

## e-shopping app

frontend ---→ angular application

backend -→ spring boot with micro service concept to develop the rest api

the communication between front end and backend is rest api

## course-3-shopping-app

frontend

backend

folder

### backend

1<sup>st</sup>

Project      Eureka – Server

With starter web and eureka server : port number 8761

The screenshot shows the Spring Initializr web application interface. The 'Project' section has 'Maven' selected under 'Project' and 'Java' under 'Language'. The 'Spring Boot' section has '3.1.8' selected. The 'Project Metadata' section has 'Artifact' set to 'Eureka-Server', 'Name' set to 'Eureka-Server', 'Description' set to 'Demo project for Spring Boot', 'Package name' set to 'com', and 'Packaging' set to 'Jar'. The 'Dependencies' section has 'Spring Web' and 'Eureka Server' selected. The 'Generate' button is highlighted.

spring initializr

**Project**  
☒ Gradle - Groovy ☐ Gradle - Kotlin ☒ **Java** ☐ Kotlin ☐ Groovy

**Spring Boot**  
☐ 3.3.0 (SNAPSHOT) ☒ **3.1.8** ☐ 3.2.3 (SNAPSHOT) ☐ 3.2.2

**Project Metadata**  
Group: com.example  
**Artifact:** Eureka-Server  
Name: Eureka-Server  
Description: Demo project for Spring Boot  
Package name: com  
Packaging: ☒ **Jar** ☐ War  
Java: ☐ 21 ☒ **17**

**Dependencies**  
**Spring Web** **WEB**  
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.  
**Eureka Server** **SPRING CLOUD DISCOVERY**  
spring-cloud-netflix Eureka Server.

**ADD DEPENDENCIES...** CTRL + B

**GENERATE** CTRL + G **EXPLORE** CTRL + SPACE **SHADE**

### application.properties

```
server.port=8761
eureka.client.register-with-eureka=false
eureka.client.fetch-registry=false
```

2<sup>nd</sup> database **caltech\_course3\_eshopping\_app**

Project login-app (admin and customer with signIn and signUp)

With starter port 8181 with database

Web

Eureka client

Jpa

Mysql

Dev tool

2<sup>nd</sup> rest api

**emailId, password and typeOfuser**

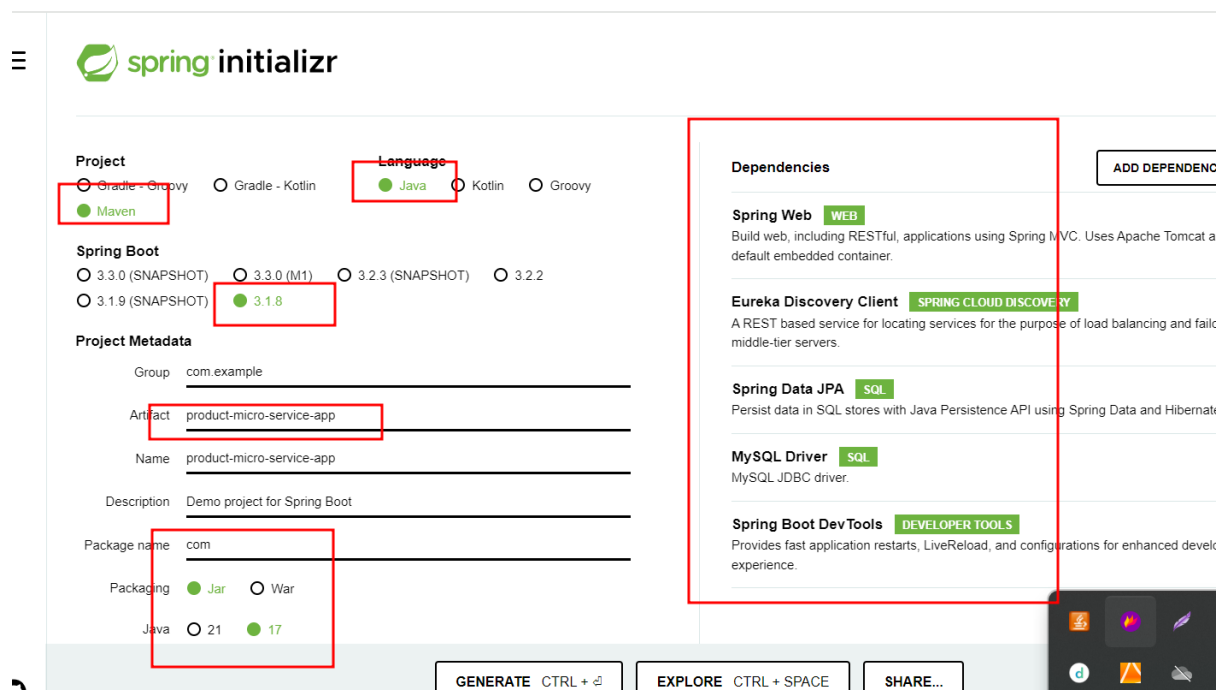
Admin user only Signin and for customer SignIn and SignUp

SignIn <http://localhost:8181/login/signIn> post with json data

SignUp <http://localhost:8181/login/signUp> post with json data

The screenshot displays the Spring Initializr web application interface. The 'Project' section has 'Maven' selected. The 'Language' section has 'Java' selected. The 'Spring Boot' section has '3.1.8' selected. The 'Project Metadata' section has 'Artifact' set to 'Login-micro-service-app', 'Name' set to 'Login-micro-service-app', 'Description' set to 'Demo project for Spring Boot', 'Package name' set to 'com', 'Packaging' set to 'Jar', and 'Java' version set to '17'. The 'Dependencies' section lists 'Spring Web', 'Eureka Discovery Client', 'Spring Data JPA', 'MySQL Driver', and 'Spring Boot DevTools'. A red box highlights the 'Language', 'Spring Boot', 'Artifact', 'Package name', 'Packaging', and 'Dependencies' sections. A green bar at the bottom indicates 'You are screen sharing'.

3rd database **caltech\_course3\_eshopping\_app**  
Project product\_app  
With starter port 8282 with database  
Web  
Eureka client  
Jpa  
Mysql  
Dev tool



The image shows the Spring Initializr web interface for configuring a new project. Several fields are highlighted with red boxes:

- Project:** The **Maven** radio button is selected.
- Language:** The **Java** radio button is selected.
- Spring Boot:** The **3.1.8** version is selected.
- Project Metadata:**
  - Artifact:** The value **product-micro-service-app** is entered.
  - Package name:** The value **com** is entered.
  - Packaging:** The **Jar** radio button is selected.
  - Java:** The **17** version is selected.
- Dependencies:** A list of dependencies is shown, including **Spring Web**, **Eureka Discovery Client**, **Spring Data JPA**, **MySQL Driver**, and **Spring Boot DevTools**. An **ADD DEPENDENCY** button is located at the top right of this section.

At the bottom of the interface, there are buttons for **GENERATE** (with a keyboard shortcut **CTRL + G**), **EXPLORE** (with a keyboard shortcut **CTRL + SPACE**), and **SHARE...**. A small system tray icon is visible in the bottom right corner.

<http://localhost:8282/product/addProduct>

<http://localhost:8282/product/viewProduct>

<http://localhost:8282/product/deleteProduct>

<http://localhost:8282/product/updateProduct>

Frontend

ng new e-shopping-app

routing -> yes

styling -> css

with bootstrap features.

ng g c login                      component

ng g c signup                    component

ng g class login                model class

ng g s login                    service class

ng g c dashboard              component              not using

ng g c adminhome              component

we need to allow to all call rest api of product ie

addProduct, deleteProduct, viewProduct and updateProduct

ng g c productbyadmin

add product, deleteproduct, updateproduct retrieveproduct

ng g c customerhome        component

viewProduct

ng g c productbycustomer

retrieveproduct

ng g class product

ng g s product

product model                      product entity                      product table  
angular model class ↔ Java bean or entity class ↔ Table in mysql

Variable name and column name must match

Cors : Cross Origin Resource Sharing

Two domain or server are communicating with each other.

4200	8181
Web server	tomcat server

Backend technologies we need to allow to  
access resources.