Spring Cloud로 개발하는 마이크로서비스 애플리케이션

+

목차

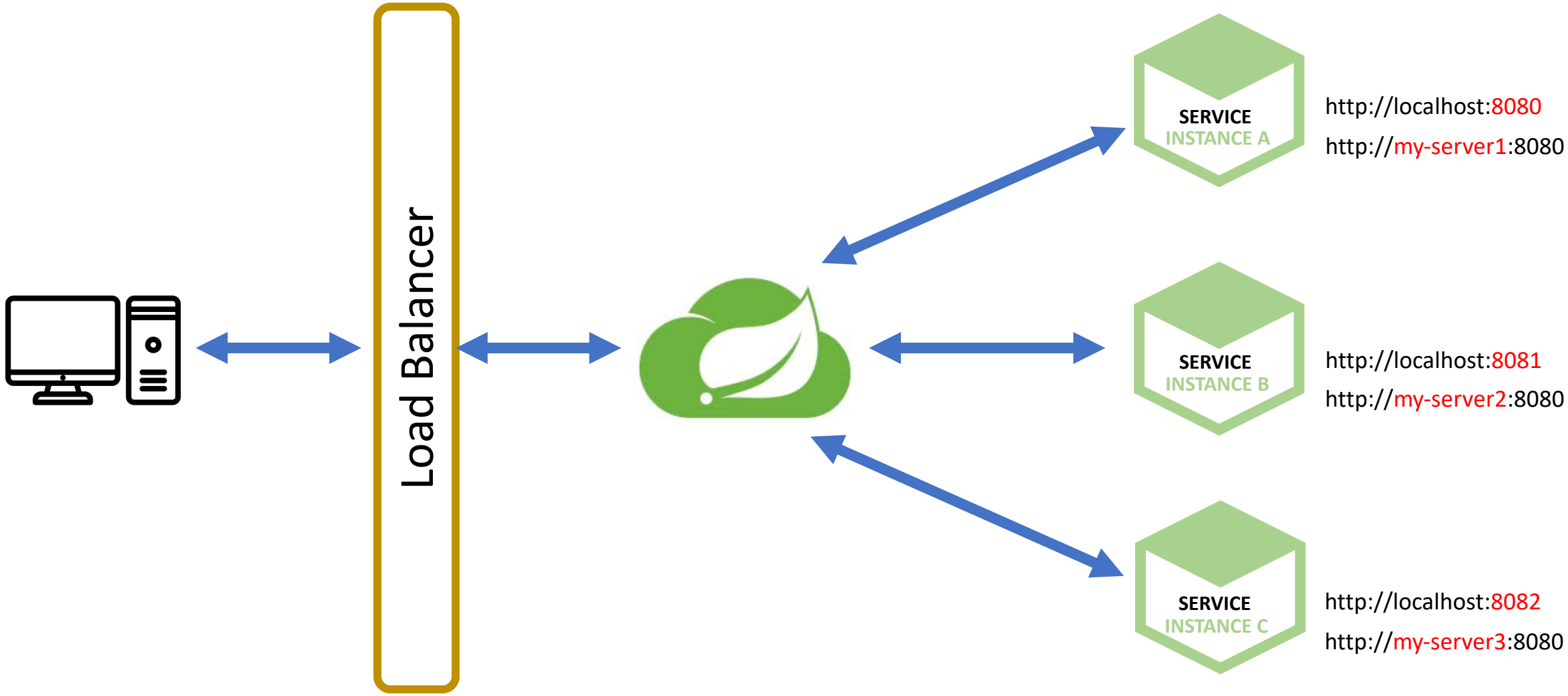
Part 1

Section 1.

Service Discovery

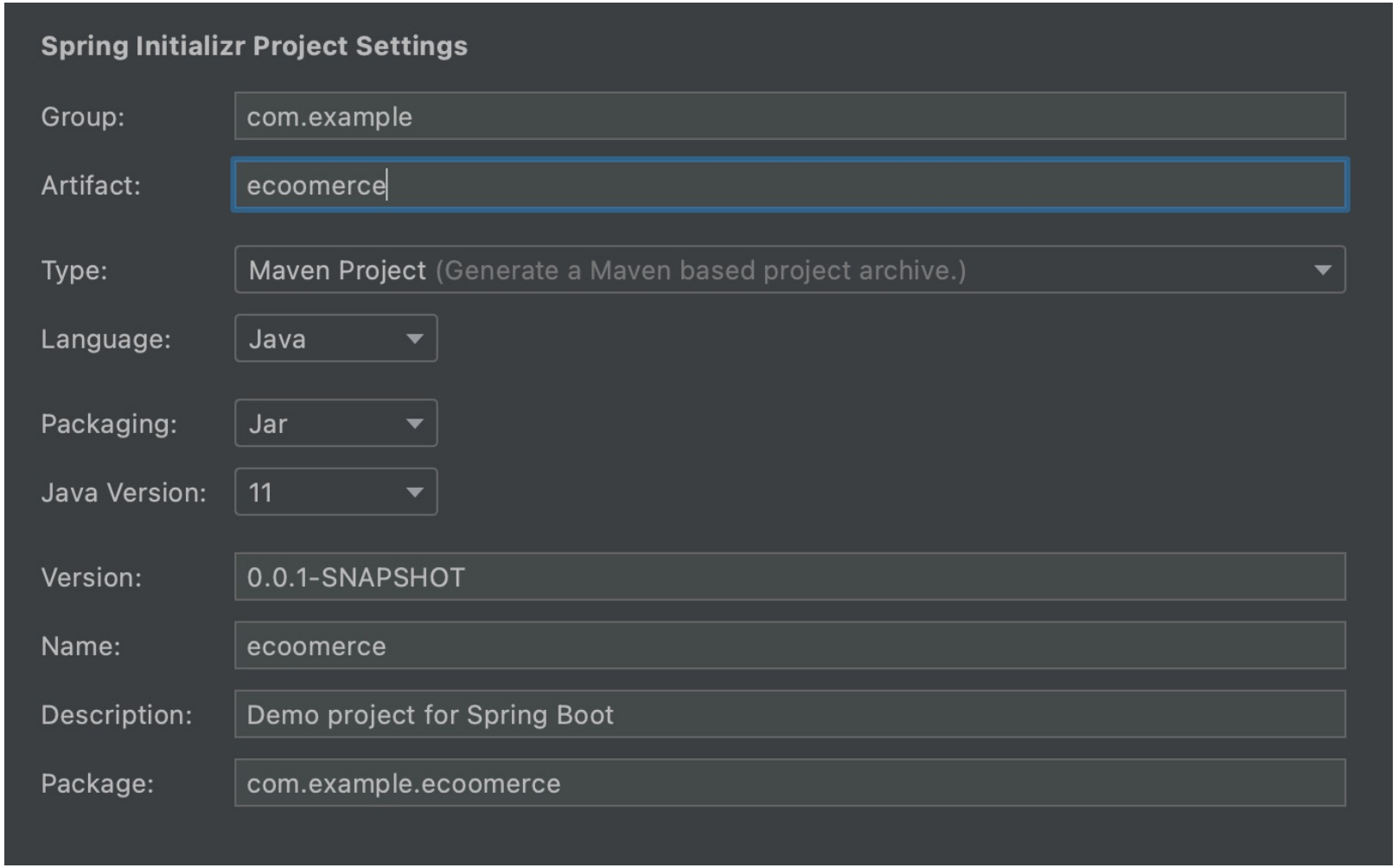
- Spring Cloud Netflix Eureka
- Eureka Service Discovery – 프로젝트 생성
- User Service – 프로젝트 생성
- User Service – 등록
- User Service – Load Balancer

Spring Cloud Netflix Eureka



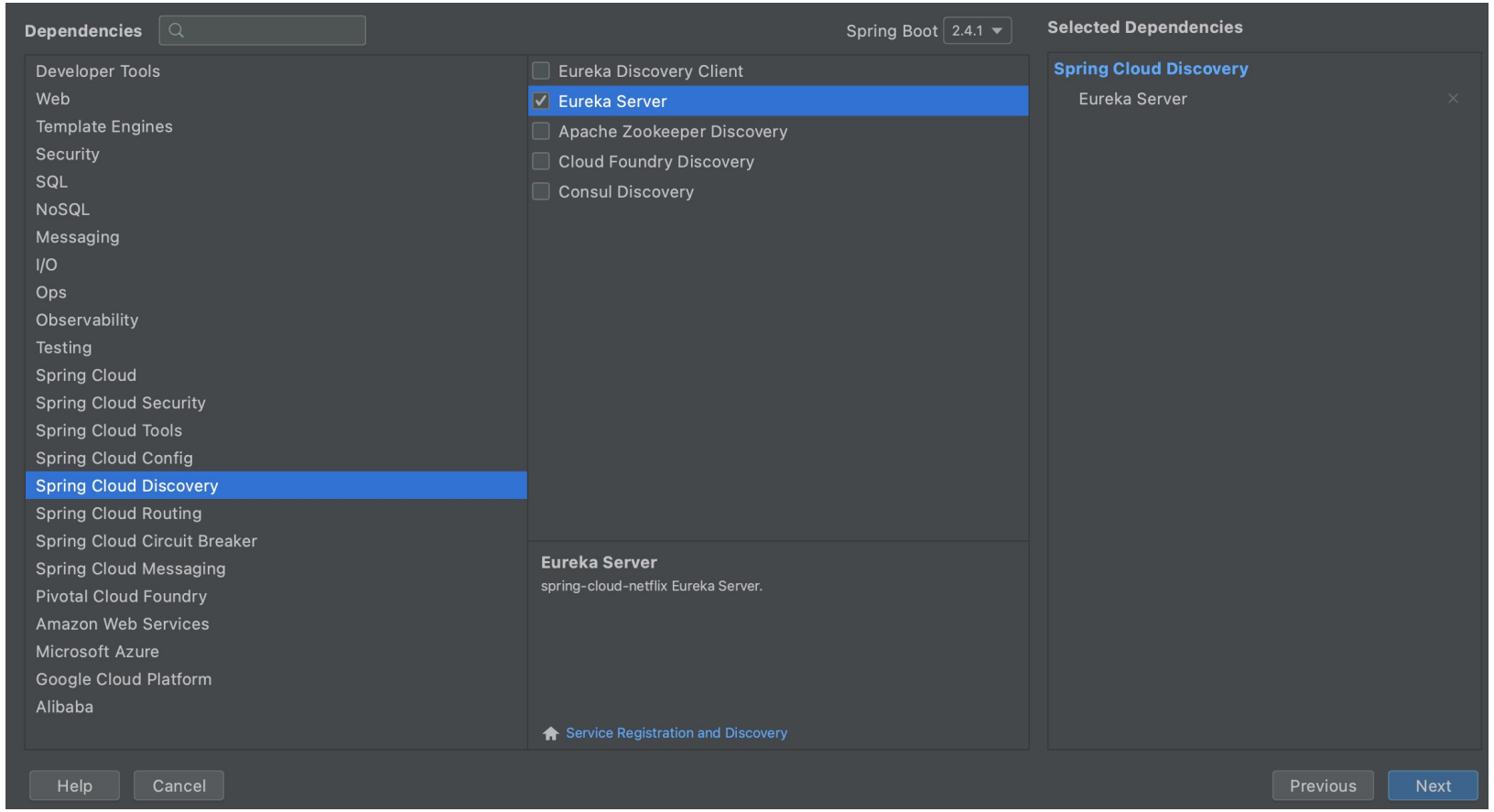
EcommerceDiscoveryService

- IntelliJ
  - Create New Project > Spring Initializr



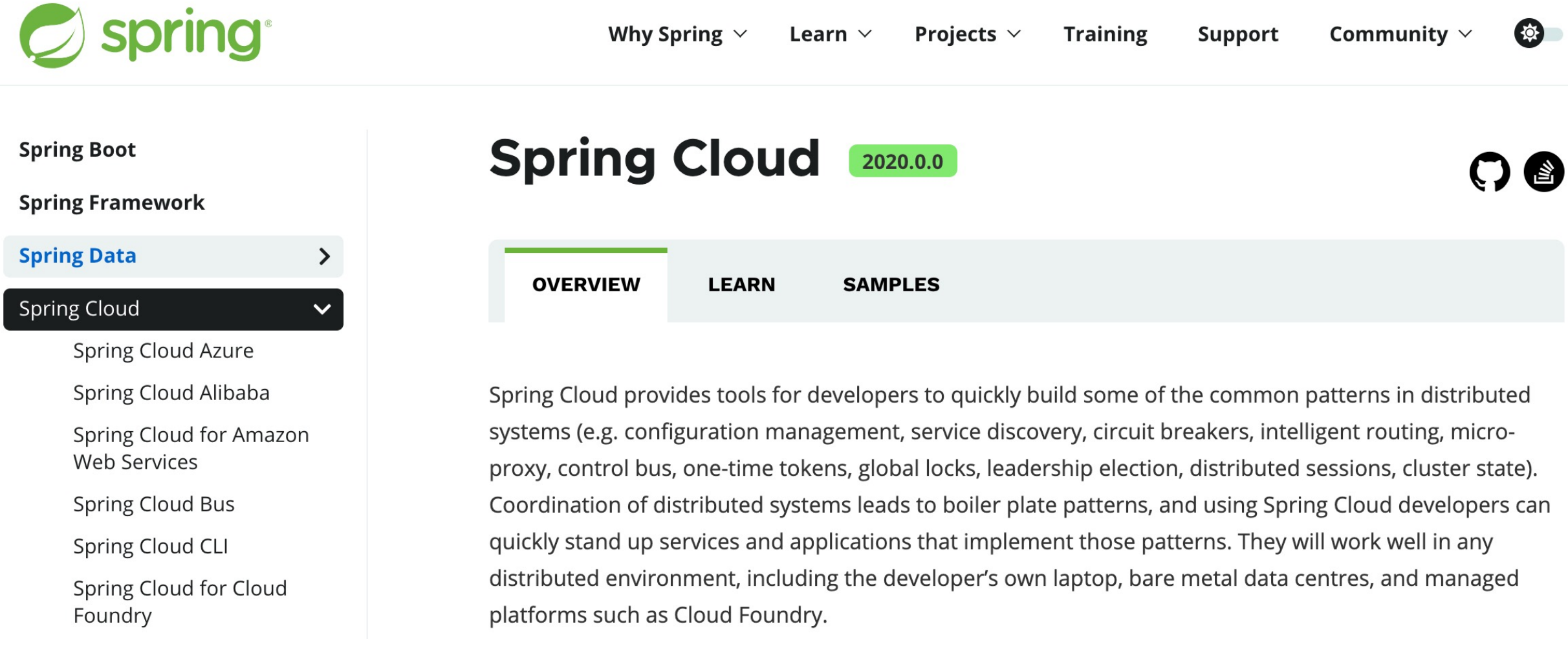
EcommerceDiscoveryService

- Dependencies
  - Spring Boot > 2.4.1
  - Spring Cloud Discovery > Eureka Server



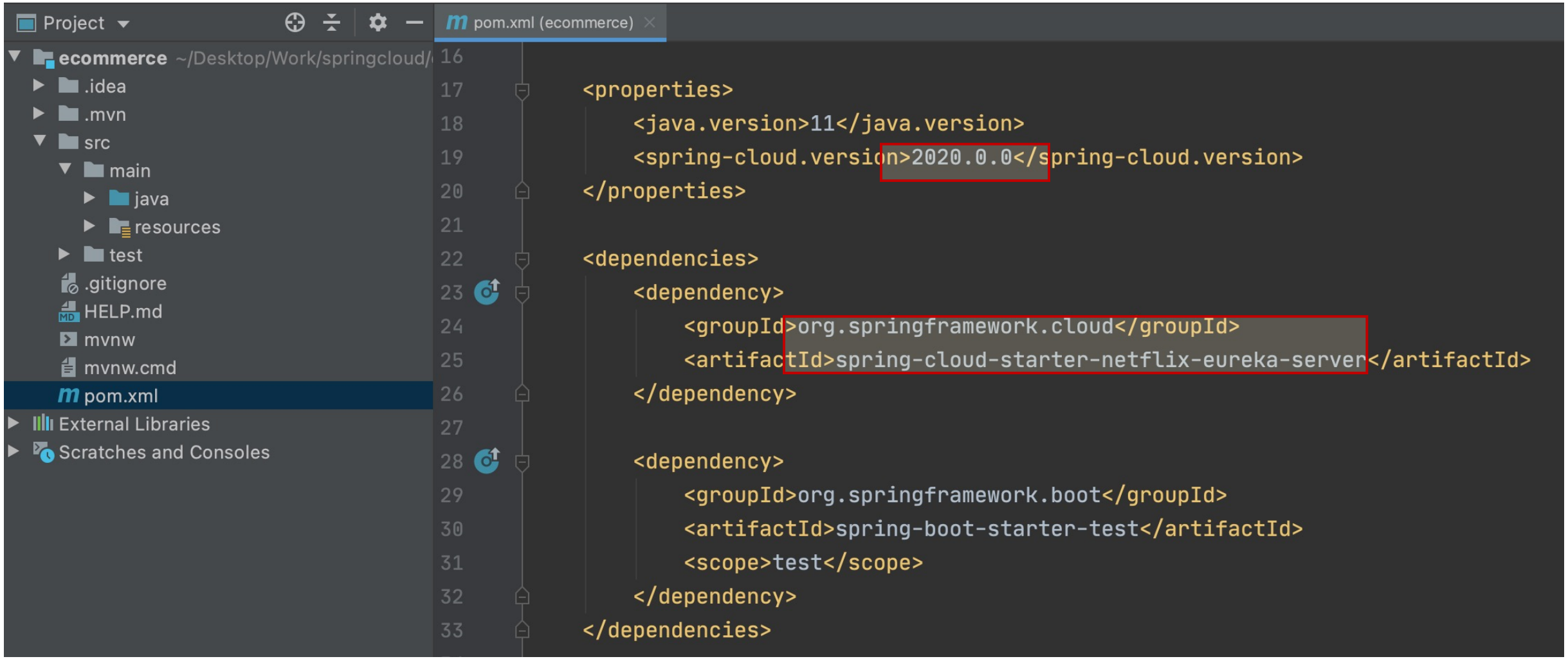
EcommerceDiscoveryService

- Spring Cloud 2020.0.X
  - Spring Boot > 2.4.1



EcommerceDiscoveryService

- pom.xml



EcommerceDiscoveryService

- EcommerceApplication.java

```
7  @SpringBootApplication
8  @EnableEurekaServer
9  public class DiscoveryserviceApplication {
10
11     public static void main(String[] args) { SpringApplication.run(DiscoveryserviceApplication.class, args); }
14
15 }
```

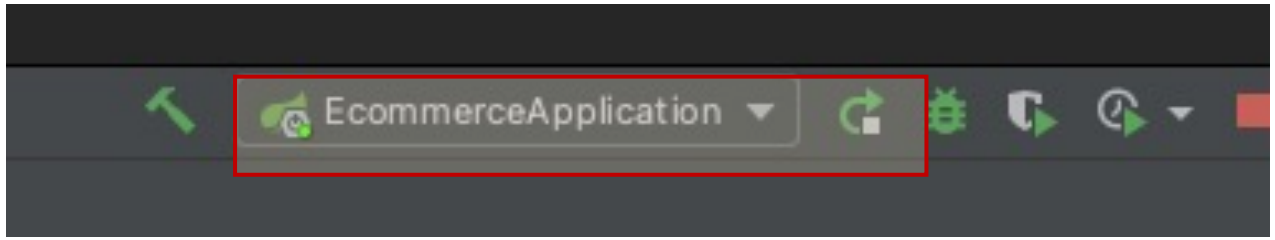
EcommerceDiscoveryService

- application.yml (or application.properties)

```yaml
server:
  port: 8761

spring:
  application:
    name: discoveryservice

eureka:
  client:
    register-with-eureka: false
    fetch-registry: false
```

EcommerceDiscoveryService

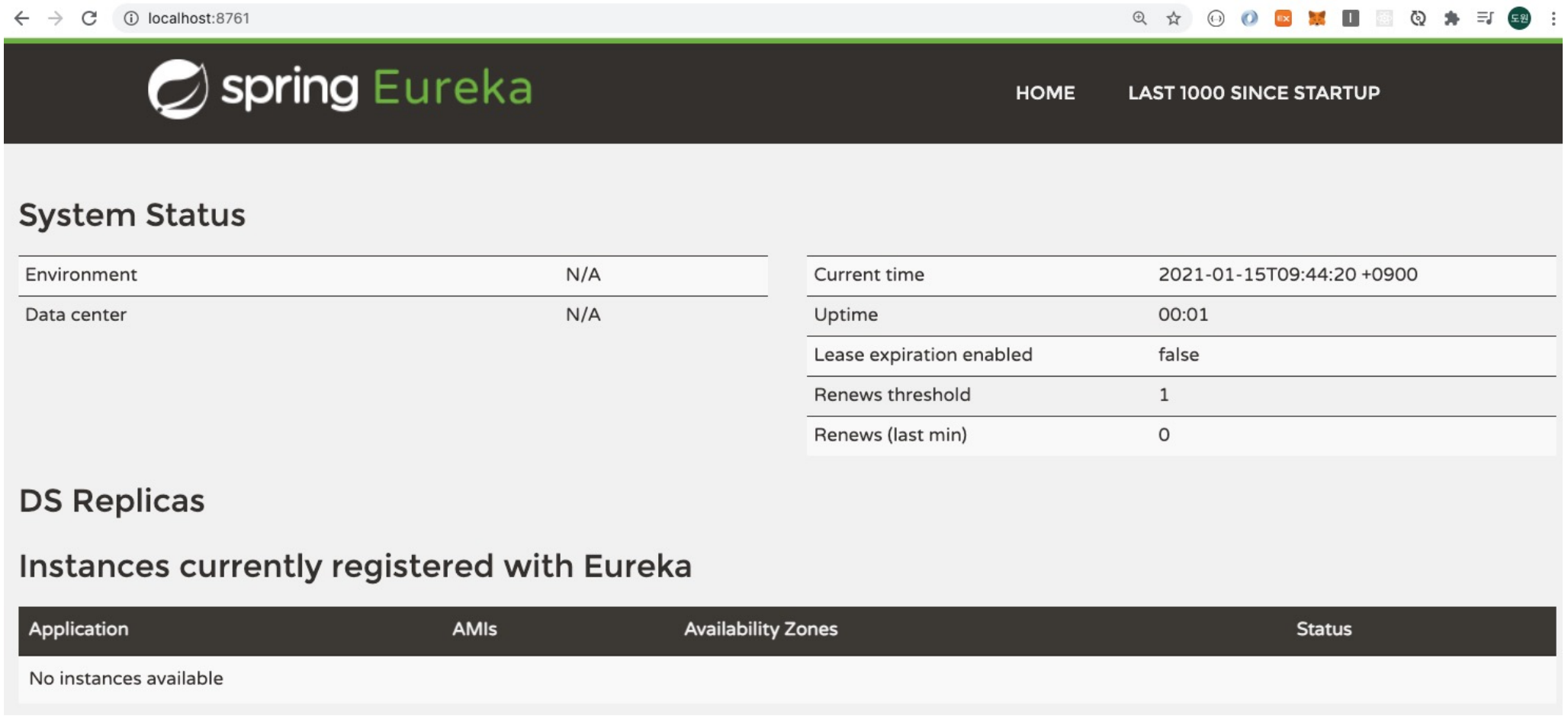
- 실행




```
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8761 (http) with context path ''
.s.c.n.e.s.EurekaAutoServiceRegistration : Updating port to 8761
c.e.ecommerce.EcommerceApplication       : Started EcommerceApplication in 3.252 seconds (JVM running for 3.799)
o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServlet'
o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
o.s.web.servlet.DispatcherServlet        : Completed initialization in 1 ms
```

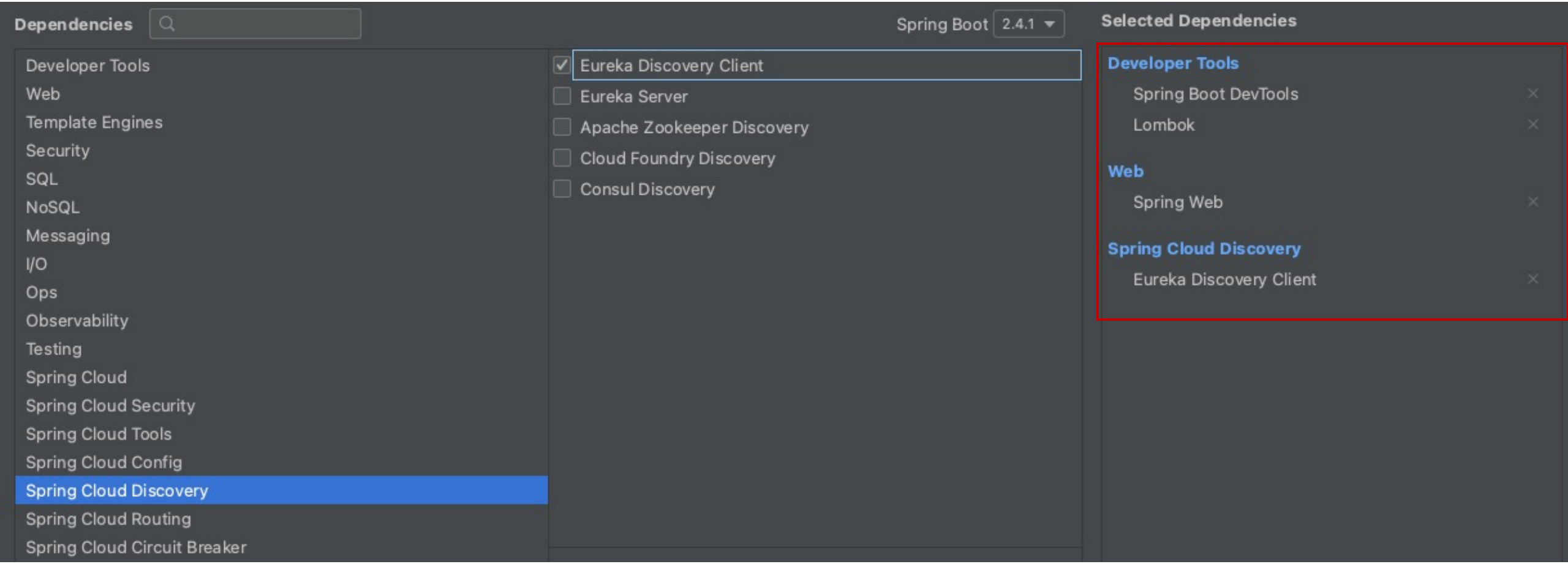
EcommerceDiscoveryService

- 실행화면
  - http://localhost:8761



User Service

- Eureka Discovery Service에 등록



User Service

- pom.xml

```xml
<properties>
    <java.version>11</java.version>
    <spring-cloud.version>2020.0.0</spring-cloud.version>
</properties>

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

```xml
<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>
```

User Service

- application.yml

```
7  @SpringBootApplication
8  @EnableDiscoveryClient
9  public class UserServiceApplication {
10
11     public static void main(String[] args) { SpringApplication.run(UserServiceApplication.class, args); }
14
15 }
```

User Service

- application.yml

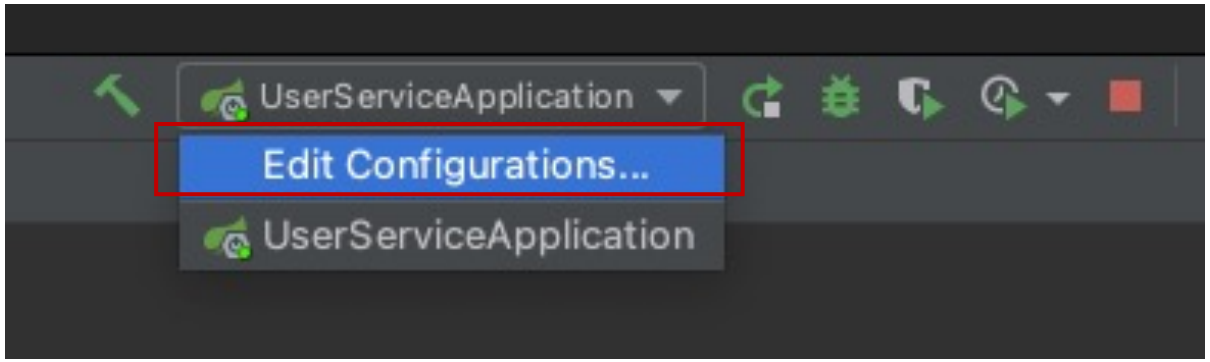
```yaml
server:
  port: 9001

spring:
  application:
    name: user-service

eureka:
  client:
    register-with-eureka: true
    fetch-registry: true
    service-url:
      defaultZone: http://localhost:8761/eureka
```

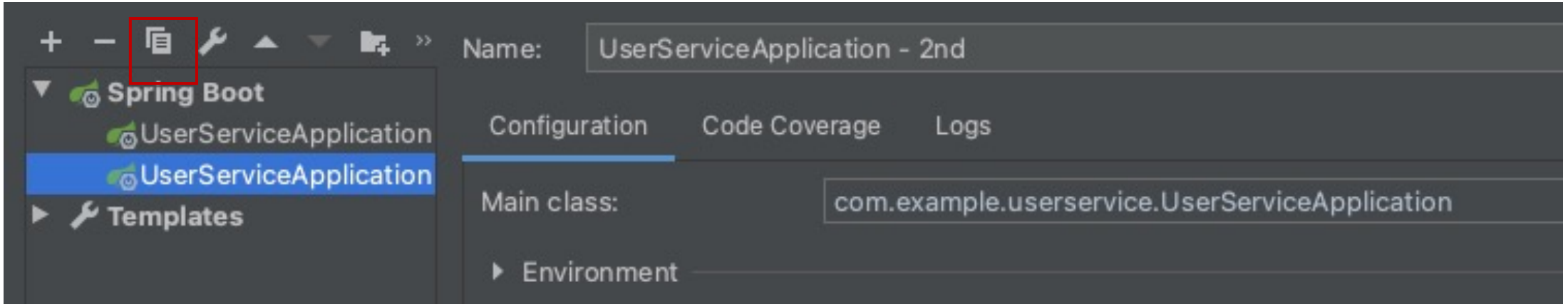
Instances currently registered with Eureka

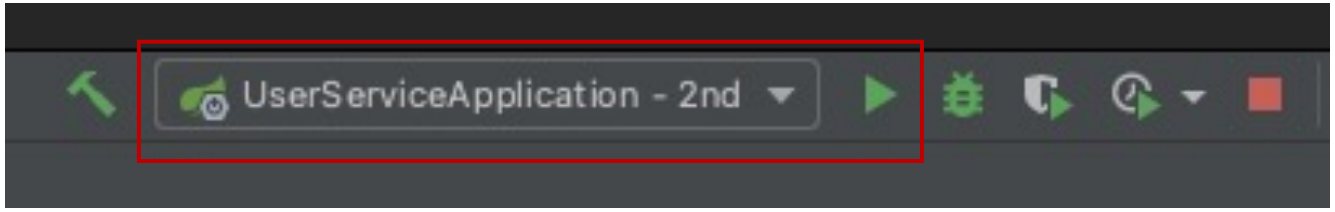
Application	AMIs	Availability Zones	Status
USER-SERVICE	n/a (1)	(1)	UP (1) - 10.90.1.91:user-service:9001

User Service

- Scaling – 같은 서비스 추가 실행 #1







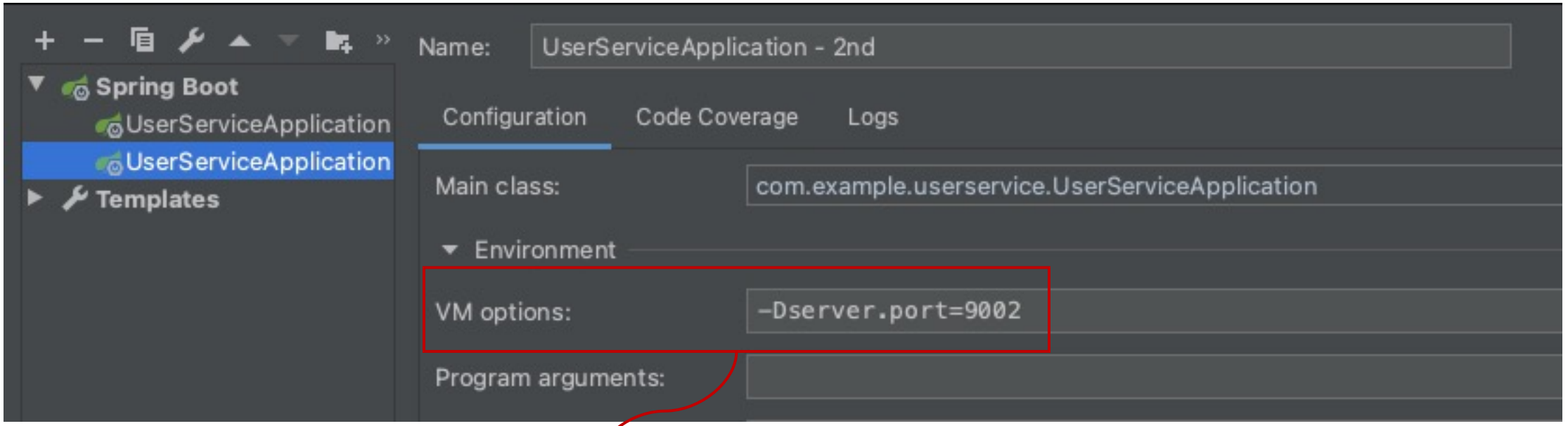
User Service

- Scaling – 같은 서비스 추가 실행 #1
  - ***Port 충돌***

```
Error starting ApplicationContext. To display the conditions report re-run your application with 'debug' enabled.
2021-01-15 10:54:05.701 ERROR 68628 --- [  restartedMain] o.s.b.d.LoggingFailureAnalysisReporter   :

***************************
APPLICATION FAILED TO START
***************************

Description:

Web server failed to start. Port 9001 was already in use.
```

User Service

- Scaling – 같은 서비스 추가 실행 #1
  - VM Options → -Dserver.port=[다른포트]




```
: Tomcat started on port(s): 9002 (http) with context path ''
: Updating port to 9002
: DiscoveryClient_USER-SERVICE/10.90.1.91:user-service:9002 - registration
: Started UserServiceApplication in 2.881 seconds (JVM running for 3.455)
```

User Service

- Scaling – 같은 서비스 추가 실행 #1

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
USER-SERVICE	n/a (2)	(2)	UP (2) - 10.90.1.91:user-service:9001 , 10.90.1.91:user-service:9002

- Scaling - 같은 서비스 추가 실행 #2

```
$ mvn spring-boot:run -Dspring-boot.run.jvmArguments='-Dserver.port=9003'
```

- Scaling - 같은 서비스 추가 실행 #3

```
$ mvn clean compile package
$ java -jar -Dserver.port=9004 ./target/user-service-0.0.1-SNAPSHOT.jar
```

User Service

- Scaling – 같은 서비스 추가 실행

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
USER-SERVICE	n/a (4)	(4)	UP (4) - 10.90.1.91:user-service:9003 , 10.90.1.91:user-service:9004 , 10.90.1.91:user-service:9001 , 10.90.1.91:user-service:9002

UP (4) - 10.90.1.91:user-service:9003 , 10.90.1.91:user-service:9004 , 10.90.1.91:user-service:9001 , 10.90.1.91:user-service:9002

User Service

- Scaling – Random 포트 사용으로 같은 서비스 추가 실행
  - application.yml

```yaml
server:
  port: 0

spring:
  application:
    name: user-service
```

```
Spring Cloud LoadBalancer is currently working with the default
Tomcat started on port(s): 55002 (http) with context path ''
Updating port to 55002
```

$ mvn spring-boot:run

```
Spring Cloud LoadBalancer is currently working with the default

Tomcat started on port(s): 55019 (http) with context path ''
Updating port to 55019
```

User Service

- Scaling – Random 포트 사용으로 같은 서비스 추가 실행
  - 같은 서비스 명으로 인해 Eureka에 하나의 앱만 등록

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
USER-SERVICE	n/a (1)	(1)	UP (1) - 10.90.1.91:user-service:0

User Service

- Scaling – Random 포트 사용으로 같은 서비스 추가 실행
  - application.yml에 instance 정보 등록
    - ***eureka.instance.instanceId***

```yaml
server:
  port: 0

spring:
  application:
    name: user-service


eureka:
  instance:
    instanceId: ${spring.cloud.client.hostname}:${spring.application.instance_id:${random.value}}
  client:
    register-with-eureka: true
    fetch-registry: true
```

User Service

- Scaling – Random 포트 사용으로 같은 서비스 추가 실행
  - application.yml에 instance 정보 등록
    - ***eureka.instance.instanceId***