1. Micro Service – SOA – Service Oriented Architecture
2. Monolith Application & Micro Service
3. Pro’s & Con’s in Micro Service Architecture
4. Challenges in Micro-Service
5. Customizing Spring Security (Token Based Auth, oAuth)

Monolith Application – Single Project where all the controllers, service, model, repo, exceptions & other

Amazon/Flipkart

1. User management service (add/deleting/updating) – Authentication & Authorization (OTP email validation, captcha validation)
2. Payment management service (add/delete/update) -- upi/netbanking/pod or cod/wallets
3. Cart mgmt. service (add/updating/deleting)
4. Order mgmt. service (get all/get by id/add/ delete/update)
5. Product mgmt. service
6. Authentication service
7. Rating service
8. Review Service

Single project – multiple small simple projects

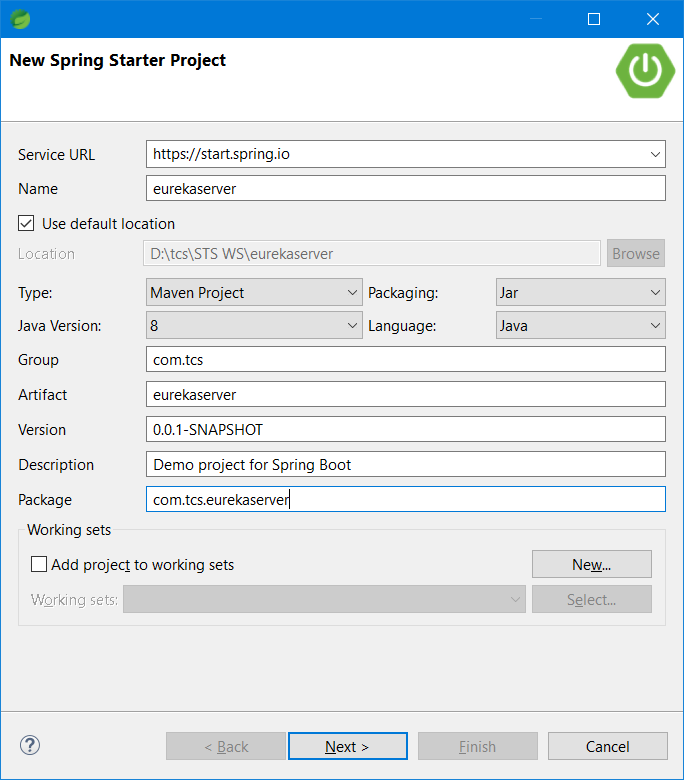
Netflix OSS – Netflix Open Source Software (MicroService)

Eureka Server – Netflix Eureka (Discovery Server) [API – Gateway]

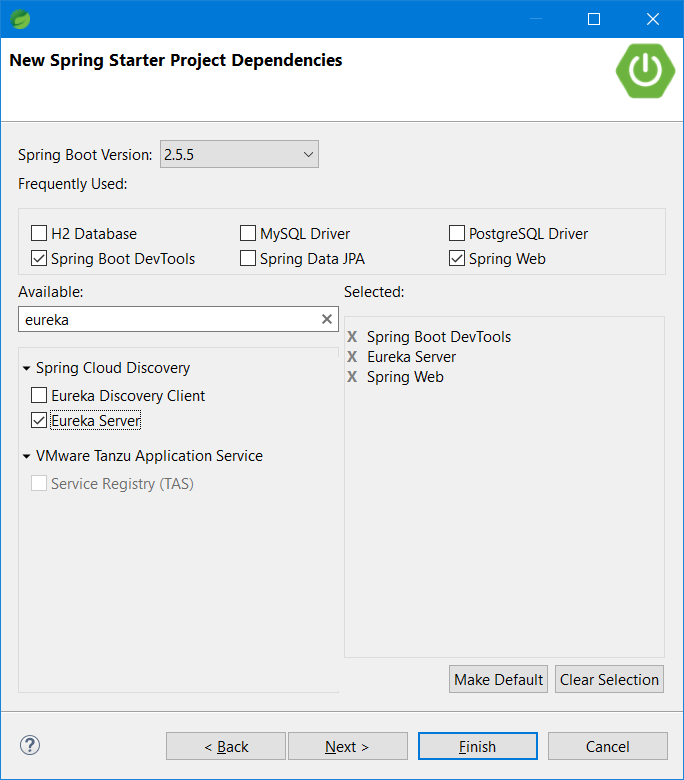
Eureka Client –

Creating Eureka Server

Step 1: create a spring starter project in STS



Step 2: add the following dependencies to the project (devtools, Eureka Server, spring web)



Step 3: add the following properties in application.properties file

server.port=8761

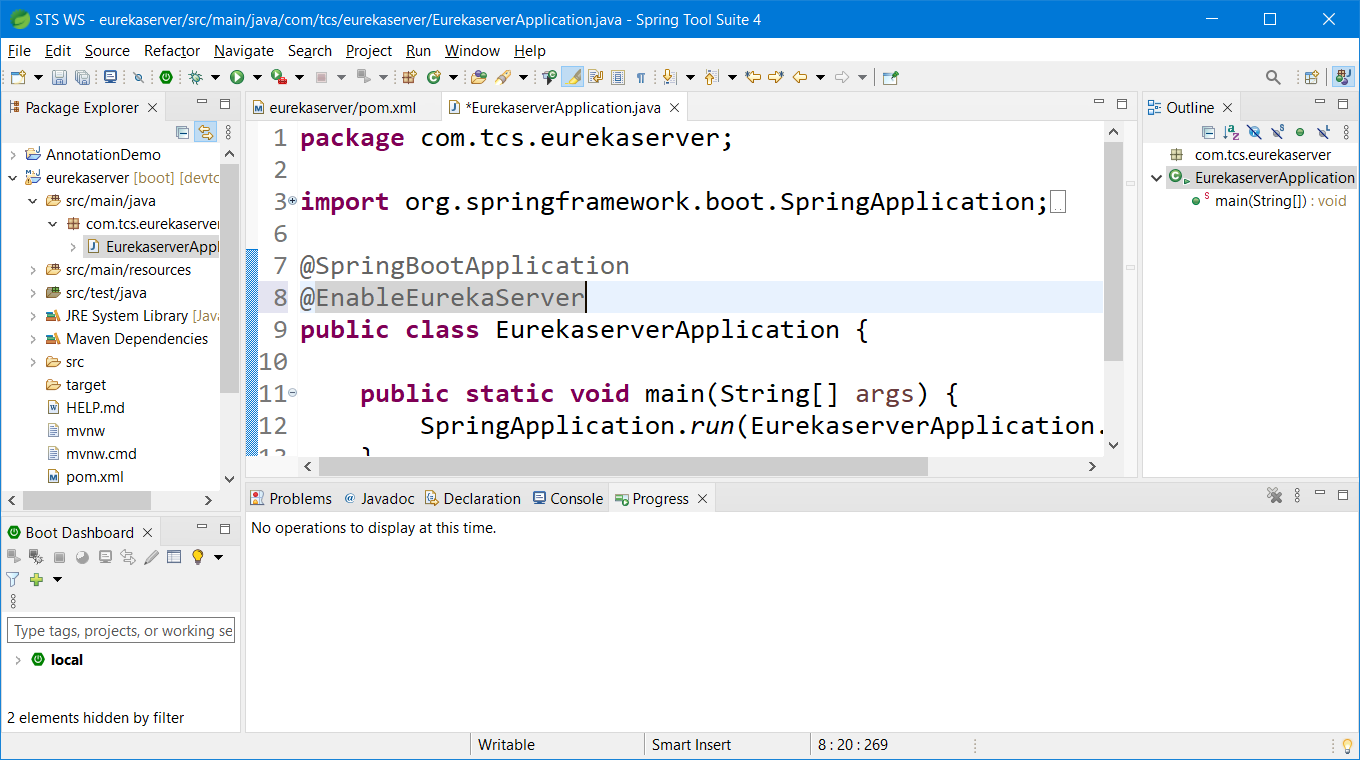
eureka.client.register-with-eureka=false

eureka.client.fetch-registry=false

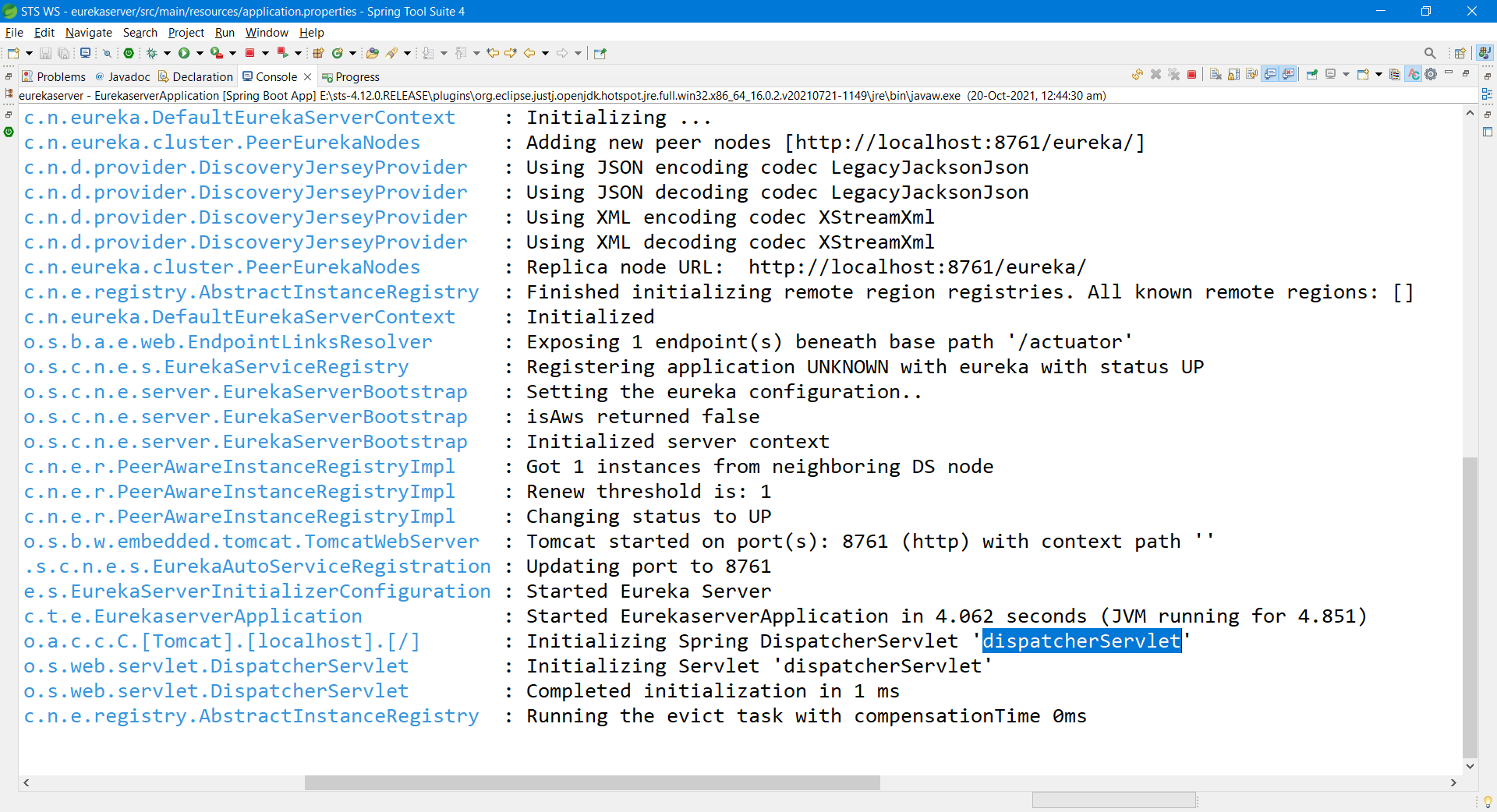
#logging.level.com.netflix.eureka=OFF

#logging.level.com.netflix.discovery=OFF

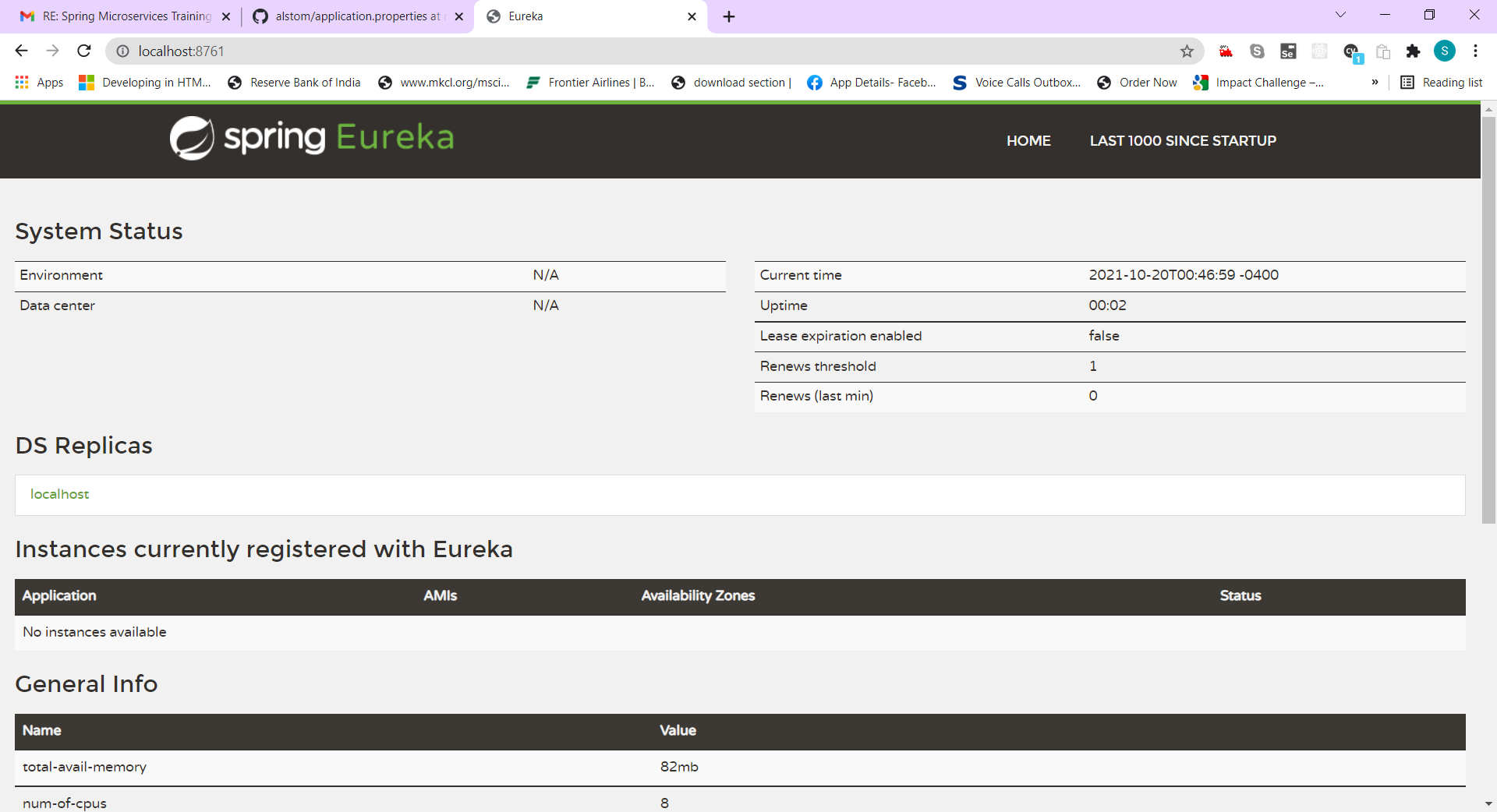
Step 4: Add @EnableEurekaServer in the Starter class



Step 5: Run the EurekaServer as a SpringBoot App



Step 6: Open the Eureka Server in browser by visiting <http://localhost:8761>



1)FoodItem

2) Cart

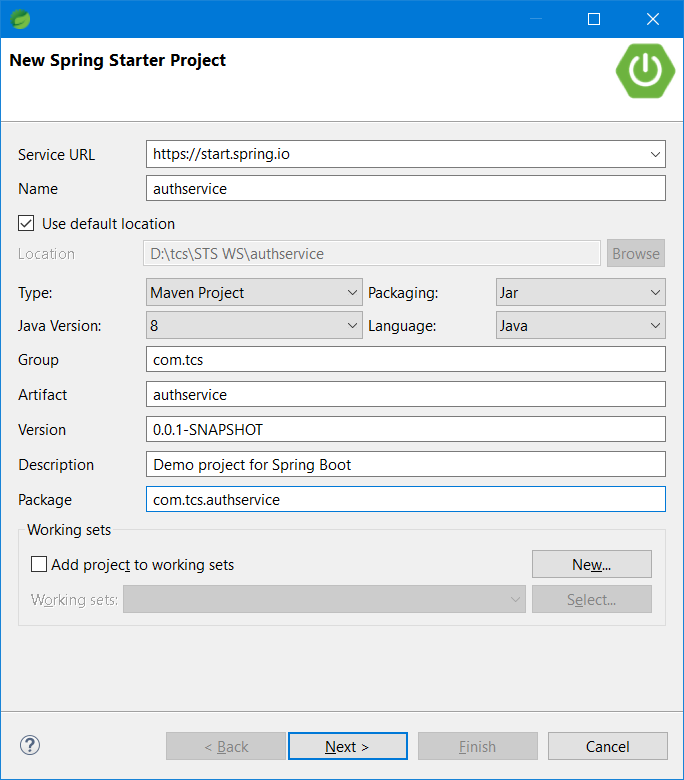
3) User

4) Rating

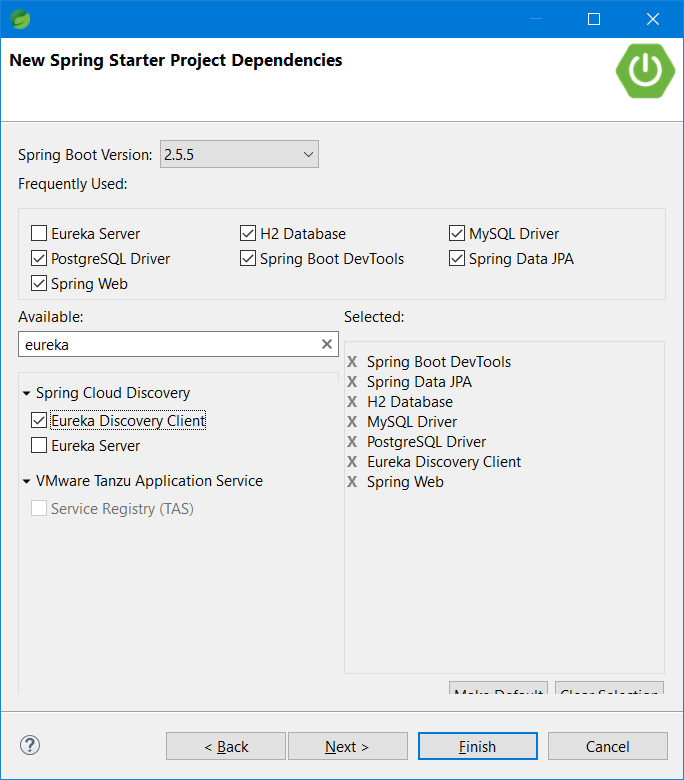
5) Role

Creating Authentication & Authorization Service (microservice -1)

Step 1: Create a new Spring Starter project in STS

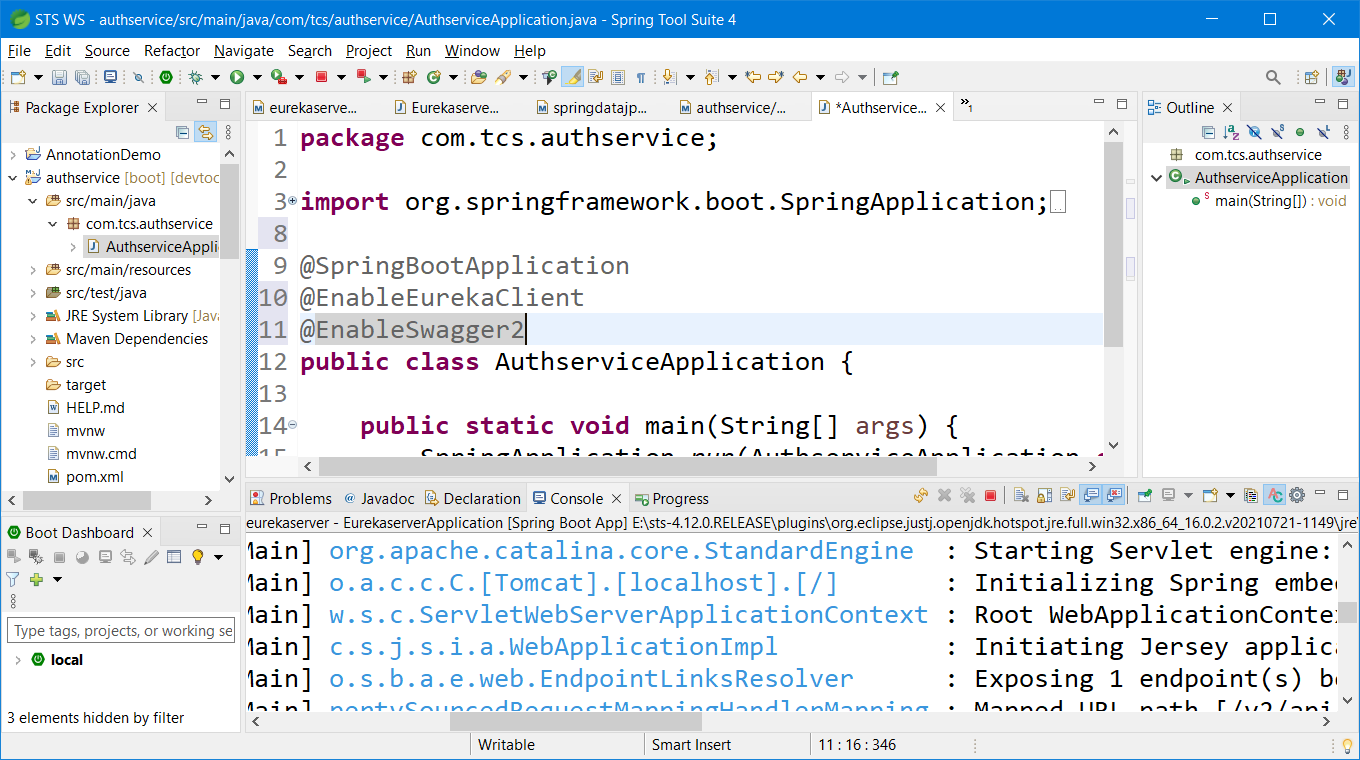


Step 2: Add the following dependencies



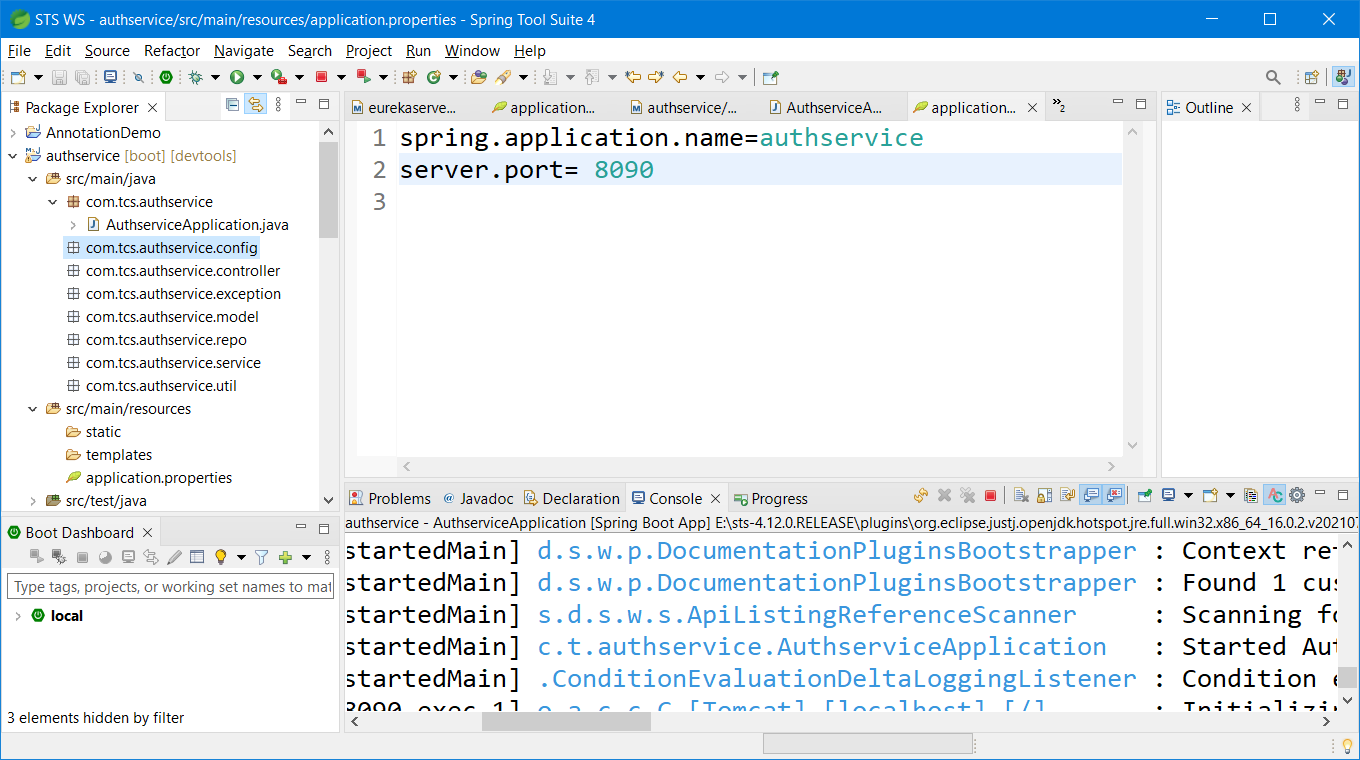
Click on Finish.

Step 3: Add Swagger2 dependencies and force update the project. Then add following annotations to the starter class.



Step 4:

<https://hellokoding.com/spring-security-login-logout-thymeleaf/> add all the necessary packages to the project and update application.properties file as shown below.



Step 5: Add Thymeleaf, security and hsql dependency to the pom.xml

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.thymeleaf.extras</groupId>

<artifactId>thymeleaf-extras-springsecurity5</artifactId>

</dependency>

<dependency>

<groupId>org.hsqldb</groupId>

<artifactId>hsqldb</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

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

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

</dependency>

Eurekaserver – 8761

Authservice – 8090 (user & role)

Fooditemservice – 8091

Cartservice -8092

Ratingservice – 8093

Userservice -8094 (add/deleteuser/update/view all user) [optional]