Microservices

Spring Cloud Eureka

Service Discovery & Registry

Using Spring Cloud Eureka

Service Discovery is a mechanism used in microservices architectures to automatically detect and connect services. It helps services find each other without hardcoding the network locations (IP addresses and ports). This is crucial in dynamic environments where services can scale up or down, and their locations can change frequently.

Service Registry is a database that keeps track of all the available service instances. When a service starts, it registers itself with the service registry, providing its network location. Other services can then query the registry to find the location of the service they need to communicate with

Spring Cloud Eureka

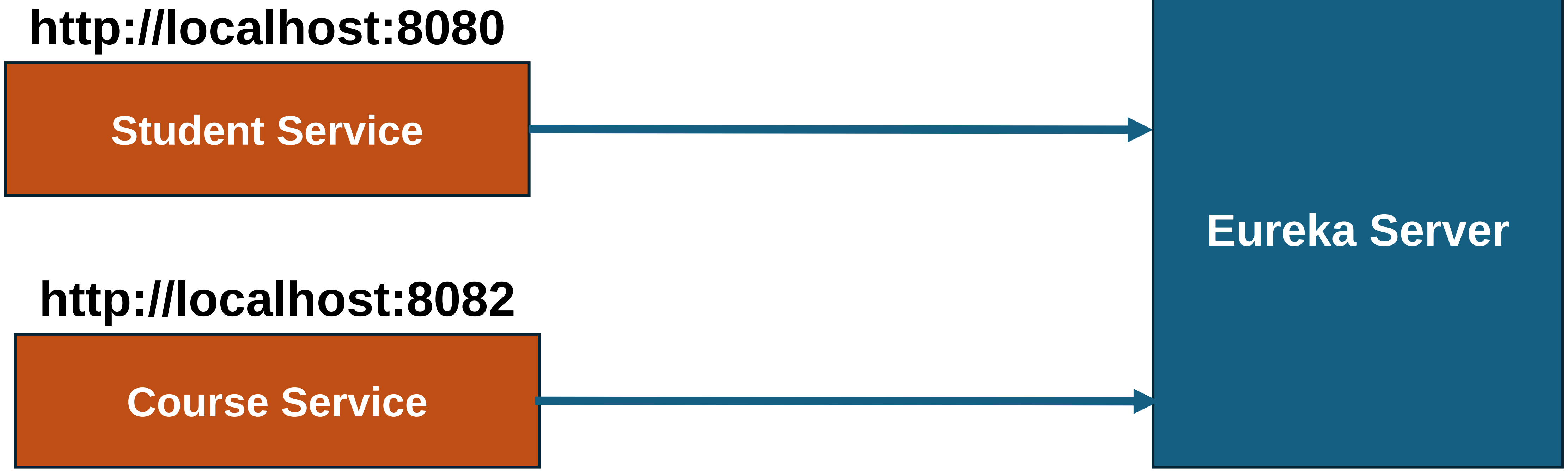
Spring Cloud Eureka is a part of the Spring Cloud Netflix project for Spring Boot applications.

Eureka is a service registry that allows microservices to register themselves at runtime and discover other services.

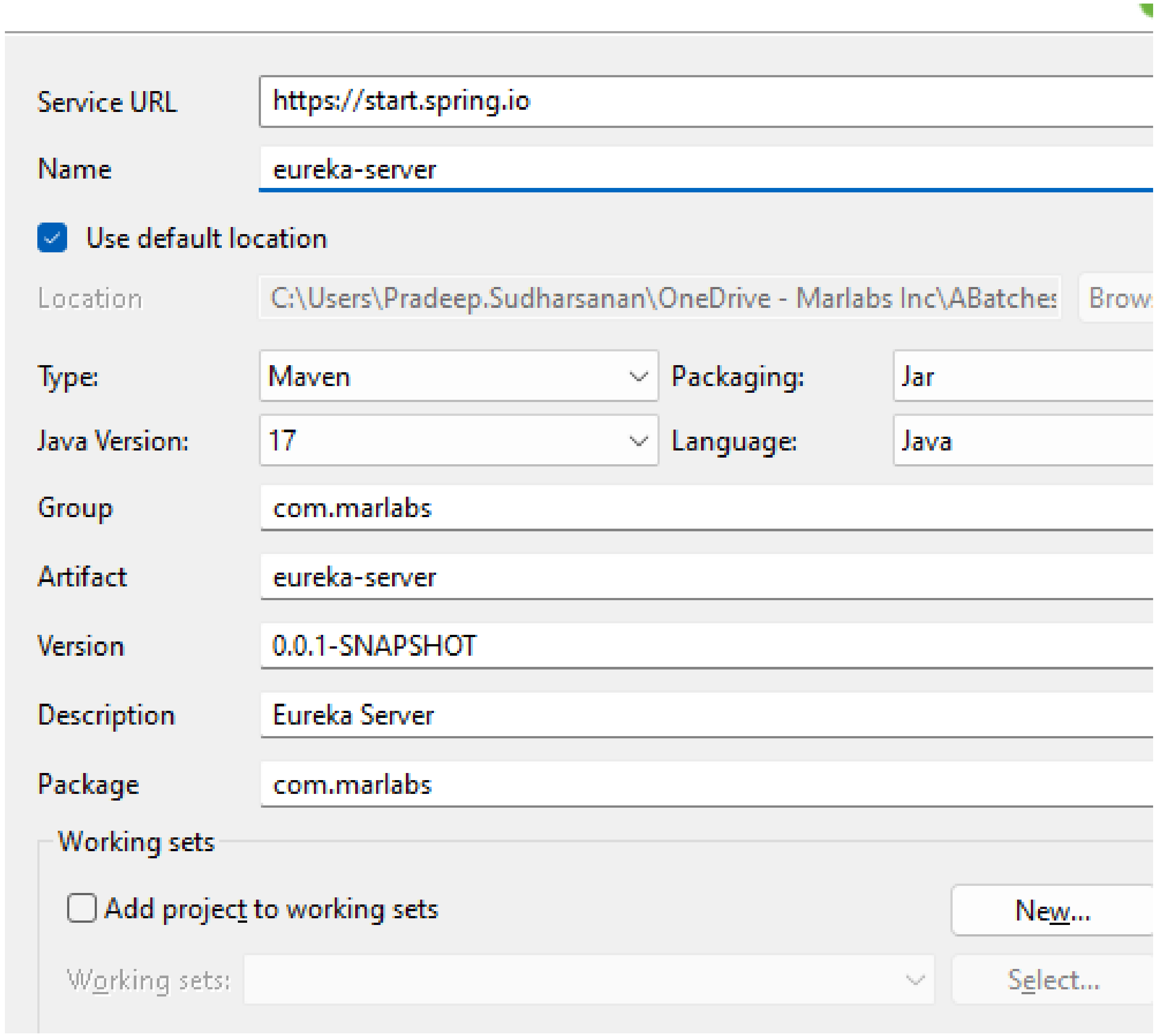
It acts as a lookup service where microservices can find each other to communicate without hardcoding the hostnames and ports

Service Discovery & Registry

Each microservice register itself with Spring Cloud Eureka with name and url.
We can communicate with microservices using their name instead of url.
In future, even the url changes, we don’t need to change in the code.
Each microservice has the name in the properties file



Create a new project



Service URL: https://start.spring.io

Name: eureka-server

[✓] Use default location

Location: C:\Users\Pradeep.Sudharsanan\OneDrive - Marlabs Inc\ABatche:

Type: Maven

Packaging: Jar

Java Version: 17

Language: Java

Group: com.marlabs

Artifact: eureka-server

Version: 0.0.1-SNAPSHOT

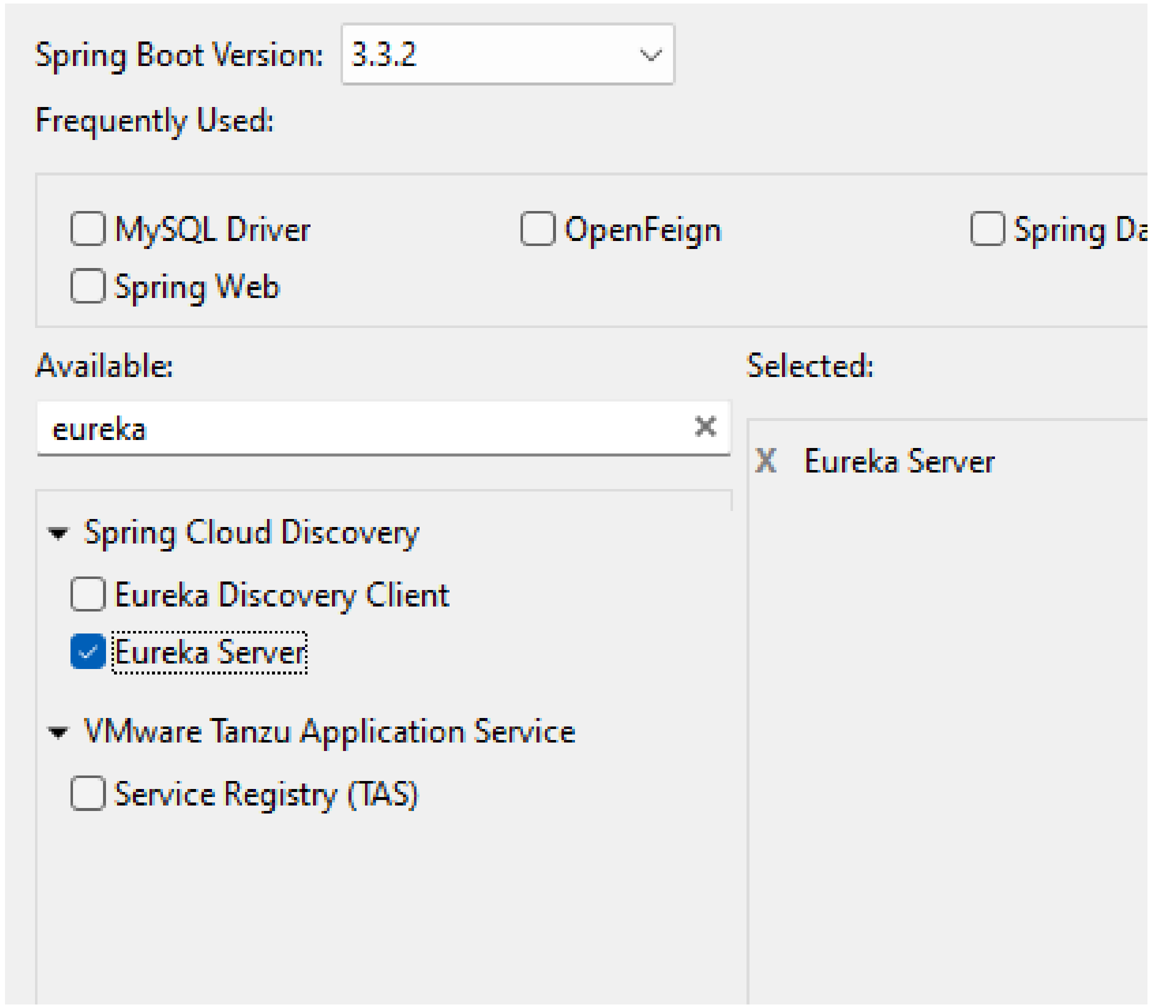
Description: Eureka Server

Package: com.marlabs

Working sets

[ ] Add project to working sets

Working sets:



Spring Boot Version: 3.3.2

Frequently Used:

[ ] MySQL Driver
[ ] OpenFeign
[ ] Spring Da
[ ] Spring Web

Available: eureka

Spring Cloud Discovery

[ ] Eureka Discovery Client
[✓] Eureka Server

VMware Tanzu Application Service

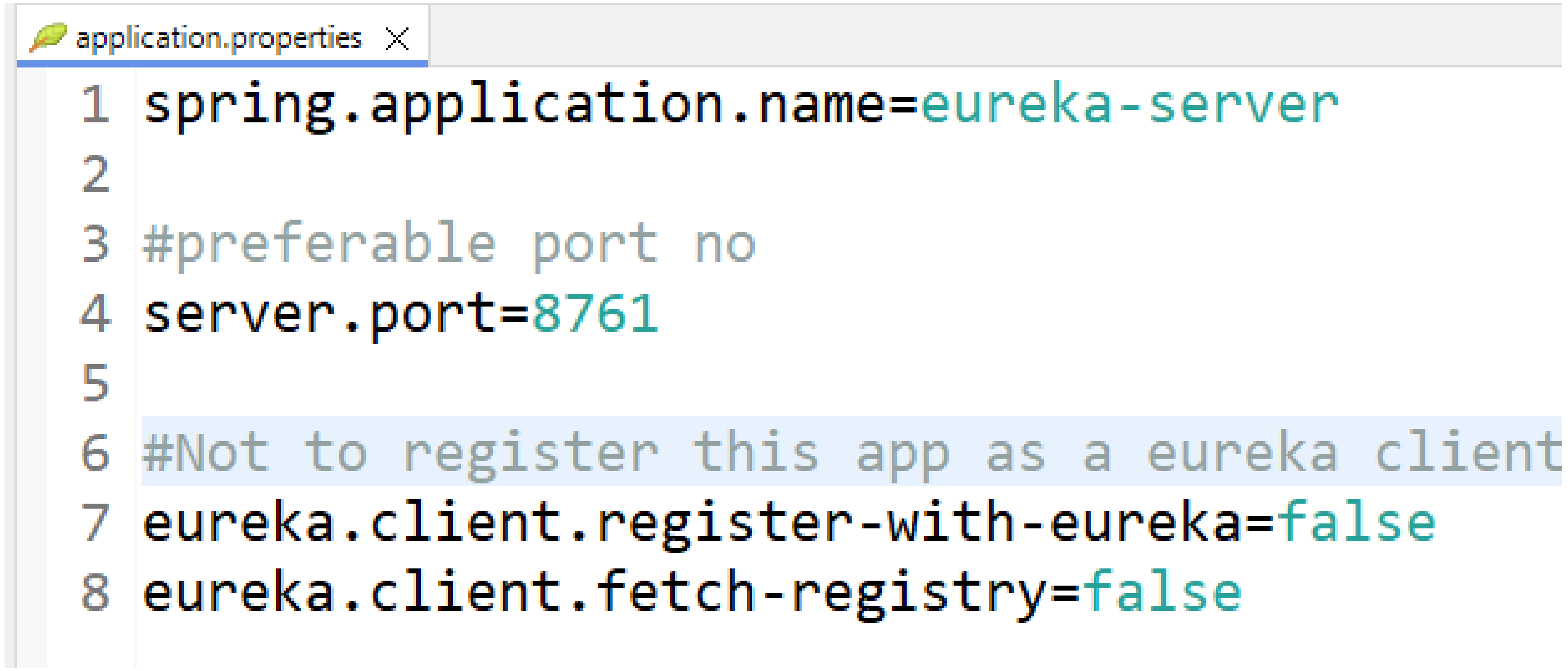
[ ] Service Registry (TAS)

Selected:

X Eureka Server

```xml
<properties>
    <java.version>17</java.version>
    <spring-cloud.version>2023.0.3</spring-cloud.version>
</properties>
<dependencies>   Add Spring Boot Starters...
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-netflix-eureka-server</art
    </dependency>

    <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>
```

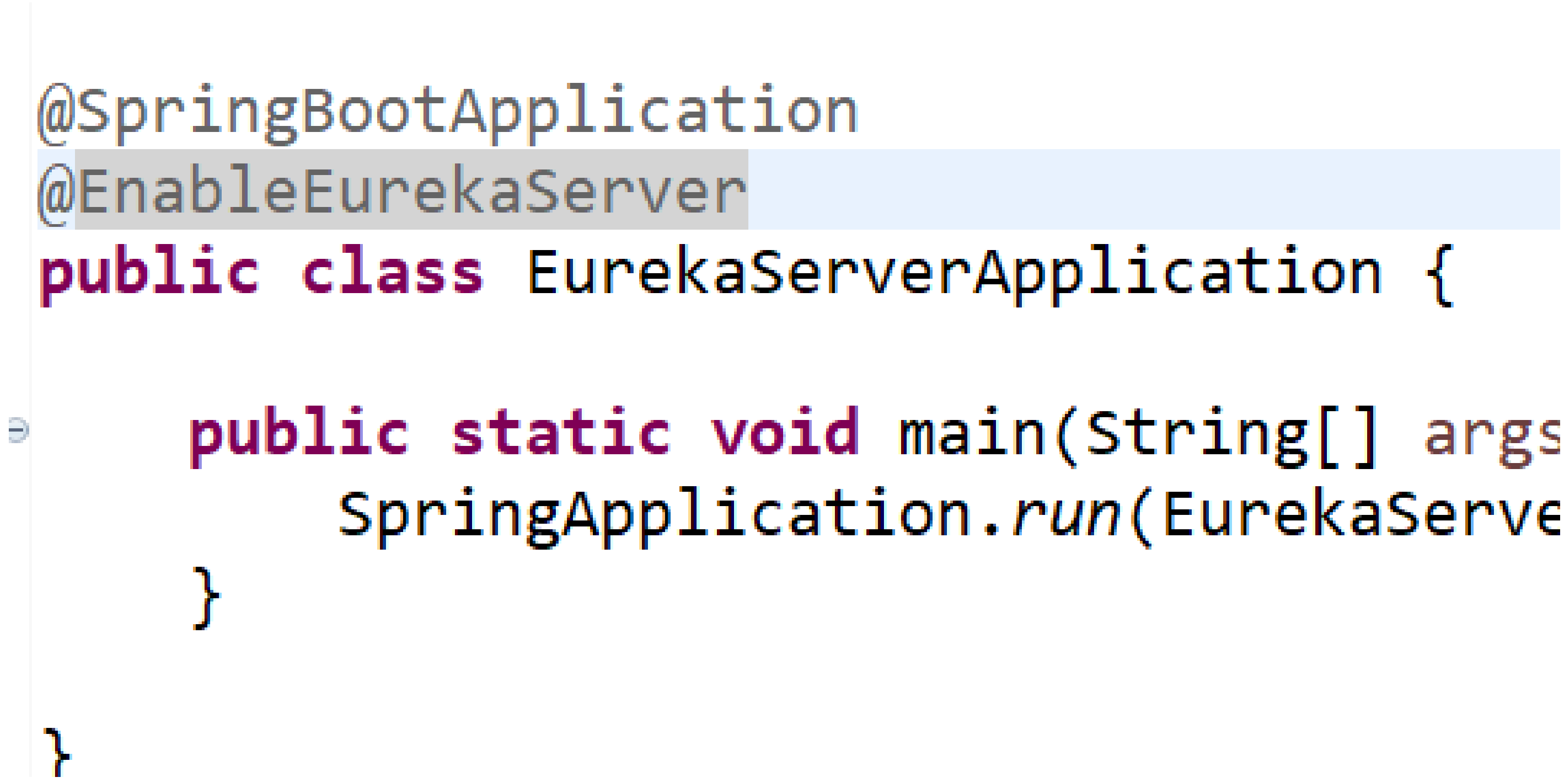
Properties file



application.properties

```
spring.application.name=eureka-server

#preferable port no
server.port=8761

#Not to register this app as a eureka client
eureka.client.register-with-eureka=false
eureka.client.fetch-registry=false
```

In main class enable eureka server

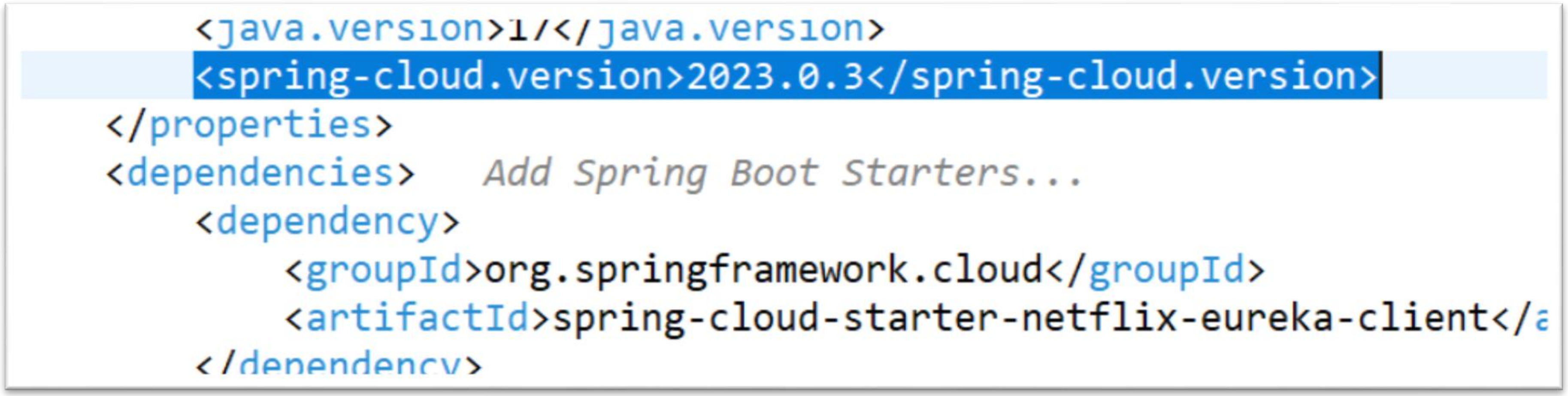


```
@SpringBootApplication
@EnableEurekaServer
public class EurekaServerApplication {

    public static void main(String[] args
        SpringApplication.run(EurekaServe
    }

}
```

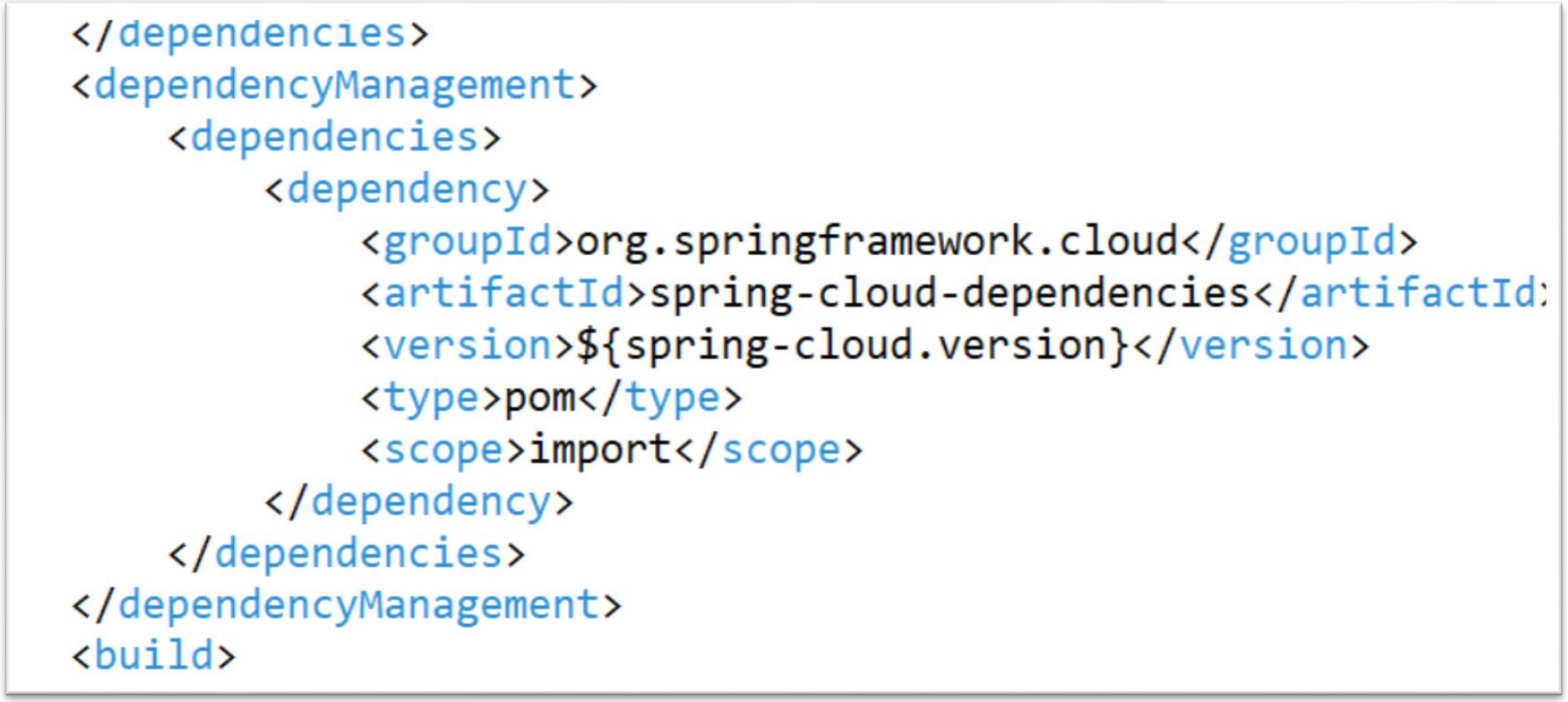
In all microservices

- Add EurekaClient dependency
- Enable EurekaDiscoveryClient
- Provide the Eureka Server url

Update pom.xml In all microservices

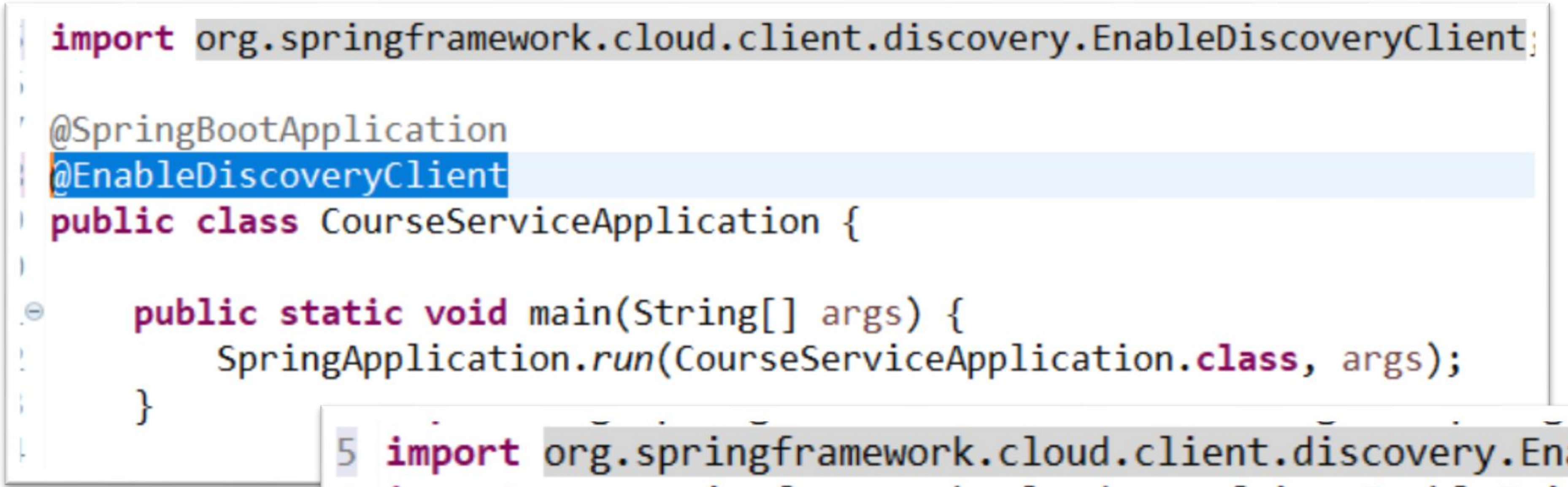


```
        <java.version>17</java.version>
        <spring-cloud.version>2023.0.3</spring-cloud.version>
    </properties>
    <dependencies>    Add Spring Boot Starters...
        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-starter-netflix-eureka-client</a
        </dependency>
```

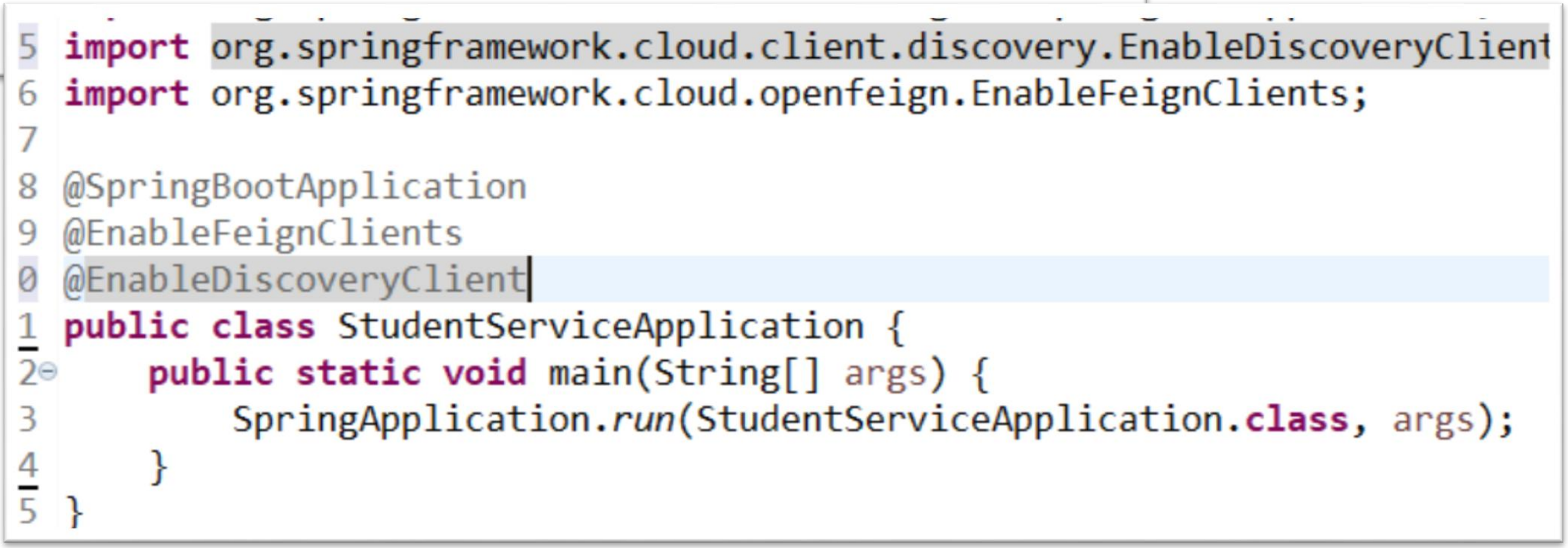


```
    </dependencies>
    <dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>org.springframework.cloud</groupId>
                <artifactId>spring-cloud-dependencies</artifactId
                <version>${spring-cloud.version}</version>
                <type>pom</type>
                <scope>import</scope>
            </dependency>
        </dependencies>
    </dependencyManagement>
    <build>
```

@EnableDiscoveryClient in all microservices



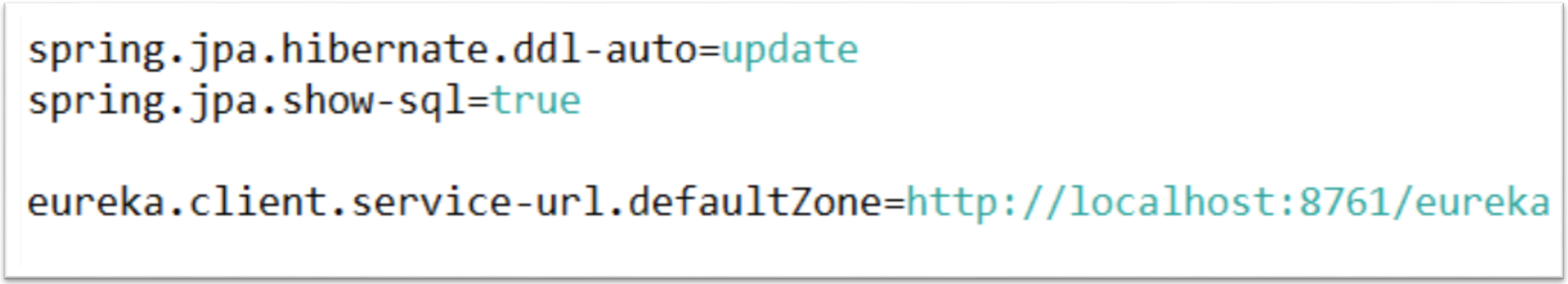
```java
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;

@SpringBootApplication
@EnableDiscoveryClient
public class CourseServiceApplication {

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



```java
import org.springframework.cloud.client.discovery.EnableDiscoveryClient
import org.springframework.cloud.openfeign.EnableFeignClients;

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

In all microservices properties file

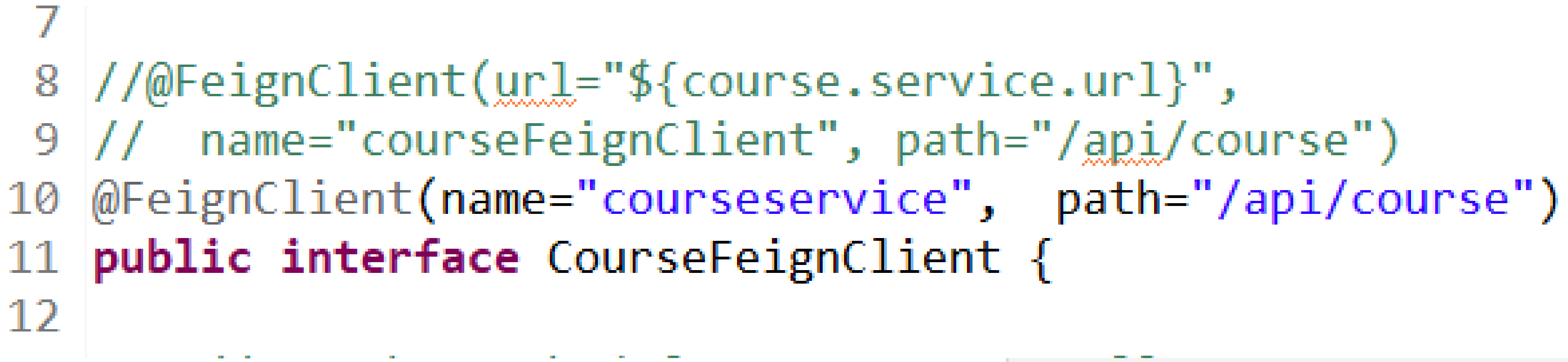
- Provide Eureka Server url



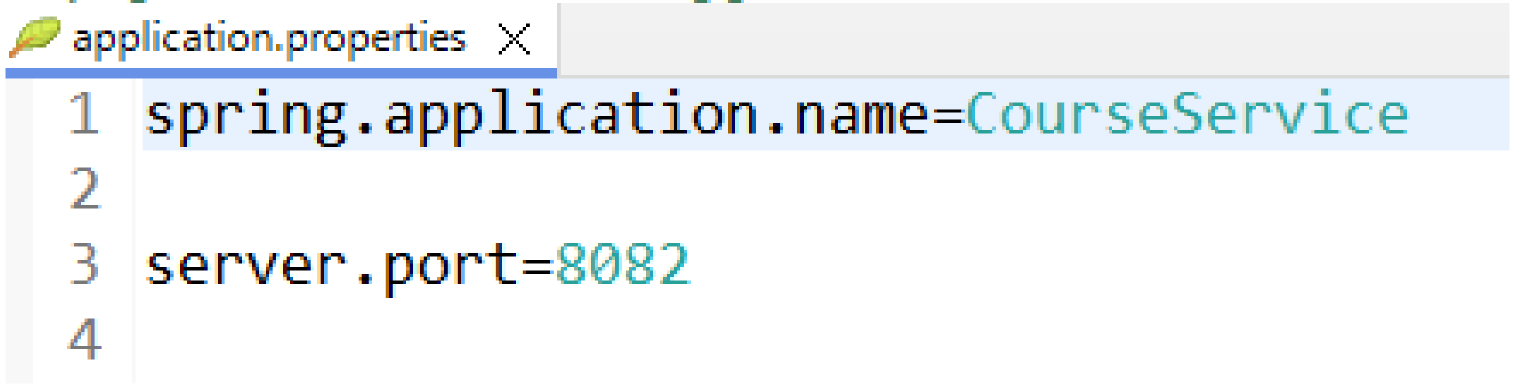
```
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true

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

Update the name in feignClient

- The name must be same as the name in the respective microservice properties file (case in-sensitive)



```java

//@FeignClient(url="${course.service.url}",
//   name="courseFeignClient", path="/api/course")
@FeignClient(name="courseservice",  path="/api/course")
public interface CourseFeignClient {

```

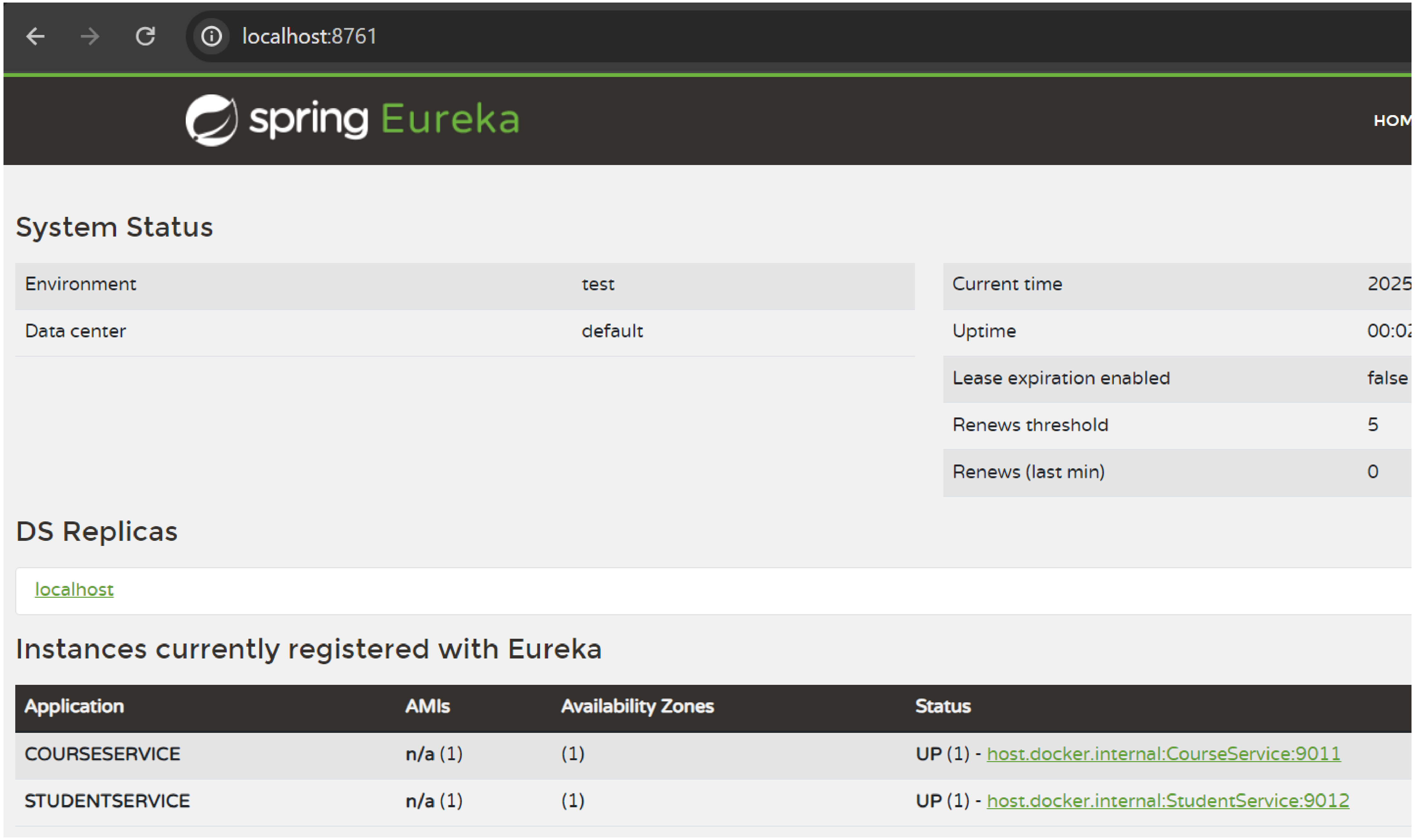


application.properties

```properties
spring.application.name=CourseService

server.port=8082

```

Eureka Server in Action

- First start the eureka server
- Then the microservices to register it with eureka



System Status

Environment	test
Data center	default

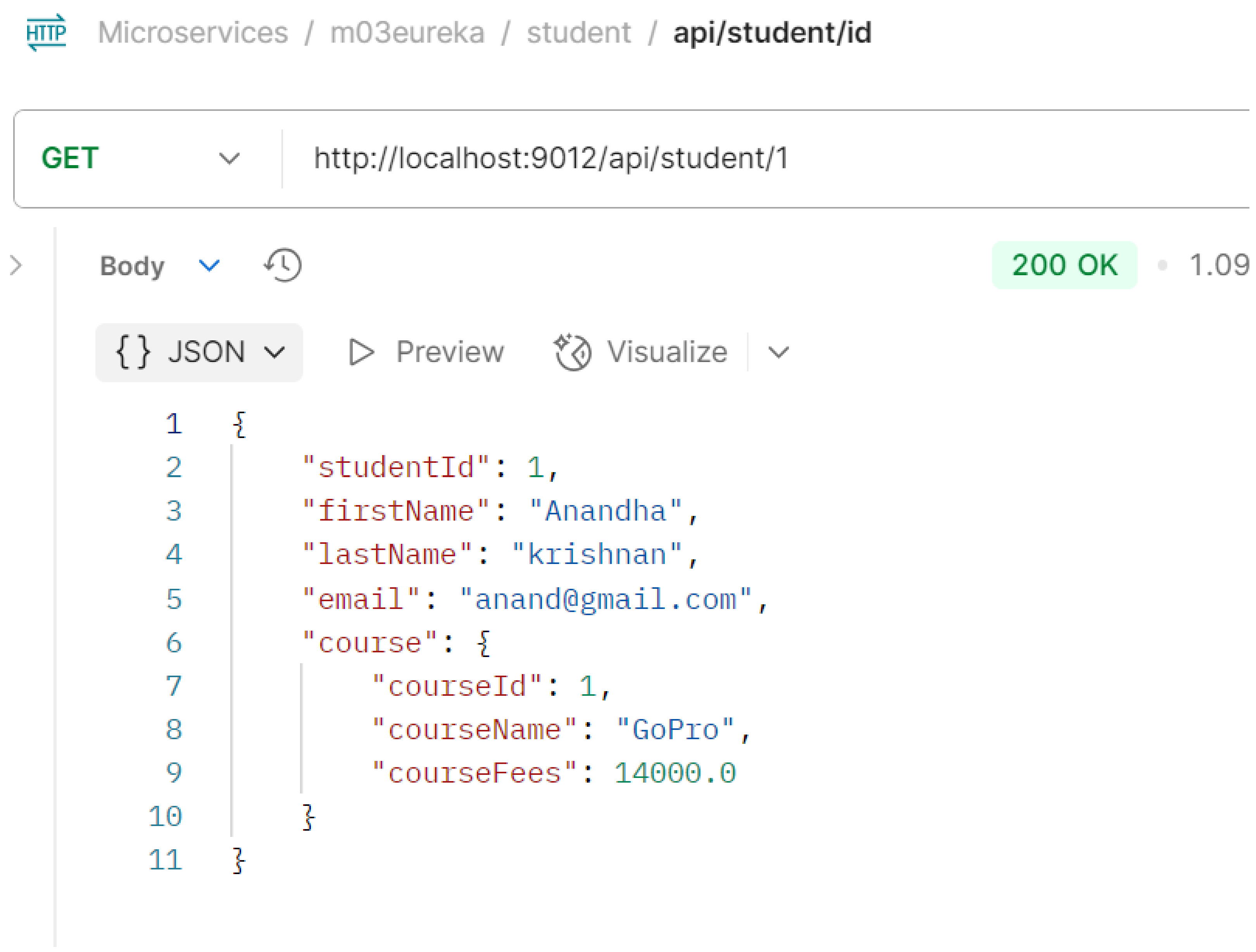
Current time	2025
Uptime	00:0
Lease expiration enabled	false
Renews threshold	5
Renews (last min)	0

DS Replicas

localhost

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
COURSESERVICE	n/a (1)	(1)	UP (1) - host.docker.internal:CourseService:9011
STUDENTSERVICE	n/a (1)	(1)	UP (1) - host.docker.internal:StudentService:9012



GET http://localhost:9012/api/student/1

```
{
    "studentId": 1,
    "firstName": "Anandha",
    "lastName": "krishnan",
    "email": "anand@gmail.com",
    "course": {
        "courseId": 1,
        "courseName": "GoPro",
        "courseFees": 14000.0
    }
}
```