AC – Food back end:

1. Requirement analysis:
   1. Requirement functions:

Menu: save menu’s image into database, get list all menu, get menu by id, create, edit, and deleted menu from database

Store: get list all store, get store information by id, create, edit and deleted store from database

User: get list all user, get user information by id, create, edit and deleted user from database

Orders: get list all orders, get list orders by user’s id, get list order from date to date, create, edit and deleted order from database

Users have roles : ADMIN and USER

With role ADMIN: user can add, edit, deleted that user, store and menu. Get List all orders, by user’s id and list order from date to date.

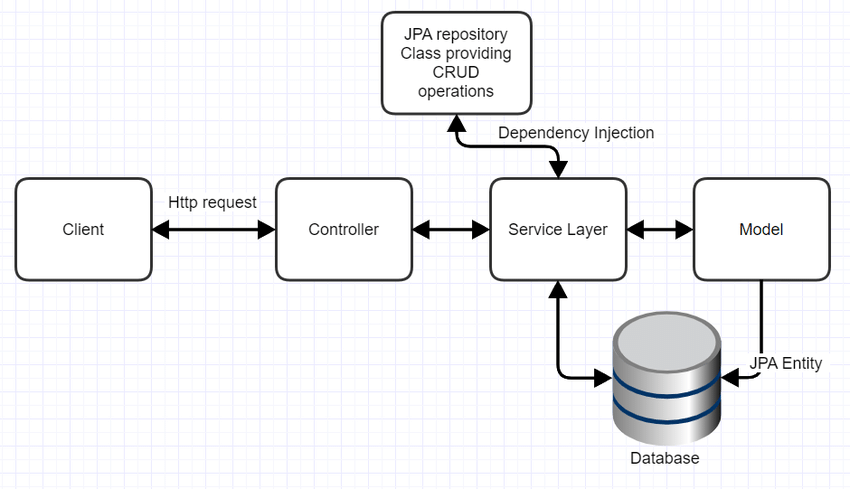
With role USER: user can create and edit orders

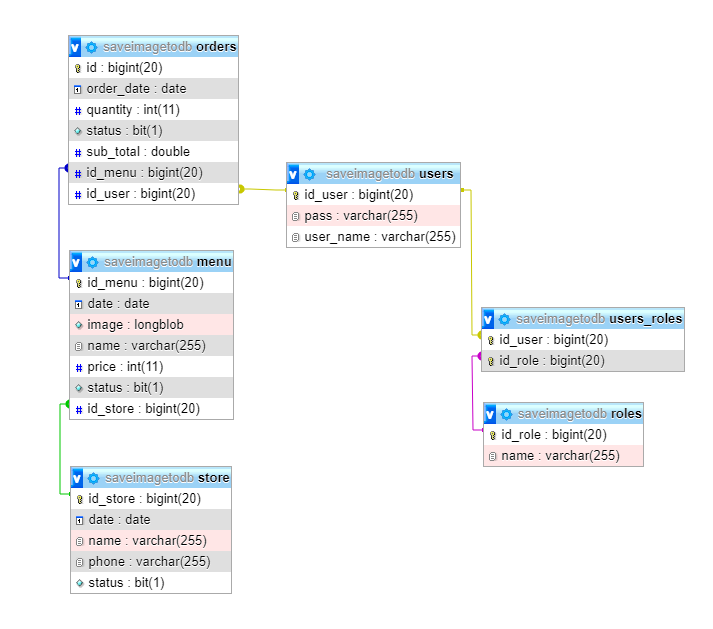
All user have to be authentication and authorization before sending request

Customize error

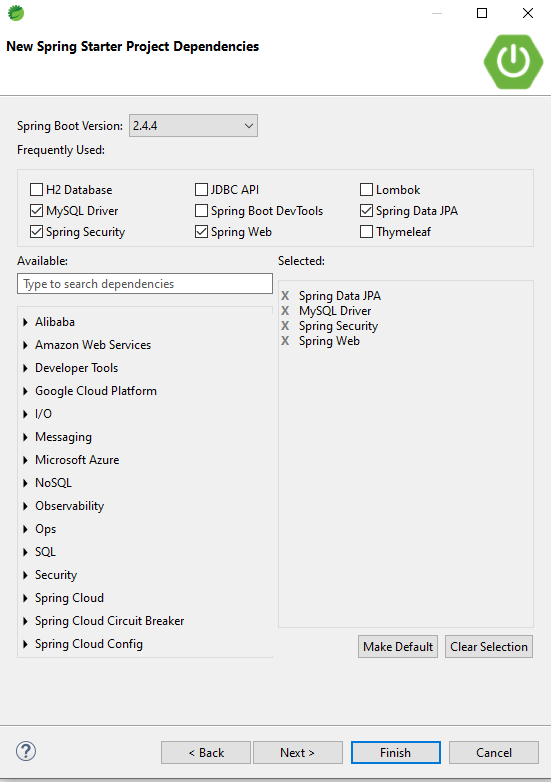
* 1. Technologies used: Spring boot version 2.4.3, JDK
  2. Tool: build tool Maven, Spring Tool Suite, Xampp, Postman.

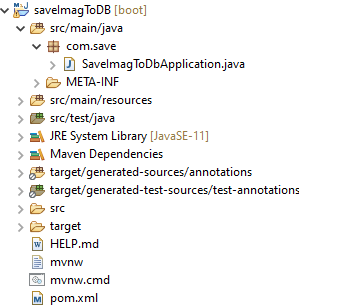
1. Database analysis:



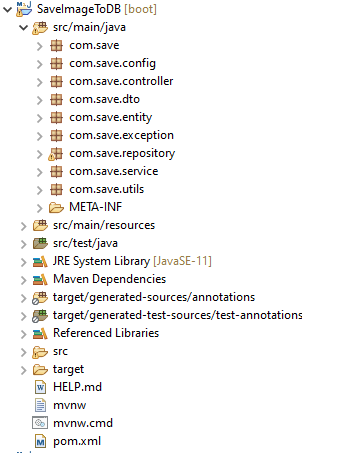


1. Set up environment:
   1. Downloading and configuration JDK and Maven
   2. Downloading and setup Spring Tool Suite, XAMPP, Postman
2. Setup Project;
   1. New a Spring Starter project’s name SaveImageToDB
   2. Adding dependences: Spring Data JPA, Spring Web, Spring Security, MySQL Driver

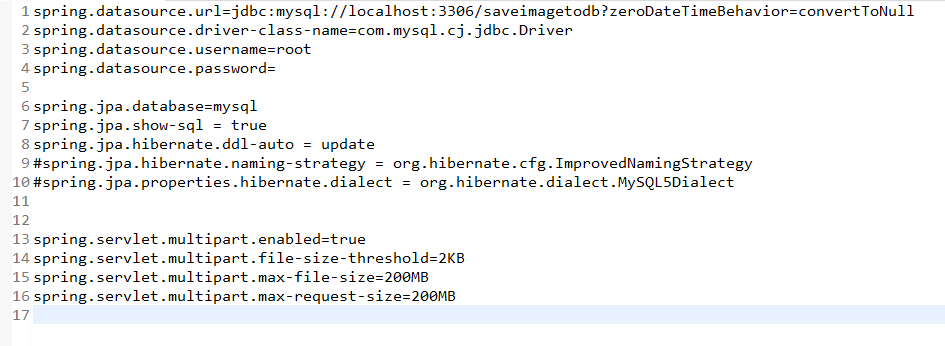


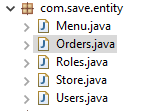
Click finish. Your project’s structure will be like this:

* 1. Create packages:

Include: com.save.config, com.save.controller, com.save.dto, com.save.entity, com.save.exception, com.save.repository, com.save.service, com.save.util

* 1. Configure src/main/resources/application.properties to connect to database and download file

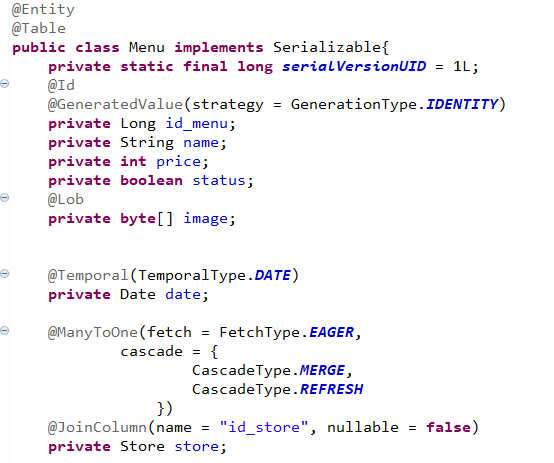
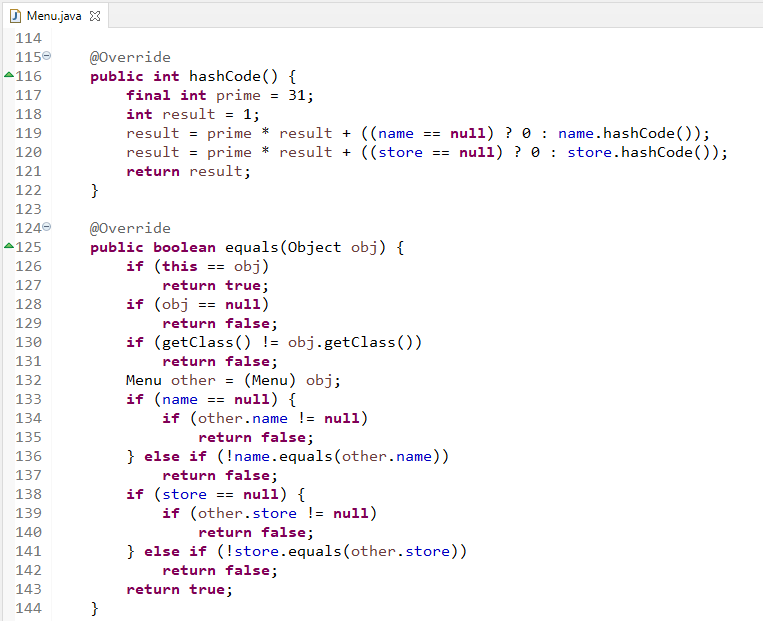


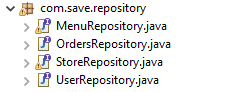
1. Before implement functions:
   1. Create entity classes in com.save.entity package:

Create entity classes and generate getter, setter, constructor for them. But Menu, Store and User we must add hashCode() and equals() methods which use for avoiding duplicate data

Notice about the annotations: if you don’t declare these annotation. There won’t be automation create table in database

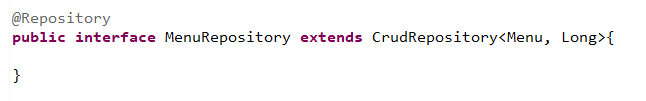
For example: Menu class

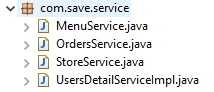
 

* 1. Declaring interfaces in com.save.repository package:

Notice to declare @Repository annotation and extends CrubRepository for each interface

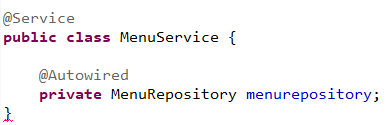
For example: MenuRepository interface



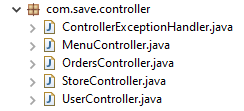
* 1. Declaring services in com.save.service package:

Notice to declare @Service annotation and @Autowired annotation to inject repository class corresponding to service class

For example: MenuService class

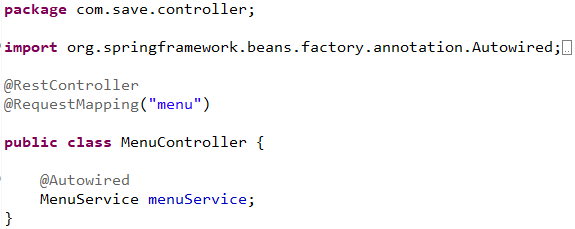


* 1. Declaring Rest controller in com.save.controller package:



Notice to declare @RestController annotation and @Autowired annotation to inject service class corresponding to rest controller class

For example: MenuController class



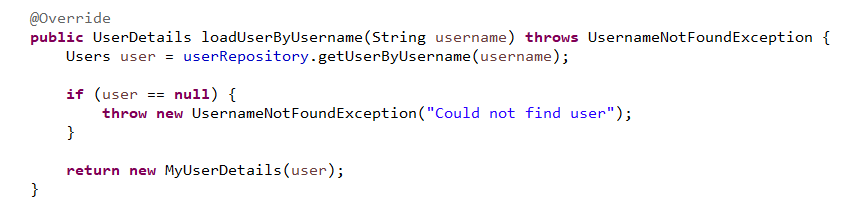
1. Implement functions:
   1. About authentication and authorization:

Create MyUserDetail class to implements interface UserDetails of [org](eclipse-javadoc:%E2%98%82=SaveImageToDB/C:%5C/Users%5C/tnyvon%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/5.4.5%5C/spring-security-core-5.4.5.jar=/maven.pomderived=/true=/=/org.eclipse.jst.component.nondependency=/=/=/maven.pomderived=/true=/=/maven.groupId=/org.springframework.security=/=/maven.artifactId=/spring-security-core=/=/maven.version=/5.4.5=/=/maven.scope=/compile=/%3Corg).[springframework](eclipse-javadoc:%E2%98%82=SaveImageToDB/C:%5C/Users%5C/tnyvon%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/5.4.5%5C/spring-security-core-5.4.5.jar=/maven.pomderived=/true=/=/org.eclipse.jst.component.nondependency=/=/=/maven.pomderived=/true=/=/maven.groupId=/org.springframework.security=/=/maven.artifactId=/spring-security-core=/=/maven.version=/5.4.5=/=/maven.scope=/compile=/%3Corg.springframework).[security](eclipse-javadoc:%E2%98%82=SaveImageToDB/C:%5C/Users%5C/tnyvon%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/5.4.5%5C/spring-security-core-5.4.5.jar=/maven.pomderived=/true=/=/org.eclipse.jst.component.nondependency=/=/=/maven.pomderived=/true=/=/maven.groupId=/org.springframework.security=/=/maven.artifactId=/spring-security-core=/=/maven.version=/5.4.5=/=/maven.scope=/compile=/%3Corg.springframework.security).[core](eclipse-javadoc:%E2%98%82=SaveImageToDB/C:%5C/Users%5C/tnyvon%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/5.4.5%5C/spring-security-core-5.4.5.jar=/maven.pomderived=/true=/=/org.eclipse.jst.component.nondependency=/=/=/maven.pomderived=/true=/=/maven.groupId=/org.springframework.security=/=/maven.artifactId=/spring-security-core=/=/maven.version=/5.4.5=/=/maven.scope=/compile=/%3Corg.springframework.security.core).[userdetails](eclipse-javadoc:%E2%98%82=SaveImageToDB/C:%5C/Users%5C/tnyvon%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/spring-security-core%5C/5.4.5%5C/spring-security-core-5.4.5.jar=/maven.pomderived=/true=/=/org.eclipse.jst.component.nondependency=/=/=/maven.pomderived=/true=/=/maven.groupId=/org.springframework.security=/=/maven.artifactId=/spring-security-core=/=/maven.version=/5.4.5=/=/maven.scope=/compile=/%3Corg.springframework.security.core.userdetails).UserDetails as below:



Declare UsersDetailServiceImp class implements UserDetailService and add method loadUserByUsername(String username), in order to check login information

As below: UsersDetailServiceImp class



Create WebSecurityConfig class to Configure security for the application. It is annotated by @Configuration, this annotation tells Spring that it is a configuration class, and so it will be parsed by Spring at the time the application is run.



After that, we declare a BCryptPasswordEncoder bean in SaveImageToDbApplication class to encode user’s password when an user is stored into database

As below:

