Name : Arshiya Tabassum A

Superset ID : 6424209

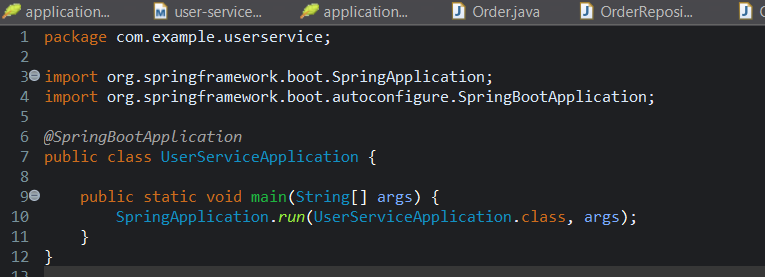
**WEEK 5**

**Exercises on Microservices with Spring Boot 3.0**

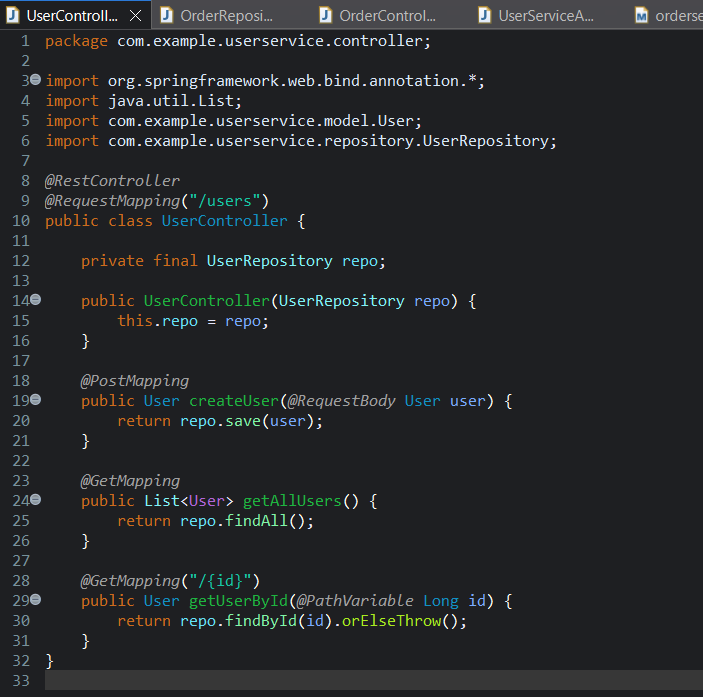
**1.Build a User and Order Management System**

**User service**

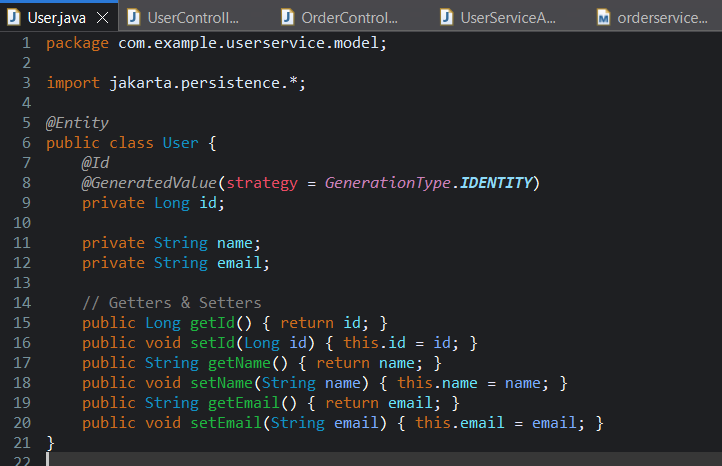
**UserServiceApplication.java**

****

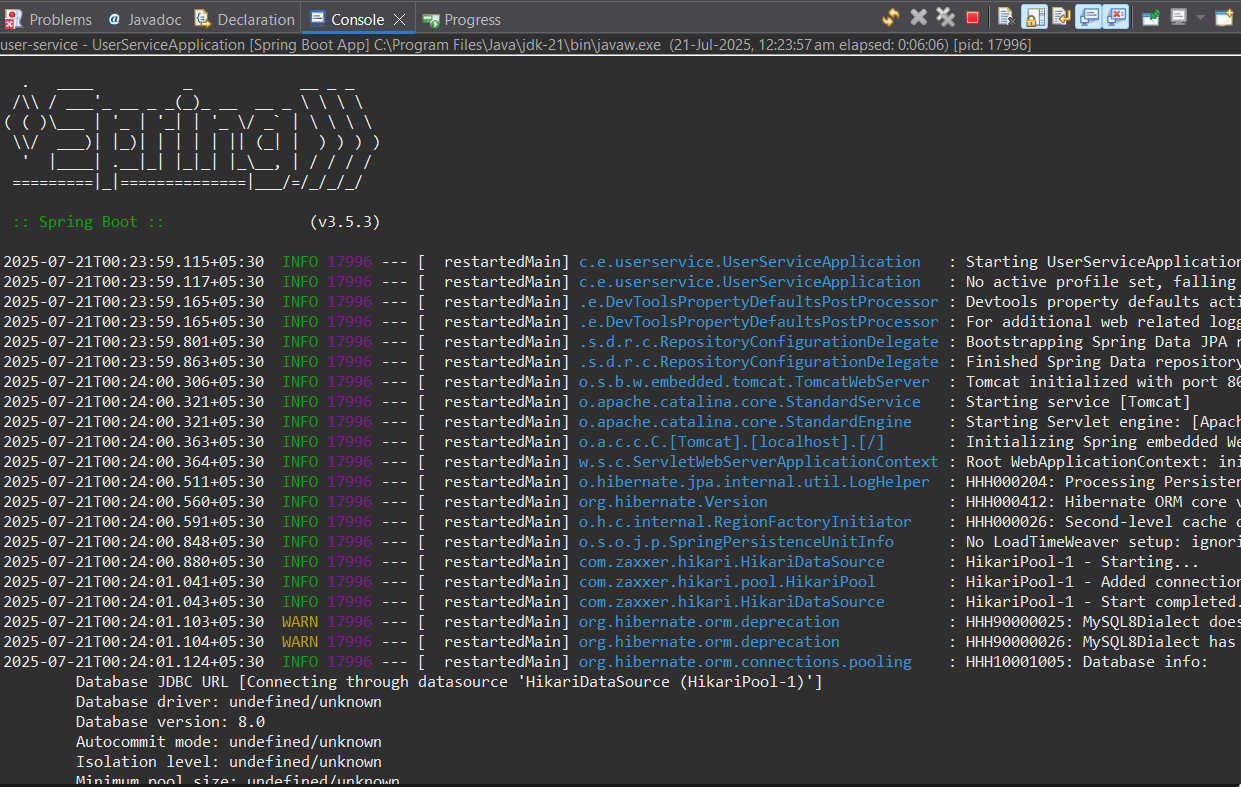
**UserController.java**

****

**User.java**

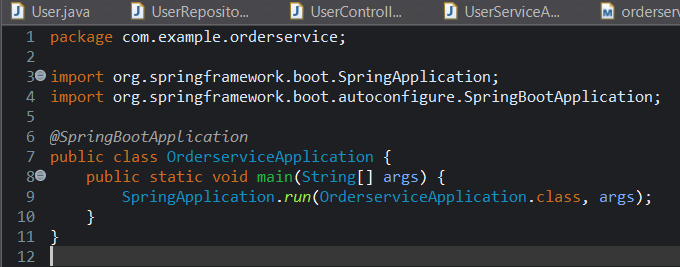
****

**Output :**

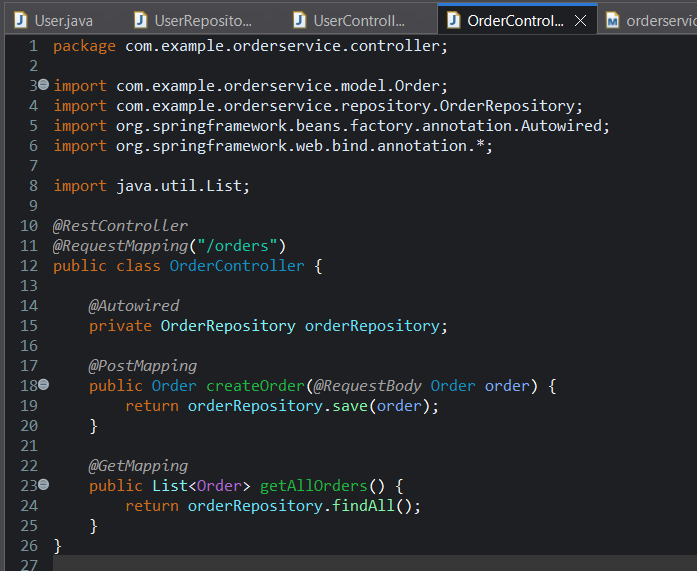
****

**OrderService**

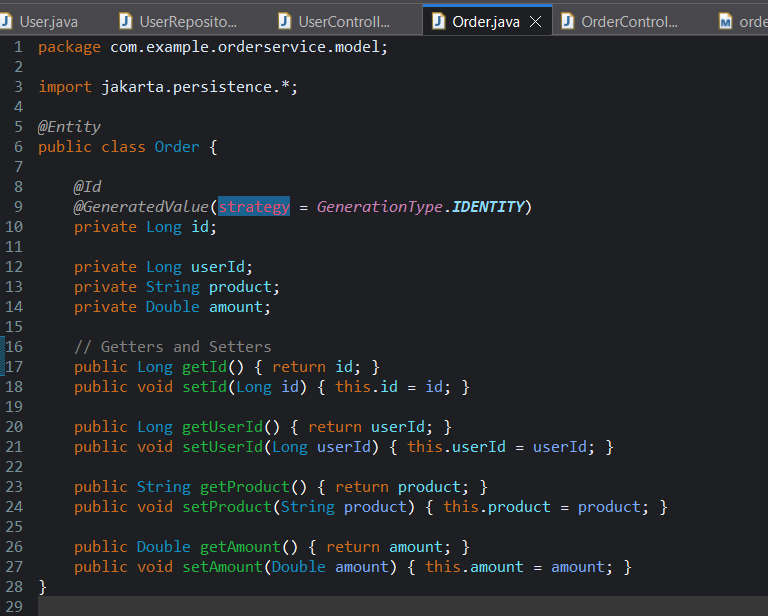
**OrderServiceApplication.java**

****

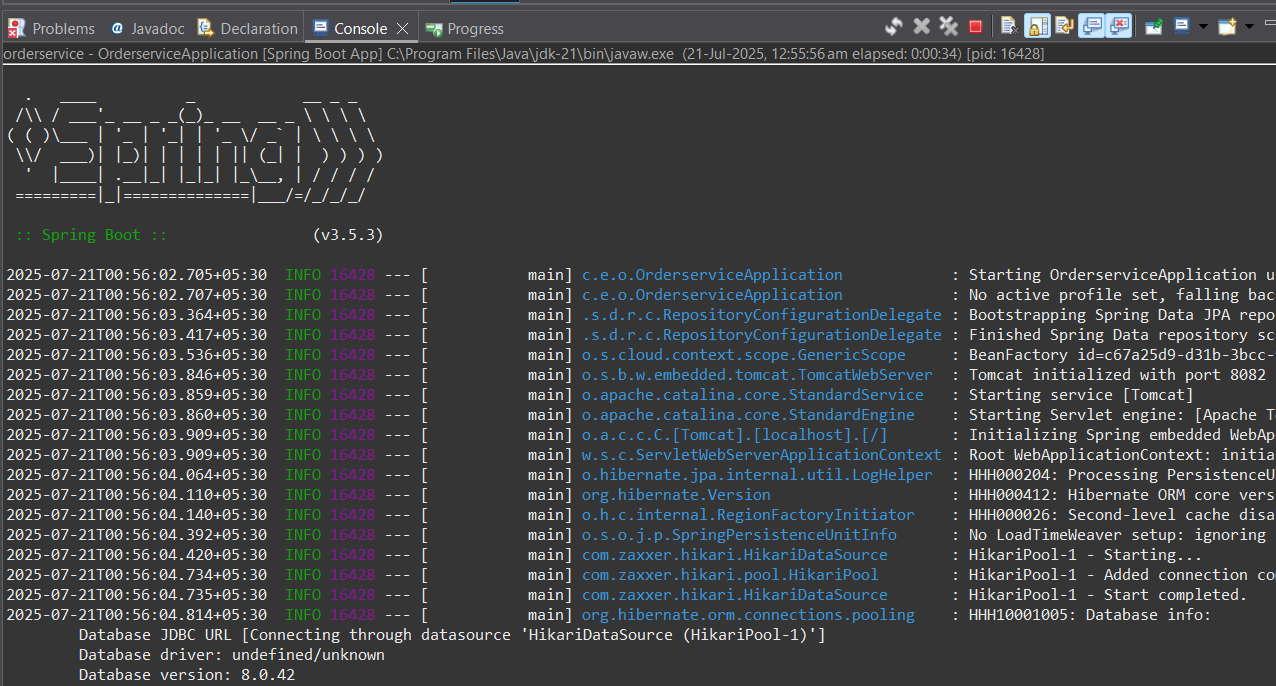
**OrderController.java**

****

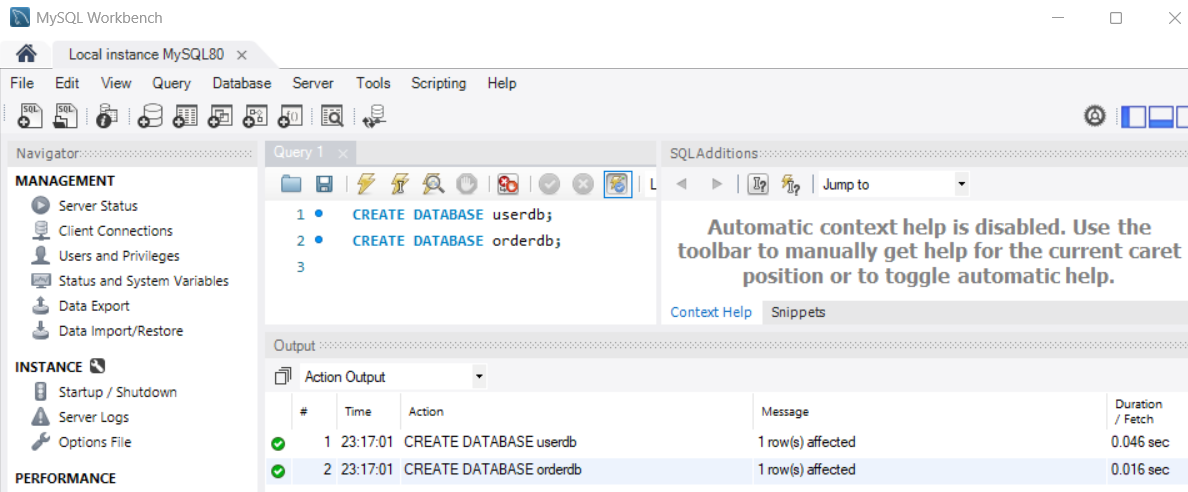
**Order.java**

****

**Output :**

****

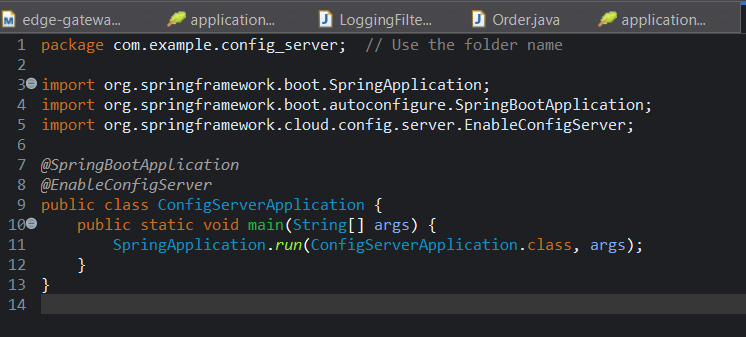
**Mysql :**

****

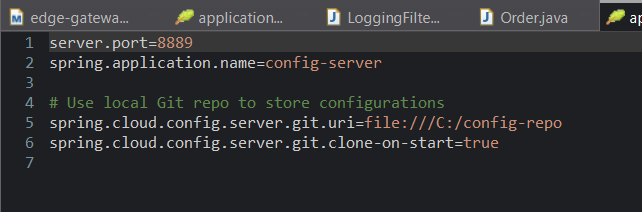
**2.Inventory Management System with Service Discovery**

**Product service**

**ConfigureServiceApplication.java**

****

**Application.properties**

****

**Pom.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>config-server</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>config-server</name>

<description>Spring Cloud Config Server</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.4.4</version>

<relativePath/>

</parent>

<properties>

<java.version>17</java.version>

<spring-cloud.version>2024.0.0</spring-cloud.version>

</properties>

<dependencies>

<!-- Config Server -->

<dependency>

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

<artifactId>spring-cloud-config-server</artifactId>

</dependency>

<!-- Web Starter -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<!-- Test -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

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

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

<version>${spring-cloud.version}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

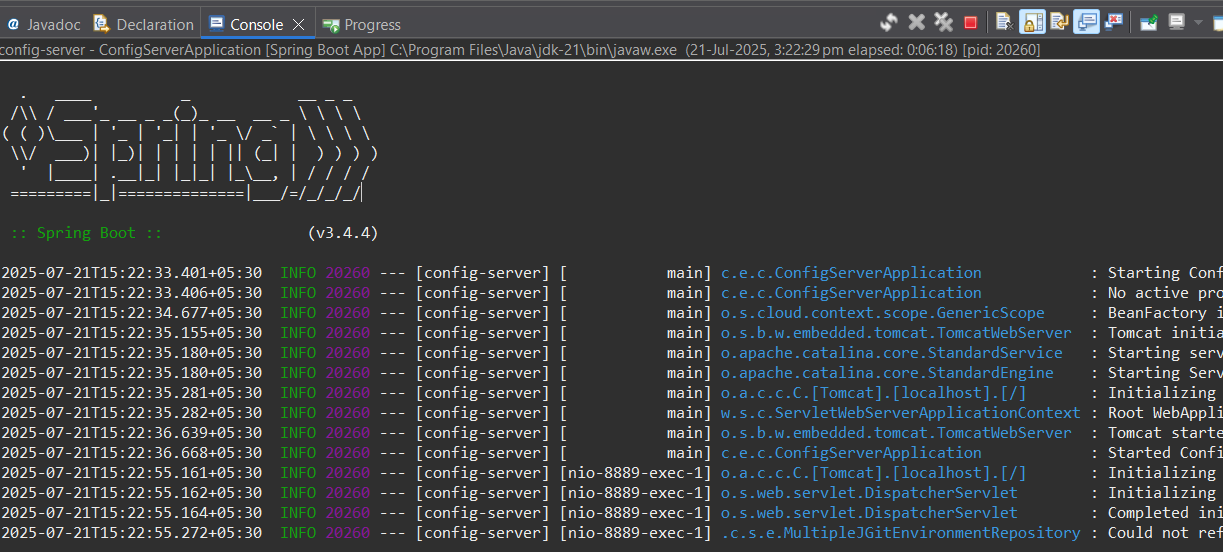
</plugin>

</plugins>

</build>

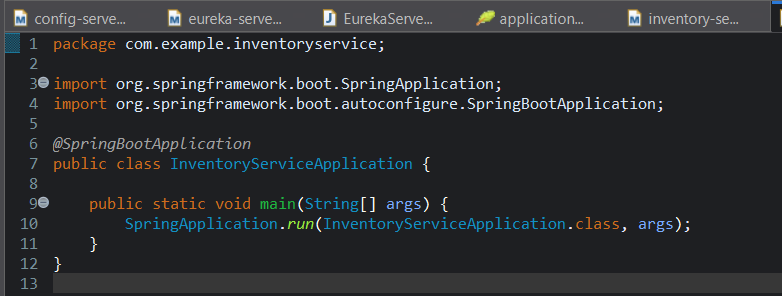
</project>

**Output :**

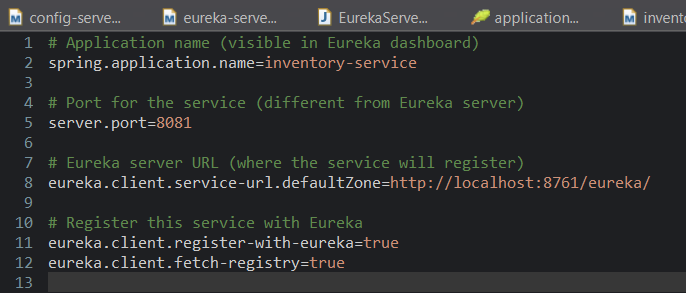
****

**Inventory-Service**

**InventoryserviceApplication.java**

****

**application.properties**

****

**Pom.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>inventory-service</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>inventory-service</name>

<description>Inventory Service (Eureka Client)</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.4.7</version>

<relativePath/>

</parent>

<properties>

<java.version>17</java.version>

<spring-cloud.version>2024.0.0</spring-cloud.version>

</properties>

<dependencies>

<!-- Spring Web -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<!-- Eureka Client -->

<dependency>

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

<artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>

</dependency>

<!-- Actuator (for health checks) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

<!-- Test -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

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

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

<version>${spring-cloud.version}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

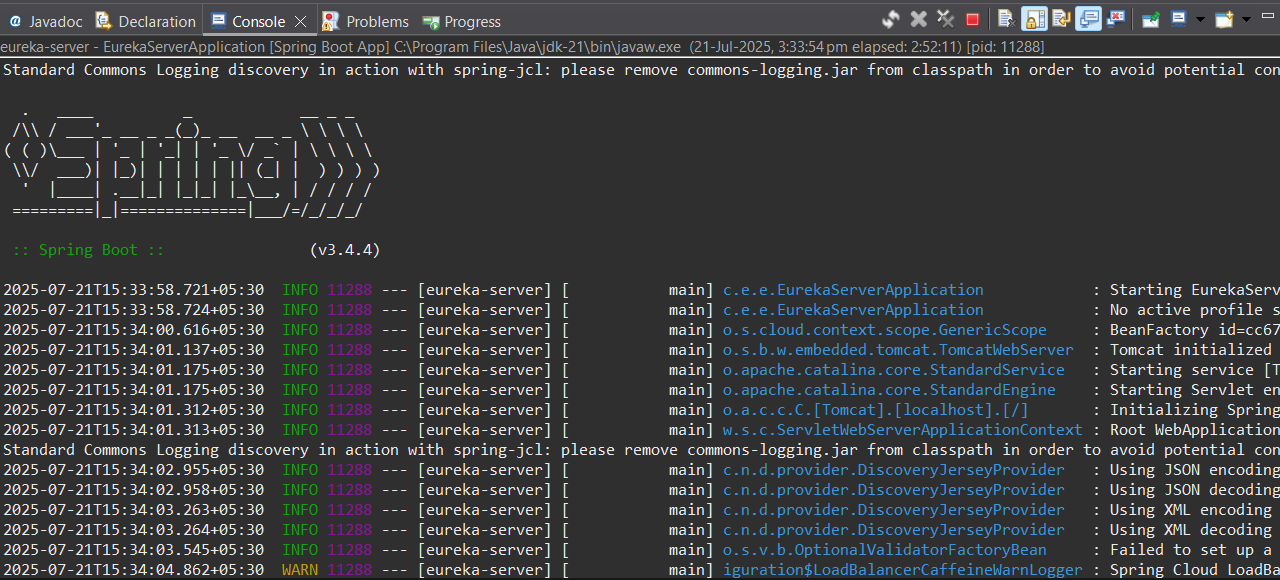
</plugin>

</plugins>

</build>

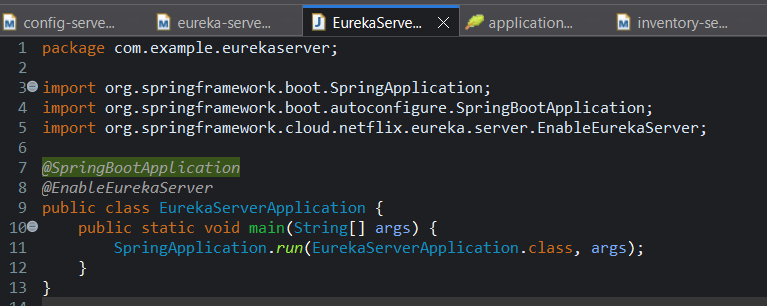
</project>

**Output :**

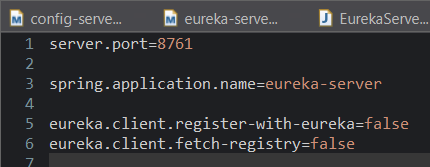
****

**Eurekaserver**

**EurekaserverApplication.java**

****

**Application.properties**

****

**Pom.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>eureka-server</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>eureka-server</name>

<description>Eureka Service Discovery</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.4.7</version> <!-- ✅ Latest compatible version -->

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<java.version>17</java.version>

<spring-cloud.version>2024.0.0</spring-cloud.version>

</properties>

<dependencies>

<!-- ✅ Eureka Server -->

<dependency>

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

<artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>

</dependency>

<!-- ✅ Web Starter -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<!-- ✅ For Validation API (optional but useful for bean validation) -->

<dependency>

<groupId>org.hibernate.validator</groupId>

<artifactId>hibernate-validator</artifactId>

</dependency>

<!-- ✅ Test -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

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

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

<version>${spring-cloud.version}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

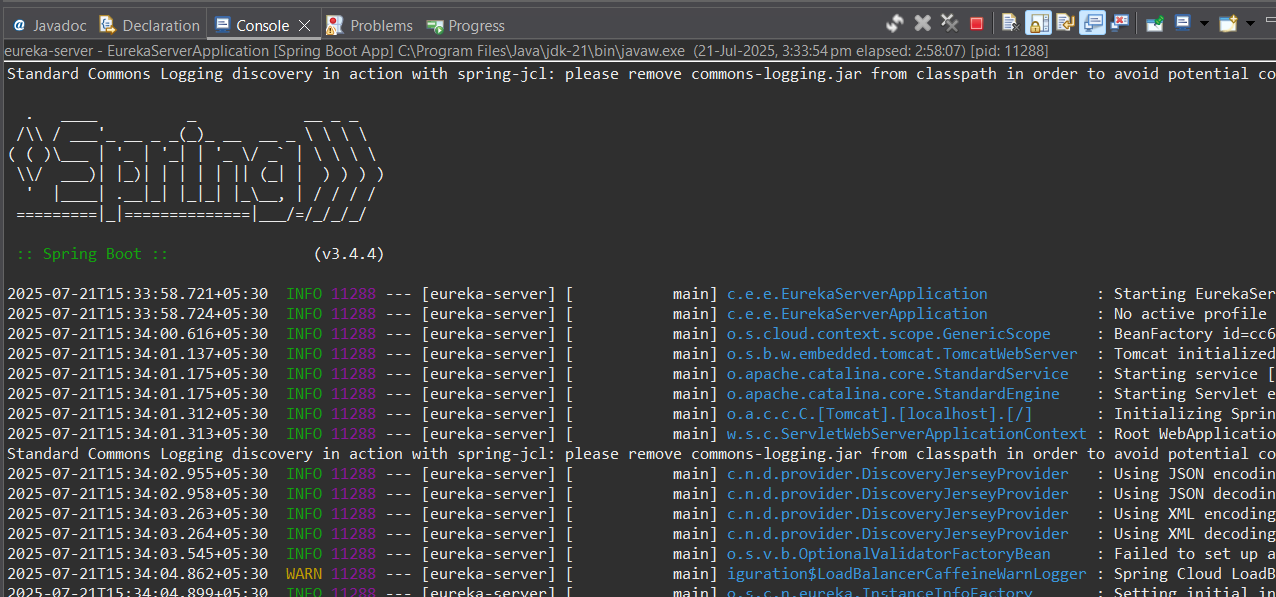
</plugin>

</plugins>

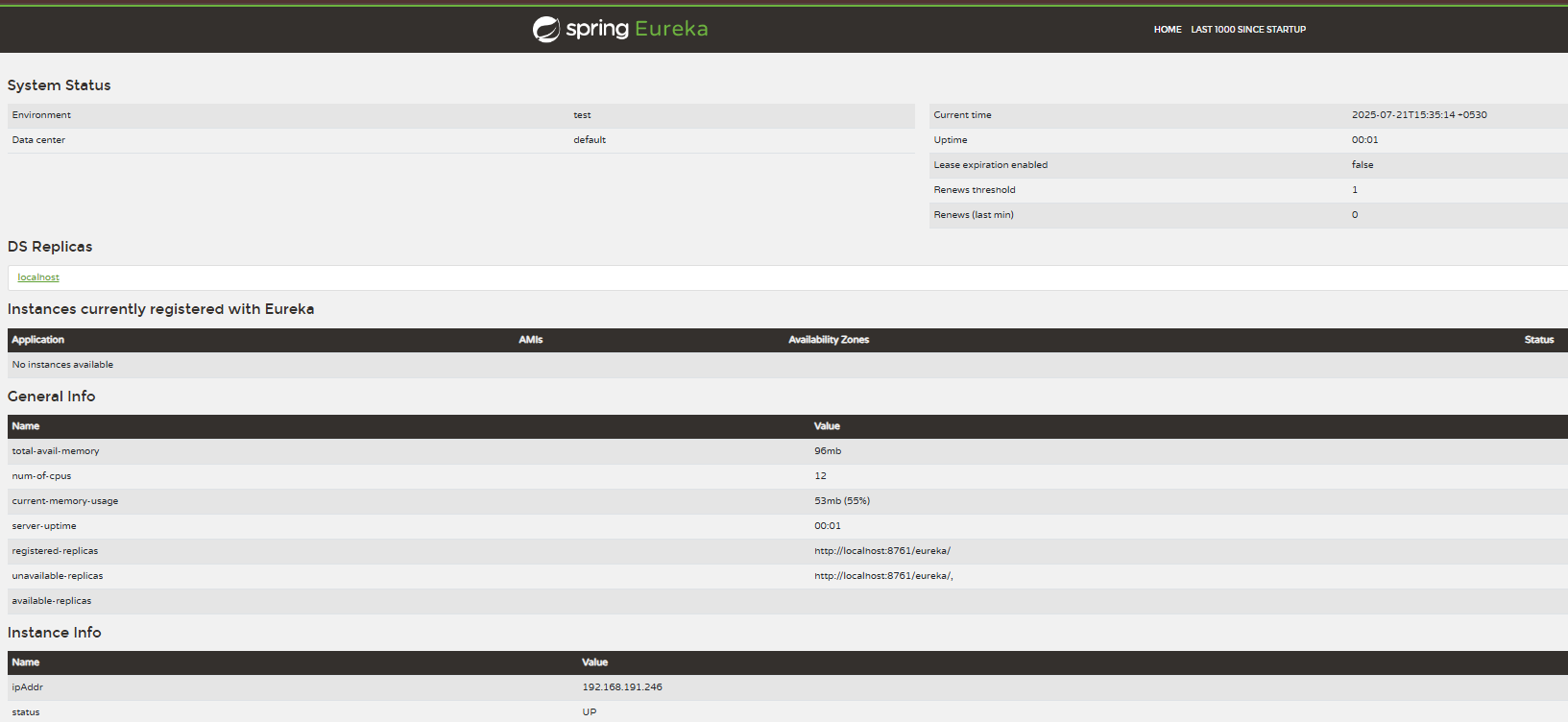
</build>

</project>

**Output :**

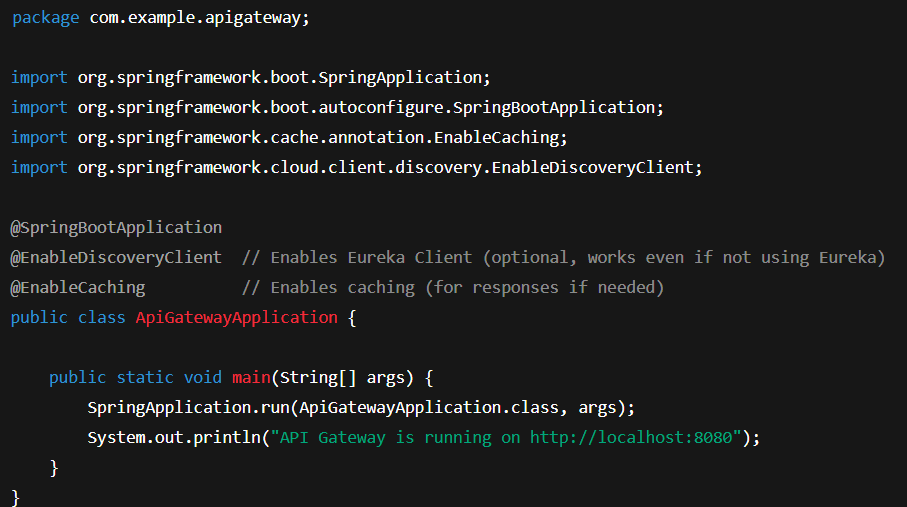
****

**EurekaServer**

****

**3. Implement an API Gateway**

**Apigatewayapplication.java**

****

**Application.yml**

server:

port: 8080

spring:

application:

name: api-gateway

cloud:

gateway:

routes:

- id: customer-service

uri: http://localhost:8081

predicates:

- Path=/customers/\*\*

filters:

- RewritePath=/customers/(?<segment>.\*), /${segment}

- RequestRateLimiter=redis-rate-limiter.replenishRate=5, redis-rate-limiter.burstCapacity=10

- id: billing-service

uri: http://localhost:8082

predicates:

- Path=/billing/\*\*

filters:

- RewritePath=/billing/(?<segment>.\*), /${segment}

- RequestRateLimiter=redis-rate-limiter.replenishRate=3, redis-rate-limiter.burstCapacity=5

default-filters:

- AddResponseHeader=X-Cache, CACHED

spring:

cache:

type: simple

spring:

redis:

host: localhost

port: 6379

**pom.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<!-- Spring Boot Parent for Dependency and Plugin Management -->

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.3.2</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.example</groupId>

<artifactId>api-gateway</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>api-gateway</name>

<description>Spring Cloud API Gateway</description>

<properties>

<java.version>17</java.version>

<spring.cloud.version>2023.0.1</spring.cloud.version>

</properties>

<dependencies>

<!-- Spring Boot WebFlux (required for Gateway) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-webflux</artifactId>

</dependency>

<!-- Spring Cloud Gateway -->

<dependency>

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

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

</dependency>

<!-- Redis Rate Limiter (for RequestRateLimiter filter) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-redis-reactive</artifactId>

</dependency>

<!-- Optional: Eureka Client for Service Discovery -->

<dependency>

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

<artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>

</dependency>

<!-- Caching Support -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-cache</artifactId>

</dependency>

<!-- Lombok (Optional for Reducing Boilerplate Code) -->

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<!-- Spring Boot Test (Optional for Testing) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

<exclusions>

<exclusion>

<groupId>org.junit.vintage</groupId>

<artifactId>junit-vintage-engine</artifactId>

</exclusion>

</exclusions>

</dependency>

</dependencies>

<!-- Spring Cloud BOM for Version Management -->

<dependencyManagement>

<dependencies>

<dependency>

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

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

<version>${spring.cloud.version}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

<build>

<plugins>

<!-- Spring Boot Maven Plugin -->

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

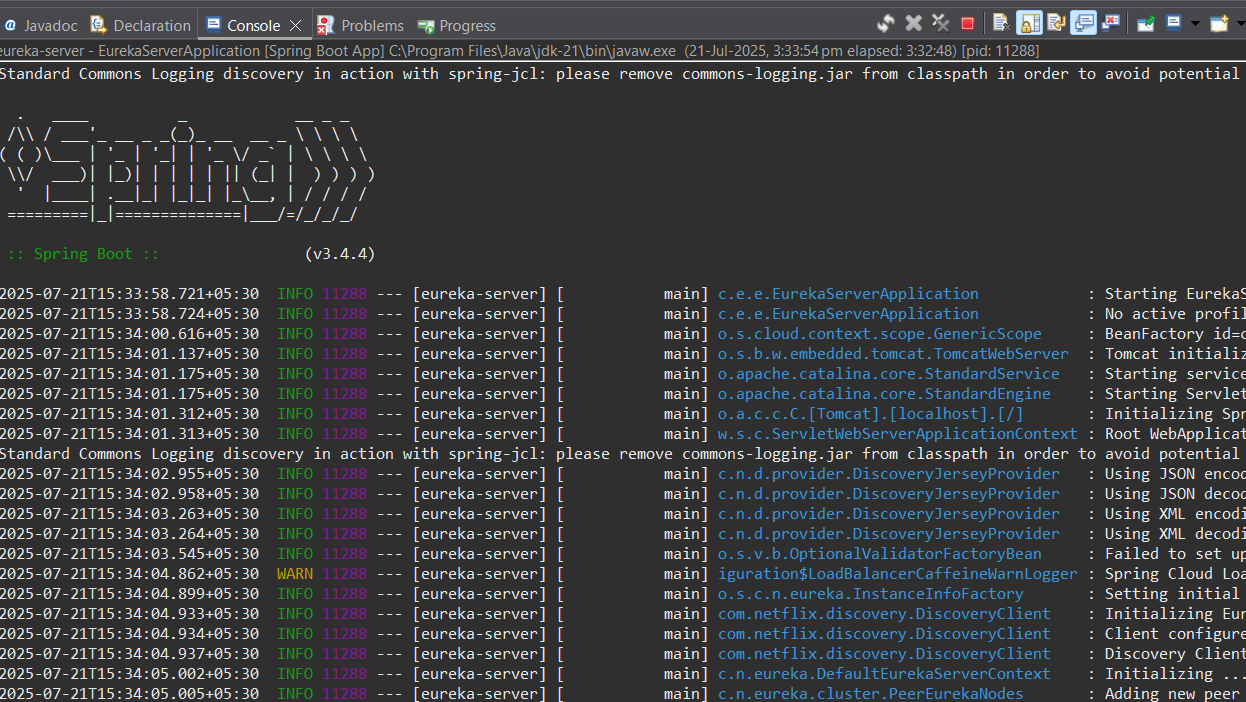
</plugin>

</plugins>

</build>

</project>

**Output :**

****

**4. Resilient Microservices with Circuit Breaker**

**PaymentServiceApplication.java**

package com.example.paymentservice;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@*SpringBootApplication

public class PaymentServiceApplication {

public static void main(String[] args) {

SpringApplication.run(PaymentServiceApplication.class, args);

System.***out***.println("Payment Service is running on http://localhost:8083");

}

}

**PaymentController.java**

package com.example.paymentservice.controller;

import com.example.paymentservice.service.PaymentService;

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

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

import java.util.concurrent.CompletableFuture;

*@*RestController

public class PaymentController {

private final PaymentService paymentService;

public PaymentController(PaymentService paymentService) {

this.paymentService = paymentService;

}

*@*GetMapping("/pay")

public CompletableFuture<String> makePayment() {

return paymentService.processPayment();

}

}

**PaymentService.java**

package com.example.paymentservice.service;

import io.github.resilience4j.circuitbreaker.annotation.CircuitBreaker;

import io.github.resilience4j.timelimiter.annotation.TimeLimiter;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Service;

import java.util.concurrent.CompletableFuture;

*@*Service

public class PaymentService {

private static final Logger ***logger*** = LoggerFactory.getLogger(PaymentService.class);

private static final String ***CB\_NAME*** = "paymentAPI";

*@*CircuitBreaker(name = ***CB\_NAME***, fallbackMethod = "fallbackPayment")

*@*TimeLimiter(name = ***CB\_NAME***)

public CompletableFuture<String> processPayment() {

return CompletableFuture.*supplyAsync*(() -> {

***logger***.info("Calling third-party payment API...");

try {

Thread.*sleep*(4000); // Simulate slow API (4 seconds)

} catch (InterruptedException e) {

throw new RuntimeException(e);

}

return "Payment processed successfully!";

});

}

// Fallback method

public CompletableFuture<String> fallbackPayment(Throwable t) {

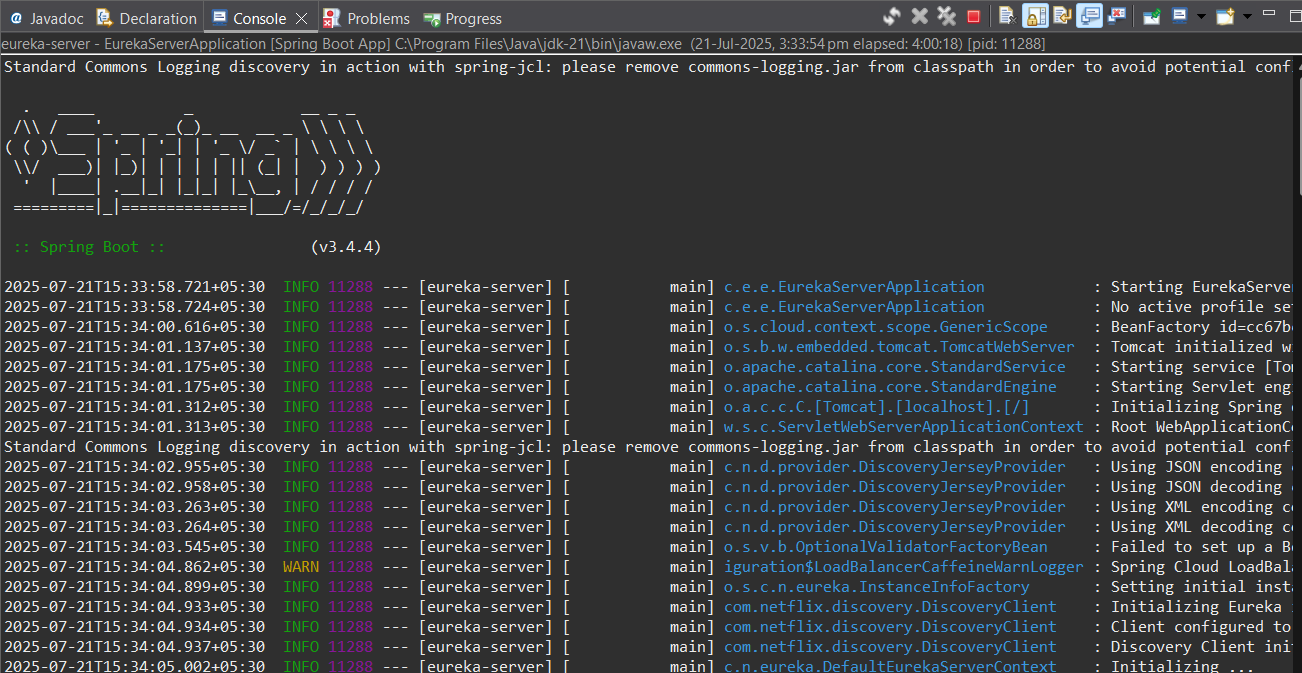
***logger***.error("Fallback triggered due to: {}", t.toString());

return CompletableFuture.*completedFuture*("Payment Service is busy. Please try again later.");

}

}

**Output :**

****