1. Download the dependency software

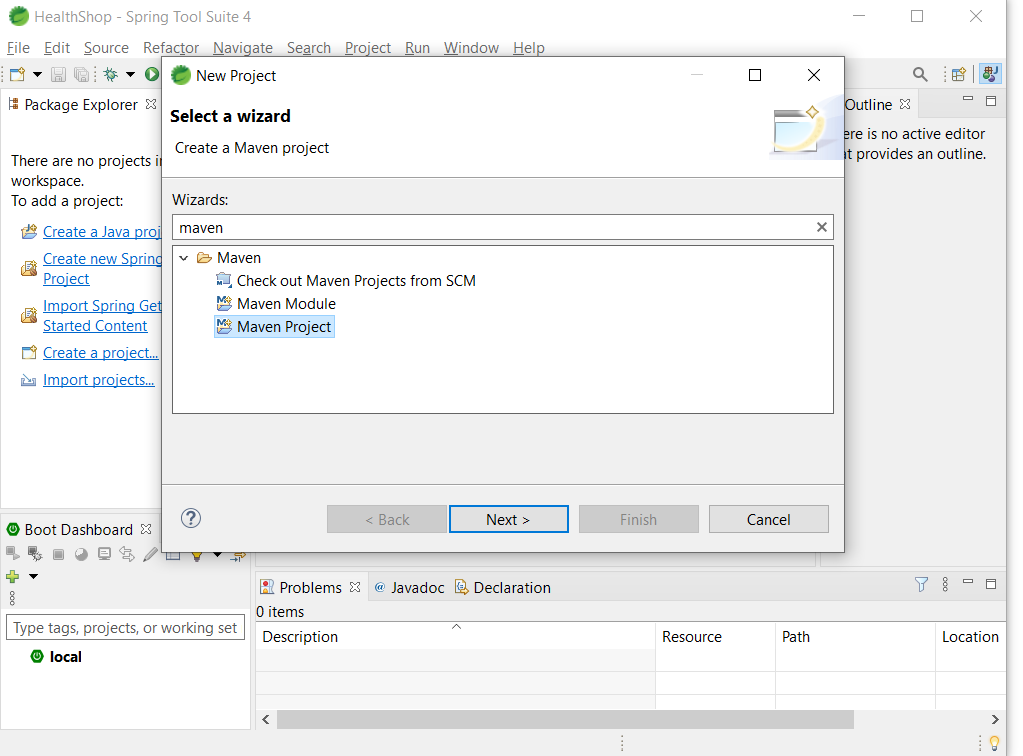
Spring Boot 4

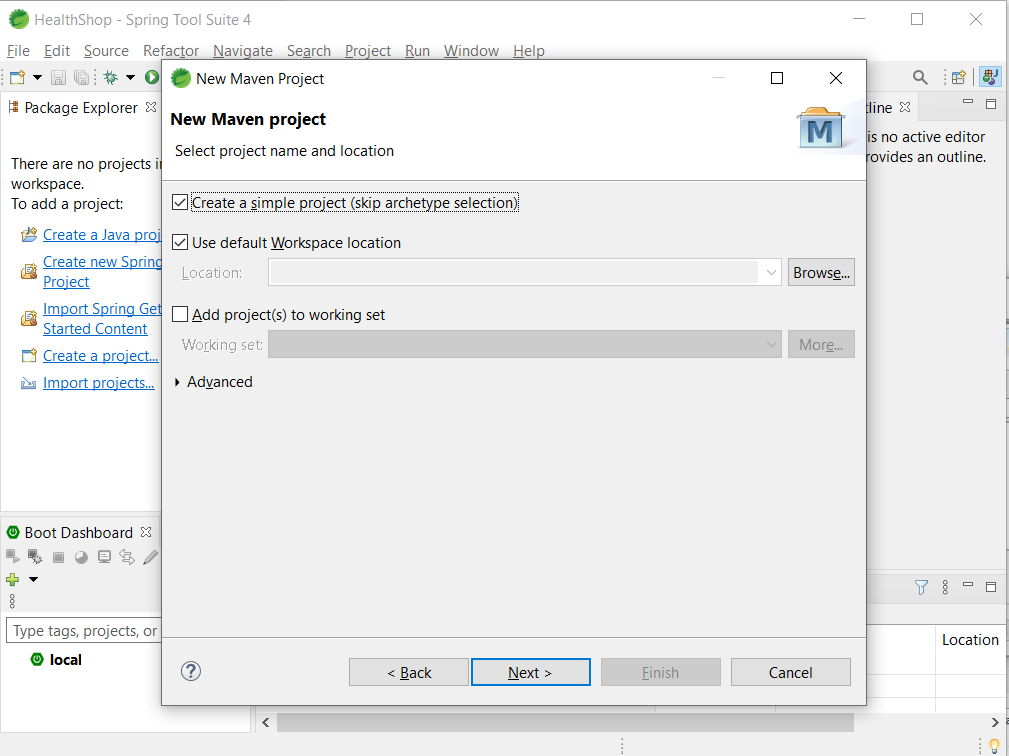
MySQL

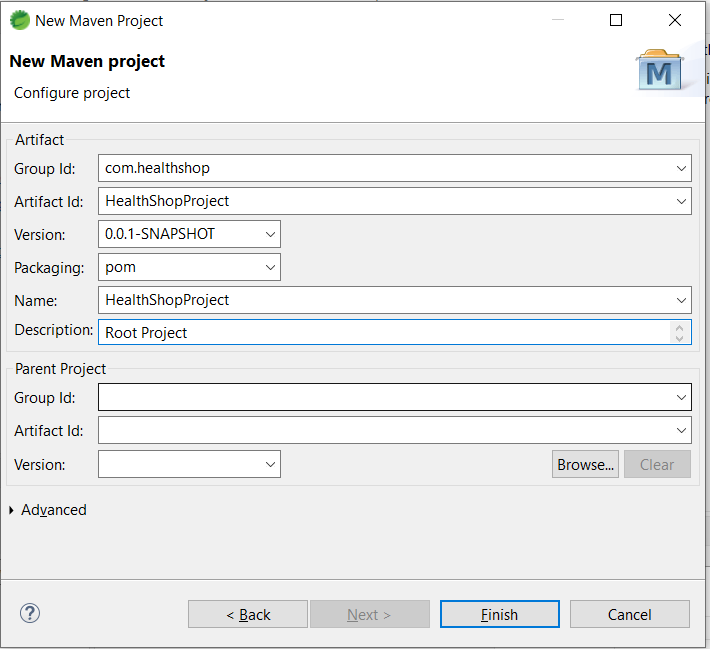
Maven

GIT

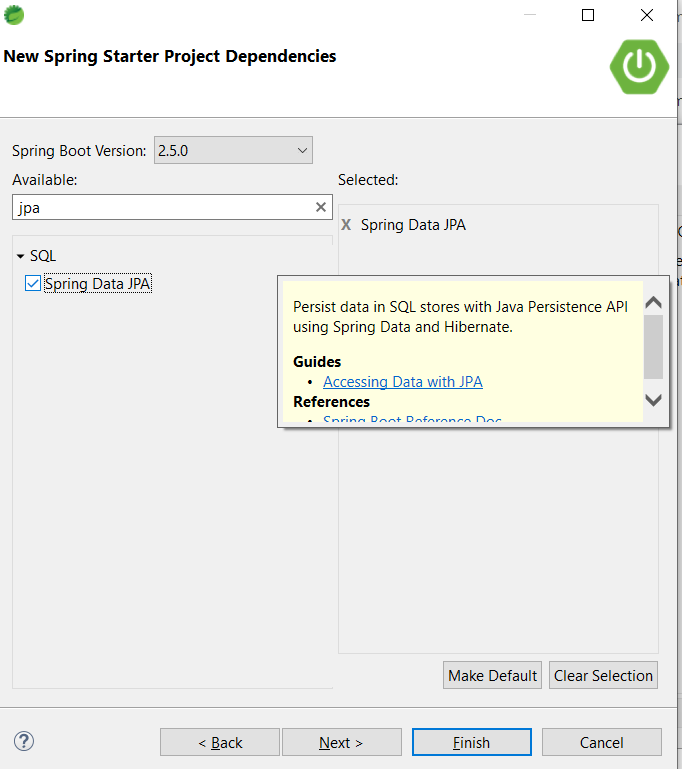
2. Create a new Project in Spring Boot



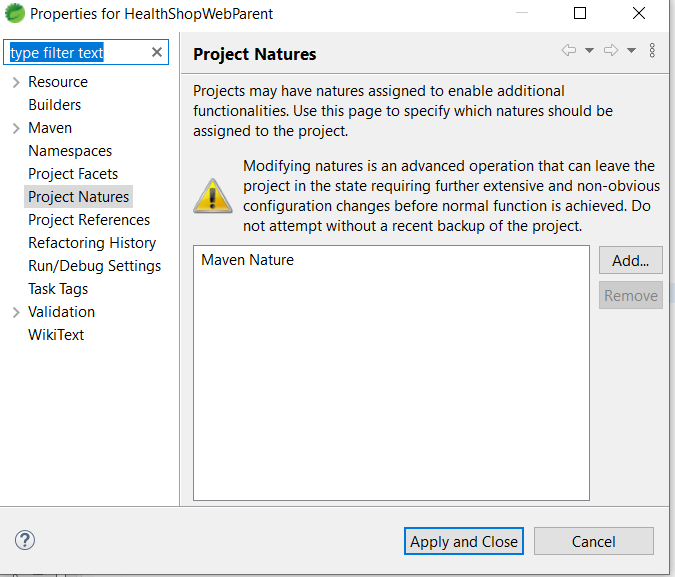








Similarly Create Spring starter project for HealthShopWebParent and remove the Java from Project Natures as it’s the root project for HealthShopFrontEnd and HealthShopBackEnd



Change the Pom.xml of HealthShopProject to include the HealthShopWebParent Module and HealthShopCommon module

<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>

<groupId>com.healthshop</groupId>

<artifactId>HealthShopProject</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>pom</packaging>

<name>HealthShopProject</name>

<description>Root Project</description>

<modules>

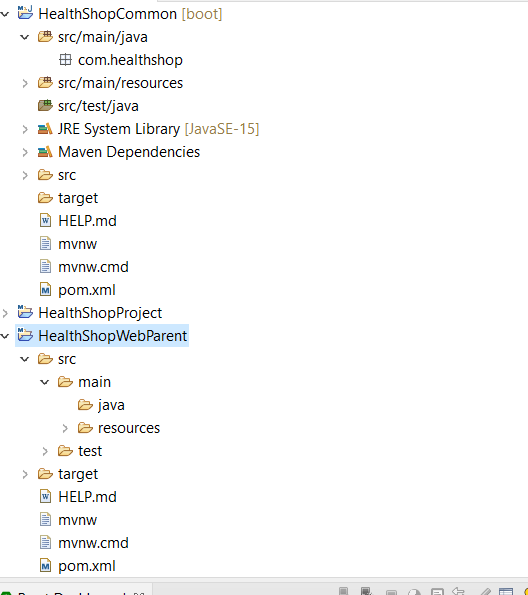
<module>HealthShopCommon</module>

<module>HealthShopWebParent</module>

</modules>

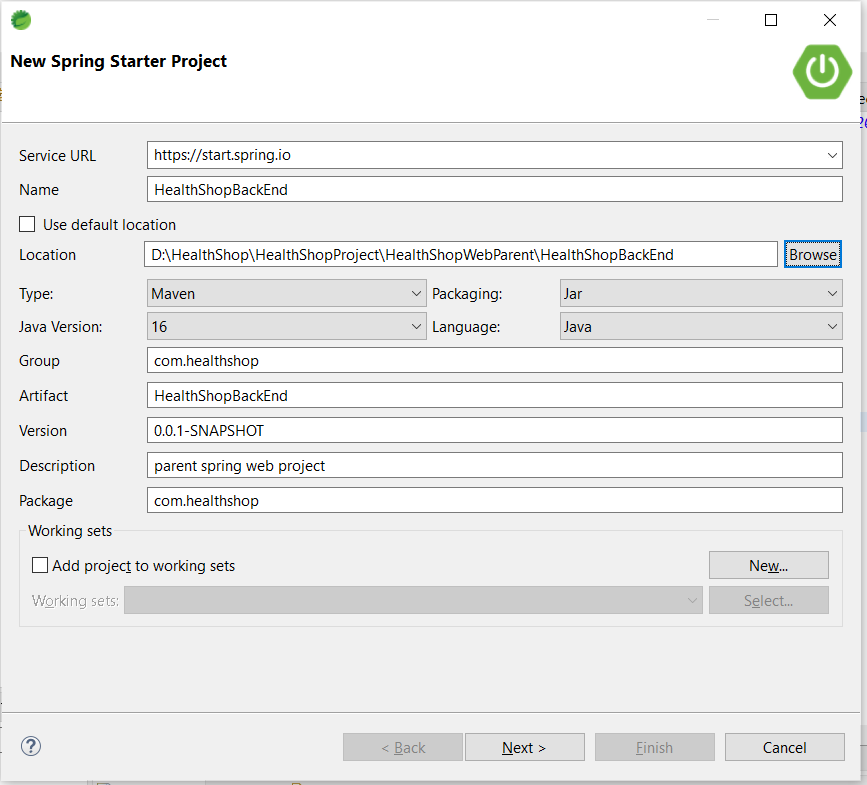
</project>

Remove the .java packages files from HealthShopCommon and HealthShopWebParent folders



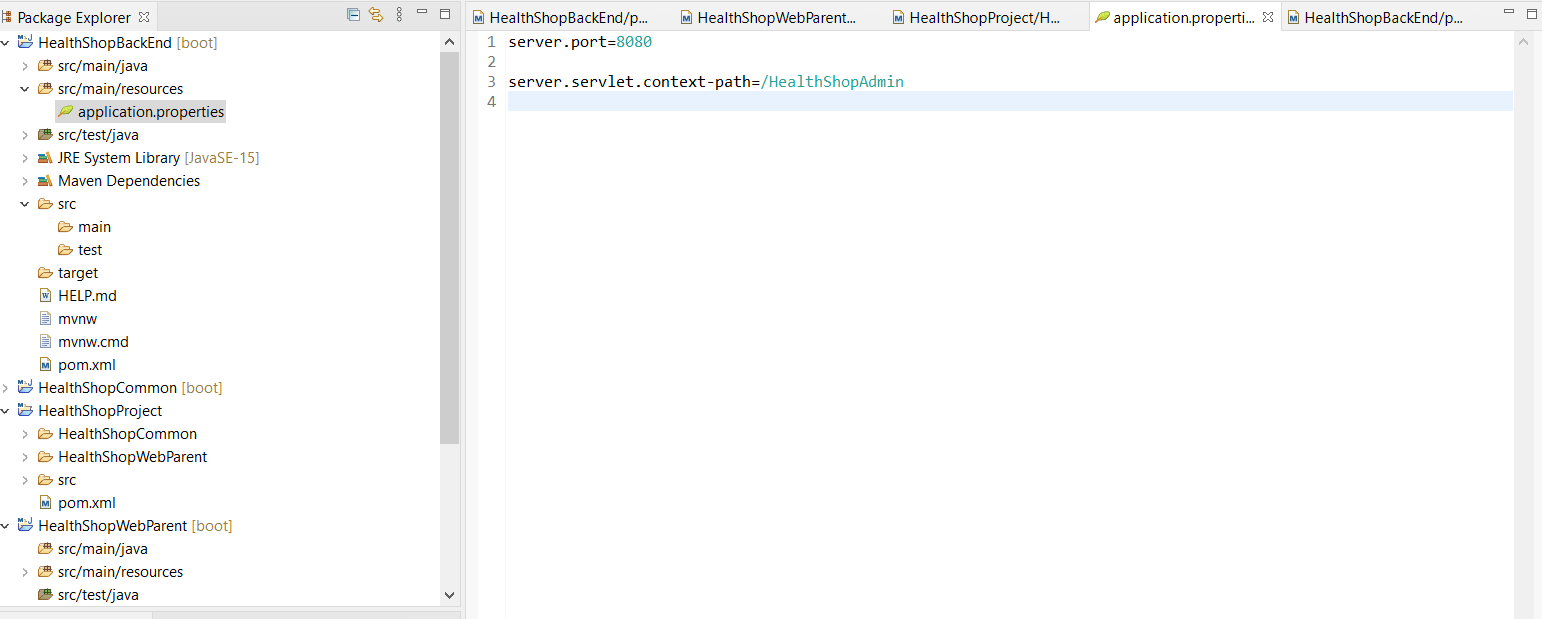
4. Create HealthShopBackend under HealthShopWebParent folder

Using Spring starter project

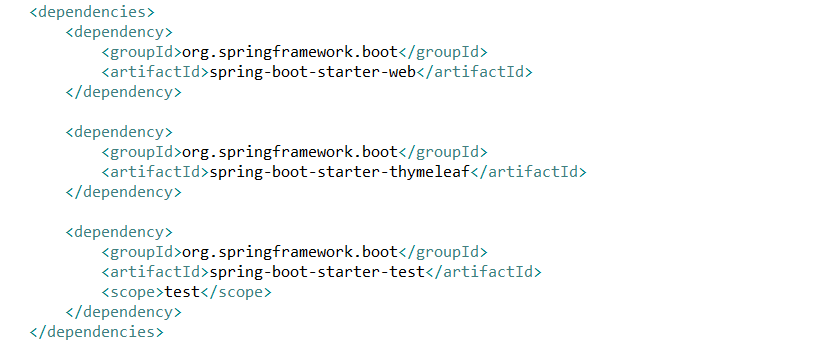


5. Edit the Pom.xml to include the HealthShopBackEnd module

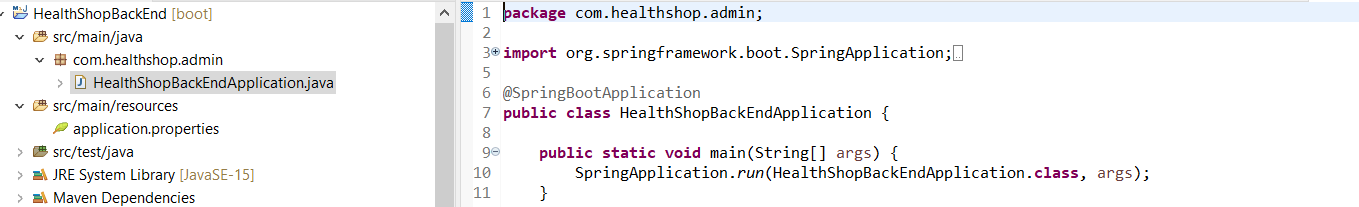
6. Edit the application.properties file in HealthShopBackEnd



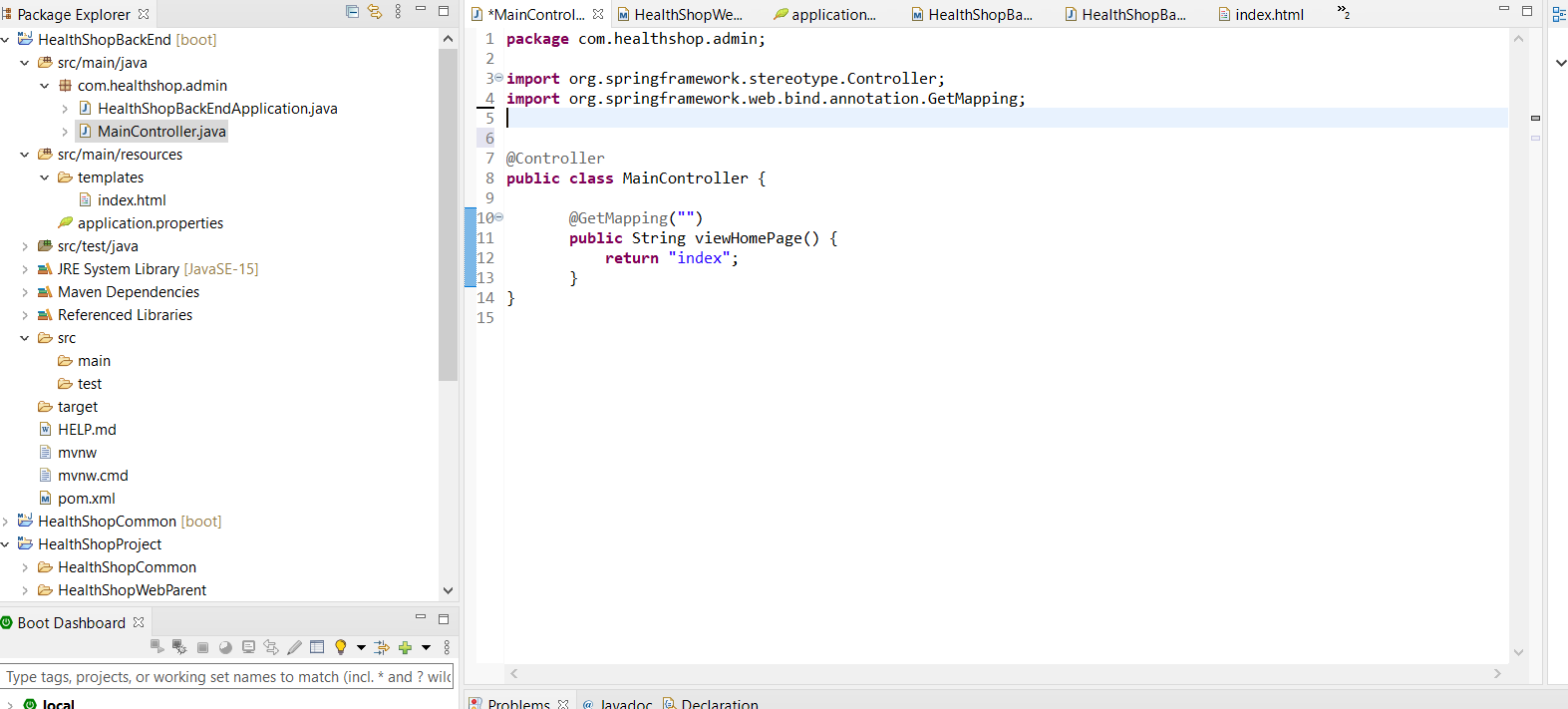
7. include Thymeleaf dependency



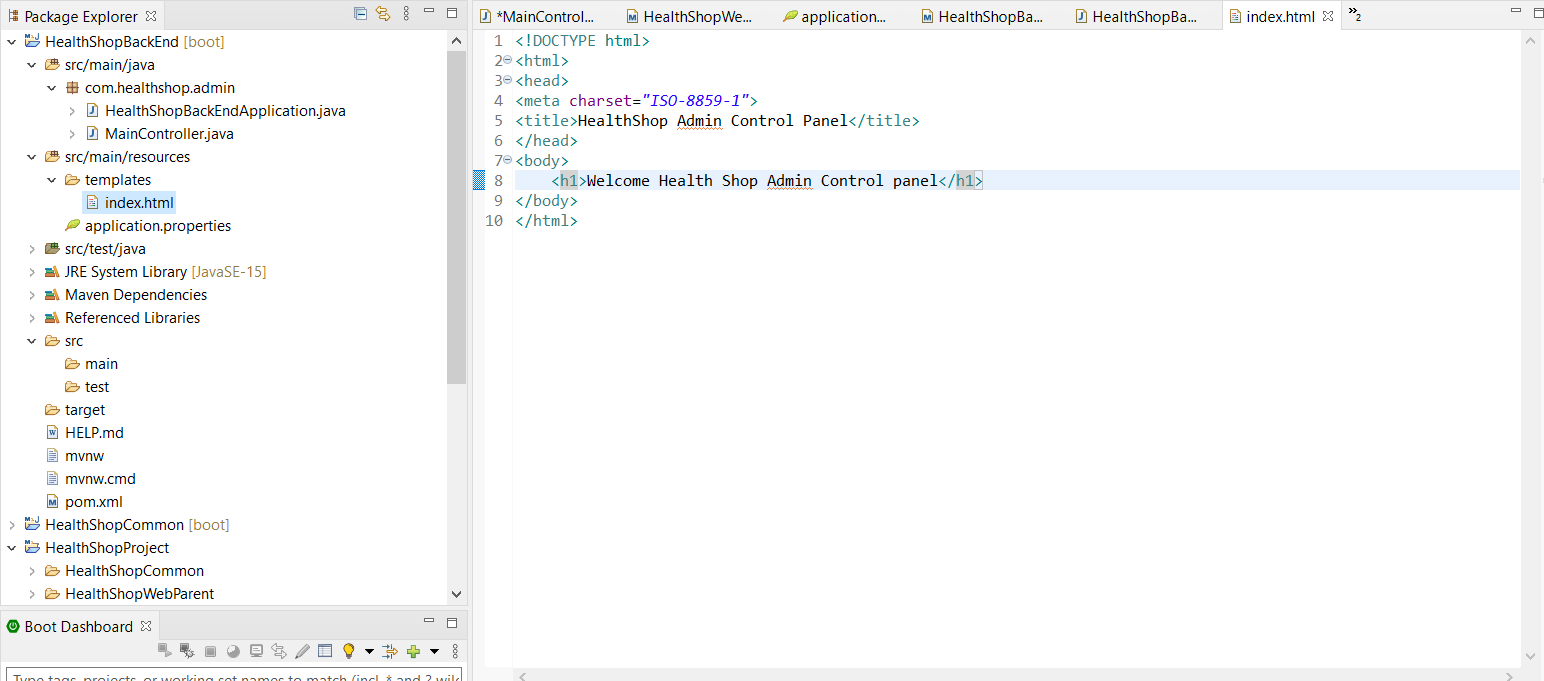
8. change the package under HealthShopAdmin from com.healthshop to com.healthshop.admin



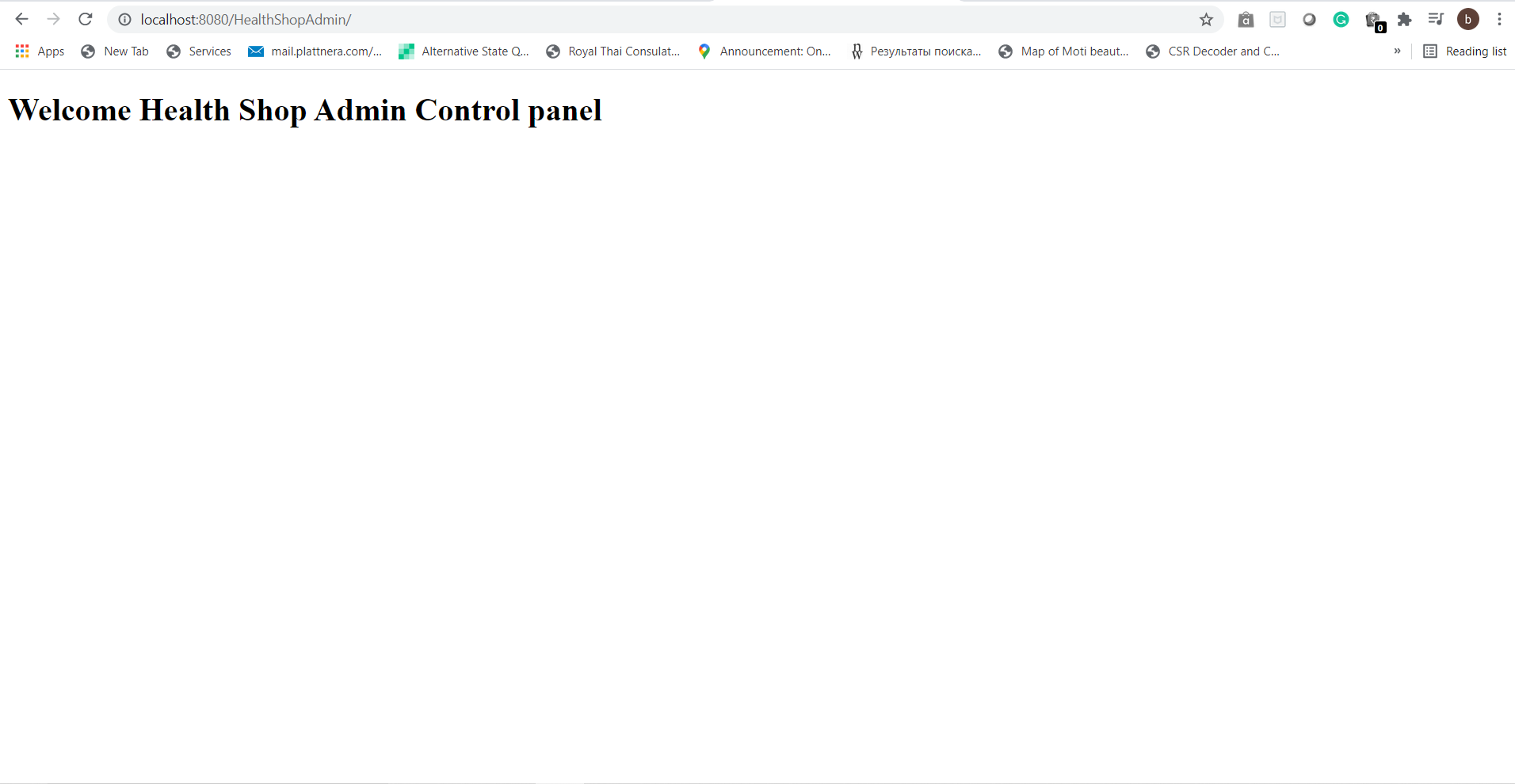
9. Create Controller Class for HealthShopBackEnd



10. Create index.html for HealthShopBackEnd



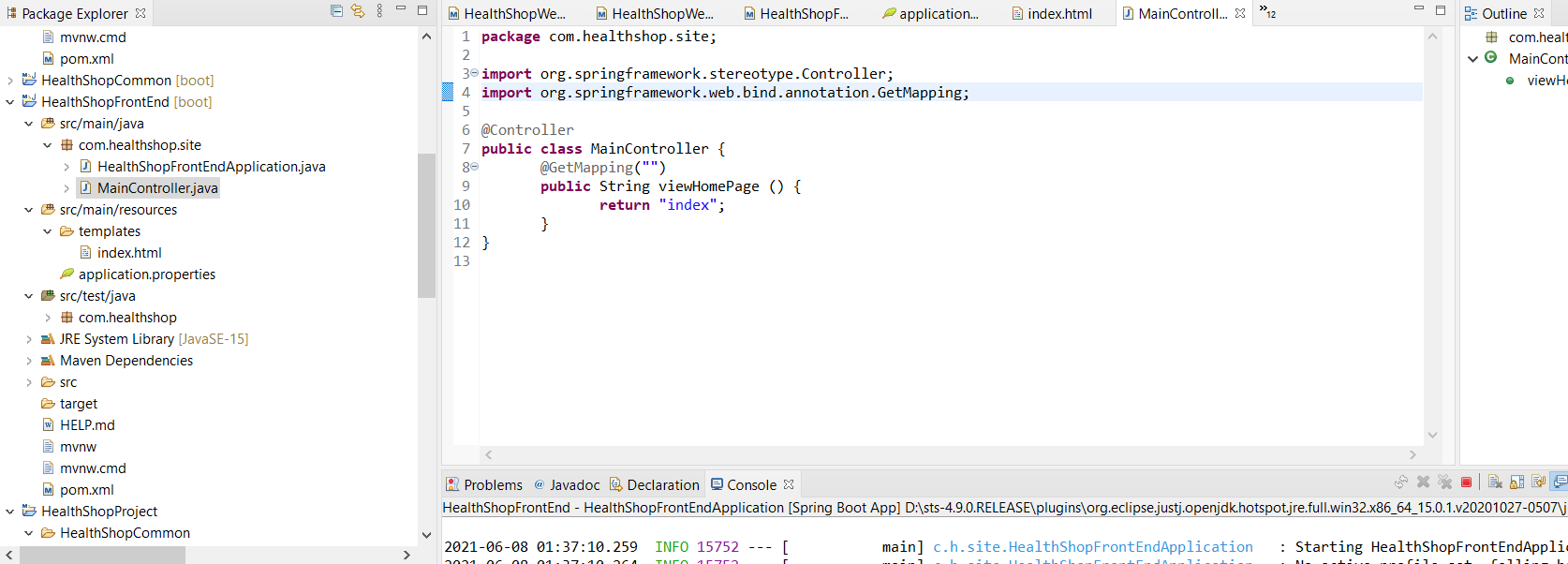
11. Run The HealthShopBackEnd and Test if its run coorectly in Browser



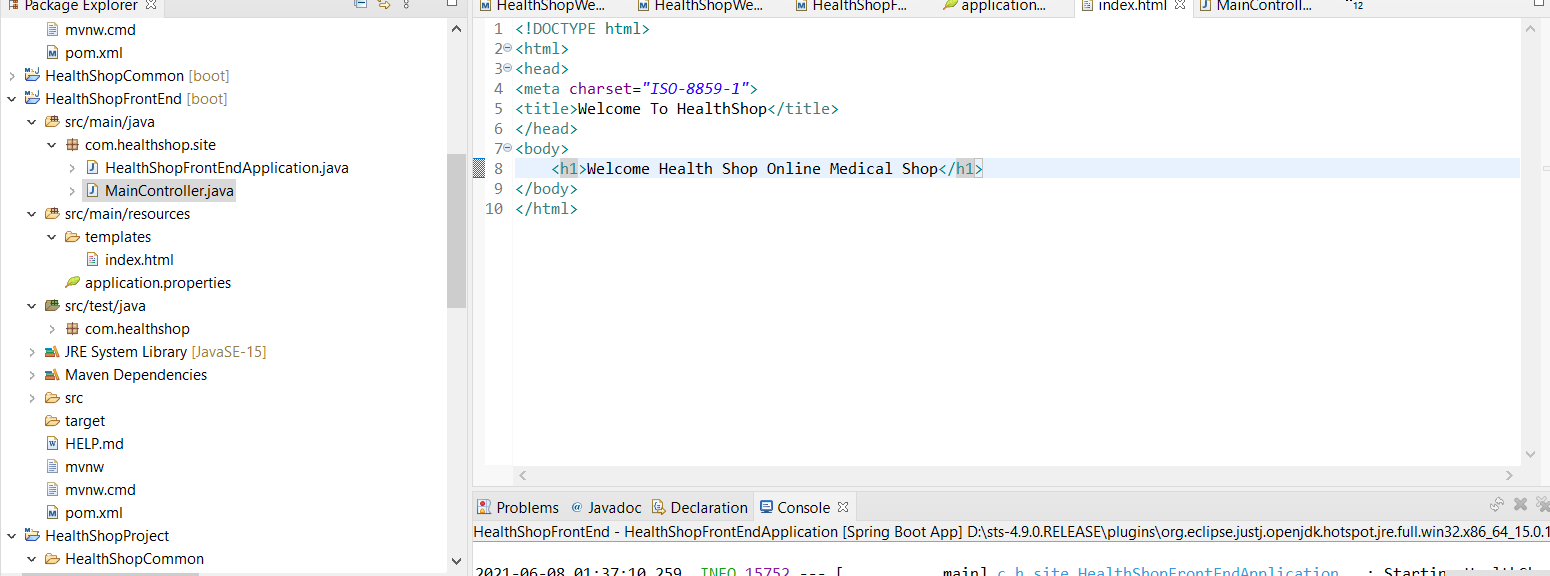
12. HealthShopFrontEnd

Edit POM.xml same as HealthShopBackEnd

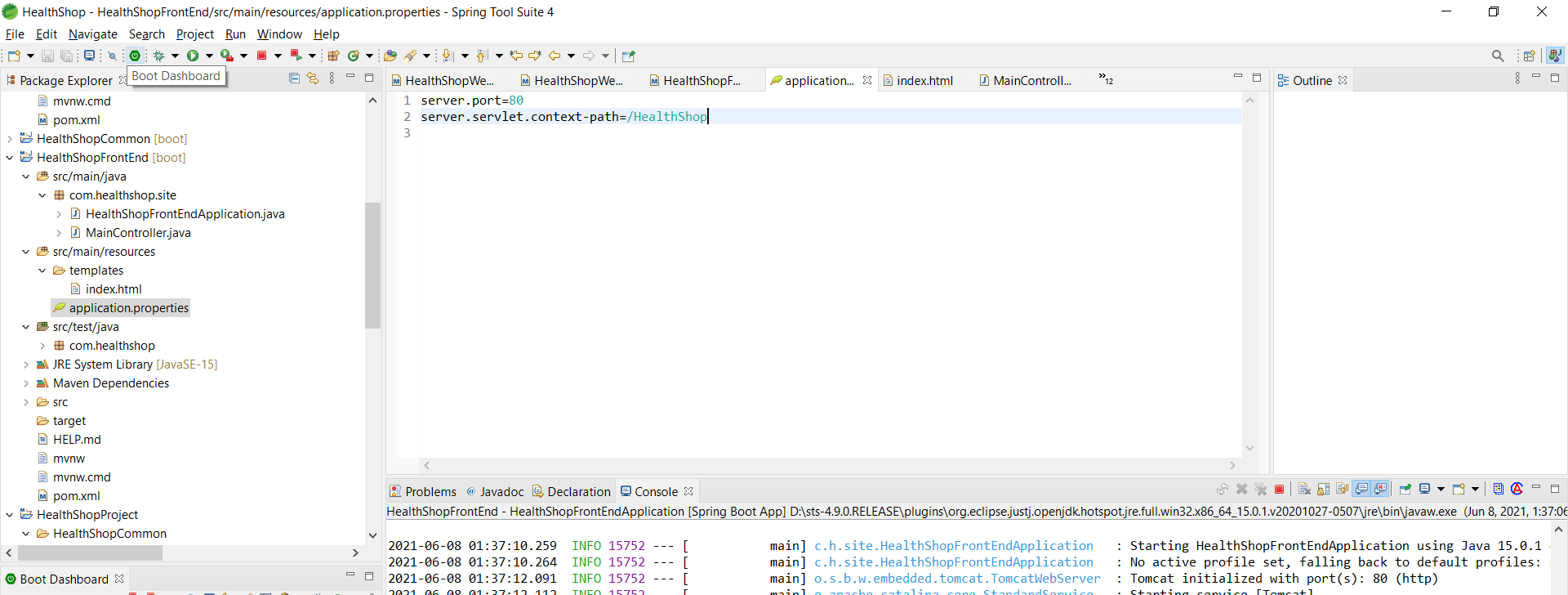
Create Controller class



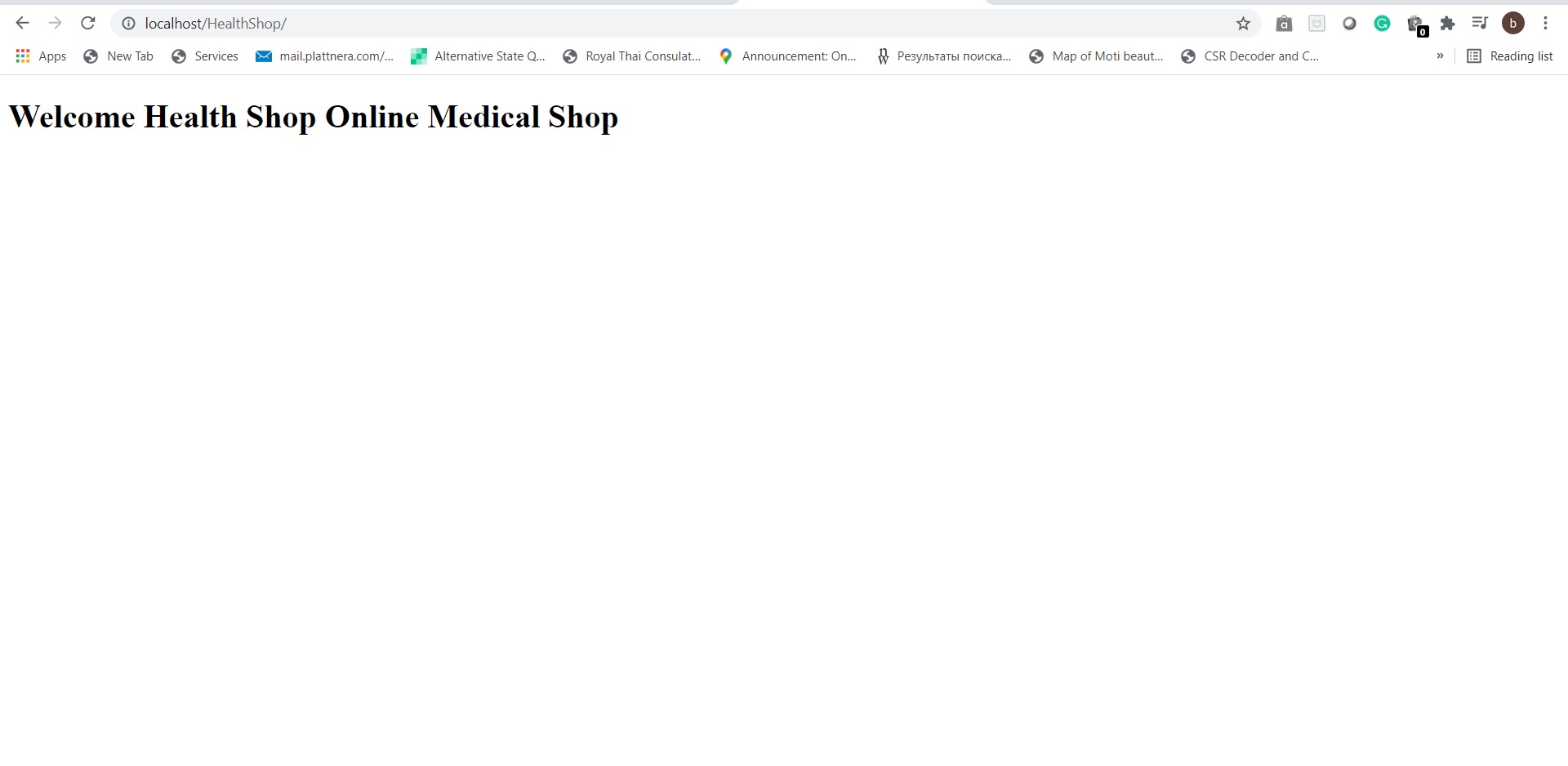
13. Create index.html



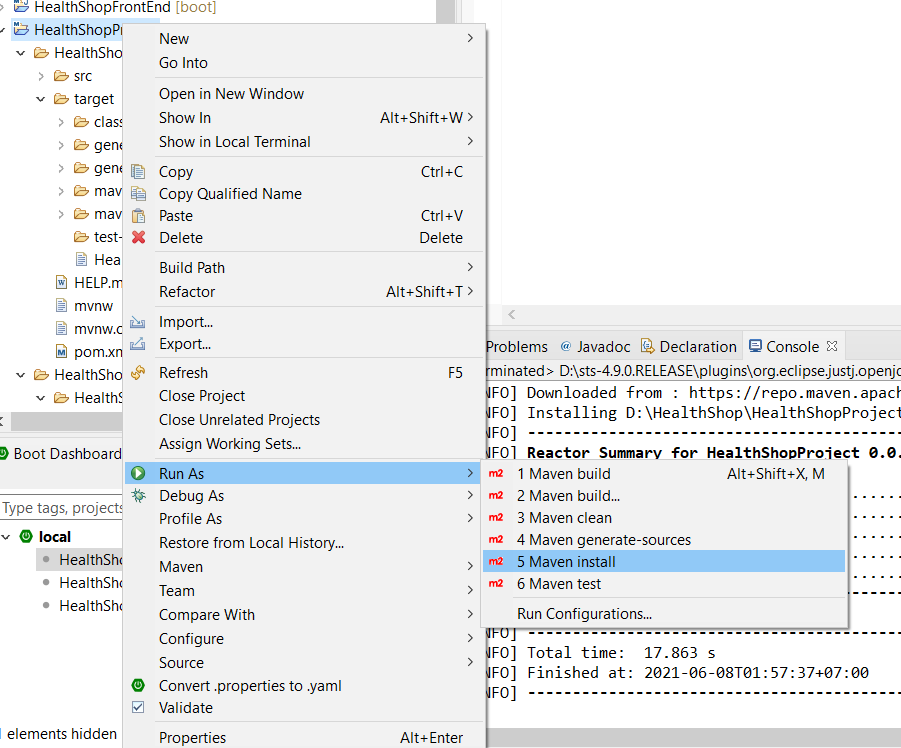
14. Add config to application.properties to HealthShopFrontEnd

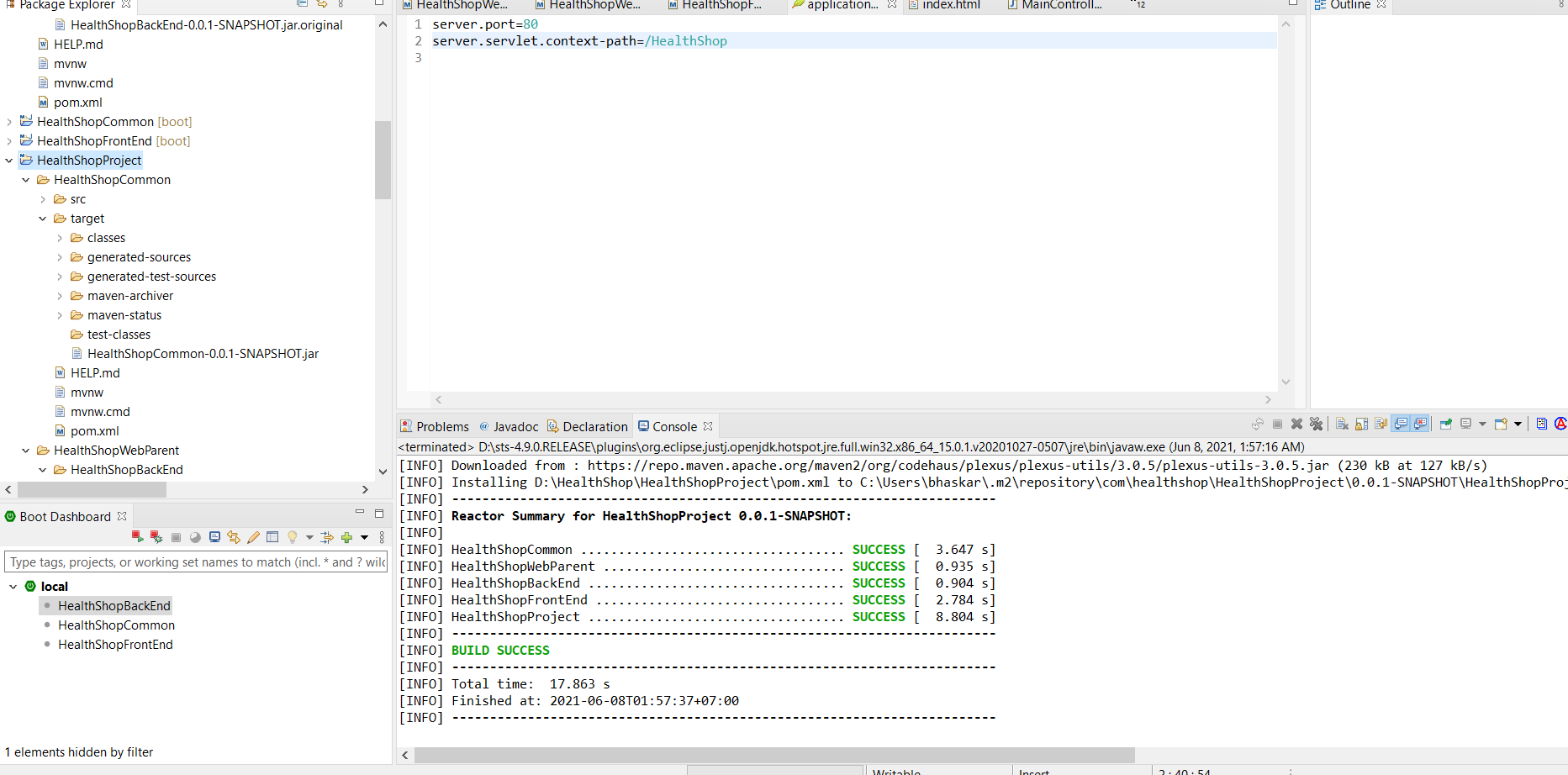


15. Run the FrontEnd to check if it starts fine

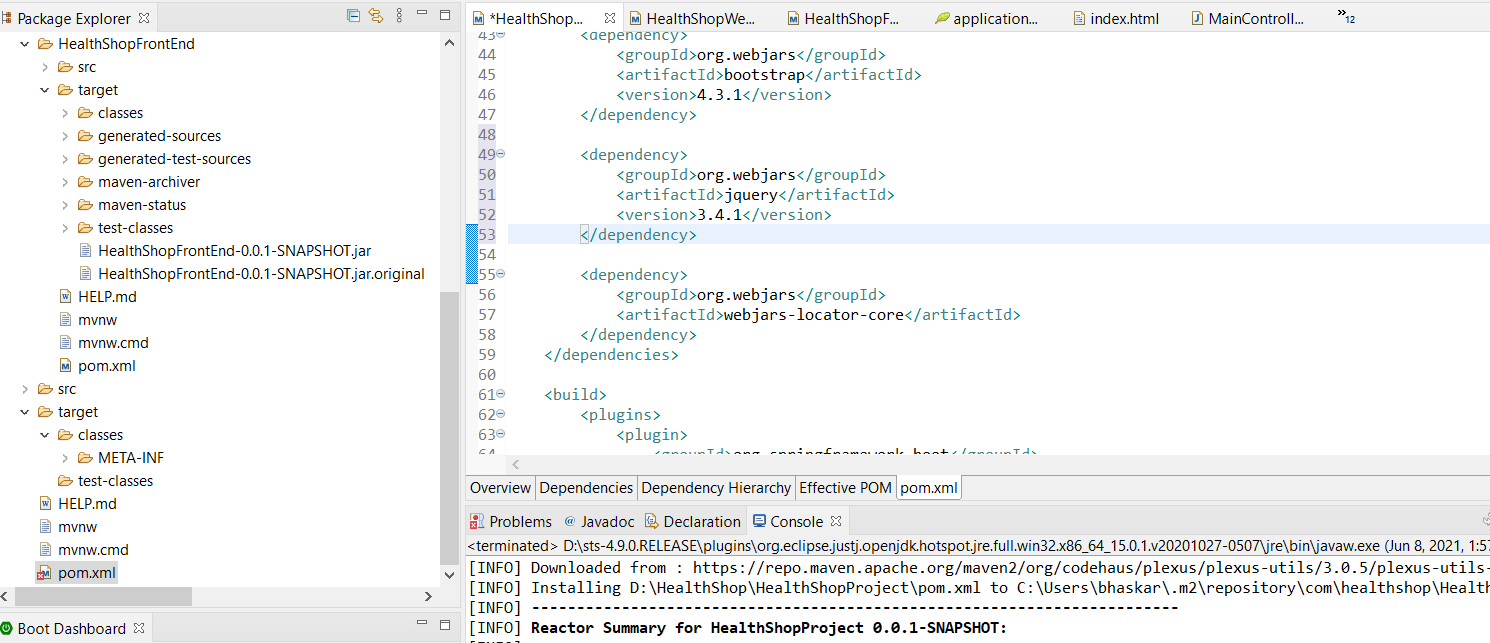


16. Compile using Maven Install right click





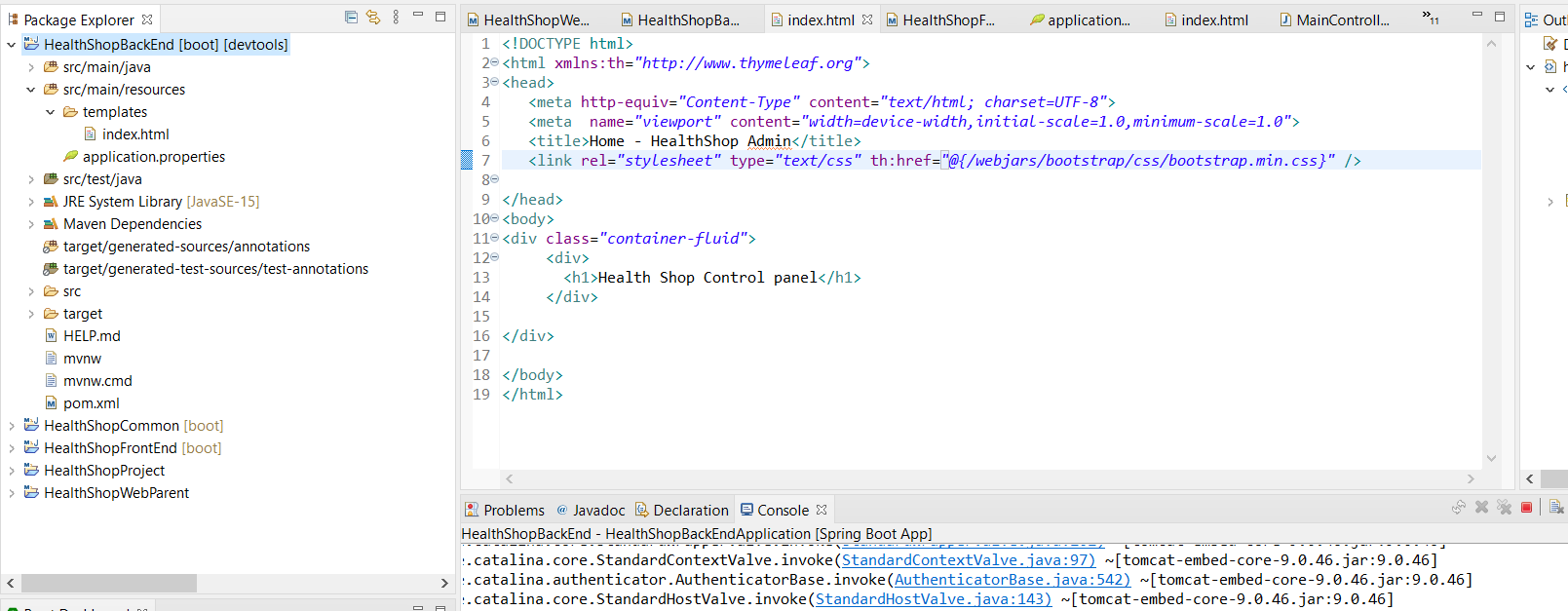
17.add bootstrap ,jquery to pom.xml of HealthShopWebParent



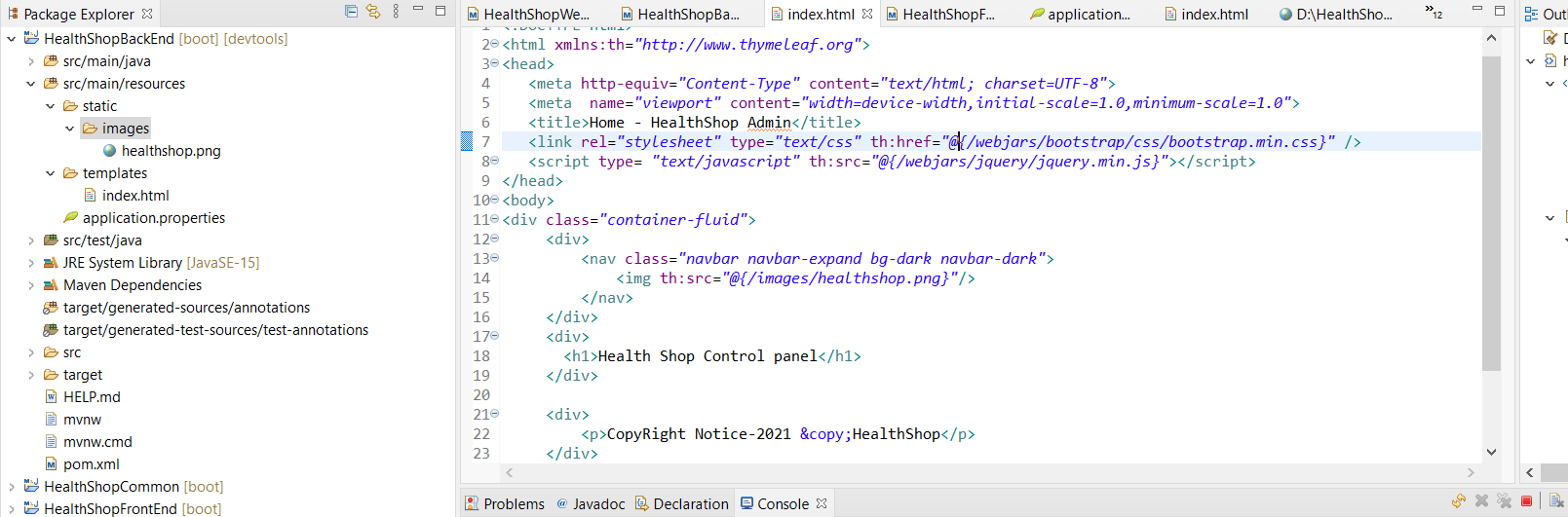
Runas-> maven install if webjars-locator-core does not work

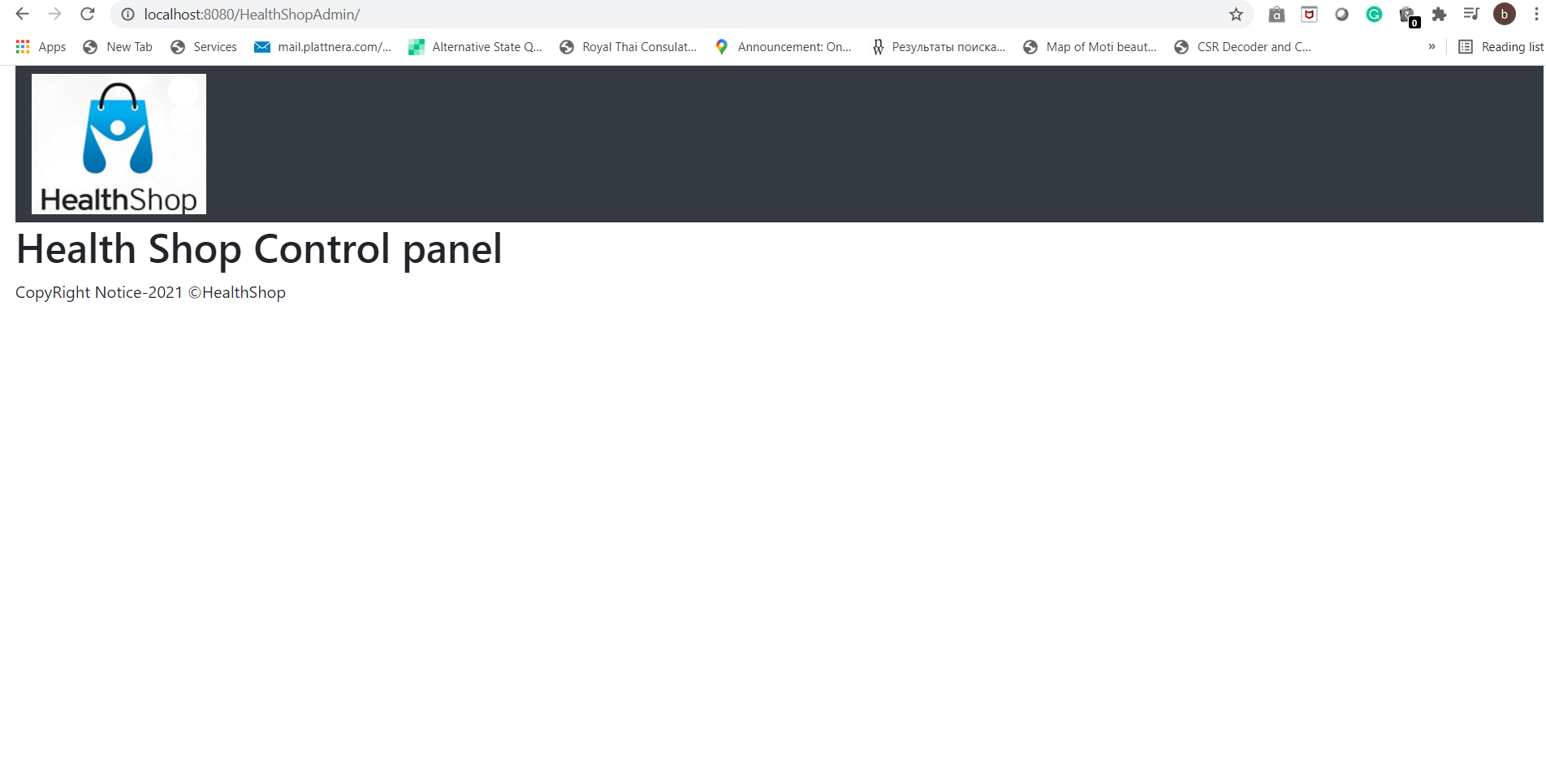
18. edit the inde.html file to include the bootstrap css and use thymeleaf to fetch the bootstrap URL

Plus right click on the HealthShopBackEnd and add devtools to project to automatically take the changes made into the project

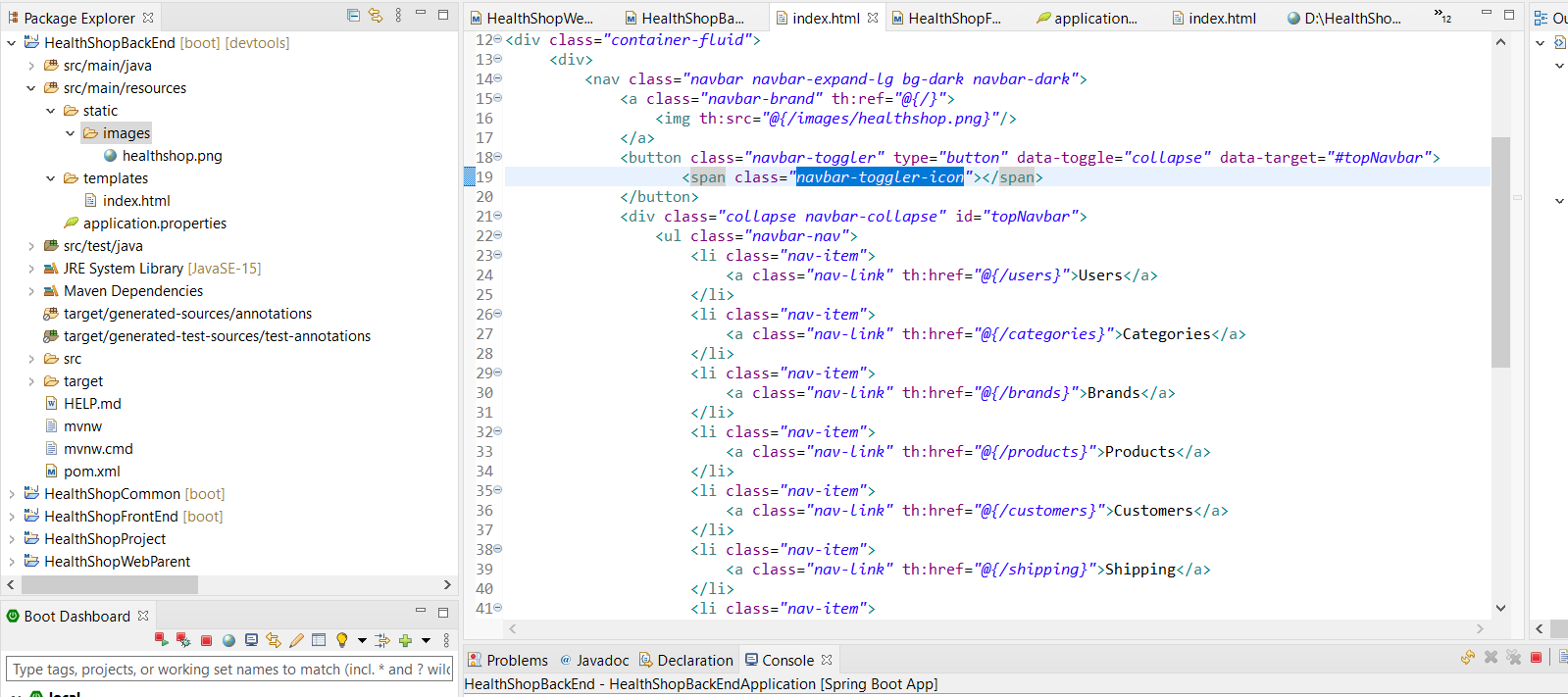


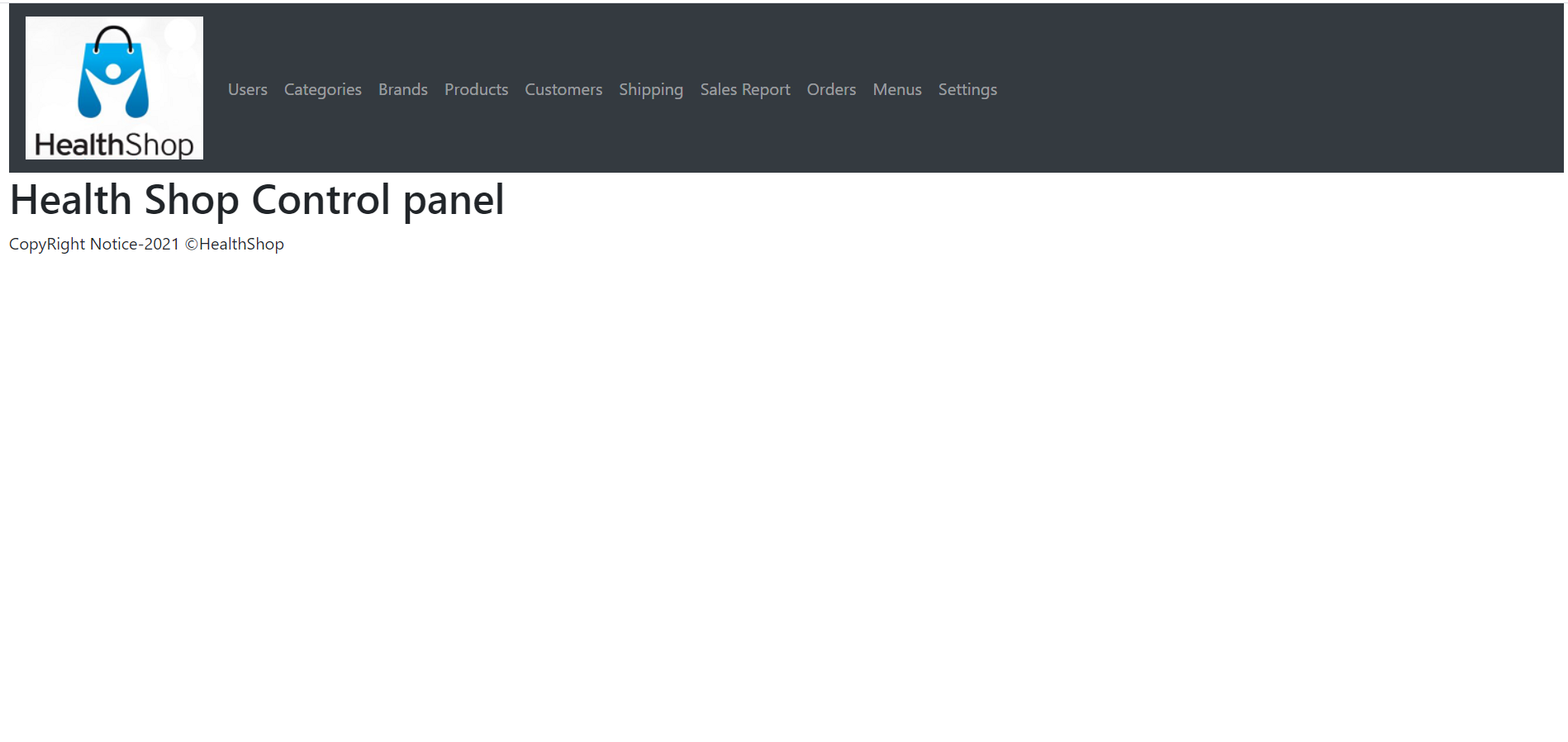
19. Edit the navbar display and logo to be displayed in index.html of HealthShopBackEnd

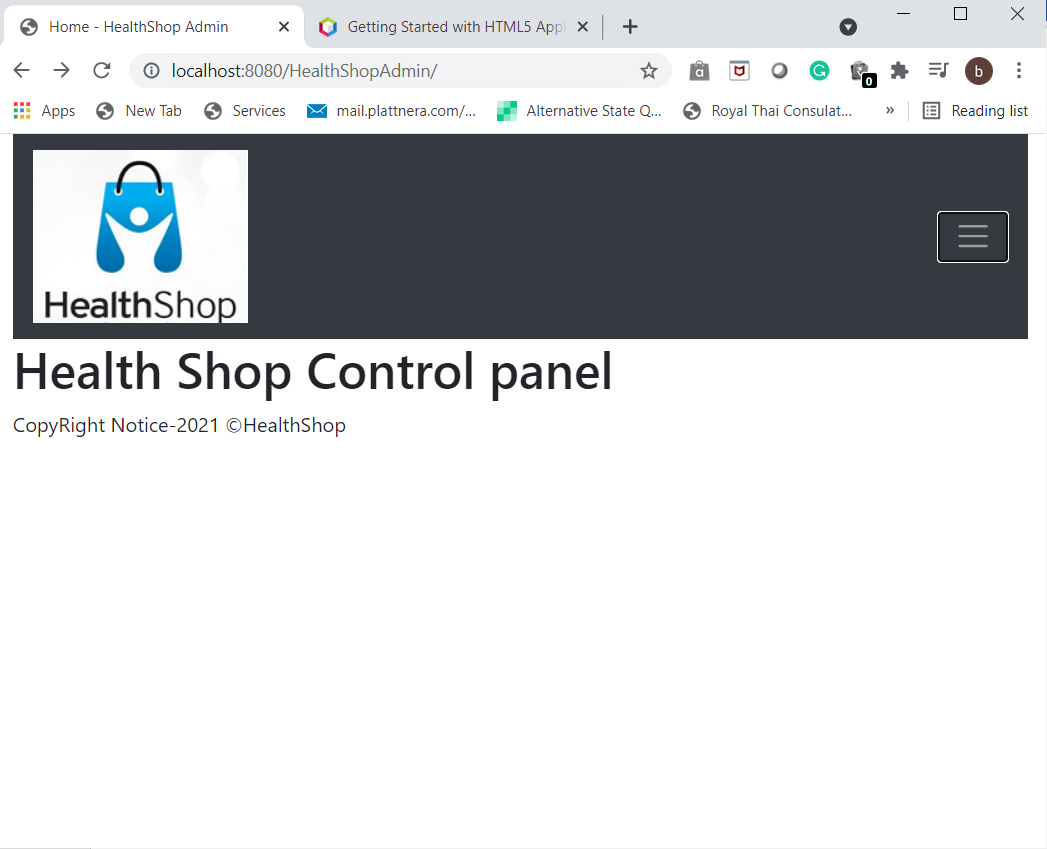


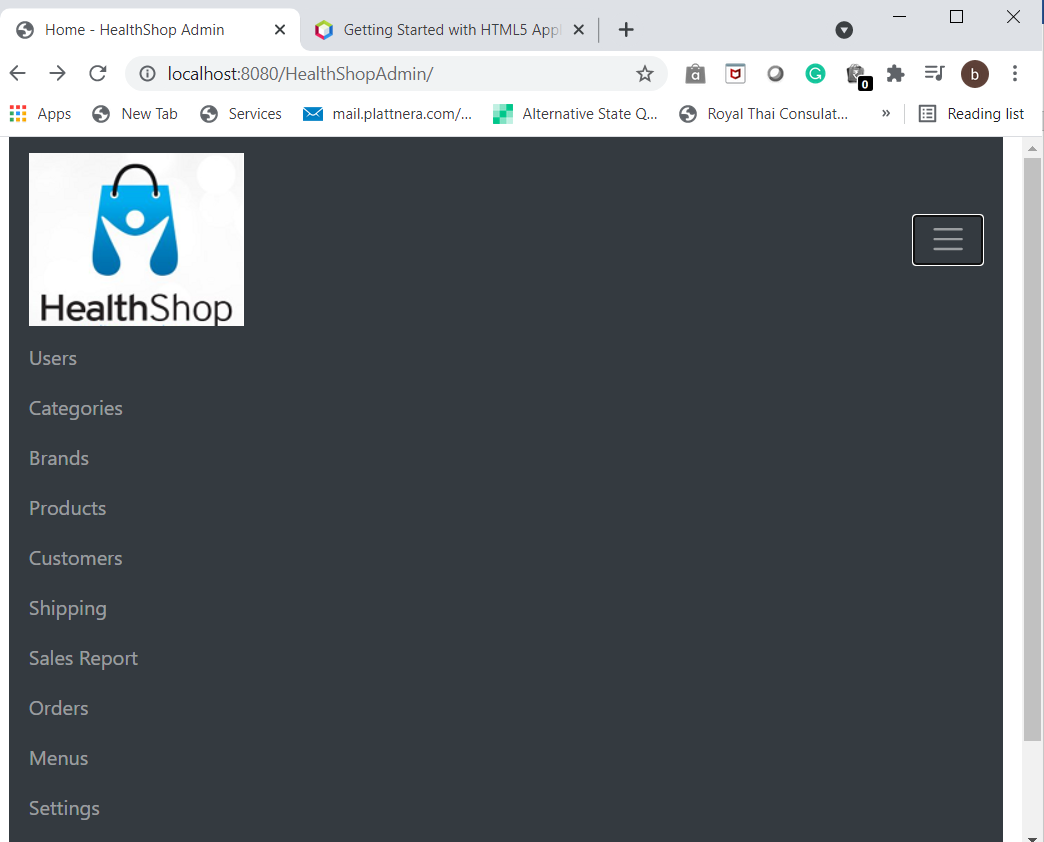


20. Put other buttons on navbar









21. Key Notes for Admin Control Panel

a. Use Bootstrap and Jquery visa Webjars dependencies

b. add Springdevtools for the project >automatic restart

c. Put static resources (eg: images ) under src/main/resources/static

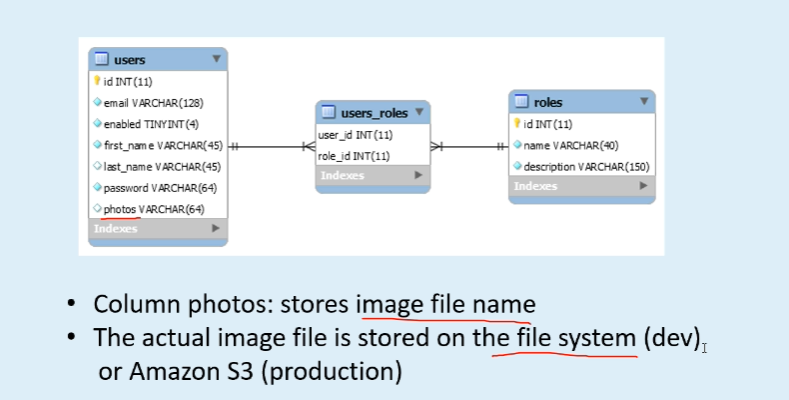
d. use Thymeleaf syntax for URL’s

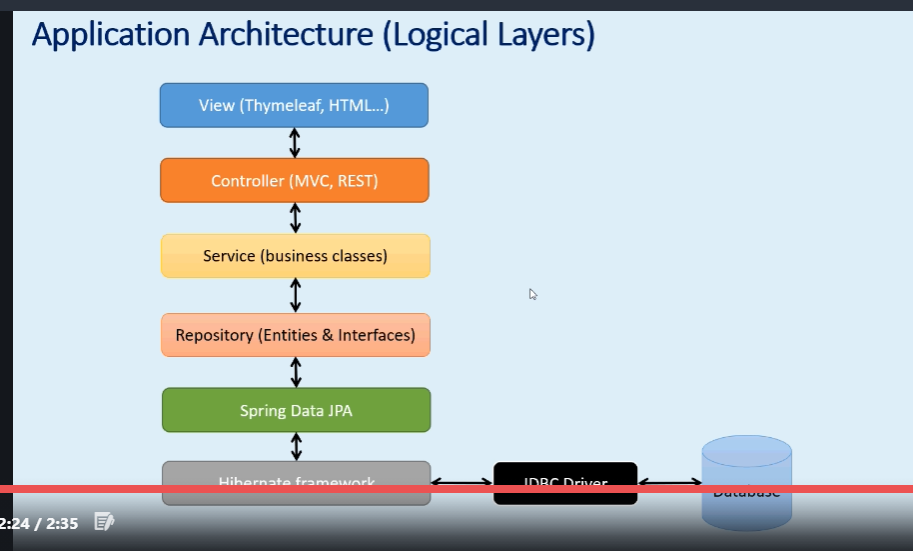
eg:

<img th:src=”@{/imageslogo.png}” />

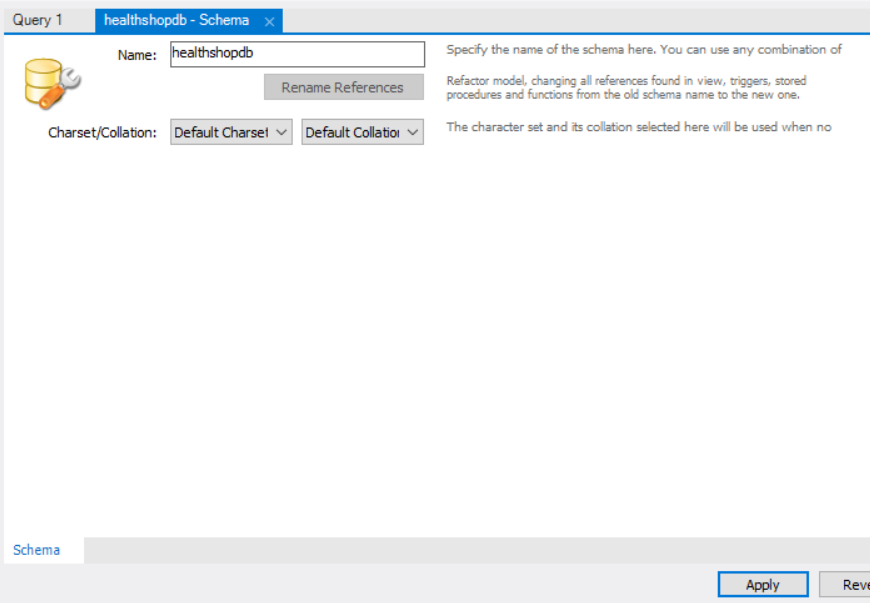
<a th:href=”@{/users}”>Users</a>

22. understanding admin control flow



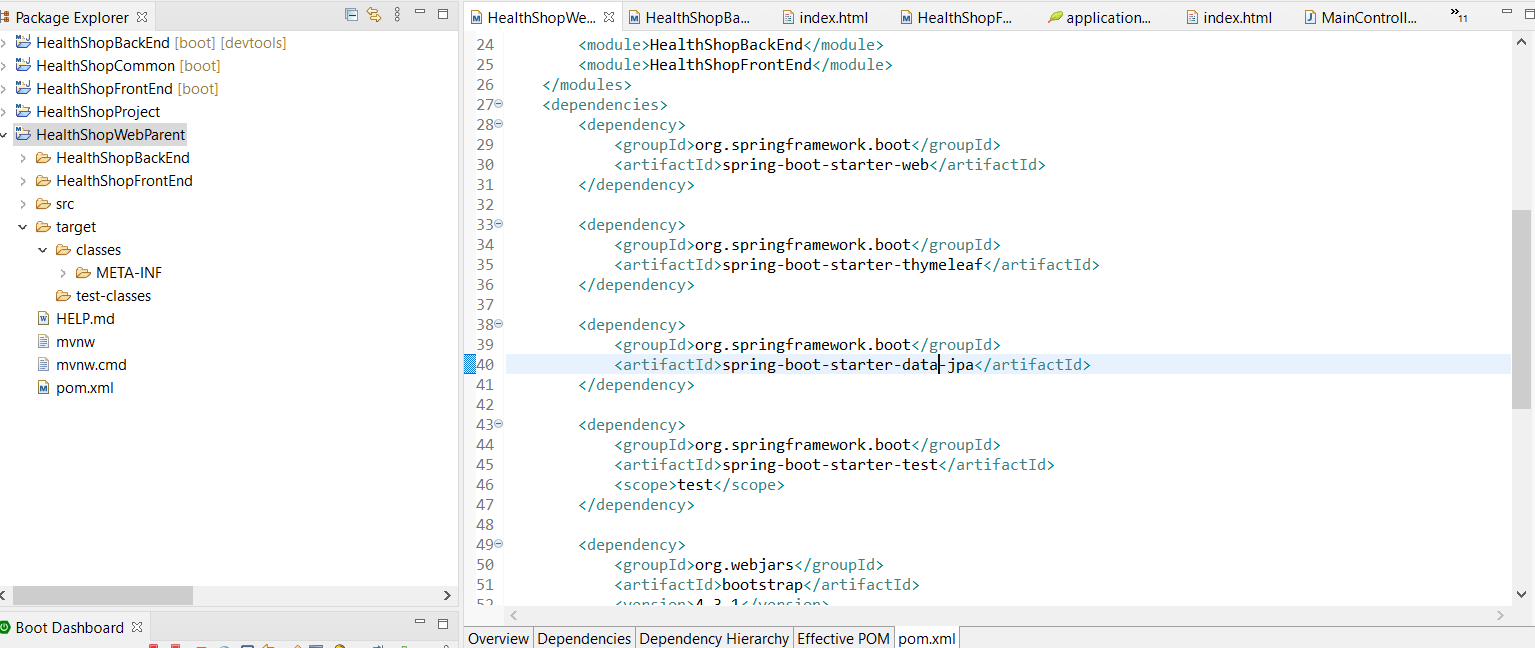


23. Create the database for HealthShop in mysql workbench

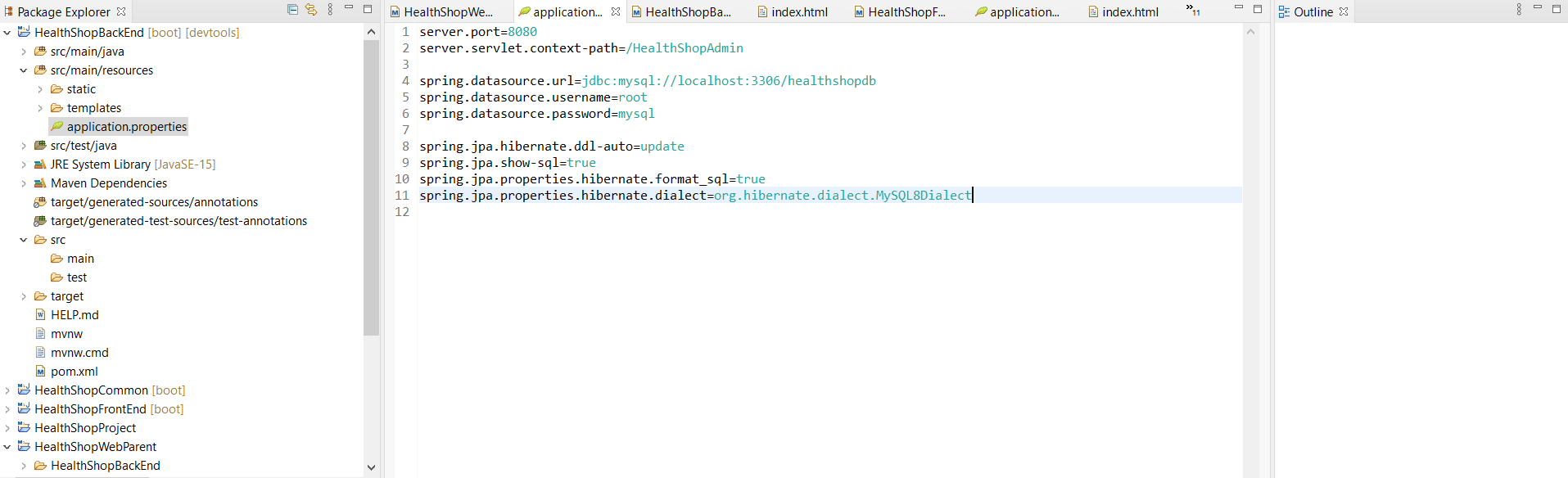


Click apply and finish

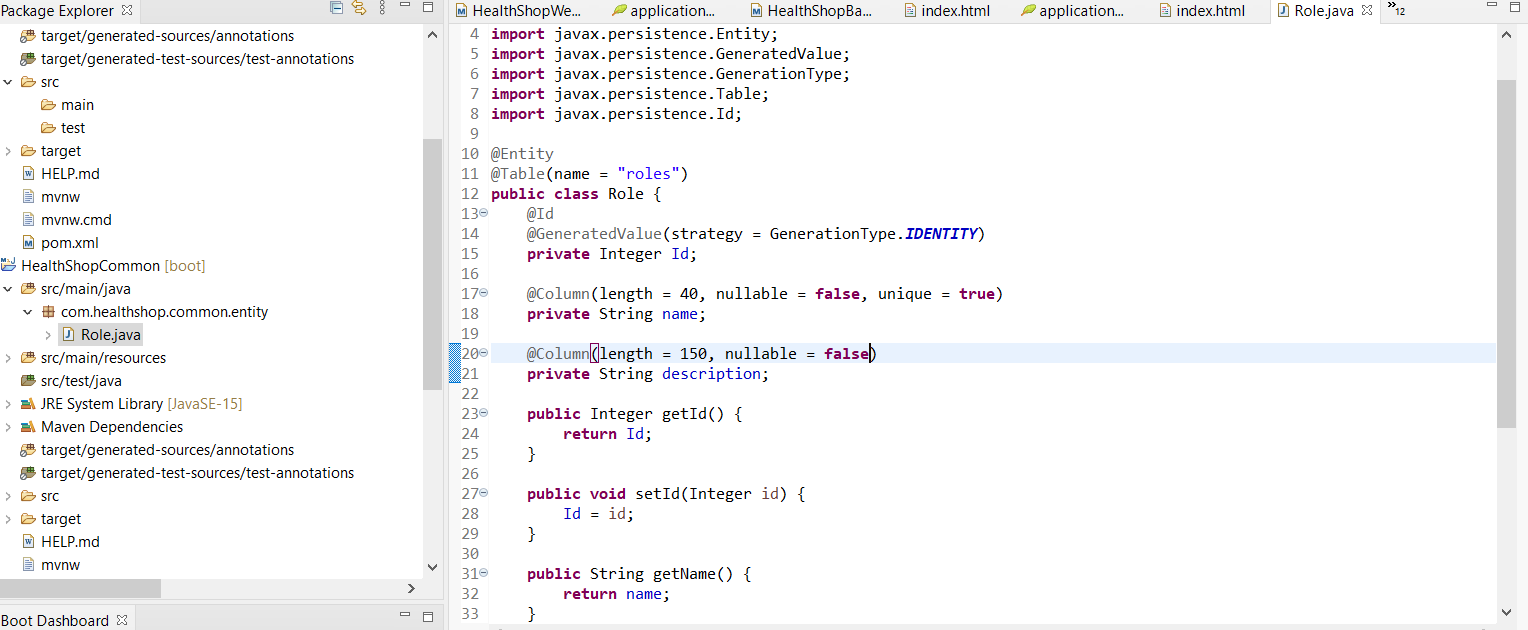
24. add the springboot jpa to HealthShopWebParent



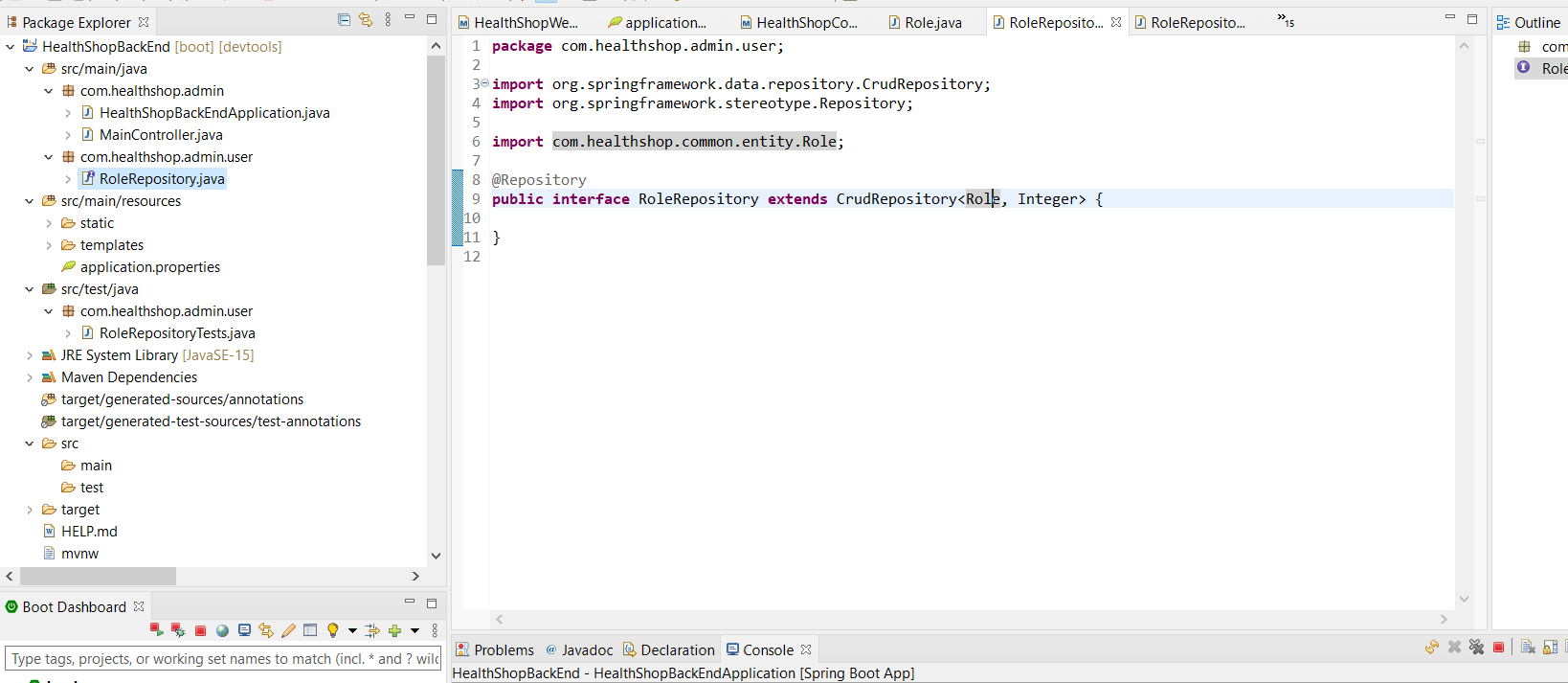
25. update the application.properties file under HealthShopBackEnd



26. create the role entity class under HealthShopCommon

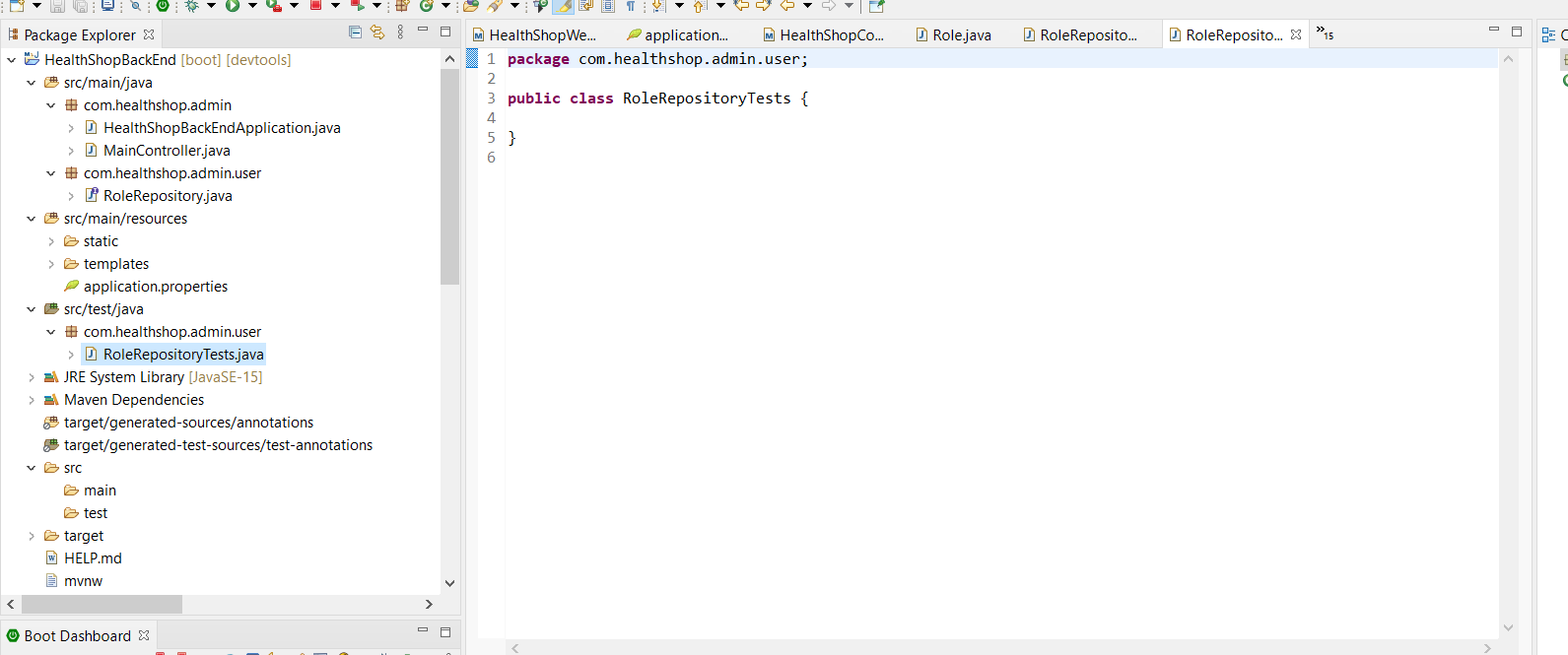


27. Create the interface for roles in HealthShopBackEnd called RoleRepository

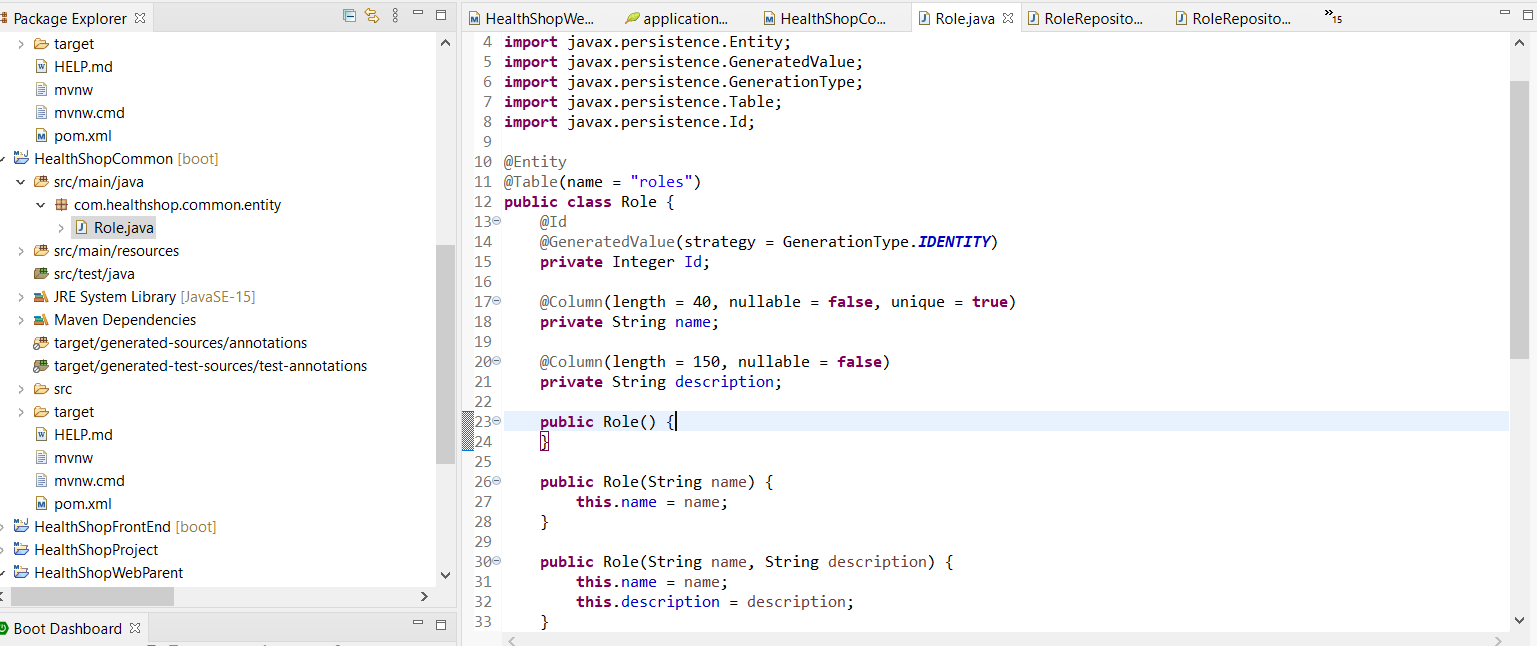


28 Create Unit test for RolesRepository under src/test/java directory

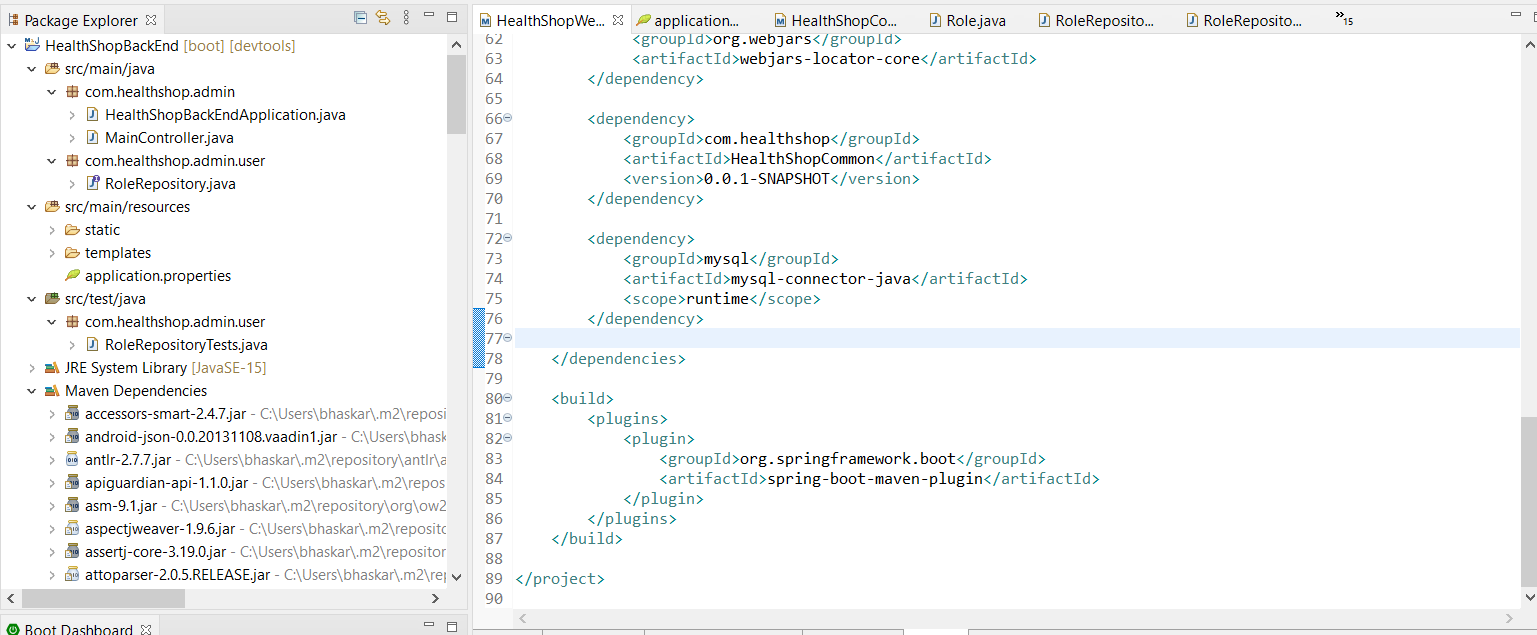
RoleRepositoryTests



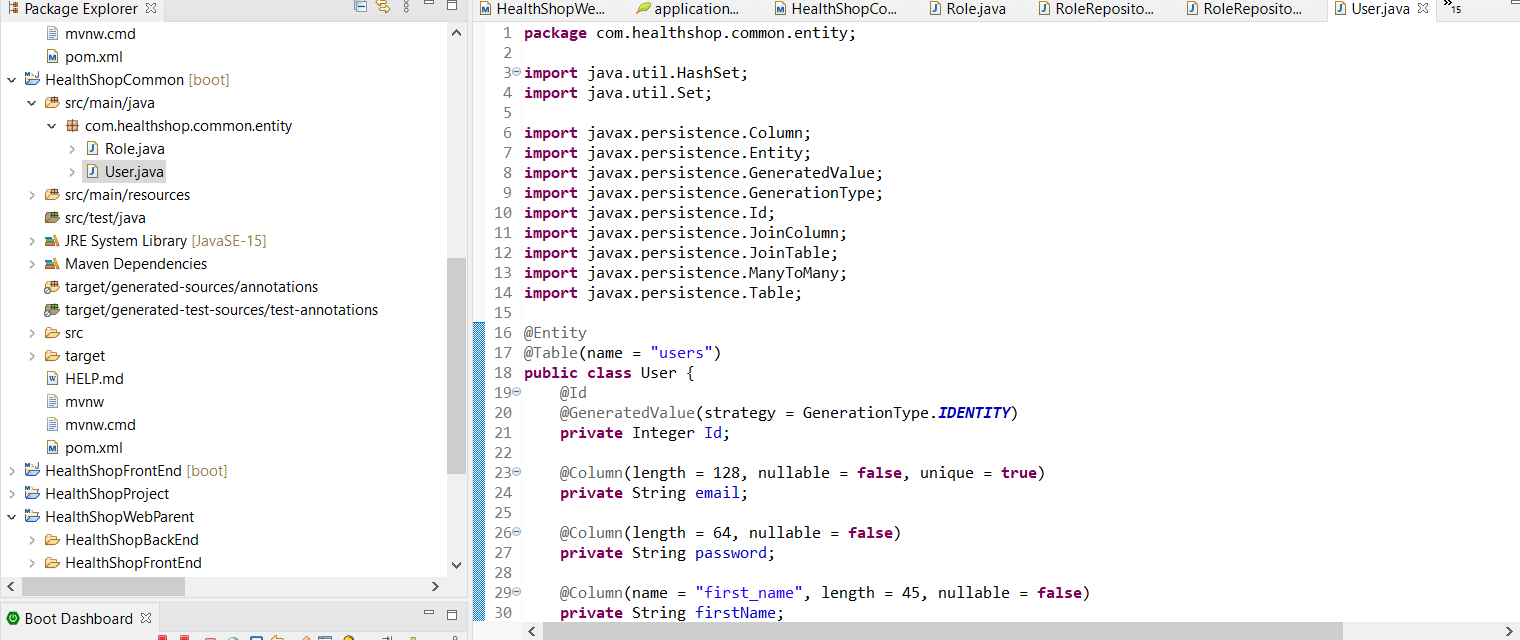
29.Create constructors for roles in Role.java

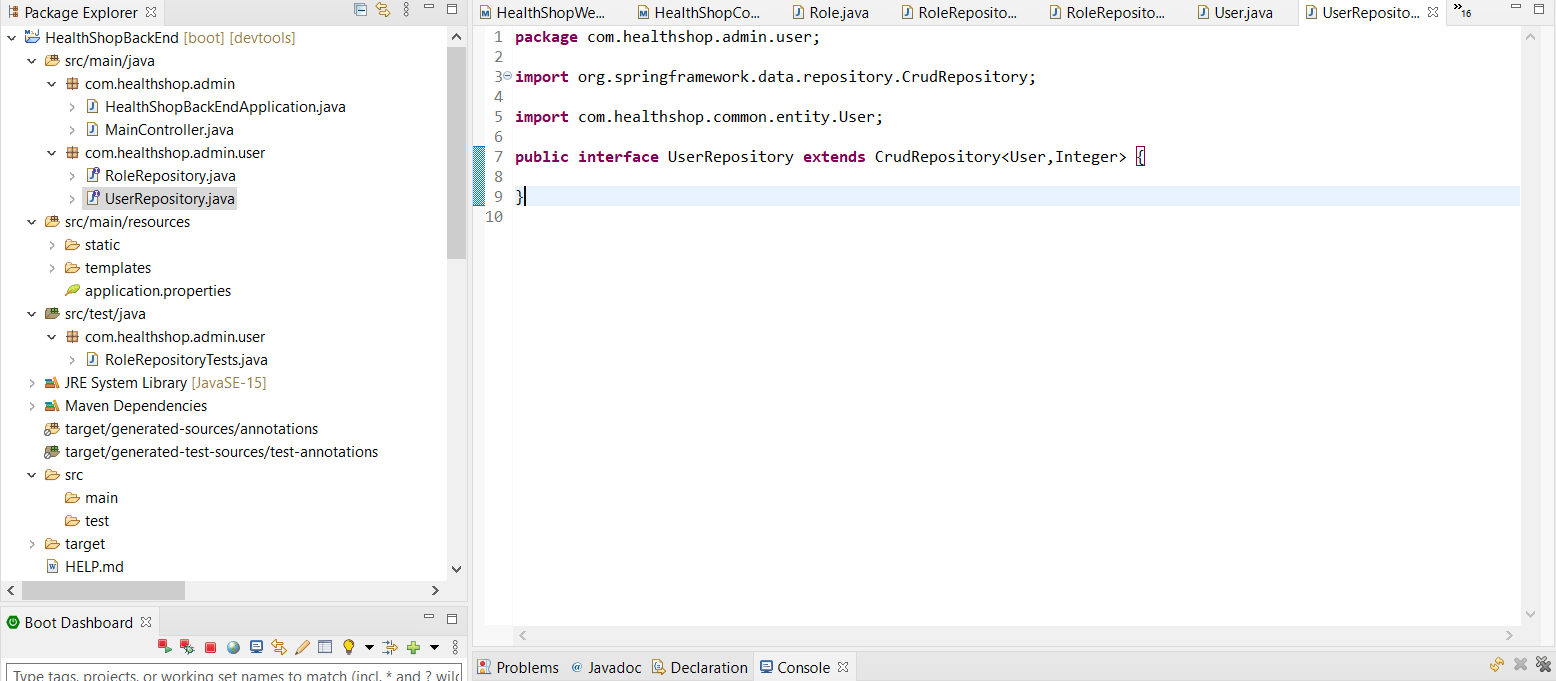


30. Update the Pom.xml to include the mysql jdbc connector

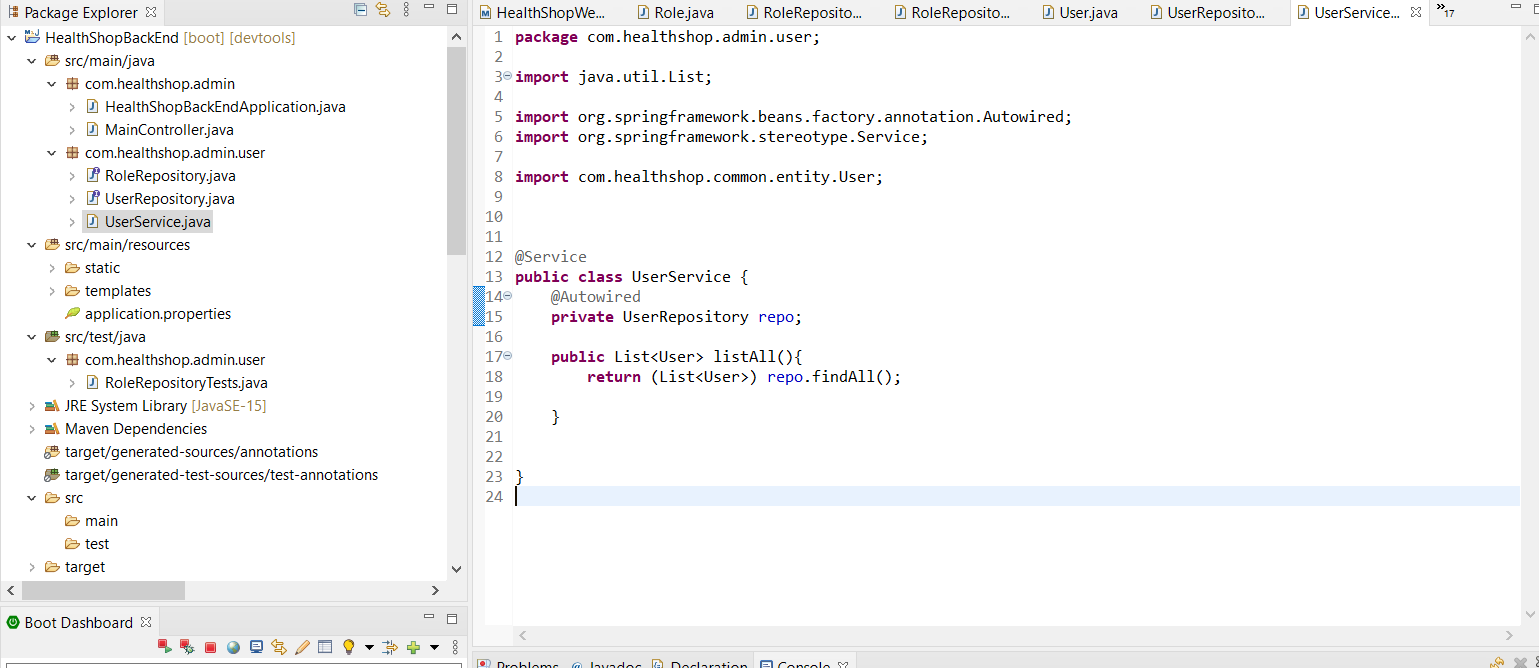


30. Create User.java entity class in HealthShopCommon

31.Create UserRepository in HealthShopBackend



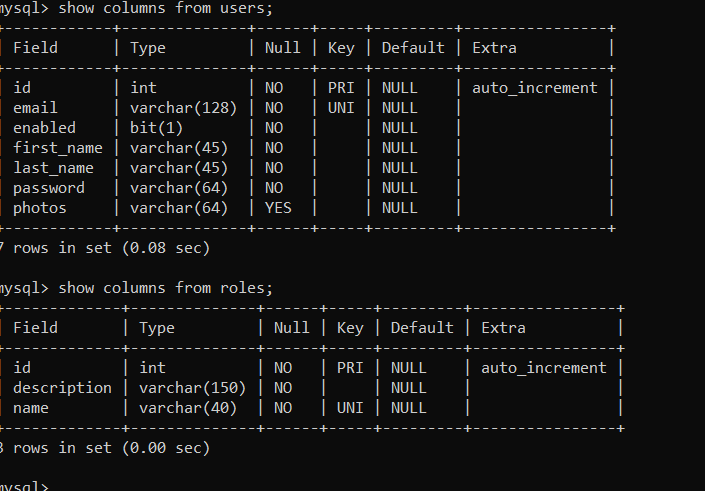
32. Add User Service Class



33. Create User Controller Class

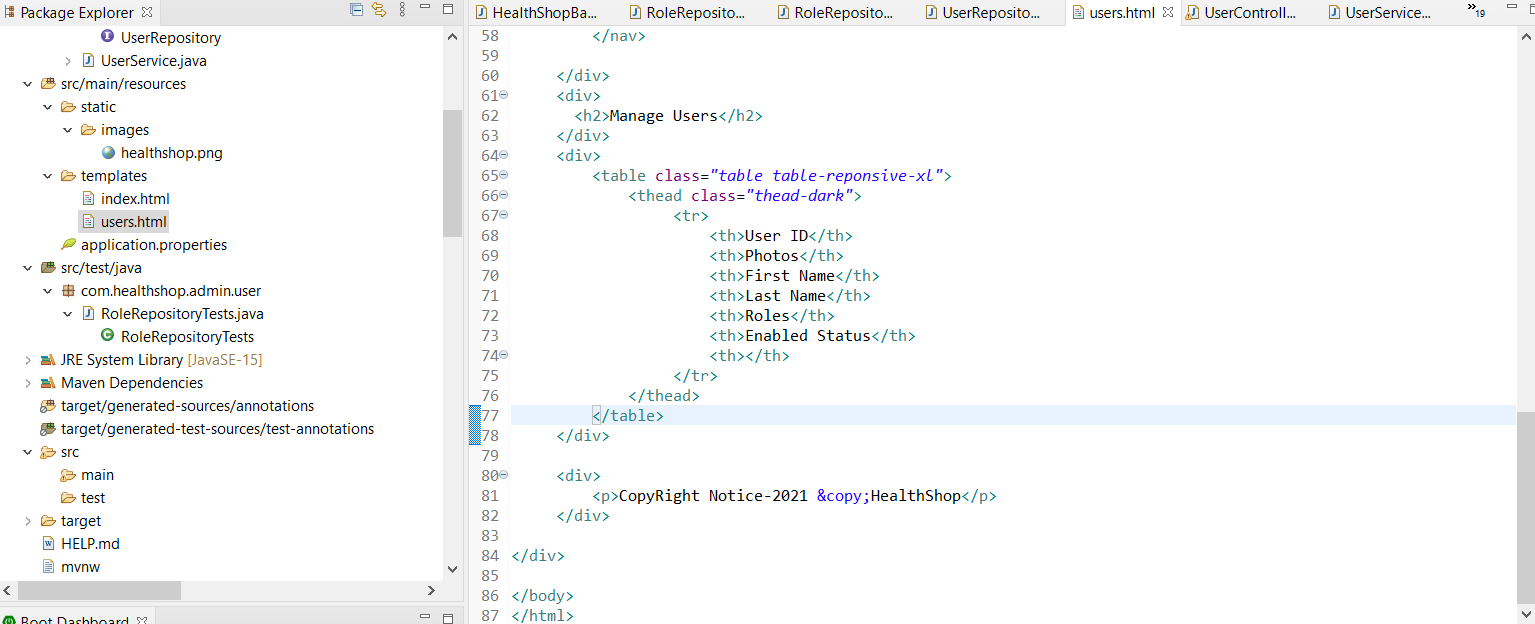


34. Create User and Roles tables

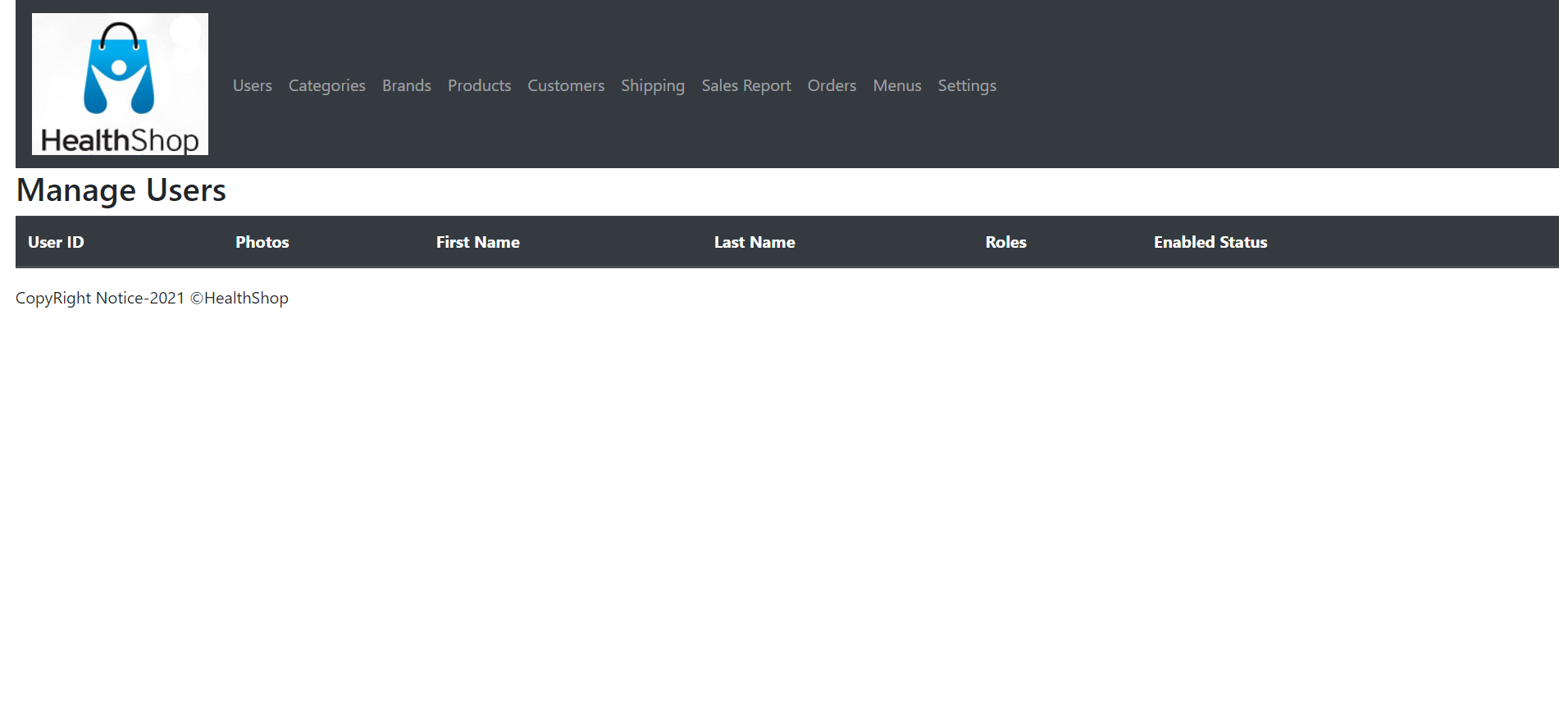


35

Code the Users.html file to display user details



36. Users Page after changes in users.html



37. Create the mysql statements to create users and roles

insert into roles

set id = 1,

description = 'manager healthshop website',

name = 'bhaskar';

insert into roles

set id = 2,

description = 'admin healthshop website',

name = 'Ramesh';

insert into roles

set id = 3,

description = 'Shipper healthshop website',

name = 'Skyrocket';

insert into users

set id = 1,

email = 'bhaskar2285@gmail.com',

enabled = true,

first\_name = 'bhaskar',

last\_name = 'singh',

password = 'secret',

photos = null;

insert into users

set id = 2,

email = 'rameshadmin@gmail.com',

enabled = true,

first\_name = 'ramesh',

last\_name = 'kapoor',

password = 'secret2',

photos = null;

insert into users

set id = 3,

email = 'skyrocket@gmail.com',

enabled = true,

first\_name = 'skyrocket',

last\_name = 'shipper',

password = 'secret3',

photos = null;

select \* from users\_roles;

select \* from roles;

select \* from users;

insert into users\_roles

set user\_id = 1,

role\_id = 1;

insert into users\_roles

set user\_id = 2,

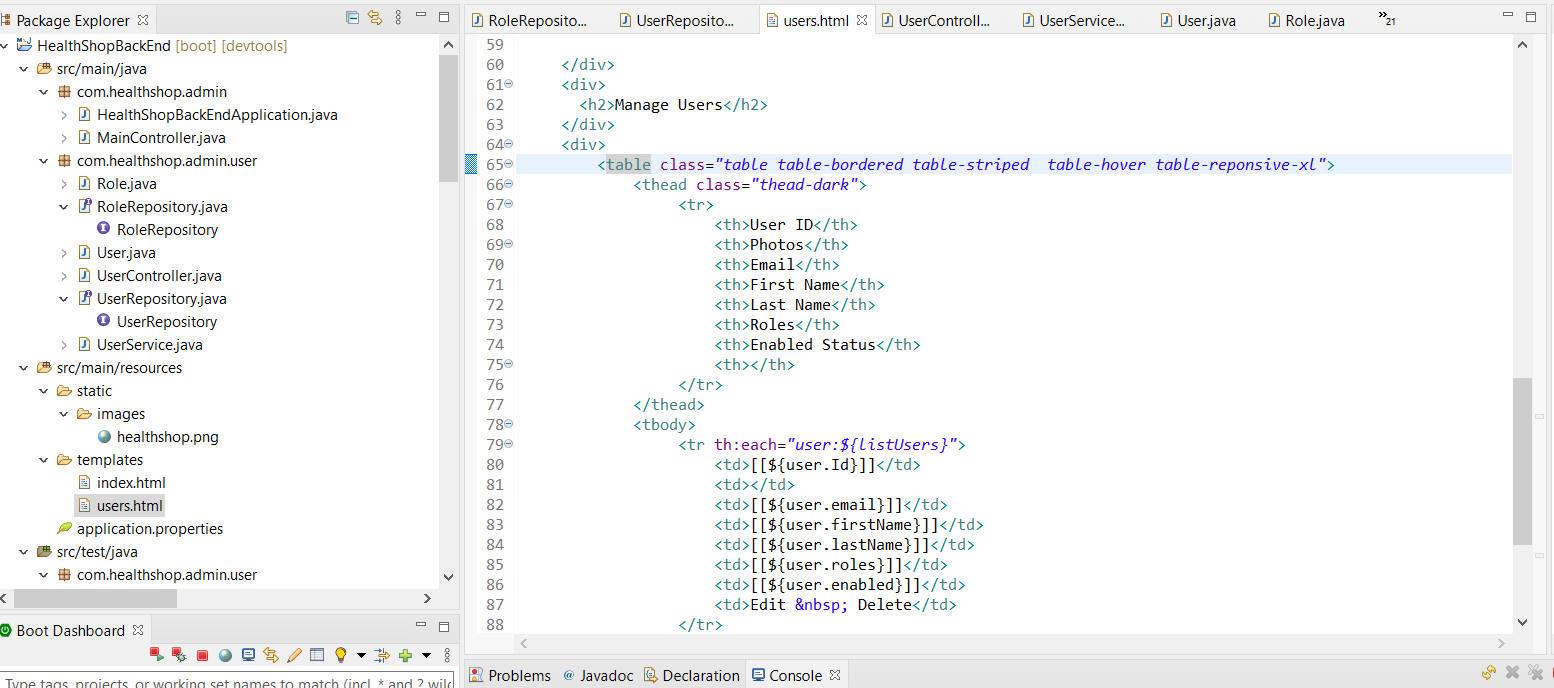
role\_id = 2;

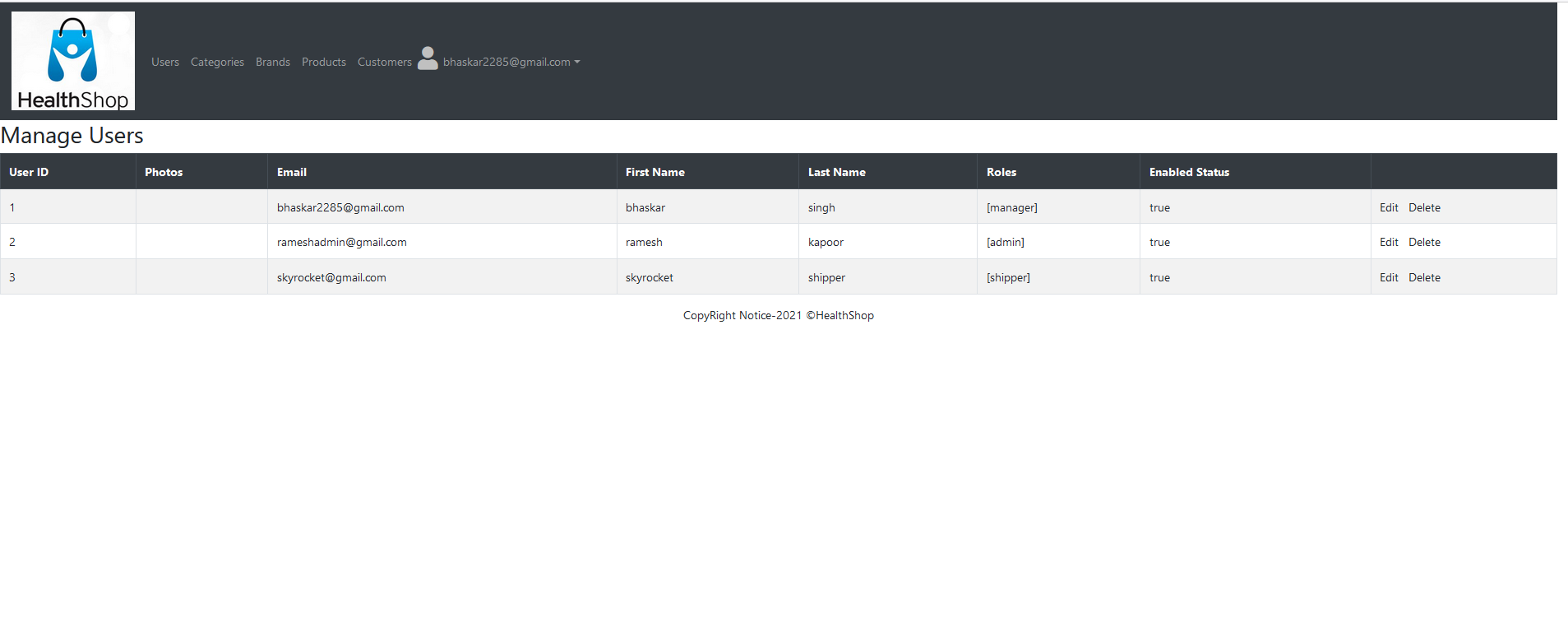
insert into users\_roles

set user\_id = 3,

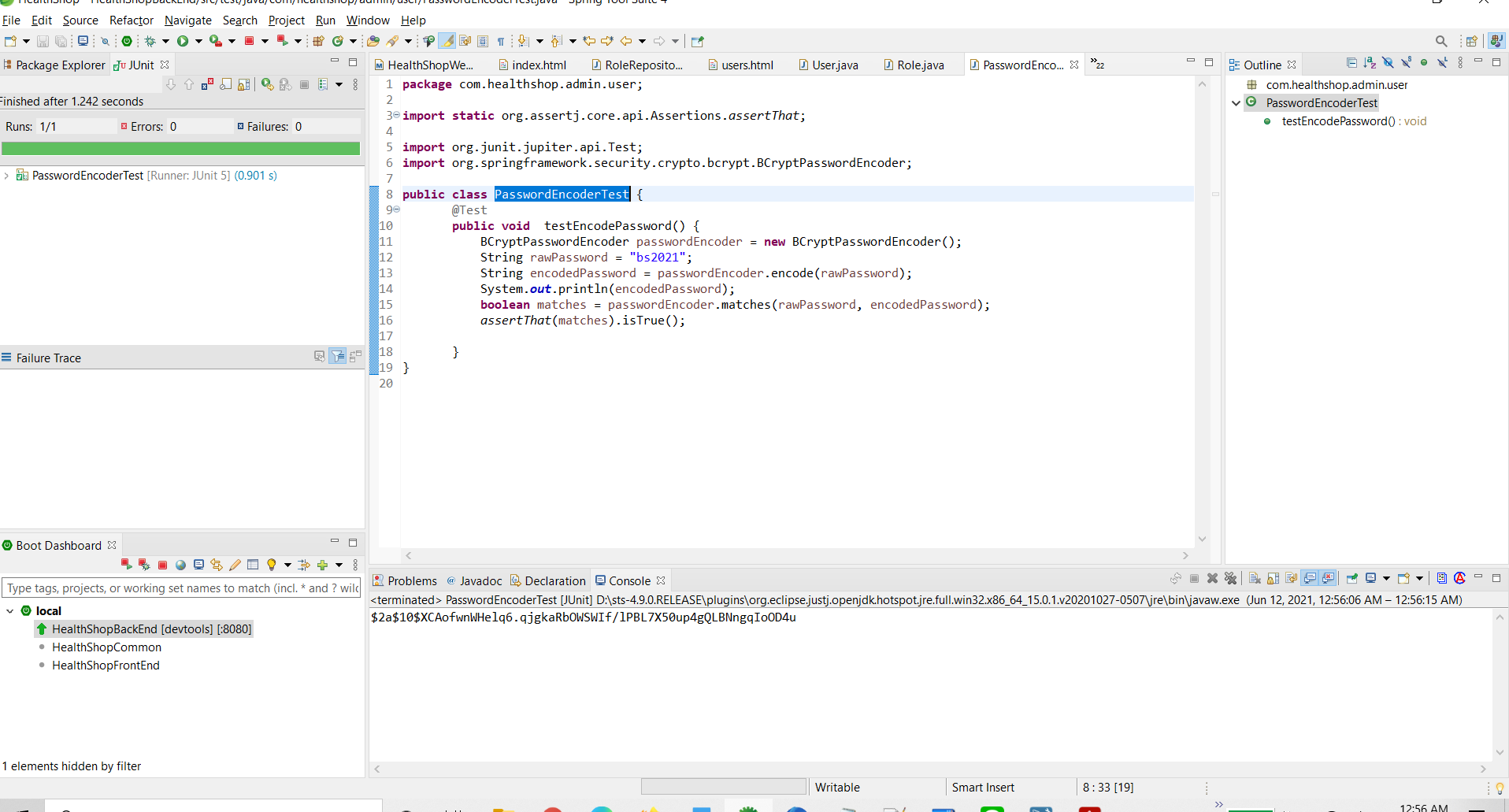
role\_id = 3;

38 Further edit the users.html to display the users detail in much better way

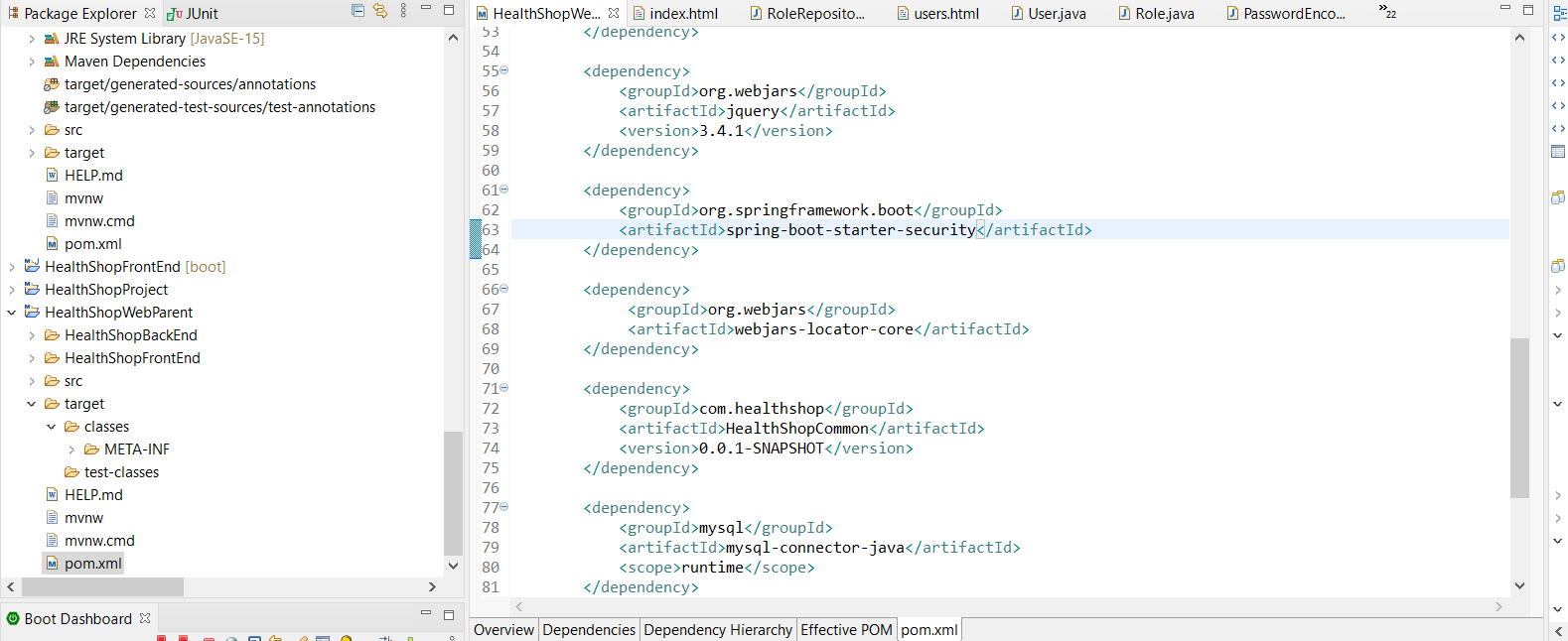


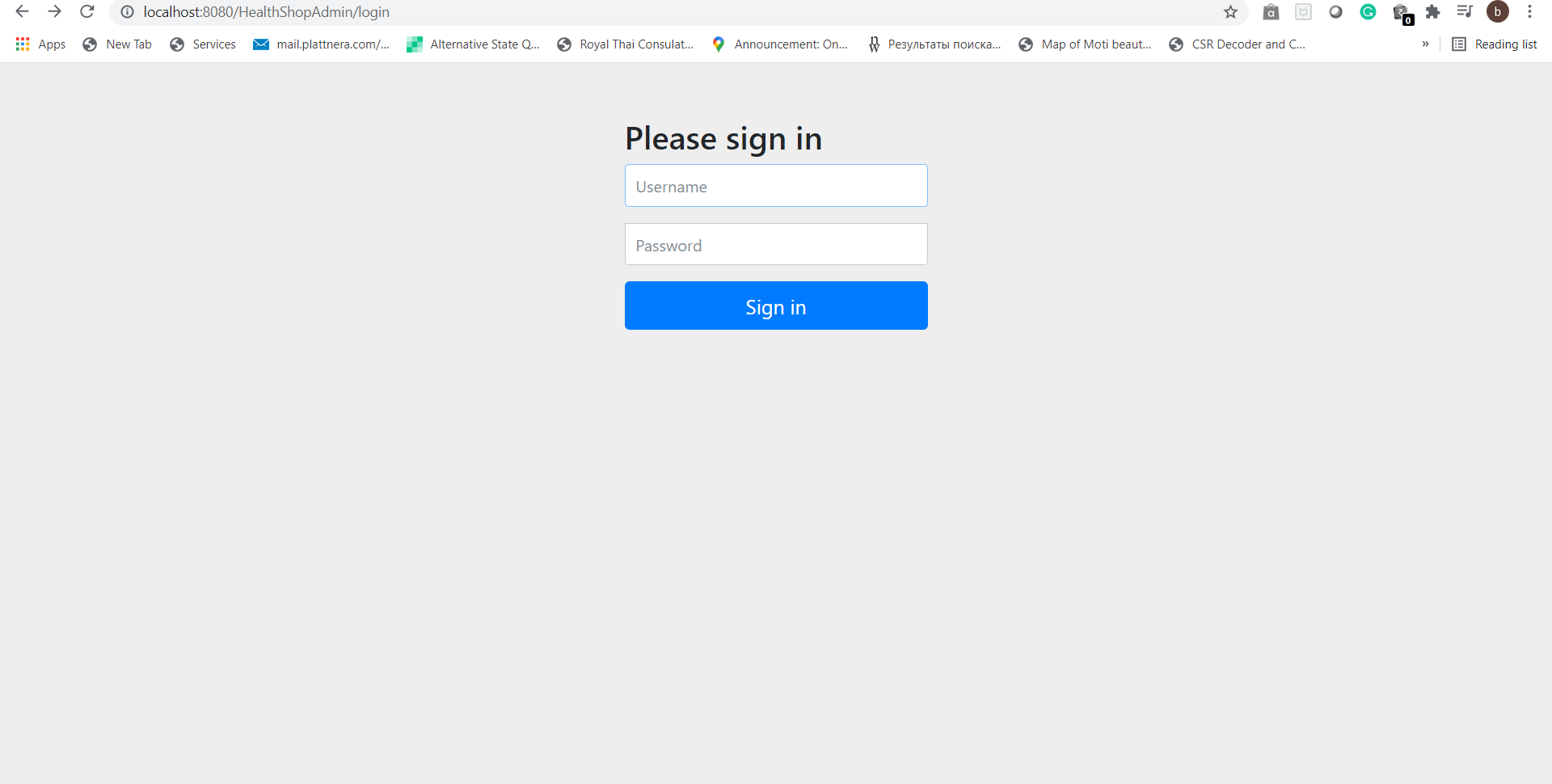


39. Run Password Encoder unit test to encode the user password in database

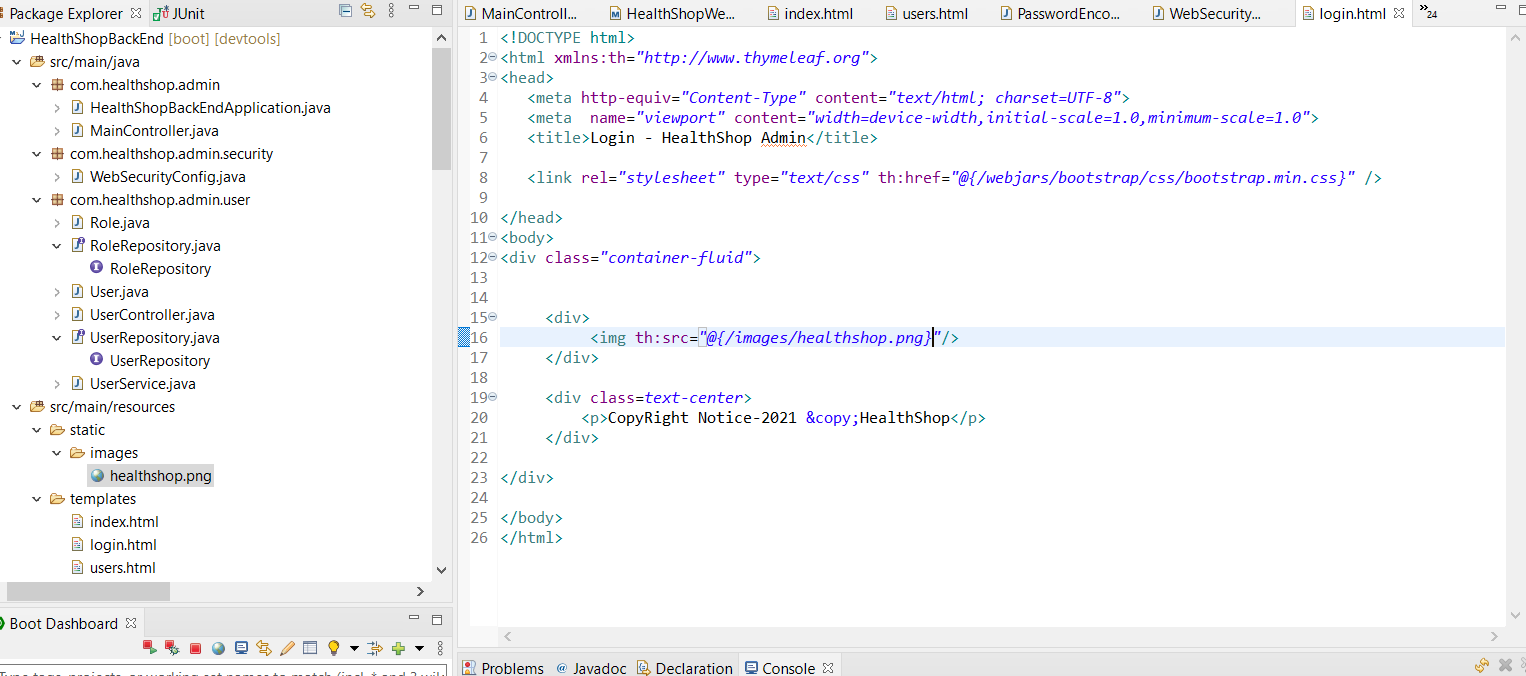


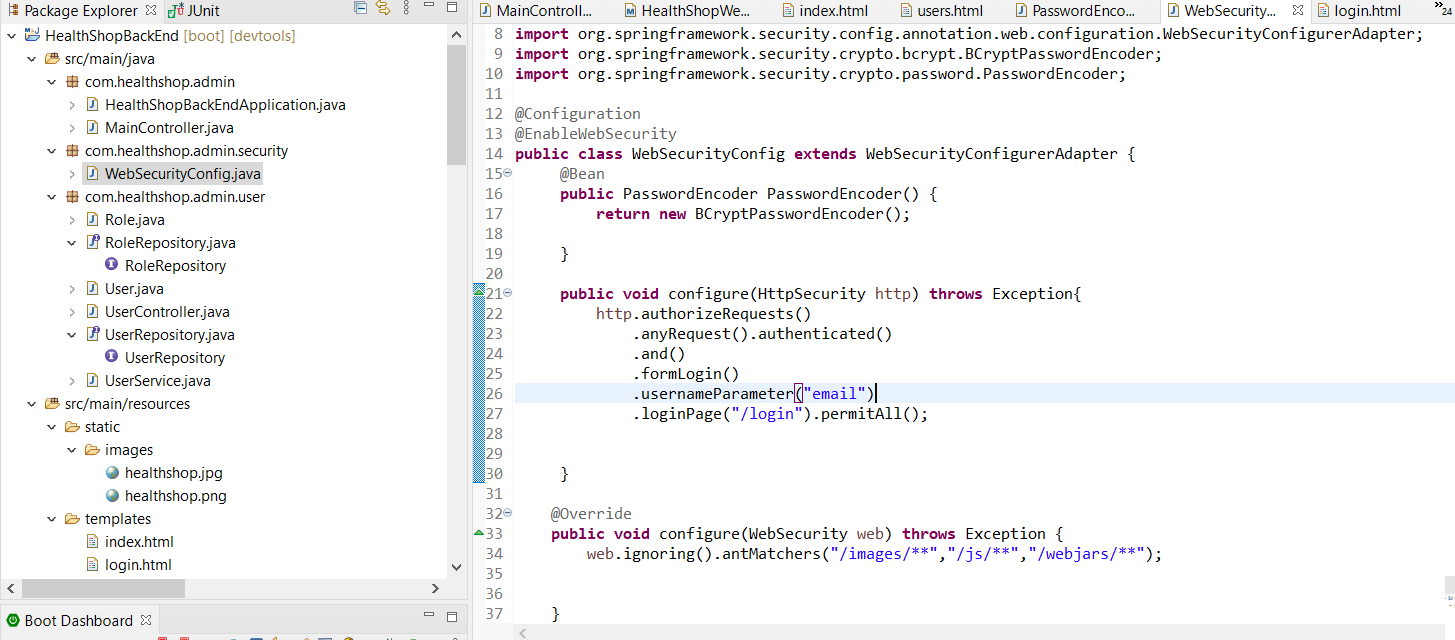
40. Implement spring security

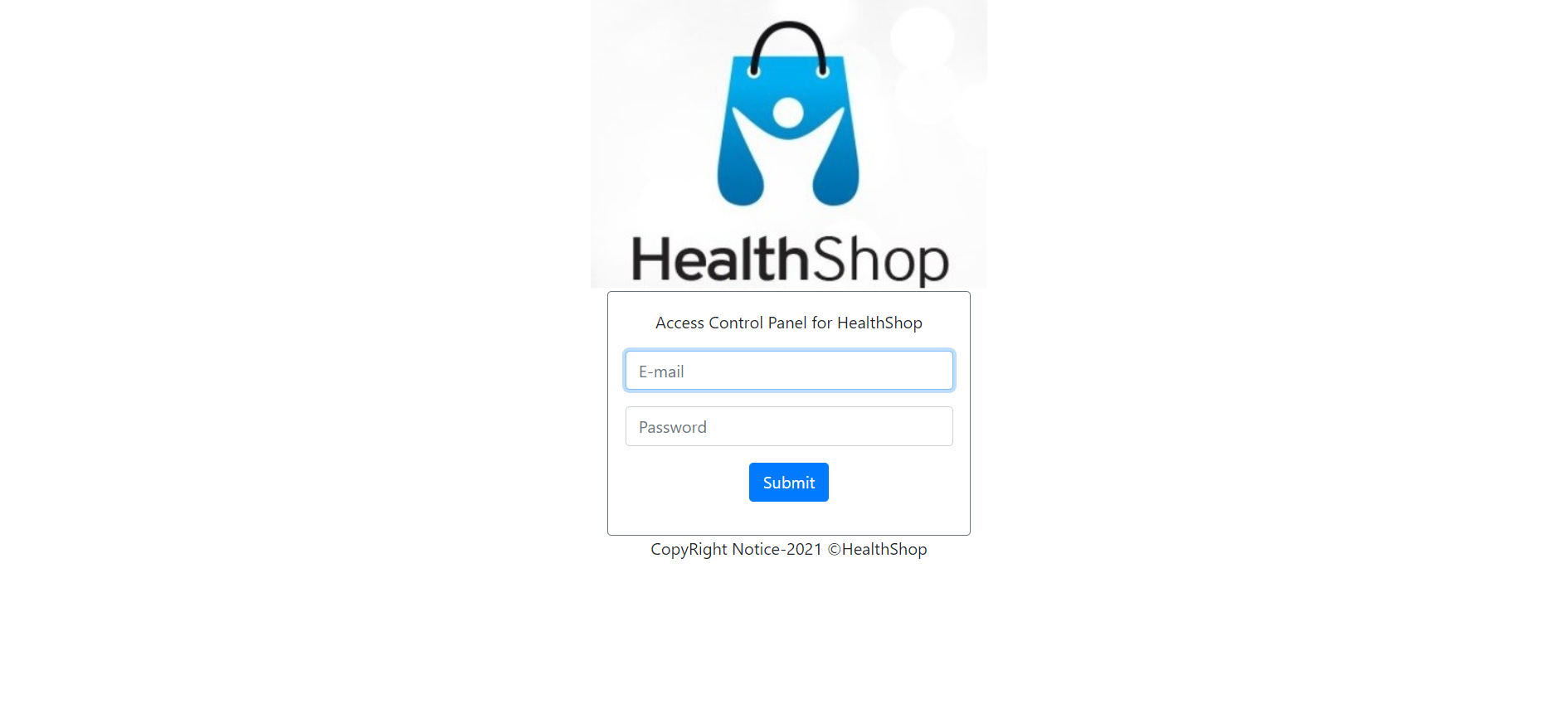


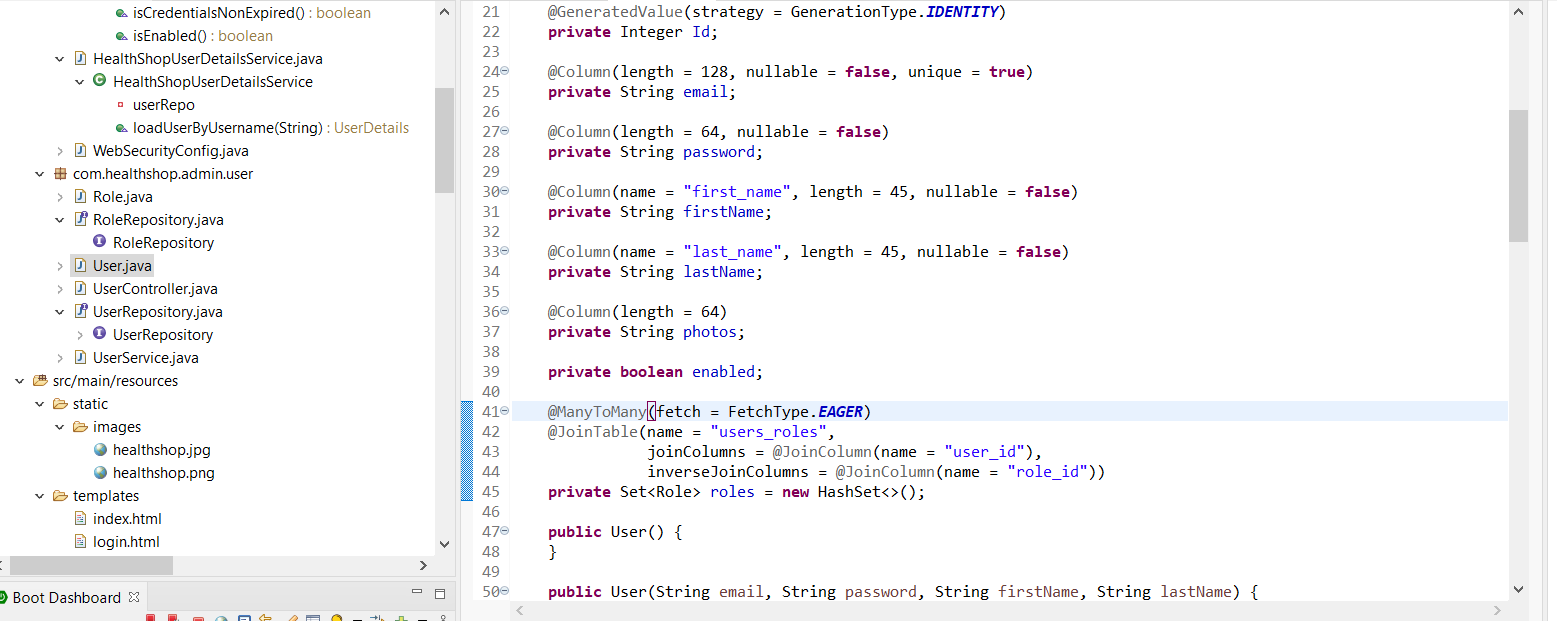


41. Customize the login page





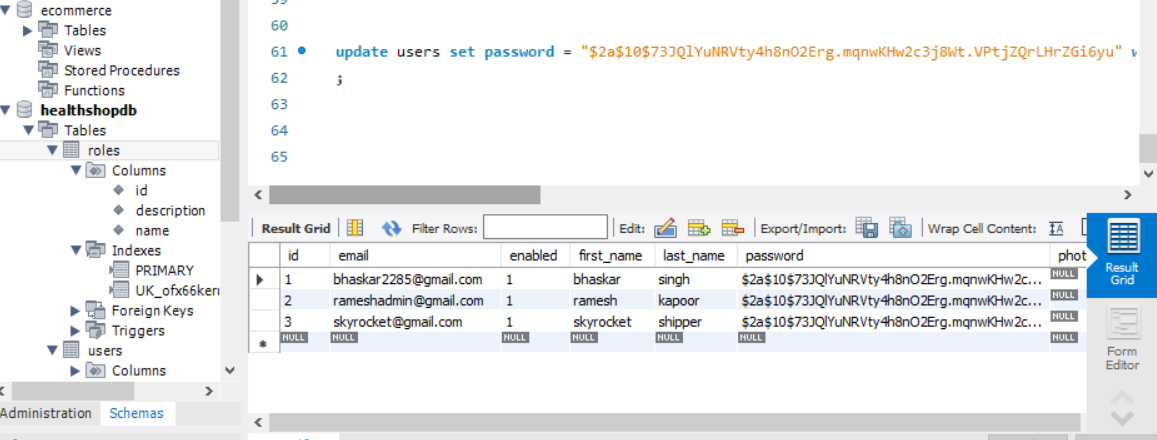




Code the Login Page to resolve the error

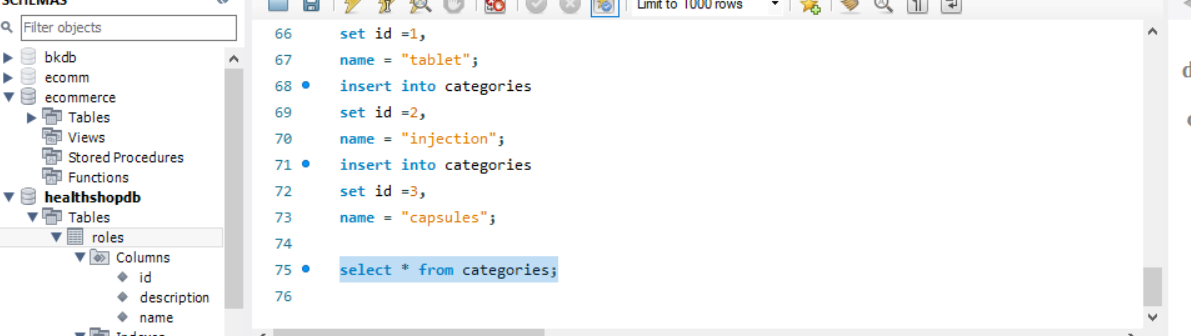


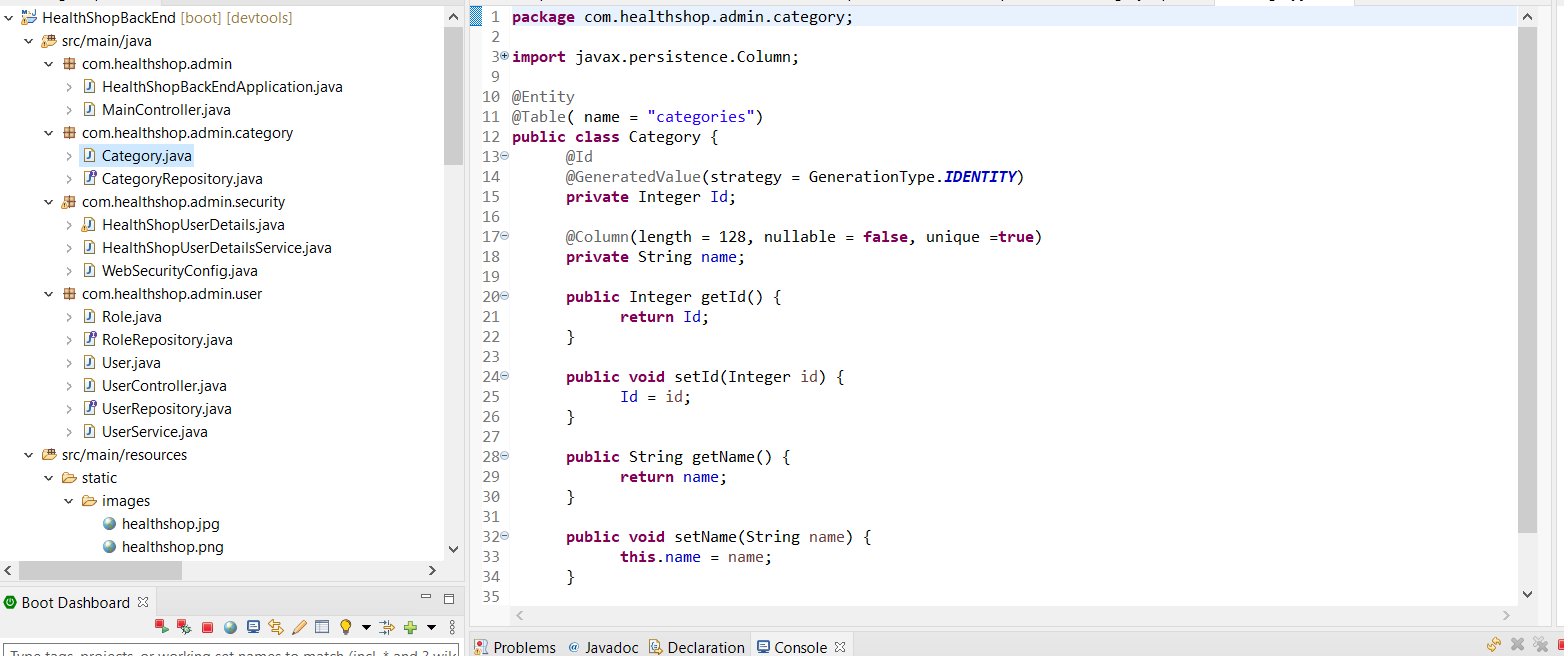
Update the users table with encoded password

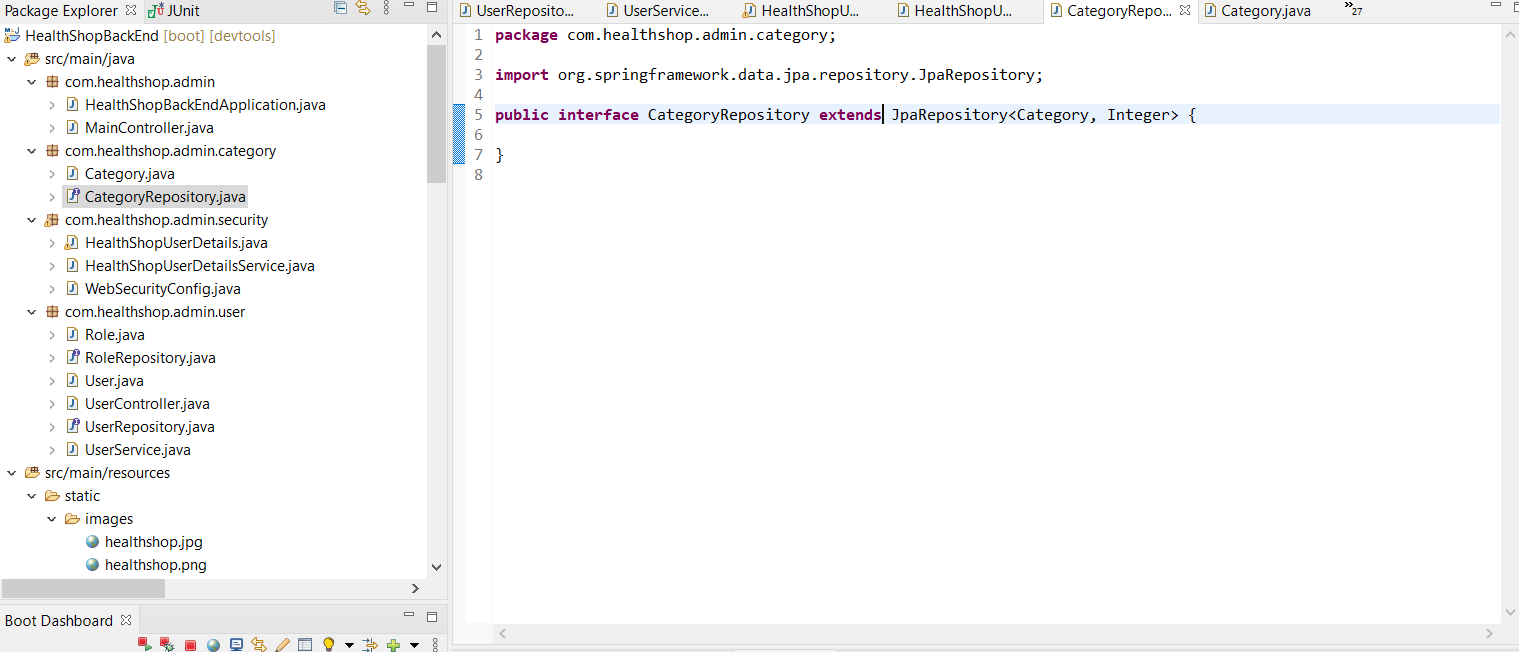


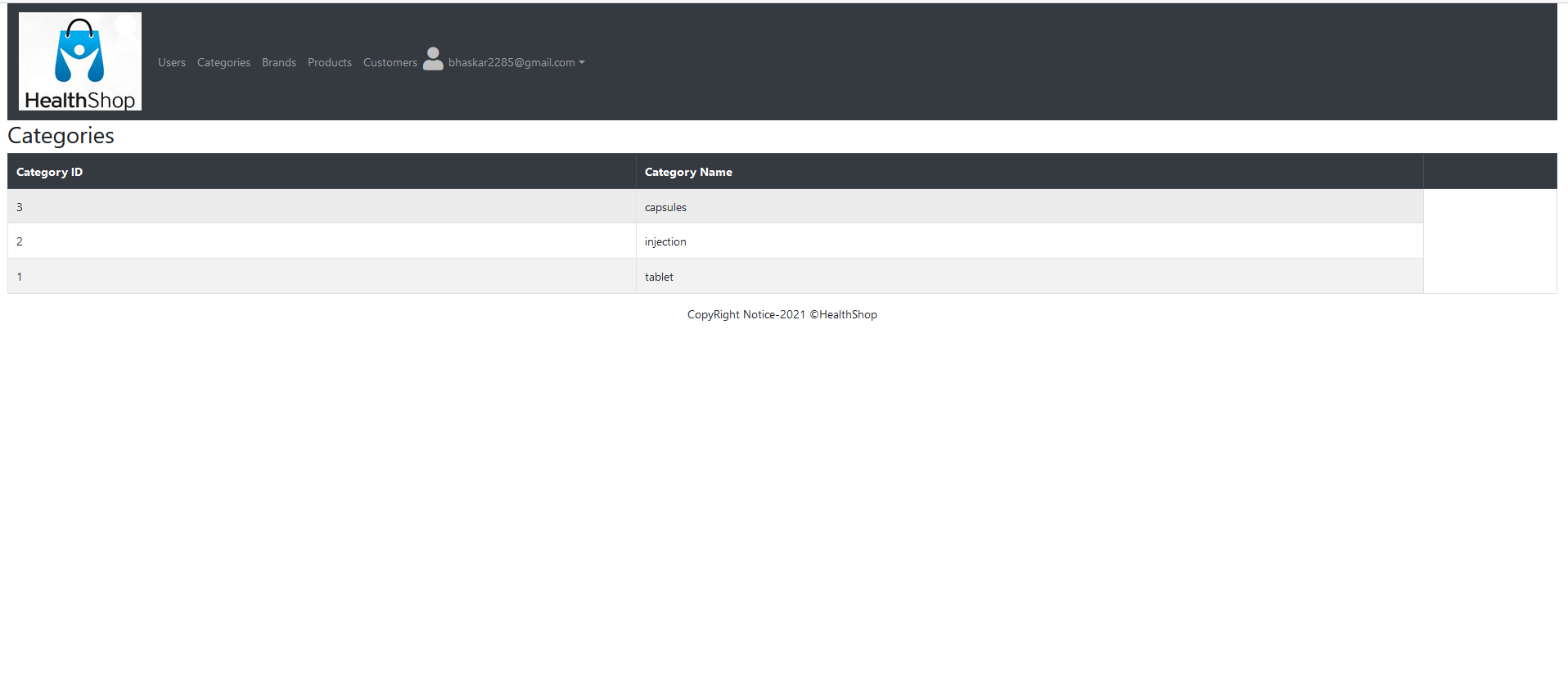
Use PasswordEncoder.java file to generate the password in encoded form

42. Code the Categories for medicines

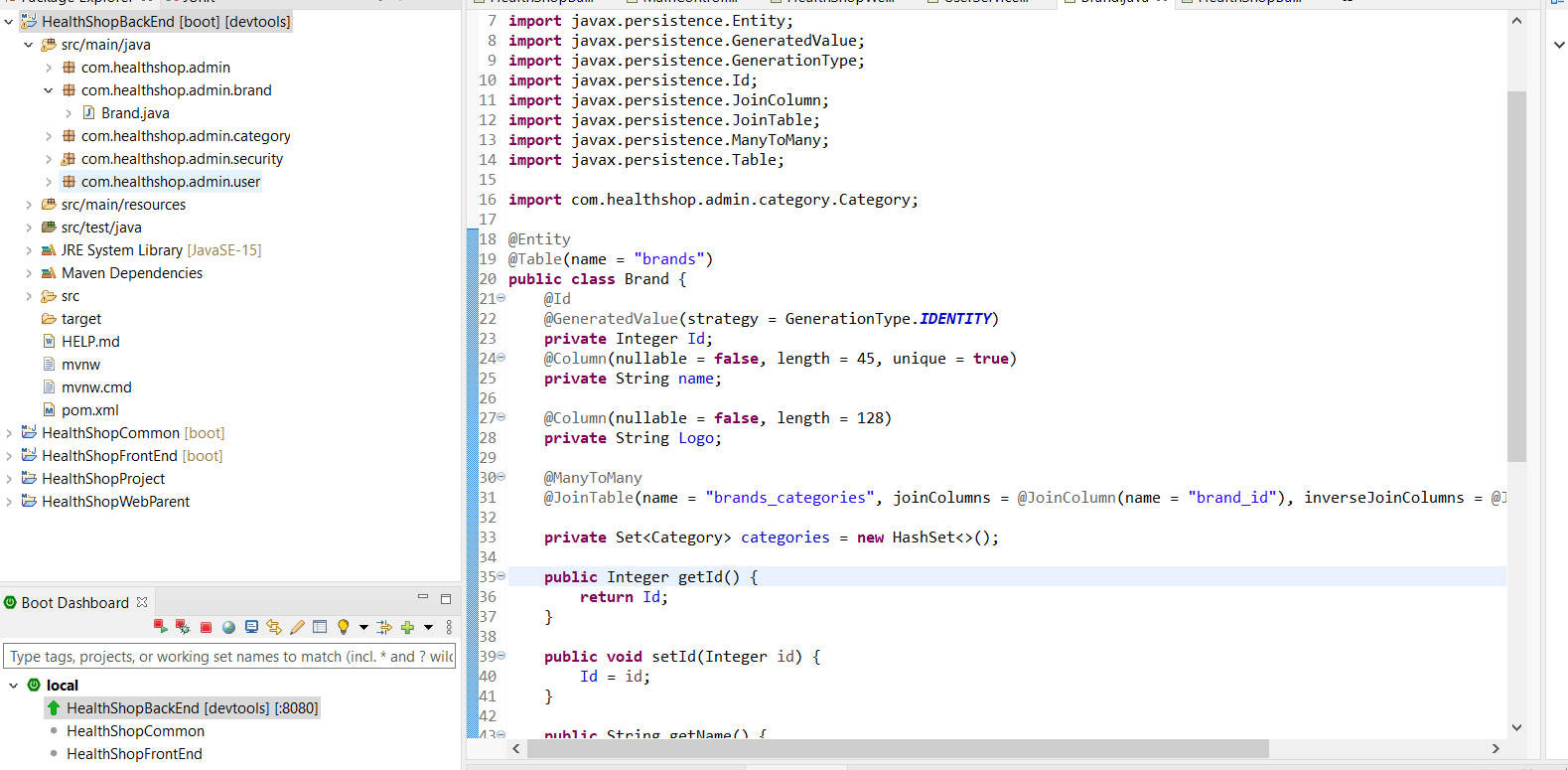








43. Code Brand Tab



My sql statements

select \* from categories;

select \* from brands;

insert into brands

set id =1,

name = "Cipla",

logo = "cipla.png"

;

insert into brands

set id =2,

name = "Glaxo",

logo = "glaxo.png";

insert into brands

set id =3,

name = "Pfizer",

logo = "pfizer.png";

select \* from brands;

select \* from brands\_categories;

insert into brands\_categories

set brand\_id =1,

category\_id =1;

insert into brands\_categories

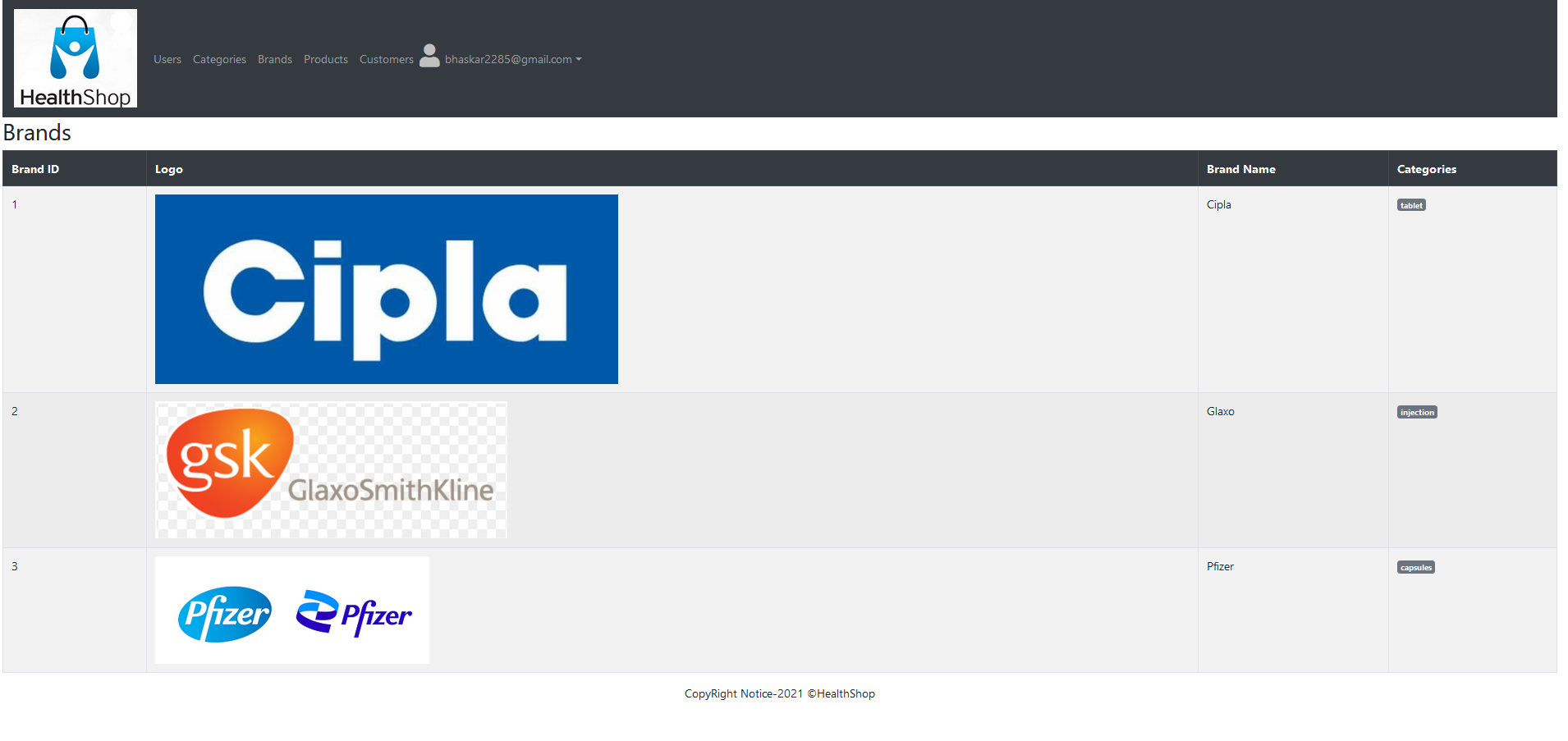
set brand\_id =2,

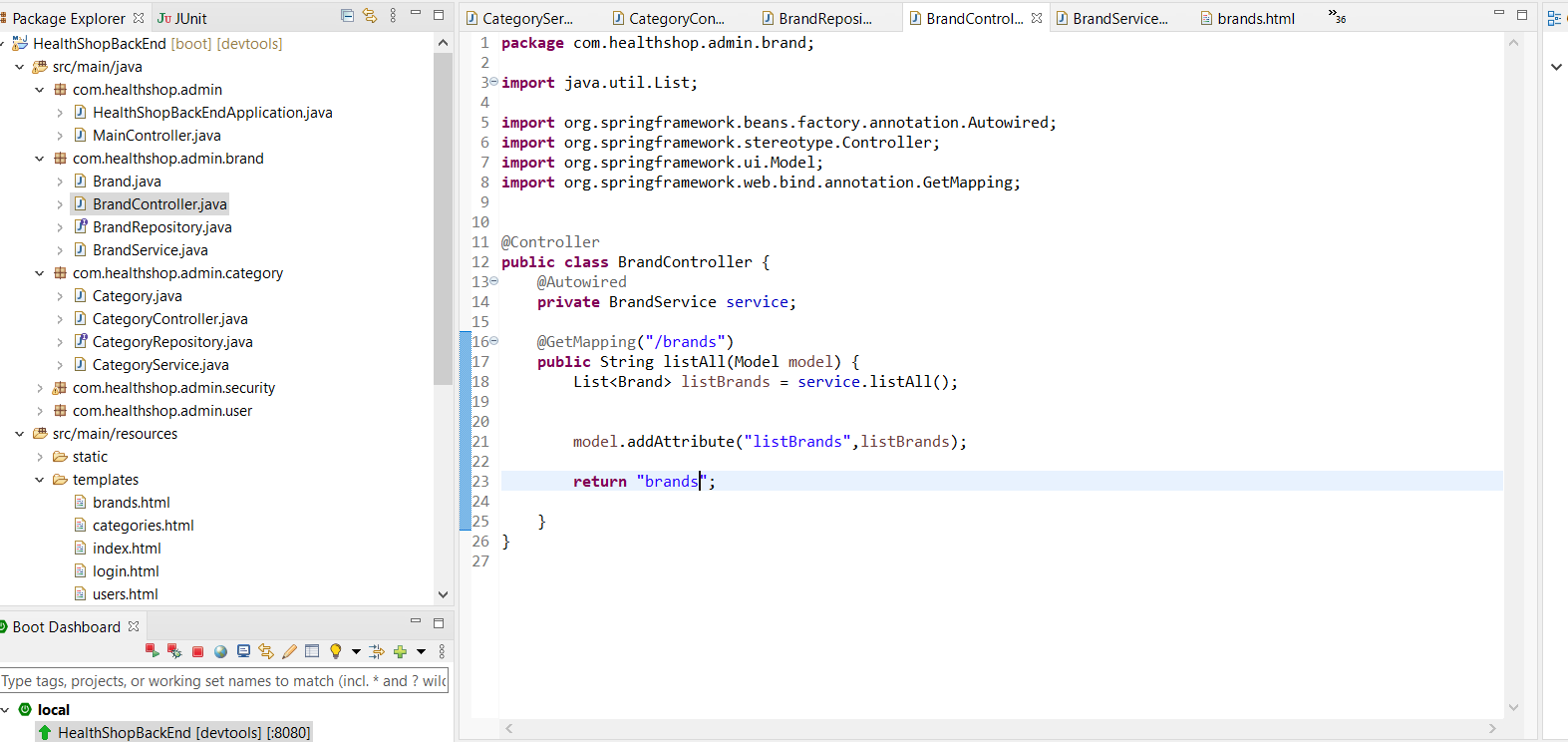
category\_id =2;

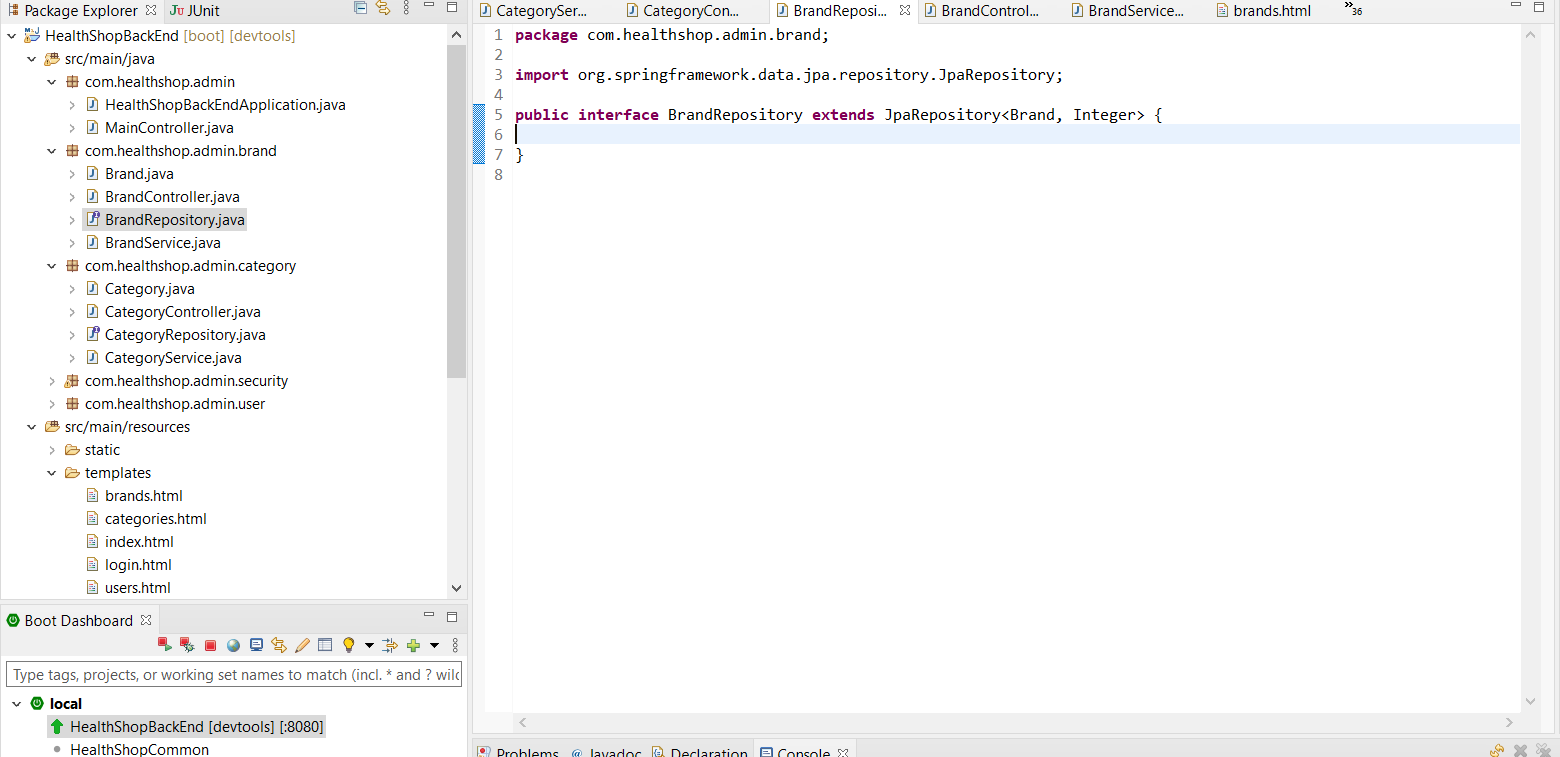
insert into brands\_categories

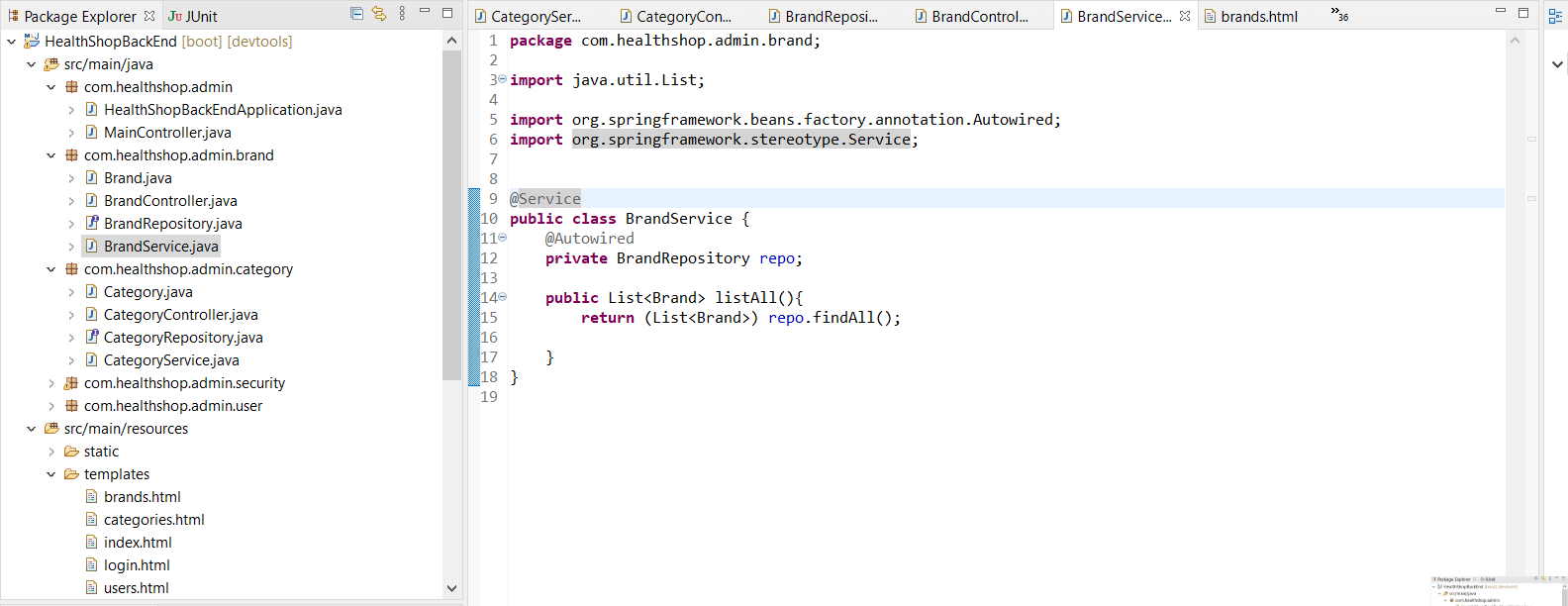
set brand\_id =3,

category\_id =3;

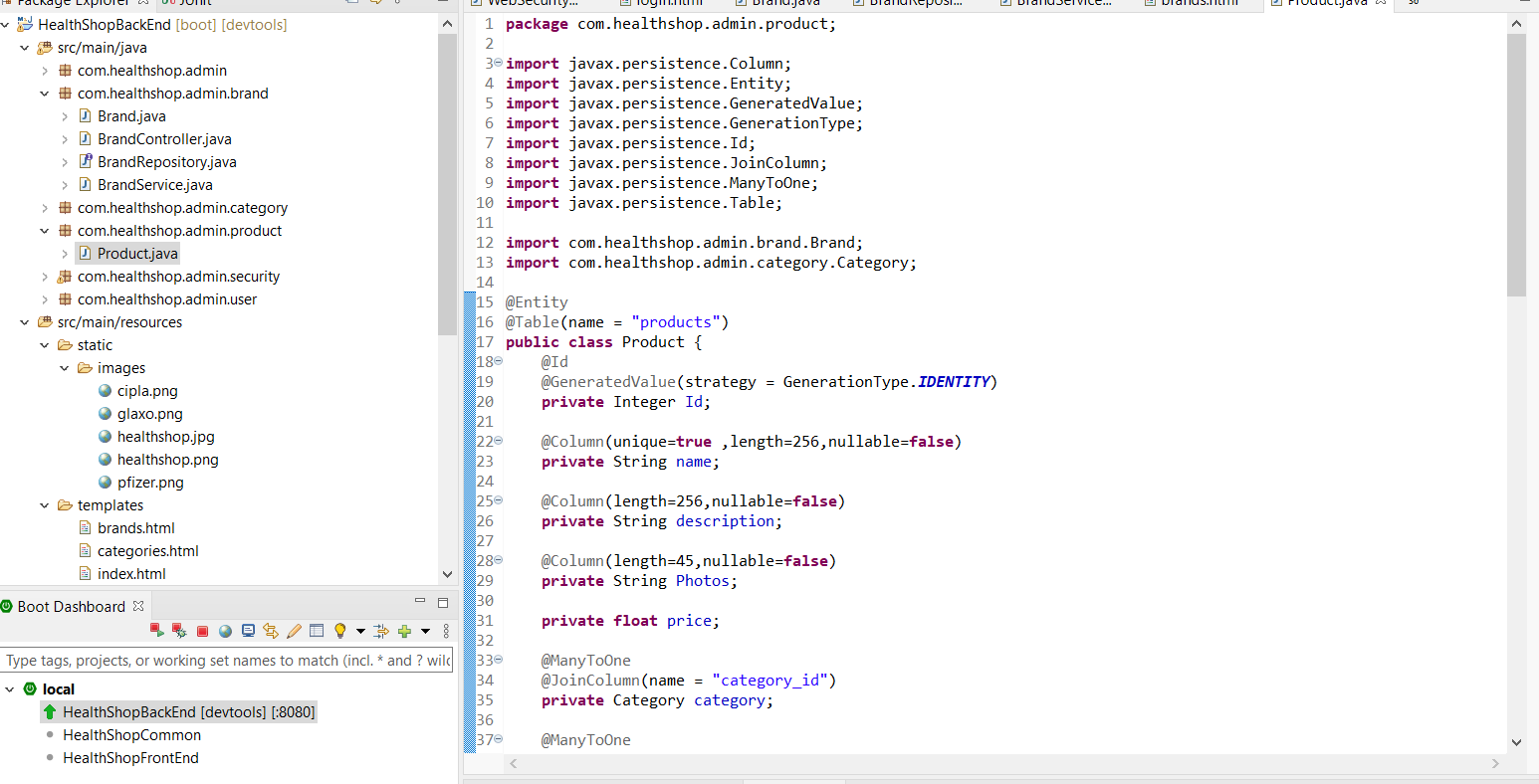








44. Code the Products tab



insert into products

set id = 1,

name = 'Cipla',

alias = 'metacin',

description = 'paracetamol',

Photos = 'metacin.png',

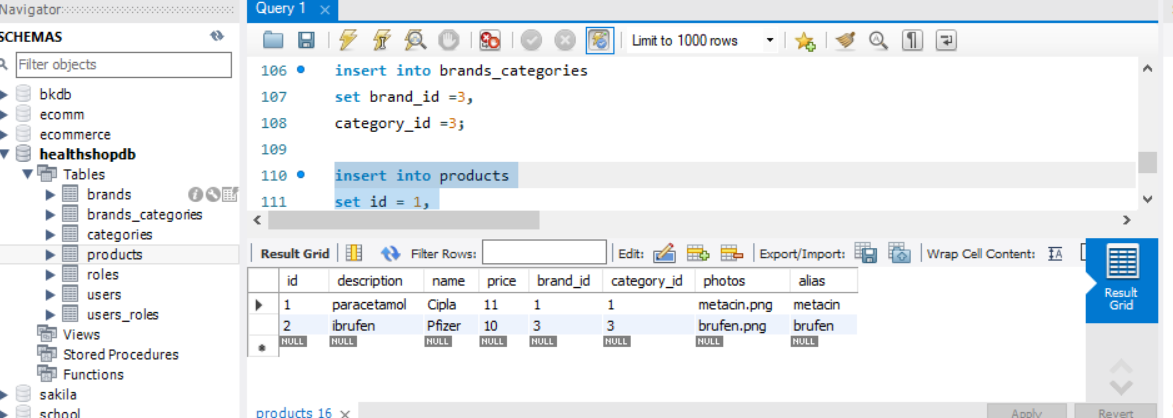
price = 11.00,

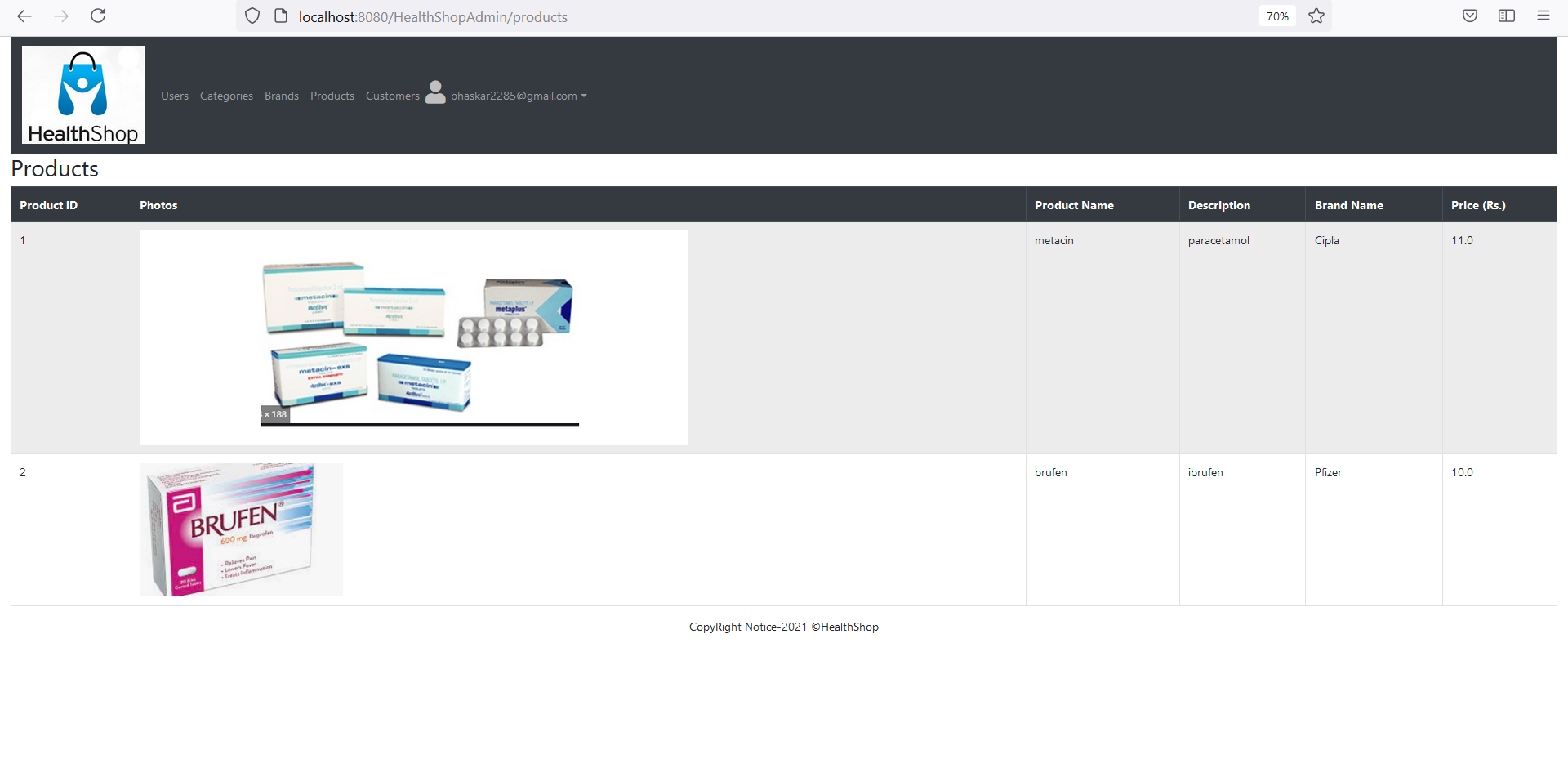
brand\_id =1,

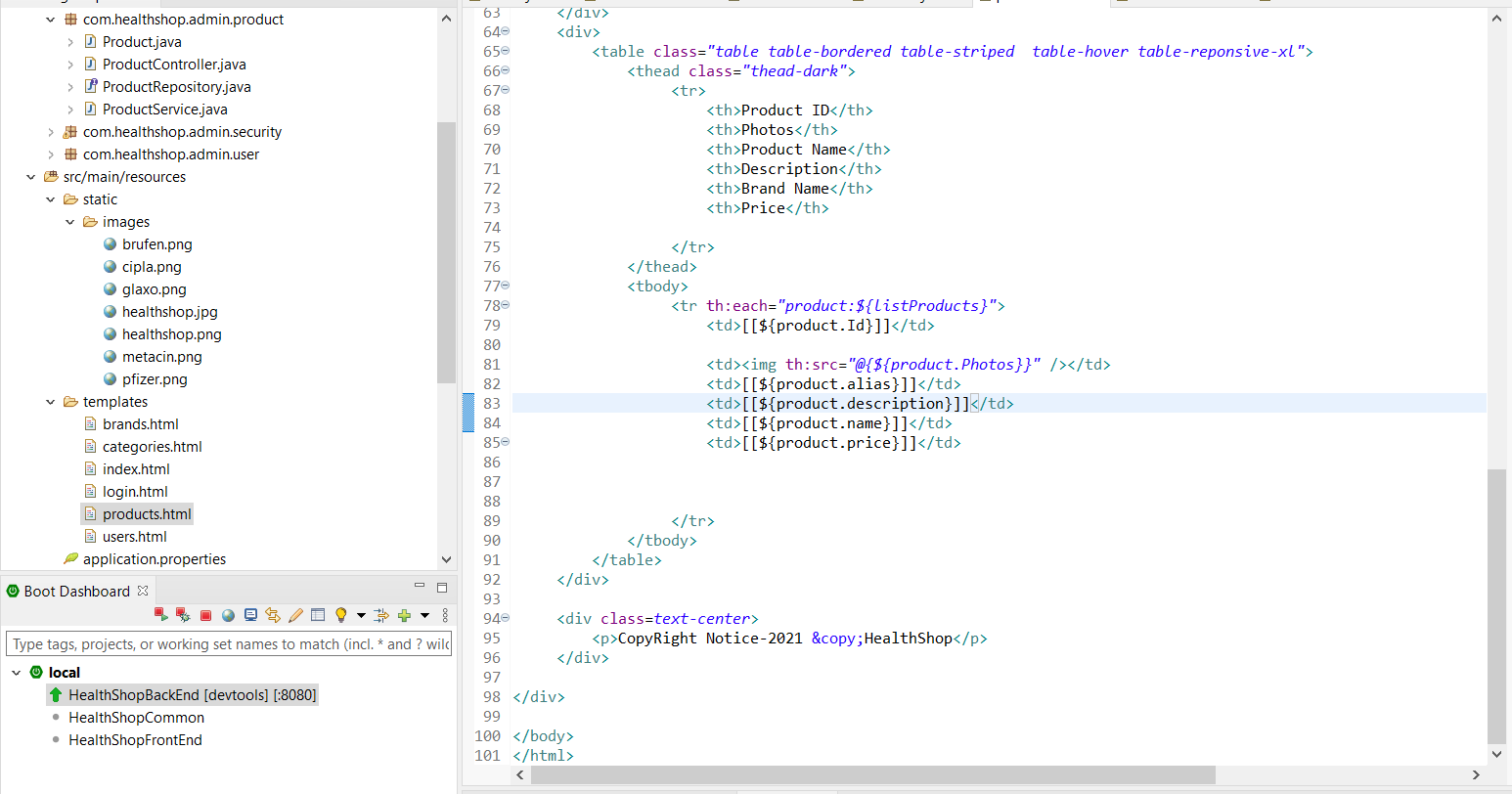
category\_id =1;

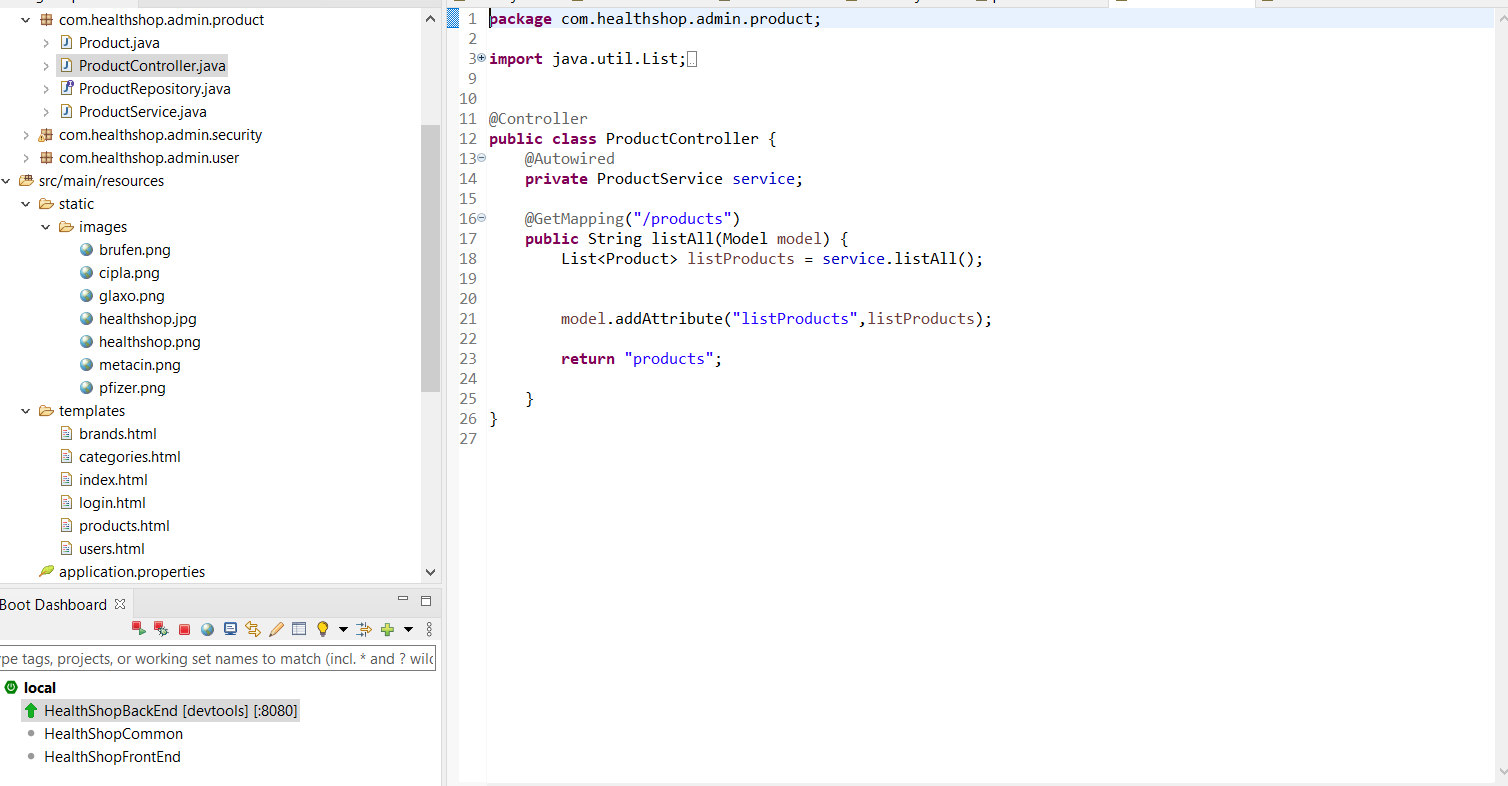
select \* from products;

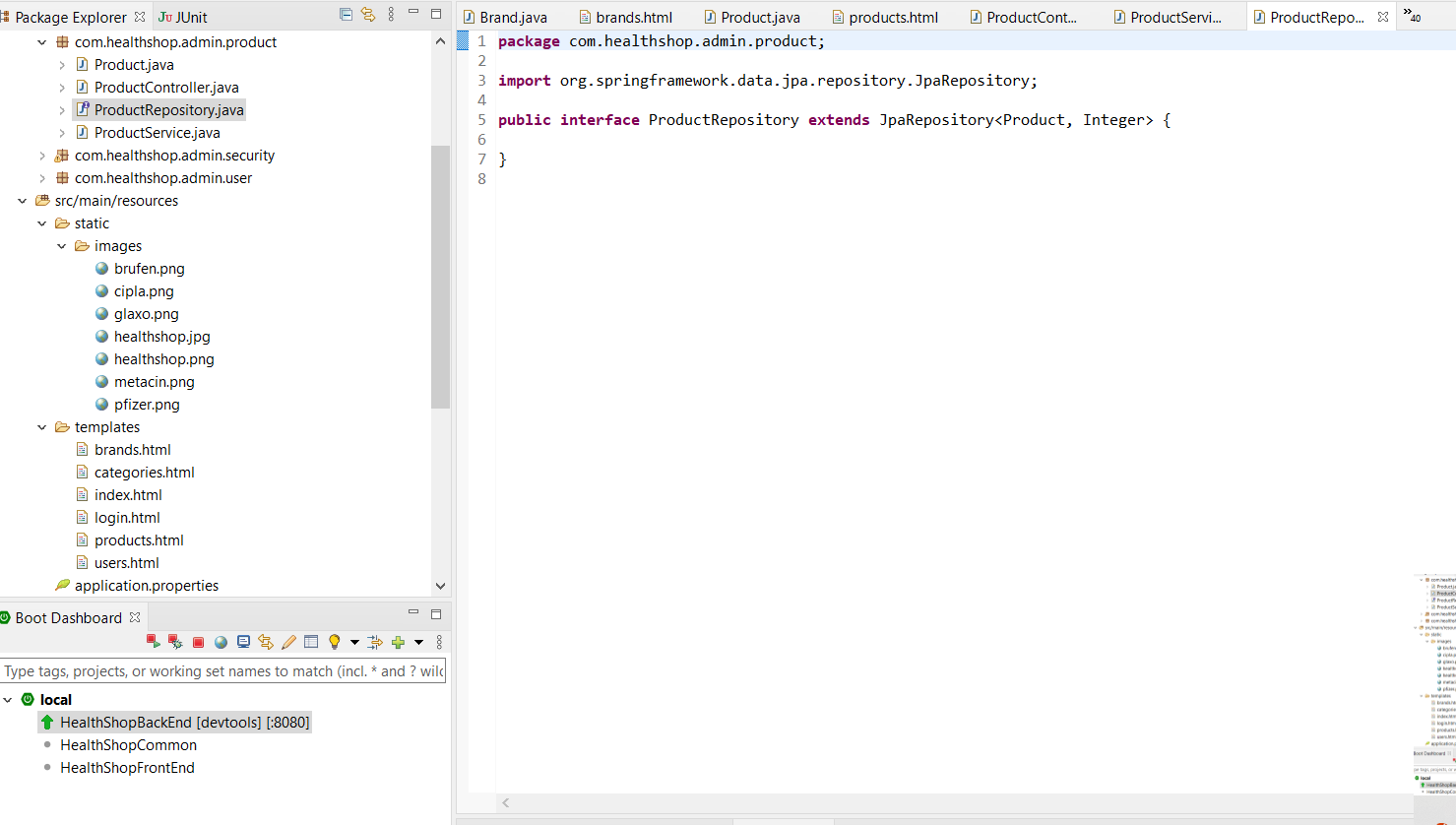
delete from products where id = 2;

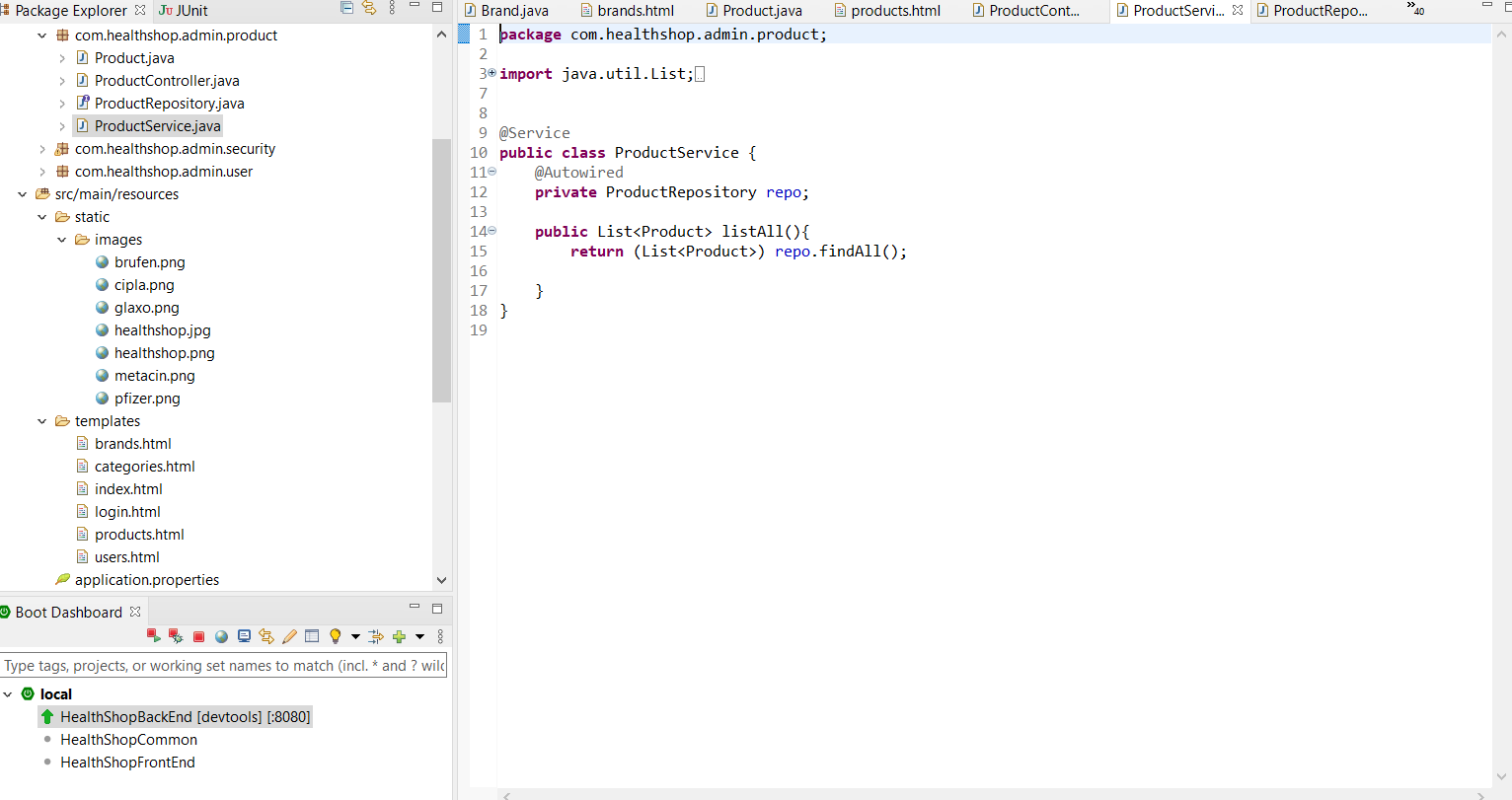




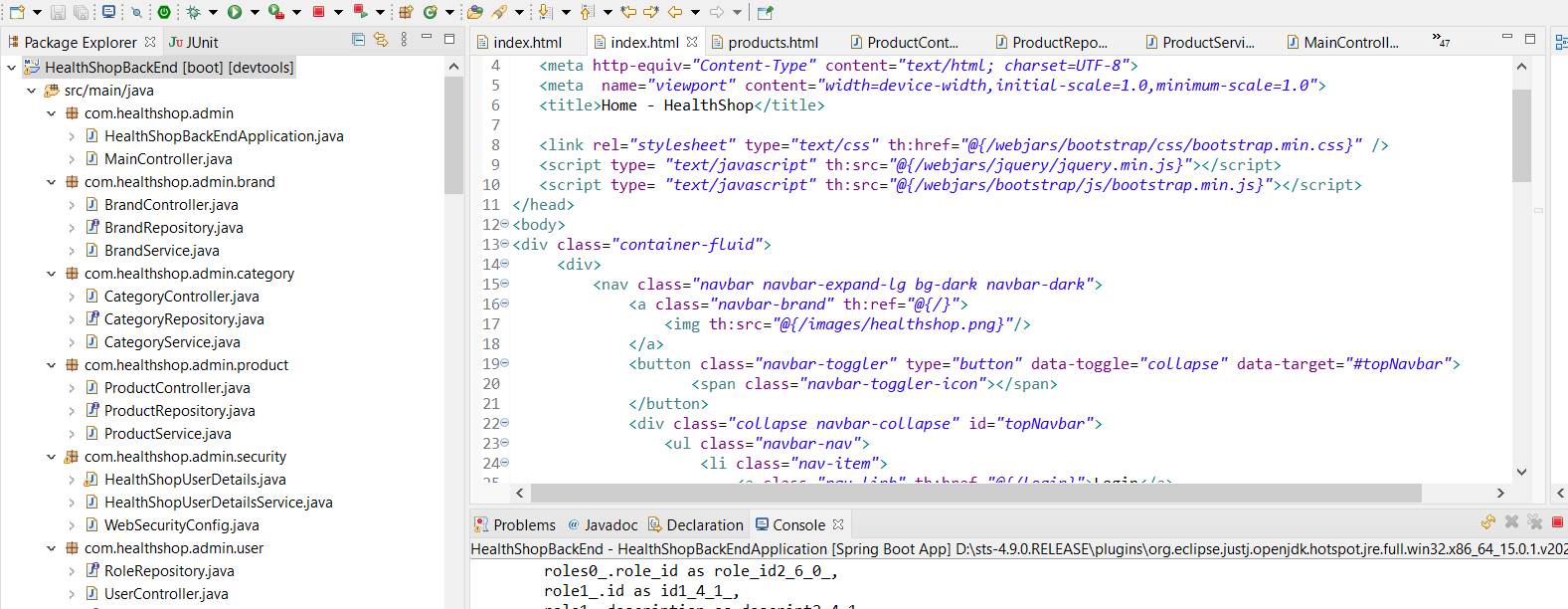


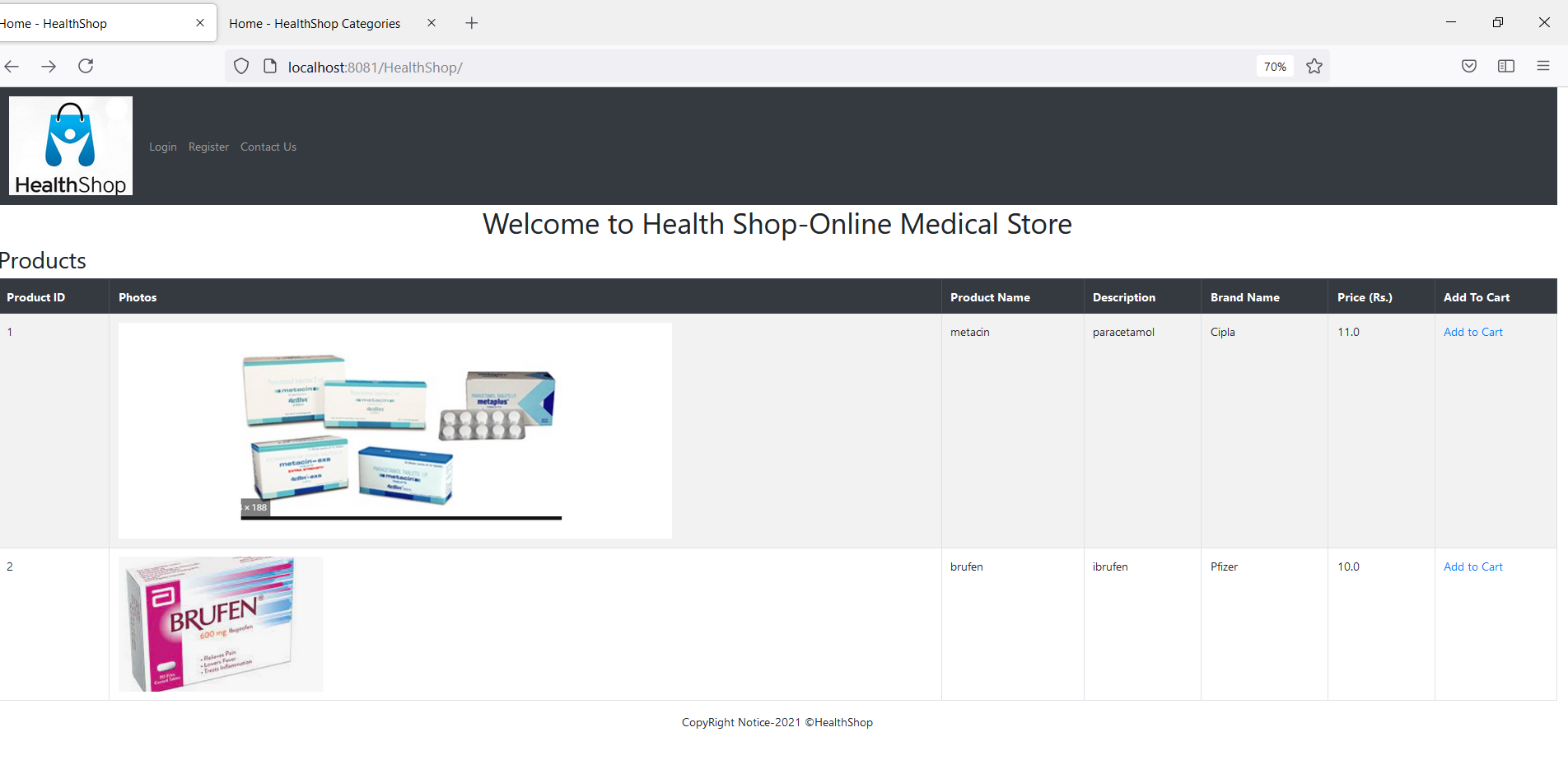






45. Code the Front End





46. Database designed uptil now

