Contents

[1 Theory 1](#_Toc94892257)

[2 Architecture 2](#_Toc94892258)

[3 Eureka Client Implementation 3](#_Toc94892259)

[3.1 First Implementation 3](#_Toc94892260)

[3.2 How to deregister Eureka Client Gracefully 4](#_Toc94892261)

[4 Feign Client 4](#_Toc94892262)

[4.1 Frist Implementation 4](#_Toc94892263)

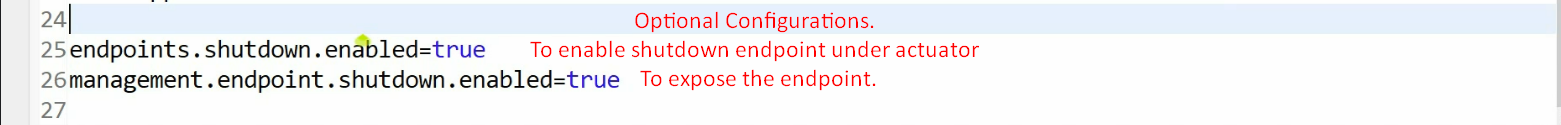
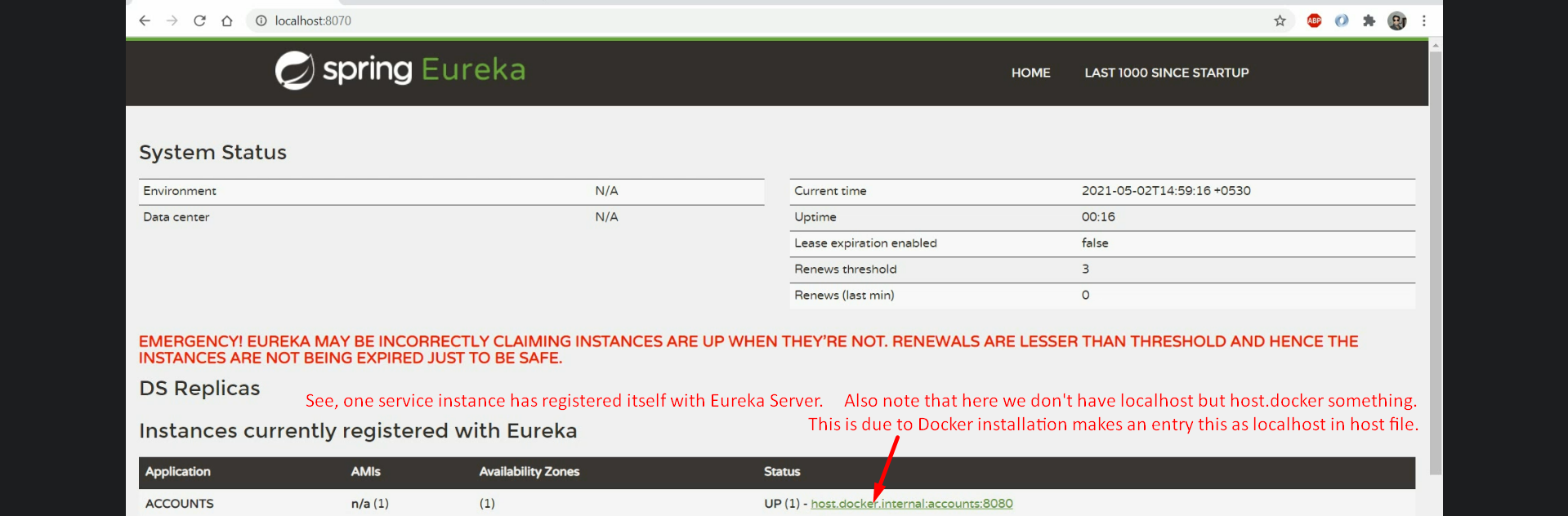
[5 How to Deregister from Eureka Server when microservices shutdown 5](#_Toc94892264)

# Theory

# Architecture

# Eureka Client Implementation

## First Implementation

1. **Step 01**: Add dependency
   1. 
      1. So that ms can register itself with Eureka Server with a logical name.
      2. So that registered ms can fetch others’ MSs details.
2. **Step 02**: Configuration inside **application.properties**
   1. 
      1. Location of Eureka Server along with the context where Service Registry Service is running.
      2. Using this info, Eureka client would register itself with the Eureka Server.  
         Actually, Eureka Server exposes some end-points using which Eureka Client registers itself.
      3. Default N/w Location along with context is   
         **localhost:8761/eureka**
   2. 
   3. 
3. **Step 03**: Run the Eureka Server then Eureka Client.  
   

## How to deregister Eureka Client Gracefully

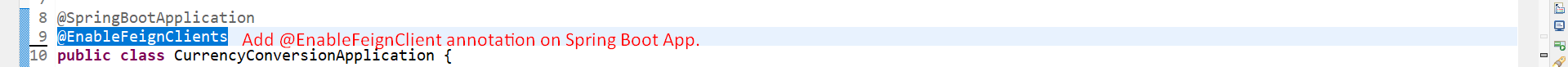
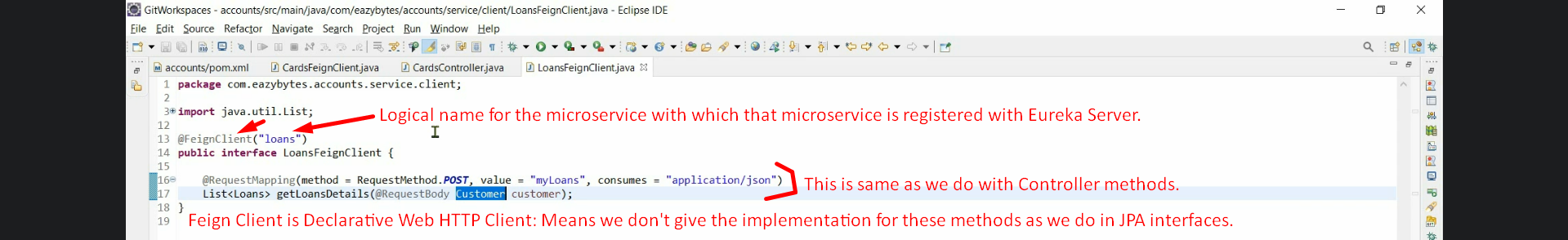
1. d

# Feign Client

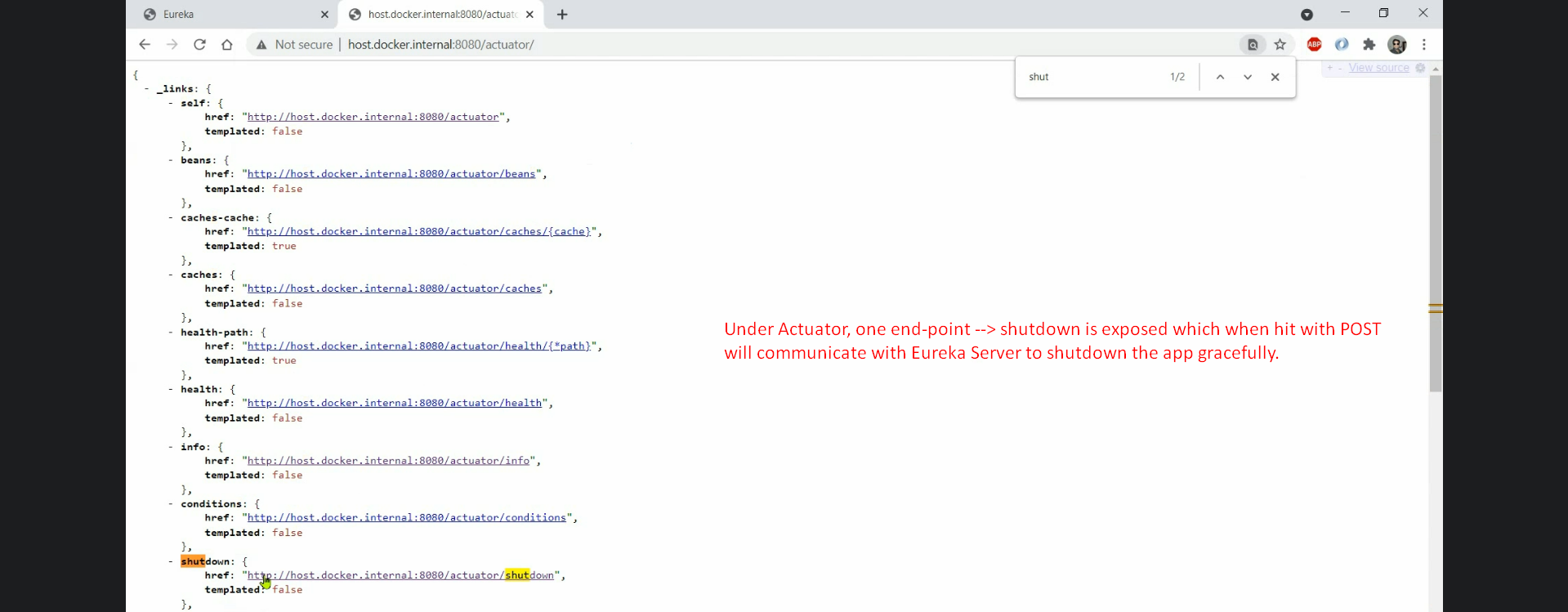
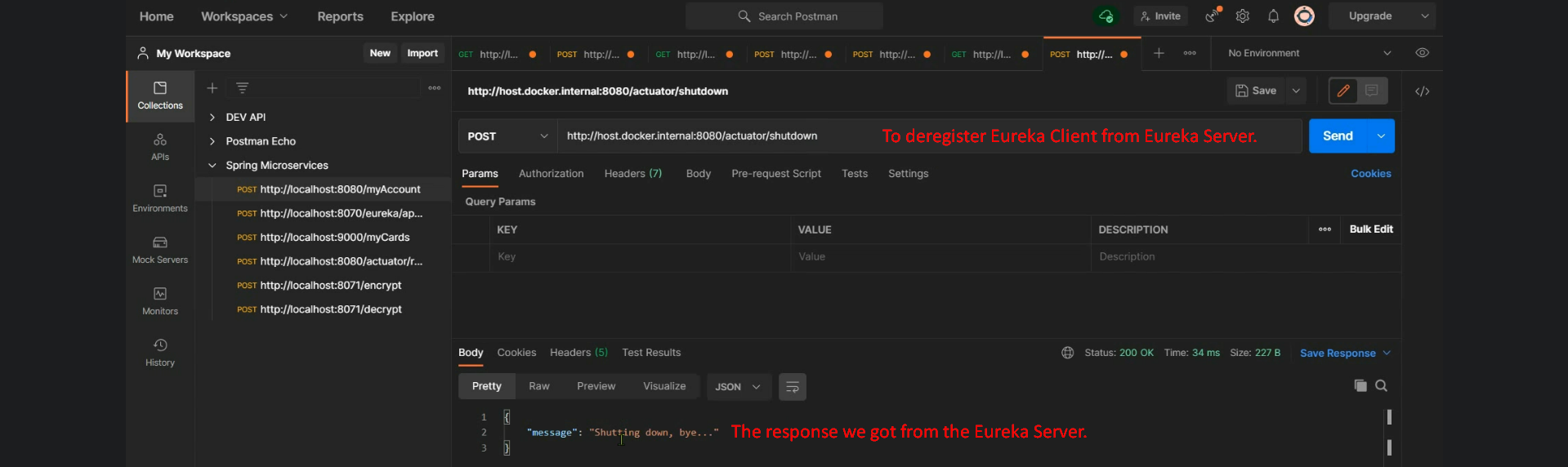
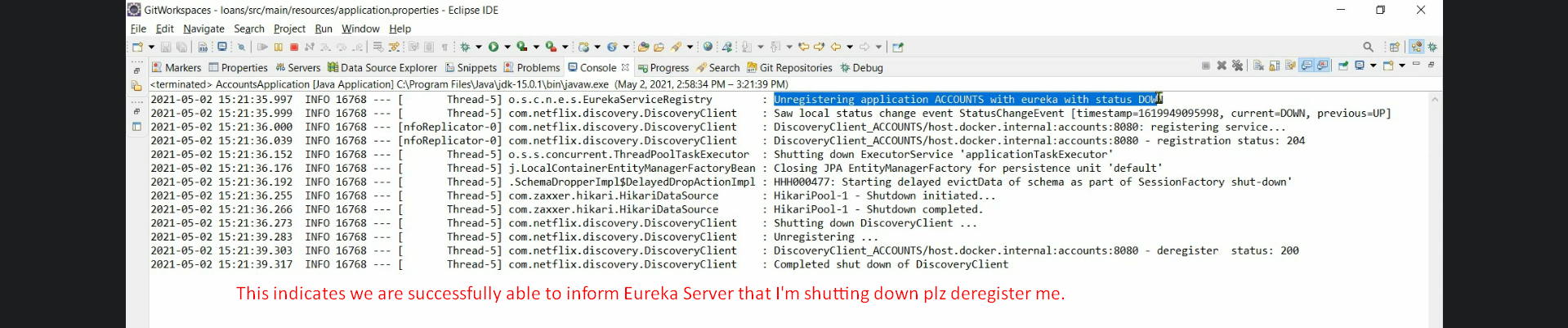
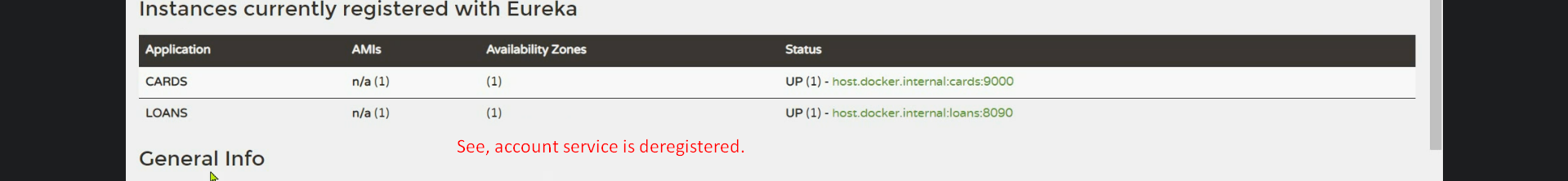
## Frist Implementation

1. **Step 01**: **Add dependencies**



1. **Step 02**: **Add @EnableFeignClient annotation to Spring Boot App**.  
   
2. **Step 03**: **Create interface annotated with @FeignClient with HTTP methods to call business logic from other microservices**.  
     
   Bean for this interface will be created automatically and inject that bean to use it.  
   This feign client will connect with Eureka Server to fetch the details about the microservice with logical name “loans”.   
   **NOTE**: To connect with Eureka Server as client, we need to add **Eureka Client Dependency**.  
   Feign-client also does load balancing by using Spring Cloud Load Balancer which is added automatically by Eureka Client/Eureka Server dependency automatically.

# How to Deregister from Eureka Server when microservices shutdown

1. 
2. 
3. 
4. 
5. What happens if fore shutdown happens which doesn’t give chance to ms instance to deregister itself?
   1. In that case, Eureka Server will not receive heart-beat from Eureka Client. It gives 3 chances to Eureka Client if after 30sec 3 times it doesn’t receive heart-beat, Eureka Server will deregister that service.
6. There is concept self-preservative. We will discuss that.