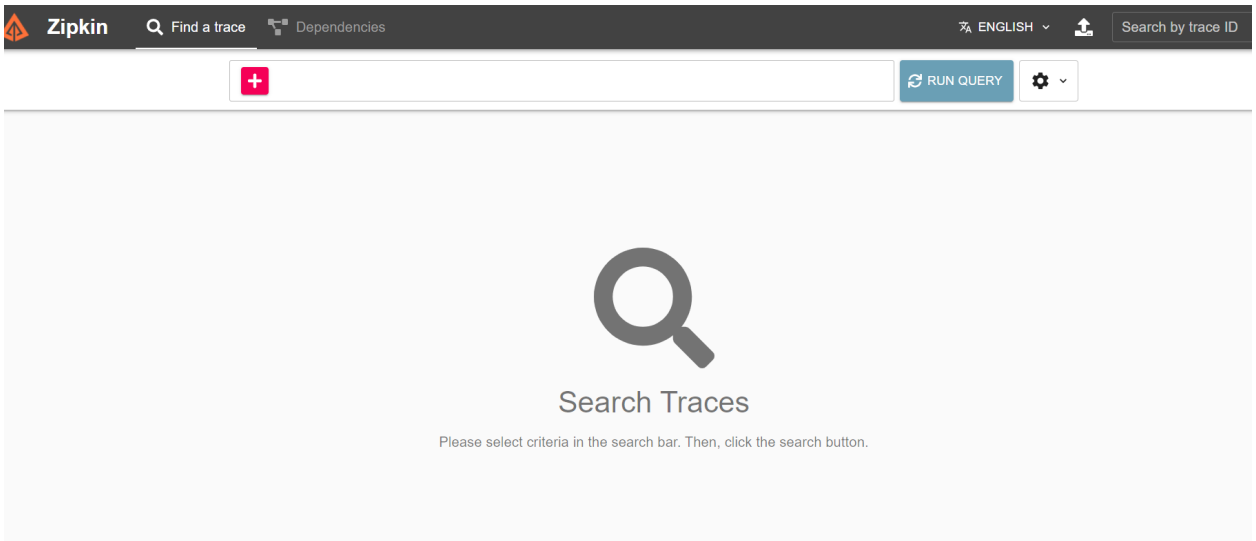
netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server

```
          ooooooooooooo
        oooooooo  oooooooo
       oooooo     oooooooo
      oooooo       oooooooo
     oooooo   o  o   oooooo
    oooooo   oo  oo   oooooo
   oooooooo  oooo  oooo  oooooooo
  oooooo   ooooo  ooooo  oooooooo
 oooooo   oooooo  oooooo  oooooooo
oooooooo      oo  oo      oooooooo
oooooooooooooo oo  oo oooooooooooooo
   oooooooooooo  oooooooooooo
      oooooooo  oooooooo
         oooo  oooo

  _________ ____  _  _____ _   _
 |__  /_ _|  _ \| |/ /_ _| \ | |
   / / | || |_) | ' / | ||  \| |
  / /_ | ||  __/| . \ | || |\  |
 |____|___|_|   |_|\_\___|_| \_|
```

:: version 2.23.2 :: commit 7bf3aab ::

2021-04-18 01:22:24.372 INFO [/] 4972 --- [oss-http-*:9411] c.l.a.s.Server : Serving HTTP at /0:0:0:0:0:0:0:0:9411 - http://127.0.0.1:9411/

4. Add 3 dependencies to pom - product-stock and enquiry service , gateway, http://localhost:9411/zipkin/

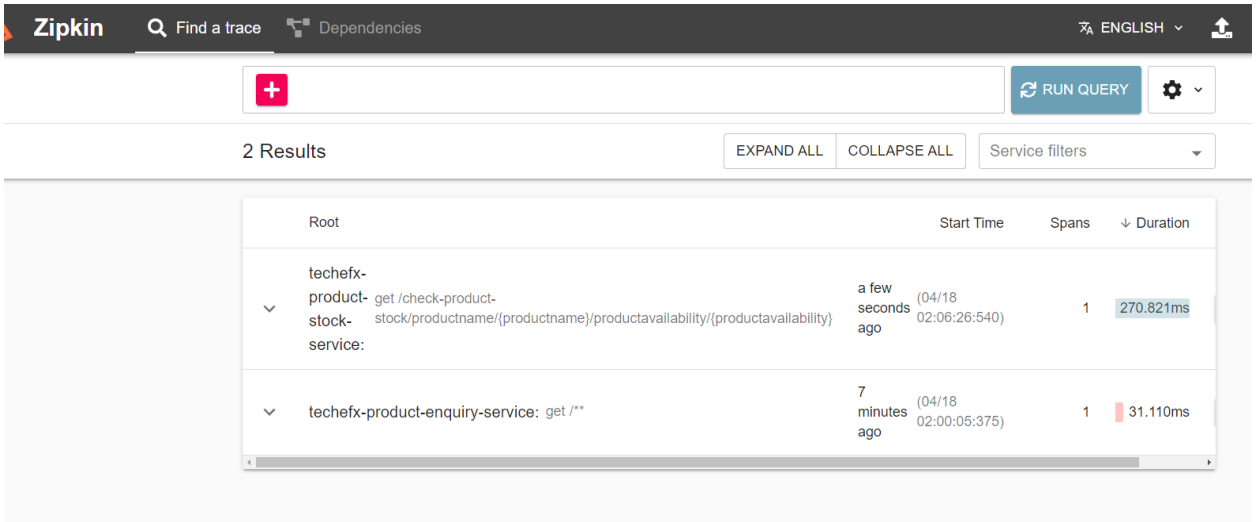


Product and enquiry running with eureka

netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server

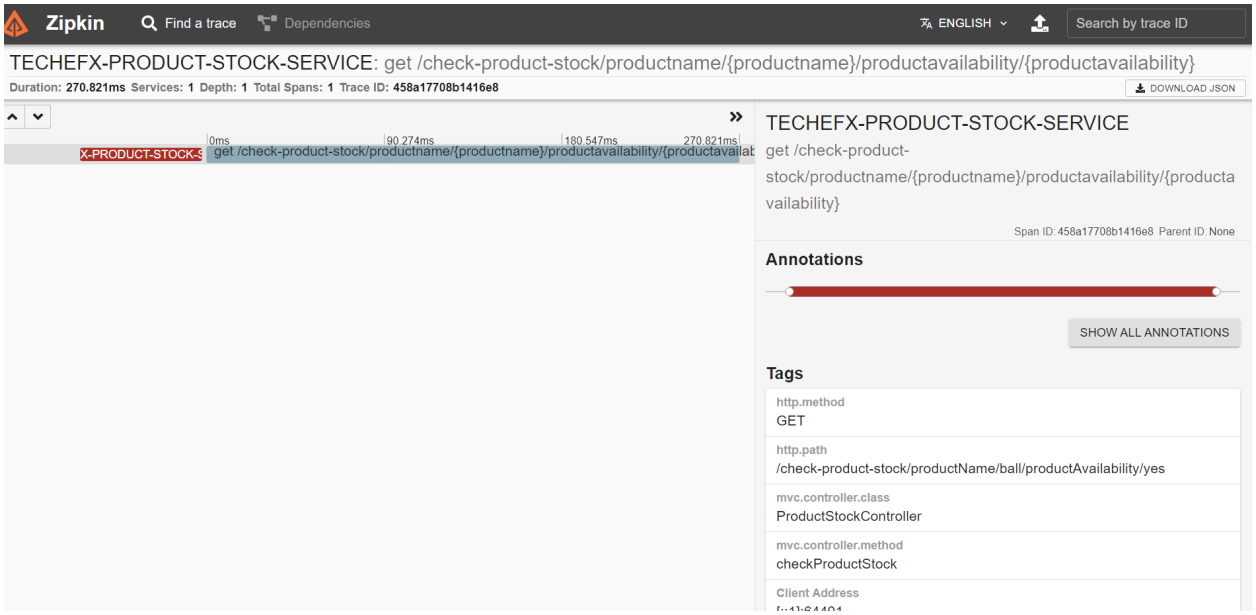
Postman

localhost:8800/check-product-stock/productName/ball/productAvailability/yes

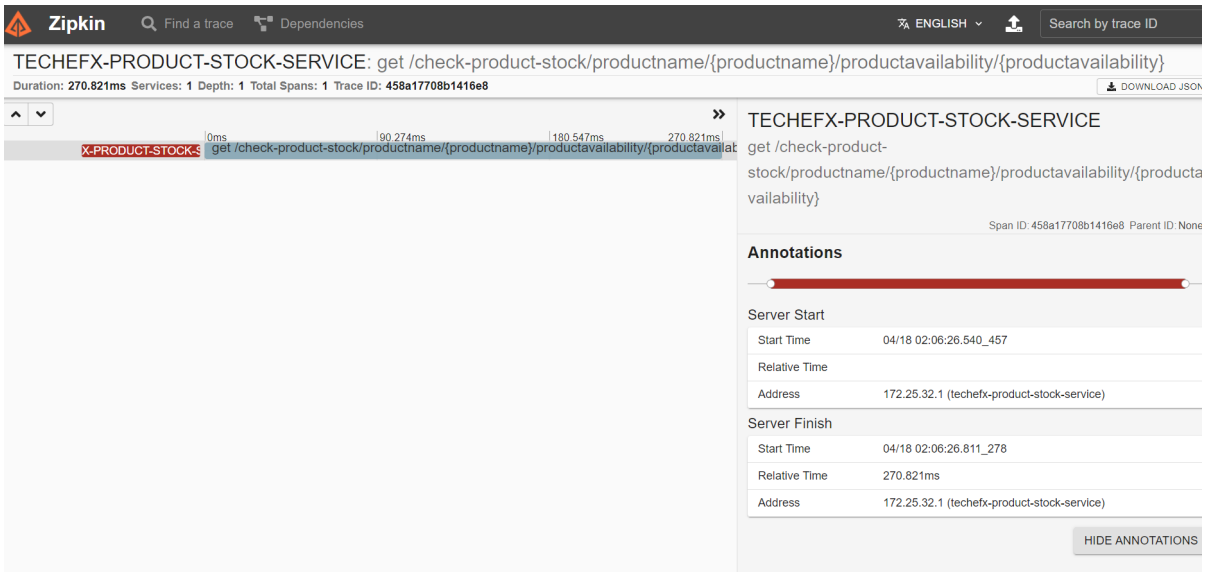


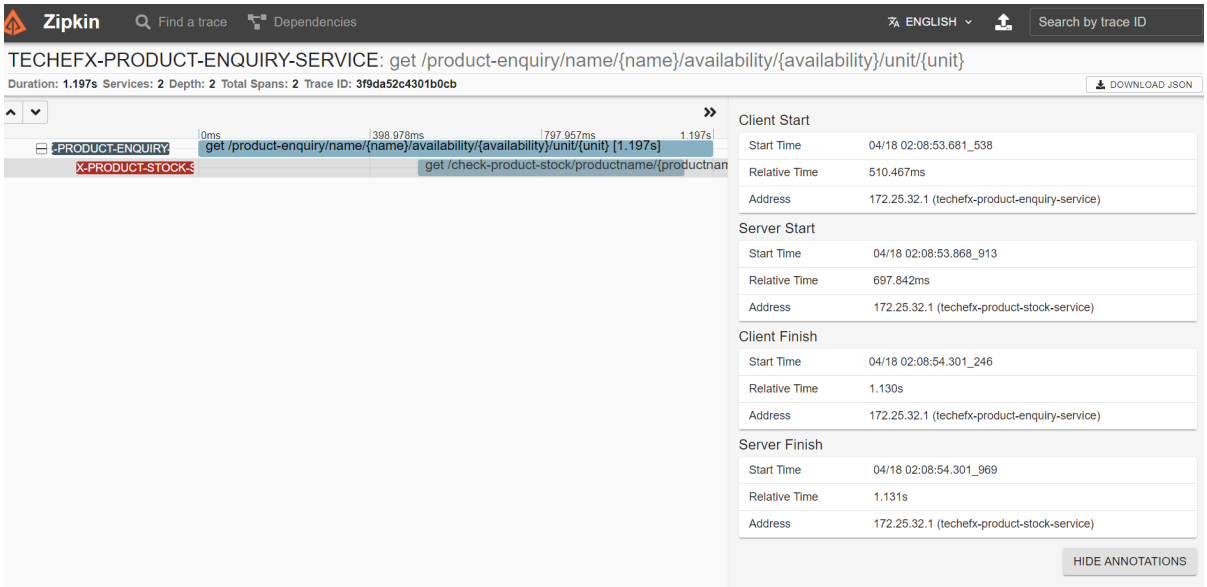
Postman

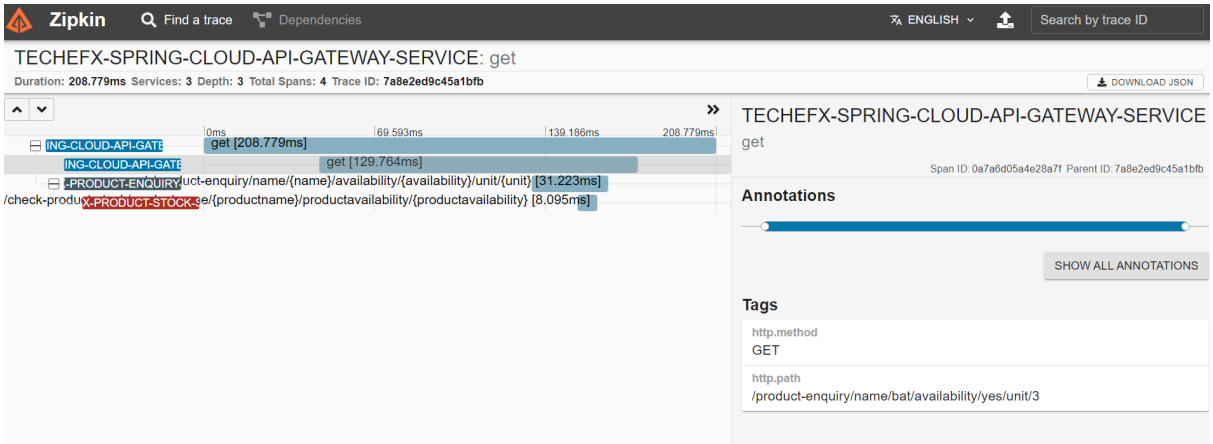
localhost:8700/product-enquiry/name/bat/availability/yes/unit/3



netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server







netflix naming server
netflix naming and discovery server
netflix eurka naming and discovery server