

CONTENTS

Sl. No	Title	Page No
1.	Introduction	3
2.	Description of	
	Existing System	3
3.	Objectives of the	4
	Project	
4.	OOAD of Project	4-9
5.	Database design of	10-11
	Project	
6.	Technologies used in	12
	Project	
7.	Limitation and Scope	12
	of future research	
8.	Working Code	12-19
	Screenshots	
9.	References	20

FOODIE APP

1. INTRODUCTION

Foodie App is online food delivery application where Restaurant owners can register their details and users can find different cuisines based on location and dishes in this application. This application is used by Foodie App admins, Restaurants and Customers. This is a website designed primarily for use in the food delivery industry. The system also allows to quickly and easily manage an online menu which customers can browse and use to place orders with just few clicks.

In this system Restaurants can register under correct category such as Bar, Cafe, Cloud Kitchen, Takeaway with location, restaurant images and contact details. They can add and remove dishes under various categories such as Vegetarian, Non-Vegetarian, Chinese etc. with available time/day. Also they can modify details of the dishes like price, description, image, available time/day. Restaurants can view online orders received and deliver the items accordingly.

This system will allow an administrator to approve Restaurants only after that it will be visible to customers and can disable restaurant from listing. Admin can view feedbacks received from customers and view status of placed orders.

Using this app user can find restaurants by name, location, dish name, rating, hygiene etc. and can place order at a particular restaurant. Users can give rating on restaurants where they placed order.

2. DESCRIPTION OF EXISTING SYSTEM

This existing system is Web application designed using JSP and HTML and it gives insight about how GUI interacts with server-side language, Java, and finally with the Oracle database. The orders are not functional on Sunday. So our project will solve this problem because this deals with 24 hours per 7 weeks. The existing system does not functional on holiday as our system is functional on every day.

3. OBJECTIVES OF THE PROJECT

The objective of this project is to change from manual system to automated system. There are

existing system which uses older technologies such as HTML, JQuery, PHP, Servlet JSP etc.

In our system we are using full stack application with angular, spring boot etc., so that we can

change the front end very easily and support many devices.

The specific objectives of this project are:

Developing user friendly computer based delivery of foods.

Enable to manage the user's request in timely manner.

Computerized ordering of food.

No need to go to buy food items.

Can find variety of foods in single application.

The main advantage of this system is that it greatly simplifies the ordering process for both the

customer and the restaurant and also greatly lightens the load on the restaurant's end, as the

entire process of taking orders is automated.

4. OOAD OF PROJECT

4.1. Use Case Diagram

4.1.1 Actor description

Restaurant: Performs adding and removing dishes, modify details of the dishes and can

view online orders received.

Administrator: Approve restaurants and can disable restaurant from listing. Admin can

view feedbacks received from customers and view status of placed orders.

Customer: User can find restaurants and can place order at a particular restaurant. Users

can give rating on restaurants where they placed order.

4.1.2 Use Case Textual Description

Use case name: Login use case

Identification: UC01

Description: use case to ensure security in system usage.

4

Actor: user: (Customer, Restaurant and Administrator.)

Precondition: the user must have username and password.

Post condition: user get access to the system according to their predefined system privilege and finally he/she logout or turn off the page.

Basic course of action:

- 1. User activates the system.
- 2. System response by displaying the login interfaces and prompts the user for the user ID and password.
- 3. User fills his or her user ID and password and presses enter.
- 4. System verifies user ID and Password.
- 5. User authenticated and gets access to the system.
- 6. System displays its home page.
- 7. Use case ends.

Alternative course of action (if user enters wrong user ID and / or password)

- A1. User is not authenticated and is denied access to the system.
- A2. System displays an incorrect user ID and password message.
- A3. System enables user to try again.
- A4. Use case returns to step 2 of main use case

Use case name: apply to register

Identification: UCO2

Description: use case to register new user and restaurant.

Actor: restaurant, customer.

Precondition: the restaurant/customer must be able to fill all the criteria perfectly and the restaurant/customer. has the ability to access the internet.

Post condition: a restaurant or new customer is registered.

Basic course of action

1. Not include login use case.

- 2. Restaurant/customer selects restaurant/customer link
- 3. Restaurant/ customer select form.
- 4. System display form.
- 5. Fills necessary data.
- 6. Submit
- 7. Use case ends

Alternative course of action (if applicant enters wrong information)

- A1. Restaurant/customer is not authenticated and is denied access to the system.
- A2. System displays an incorrect message.
- A3. Use case returns to step 5 of main use case

Use case name: Search Restaurant

Identifier: UC03

Description: use case to retrieve all restaurant in a particular location.

Actor: customer

Precondition: the customer must be a member.

Post condition: System display information about the restaurants available.

Basic course of action:

- 1. Include login use case.
- 2. Customer activates the user interface.
- 3. Customer selects restaurant from the displayed link.
- 4. System responds by displaying the browse user interface.
- 5. Customer selects the search criteria and enters the search key word.
- 6. System consults the database and displays the collection matching the search key word.
- 7. Use case ends.

Alternative course of action (user entered a search key word that doesn't exist in the system)

A1. System responds stating there is no resource matching the search in its result display.

A2.use case returned to step 5.

Use case name: Order Food

Identifier: UC04

Description: use case to order a food to browse from food collection.

Actor: customer

Precondition: the customer must be a member.

Post condition: system displays information about the available dishes.

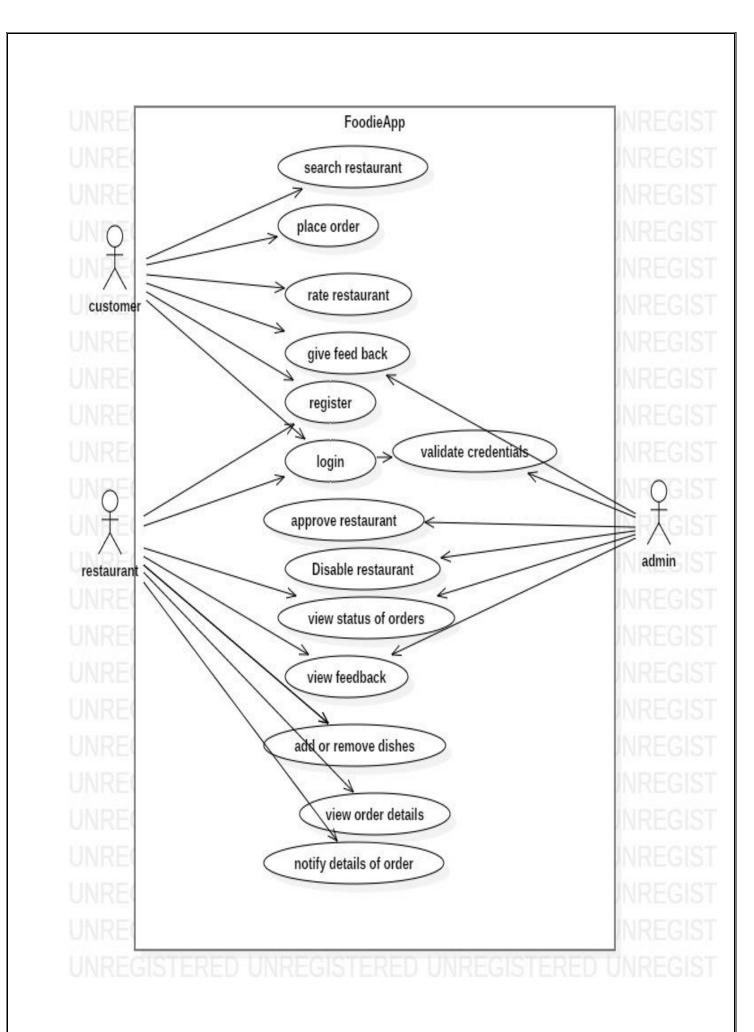
Basic course of action:

- 1. Include login use case.
- 2. Customer activates the order interface.
- 3. Customer selects brows order from the displayed link.
- 4. System responds by displaying the browse order interface.
- 5. Customer selects the order criteria and enters the order key word
- 6. System consults the database and displays message.
- 7. Use case ends.

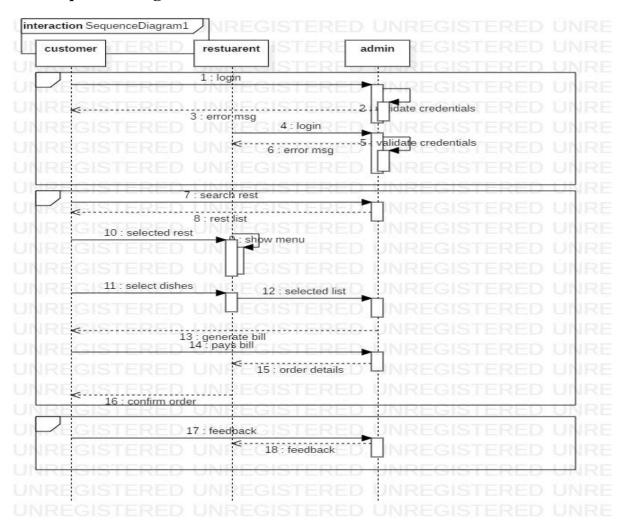
Alternative course of action (user entered an order key word that doesn't much in the system)

A1. System responds to fill the criteria again message.

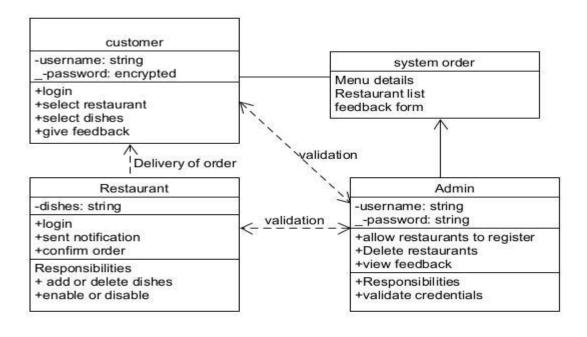
A2.use case returned to step 5 case.



4.2. Sequence Diagram



4.3. Class Diagram



5. DATABASE DESIGN OF PROJECT

Table Structure

- 1. **tbladmin** information of administrator
 - admin_id int, primary key
 - admin username- varchar
 - admin password- varchar
- 2. **tblrestaurant** information of authorized restaurant
 - user_id primary key, auto_increment, int
 - name varchar
 - phone int
 - email_address varchar
 - username varchar
 - password varchar
 - confirm_password varchar.
 - location varchar
 - image file
- 3. **tbluser** information of authorized user of the system.
 - user_id primary key, auto_increment ,int
 - name varchar
 - phone int
 - email_address varchar
 - username varchar
 - password varchar
 - confirm_password varchar.
 - Address varchar
- 4. **tblmenu** this table will store the list of menu/food.
 - menu_id primary key, auto_increment, int
 - menu_name varchar.

- price –float.
- menu_image file
- available_quantity int
- available_time time
- 5. **tblorder** customer orders are stored in the tblorder.
 - order_id primary key, auto_increment, int
 - customer_id int.
 - order_date date.
 - total_amount -float
 - order_status boolean
 - delivery_time time.
 - menu_id -int.
 - amount –float
 - quantity-int
- 6. **tblfeedback-** customer feedback
 - menu_id-int
 - menu_name-varchar
 - date-date
 - content-varchar
- 7. **tblrating** rating and comments of the customers are stored in this table.
 - rating_id primary key, auto_increment
 - menu_id -int.
 - score int
 - remarks- varchar.
 - date_recorde- date
 - customer_id -int.

6. TECHNOLOGIES USED IN PROJECT

- ➤ Front End Angular, CSS, Bootstrap etc.
- ➤ Back End Java 11, Spring Boot, Micro service, JWT, AWS etc.

7. LIMITATIONS AND SCOPE OF FUTURE RESEARCH

Right now we have not integrated payment gateway as PayPal as it will take more time. If we get more time we will do the payment gateway as to make our project to complete application in running stage. As a part of future research we will try to develop our system using modern technologies.

8. WORKING CODE SCREENSHOTS

```
🧓 workspace-spring-tool-suite-4-4.11.0.RELEASE - FoodieAppLocal/src/main/java/com/foodieApp/food/controller/UserRestuarantController.java - Spring Tool Suite 4
                                                                                                                                                                                                          ð
   File Edit Source Refactor Navigate Search Project Run Window Help
  Q 🔡 🐉
  # Package Explorer ⋈ □ □ UserRestuarantController.java ⋈
                                                                                                                                                                                                               - -
                        1 package com.foodieApp.food.controller;
2⊕ import java.util.List;
private RestuarantServiceImpl restuarantDaoImpl;
                                                     @PostMapping("/userregister")
public User addDish(@RequestDody User user) {
   System.out.println(user.toString());
   userDos.register(user);
   return user;
                                                       @PostMapping("/checkUserName")
public boolean checkAvailability(@RequestBody String username){
         return userDao.usernameExists(username);
          Maven Dependencies
         > 😂 src
                                                     @PostHapping("/registerRestuarant")
public Restuarant addRestuarant(@RequestBody Restuarant restuarant) {
    System.out.println(restuarant.toString());
    restuarantDaoImpl.register(restuarant);
    return restuarant;
           pom.xml
                                                      public List<Restuarant> adminApprove(@PathVariable("restuarantName") String restuarantName) {
                                                         List<Restuarant> restuarantData = restuarantDaoImpl.approveRestuarant(restuarantName);
System.out.println(restuarantData.toString());
return restuarantData;
                                                                                                                              Smart Insert
                                                                                                         Writable
                                                                                                                                                     ② 25°C Light rain へ © ■ (条 4) ENG 09-08-2021 09-08-2021
    Type here to search
```

8.1 UserRestaurantController.java

```
🌑 workspace-spring-tool-suite-4-4.11.0.RELEASE - FoodieAppLocal/src/main/java/com/foodieApp/food/controller/UserRestuarantController.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    O
Q া 🖭
                                                                                    🗀 🗇 UserRestuarantController.java 🛭 📝 User.java
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     - -
 ₽ Package Explorer ⊠
                                                                [devtools] 33 349
          > $\frac{10}{2}$ bookapp [boot] [devtools] $\frac{10}{2}$ FoodleAppLocal [boot] [devtools] $\frac{10}{2}$ erc/main/java $\frac{10}{2}$ erc/main/java $\frac{10}{2}$ erc/modleApp.food $\frac{10}{2}$ com.foodleApp.food.controller $\frac{10}{2}$ Discontroller[size] $\frac{10}{2}$ Discontroller[size]
                                                                                                                                         @PostHopping("/registerRestuarant")
public Restuarant addRestuarant(@RequestBody Restuarant restuarant) {
    System.out.println(restuarant.toString());
    restuarantBoalmpl.register(restuarant);
    return restuarant;
⊜
                                                                                                                                         @PostMapping("/approve/(restuarantName)")
public List<Restuarant> adminApprove(@PathVariable("restuarantName") String restuarantName) {
                                     LoginController.java
                                                                                                                                                     List<Restuarant> restuarantData = restuarantDatImpl.approveRestuarant(restuarantName); System.out.printIn(restuarantData.toString()); return restuarantData;

    → LoginController, Java
    → Donincorders/Controller, Java
    → Donincorders/Controller, Java
    → BoundoideApp, Food.dto
    → BoundoideApp, Food.entity
    → Dishes, Java
    → Declarations

                                                                                                                                         @PostMapping("/availability/{restuarantName}/{option}")
public ListRestuarant> adminApprove(@PathVariable("restuarantName") String restuarantName,@PathVariable("option") String option) {
    ListRestuarant> restuarantData * restuarantDaoImpl.restuarantAvailability(restuarantName,option);
    System.out.println(restuarantData.toString());
    return restuarantData;
}
                                     Login.java
                             src/main/resources
                        src/test/java
                      ■ JRE System Library [JavaSE-1.8]
                       Maven Dependencies
                      src

target

pom.xml
                                                                                                                                         Writable
                                                                                                                                                                                                                                                                                                                                                                                                70:1:2447
                                                                                                                                                                                                                                                                                                                                            Smart Insert
                                                                                                                                                                                                                                                                                                                                                                                                                 ② 25°C AQI 78 へ 空 回 (編句) ENG 01:14 AM 09-08-202*
     Type here to search
                                                                                                                                                                                                                                                                                                                      w
```

```
🧓 workspace-spring-tool-suite-4-4.11.0.RELEASE - FoodieAppLocal/src/main/java/com/foodieApp/food/entity/User.java - Spring Tool Suite 4
                                                                                                                                                                                                                  đ
File Edit Source Refactor Navigate Search Project Run Window Help
Q 🔡 🐉
₽ Package Explorer ⊠
                               □ 🗓 UserRestuarantController.java 🖟 User.java 🛭
                                                                                                                                                                                                                           🖹 💲 🖁 🚪 1 package com.foodieApp.food.entity;
> bookapp [boot] [devtools]

FoodieAppLocal [boot] [devtools]
                                                3⊕ import javax.persistence.Entity;[
9 @Entity
10 @Table(name = "user"
11 public class User {
> Æ com.foodieApp.food
            # com.foodieApp.food.config
          ✓ Æ com.foodieApp.food.controller
              DishController.java
                                                       @GeneratedValue(strategy=GenerationType.IDENTITY)
                                                      @GeneratedValue(strategy=f
private int id;
private String username;
private String password;
private String firstname;
private String email;
private String email;
private String anddress;
private String phone;
private String phone;
private String roles;
              feedbackController.java
              LoginController.java
OnlineOrdersController.java
              UserRestuarantController.jav
          > 🔠 com.foodieApp.food.dto

√ № com.foodieApp.food.entity

            > 🔝 Dishes.java
              feedback.java
              Login.java
             > 🔃 onlineOrders.java
                                                       public String getRole() {
             >  Restuarant.java
                                                       }
            # com.foodieApp.food.repo
                                              30⊖
31
32
                                                       public void setRole(String roles) {
          > # com.foodieApp.food.service
                                                           this.roles = roles;
       > 🕭 src/main/resources
                                                       }
         src/test/java

▲ JRE System Library [JavaSE-1.8]

                                                       public User() {
       > Maven Dependencies
        🐎 src
                                                       public User(String username, String password, String firstname, String lastname, String email, String address, String phone, String roles) {
    this.username = username;
    this.password = password;
    this.firstname = firstname;
    this.lastname = lastname;
    this.email = email;
       > 🇀 target
         pom.xml
    < >
                                                                                                             Writable
                                                                                                                                   Smart Insert
                                                                                                                                                             Type here to search
                                                                                                                          w
                                                                                         @ 🔚 💼
```

8.2 User.java

```
🧓 workspace-spring-tool-suite-4-4.11.0.RELEASE - FoodieAppLocal/src/main/java/com/foodieApp/food/entity/User.java - Spring Tool Suite 4
                                                                                                                                                                                                П
                                                                                                                                                                                                     X
File Edit Source Refactor Navigate Search Project Run Window Help
Q 🔡 🐉
# Package Explorer ⋈ □ □ UserRestuarantController,java □ User,java ⋈
                                                                                                                                                                                                     - -
                                                      this.email = email;
this.address = address;
this.phone = phone;
this.roles = roles;
    > Bookapp [boot] [devtools]
                                          45
46
47
48
    ✓ 🎏 FoodieAppLocal [boot] [devtools]
      ▣
                                                  }
         > # com.foodieApp.food
          tom.foodieApp.food.config

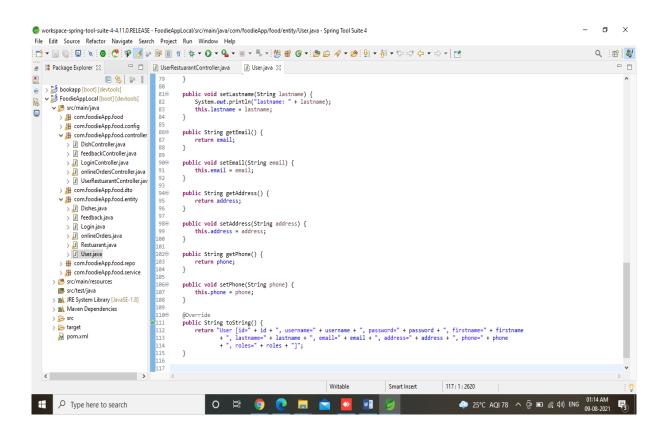
√ 

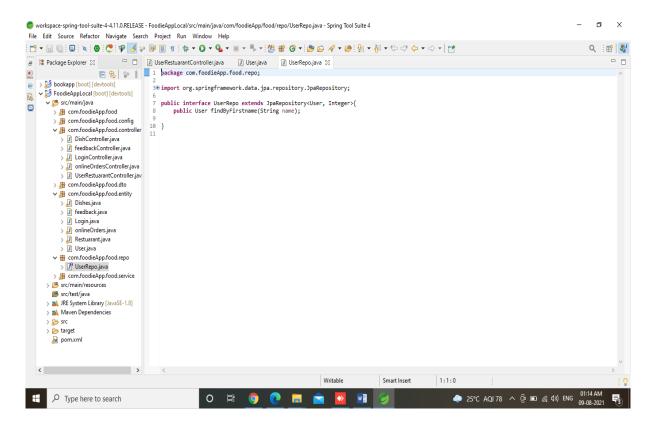
⊞ com.foodieApp.food.controller

                                          51⊝
                                                  public String getUsername() {
           > DishController.java
             feedbackController.java
                                                  }
           > II LoginController.java
                                                  public void setUsername(String username) {
    System.out.println("username: " + username);
    this.username = username;
           > 🔝 onlineOrdersController.java
            > 🔠 com.foodieApp.food.dto
                                          50
59
60⊜
61
62

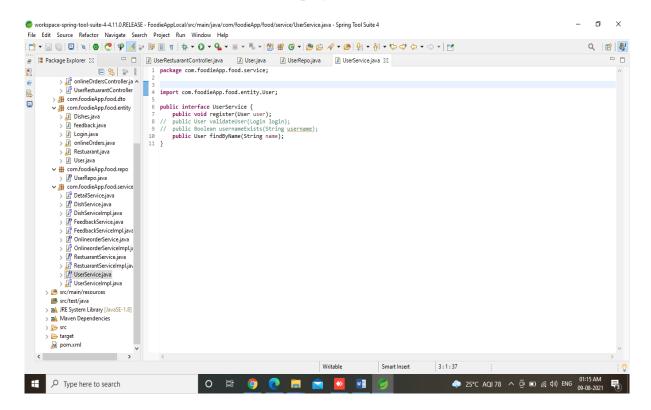
    ## com.foodieApp.food.entity

           > 🔊 Dishes.java
                                                  public String getPassword() {
             feedback.java
                                                      return password;
                                                  }
           > D Login.java
           > 🔝 onlineOrders.java
                                                  public\ void\ setPassword(String\ password)\ \{
           > Restuarant.java
                                          65
66
67
                                                  }
         > 🌐 com.foodieApp.food.repo
                                                  public String getFirstname() {
    return firstname;
         > A com.foodieApp.food.service
                                          69
                                                  }
        # src/test/iava
                                                  public void setFirstname(String firstname) {
   System.out.println("firstname: " + firstname);
   this.firstname = firstname;
      > M JRE System Library [JavaSE-1.8]
      > 🚵 Maven Dependencies
      > 🗁 src
      > 🗁 target
                                                  public String getLastname() {
    return lastname;
        mx.moq 🔝
                                                  }
                                                  public void setLastname(String lastname) {
                                          81⊖
                                >
                                                                                                                                           43:1:1000
                                                                                                    Writable
                                                                                                                       Smart Insert
                                                                                                                                                Ф 25°C AQI 78 ^ @ □ //, (1)) ENG 09-08-2021
  Type here to search
```





8.3 UserRepo.java



8.4 UserService.java

```
🏀 workspace-spring-tool-suite-4-4.11.0.RELEASE - FoodieAppl.ocal/src/main/java/com/foodieApp/food/controller/LoginController.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               o
 Q া 🖭
                                                                         | CoginController,java | Deckage com.foodieApp.food.controller;
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        ₽ Package Explorer ⊠
            mAutowired
private RestuarantService restuarantDao;
                                    onlineOrdersController.ja
UserRestuarantController
                              > # com.foodieApp.food.dto

> # com.foodieApp.food.entity

> # Dishes.java

> # feedback.java
                                                                                                                                                           @GetMapping("login")
public AuthResponse validateLogin(@AuthenticationPrincipal SecUser secUser) {
   System.out.println(""");
   System.out.println(secUser.getUser());
   System.out.println("""");
   return new AuthResponse("User successfully authenticated", secUser.getUser().getRole());
}
                            @PostMapping("register")
public UserRegResponse userRegister(@RequestBody UserRegRequest userReq) {
    System.out.println(userReq)
    User user = mew User();
    user.setfirstname(userReq, getFirstname());
    user.settirstname(userReq, getFarsnowed());
    user.settAastname(userReq, getFarsnowed());
    user.settAaddress(userReq, getAddress());
    user.setTamil(userReq, getBandl());
    user.setTamil(userReq, getBandl());
    user.setUsername(userReq.getBandl());
    user.setUsername(userReq.getBandl());
                                            DishServiceImpl.java

| DinisferviceImpljava | 50 |
| FeedbackServicejava | 51 |
| FeedbackServiceimpl.jav | 51 |
| OnlineorderServiceimpl.ja | 54 |
| OnlineorderServicejava | 55 |
| RestuarantServicejava | 55 |
| RestuarantServiceimpl.ja | 57 |
| OnlineorderServiceimpl.ja | 58 |
| OnlineorderServiceimpl.ja | 59 |
| OnlineorderServiceimpl.ja | 50 |
|
                                                                                                                                                                                                                                                                                                                                                                                         Smart Insert
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 Type here to search
```

8.5 UserServiceImpl.java

```
🧓 workspace-spring-tool-suite-4-4.11.0.RELEASE - FoodieAppLocal/src/main/java/com/foodieApp/food/controller/LoginController.java - Spring Tool Suite 4
 File Edit Source Refactor Navigate Search Project Run Window Help
Q 🔡 🖏
         # Package Explorer ⋈ □ □ DoginController.java ⋈
                                                                                                                                                                                                                                                      - -
.
     > $\frac{\text{\begin{subarray}{c}}{\text{ bookapp [boot] [devtools]}}} \ \frac{2}{\text{\begin{subarray}{c}}{\text{ bookapp [boot] [devtools]}}} \ \frac{2}{\text{\begin{subarray}{c}}{\text{ a}}} \ \int \text{import org.springframework.beans.factory.annotation.Autowired;}} \ \frac{2}{2} \ \text{\text{a}} \ \text{import org.springframework.beans.factory.annotation.Autowired;}} \ \]
        @RequestMapping("/login")
public Login showLogin() {
   return new Login();
            com.foodieApp.food.entity
              > 🔝 Dishes.java

    ∫ feedback.java

                 Login.java
                                                            > 🚺 onlineOrders.java
                🔝 Restuarant.java
                User.java
           UserRepo.java

✓ 

    Com.foodieApp.food.service

                                                             PostHapping("register")
public UserRegResponse userRegister(@RequestBody UserRegRequest userReq) {
    System.out.println(userReq);
    User user = new User();
    user.setFirstname(userReq.getFirstname());
    user.setEinstname(userReq.getLastname());
    user.setBassword(userReq.getDastword());
    user.setAddress(userReq.getAddress());
    user.setEmail(userReq.getDastword());
    user.setPhone(userReq.getPhone());
    user.setUsername(userReq.getUsername());
                 DetailService.java
                DishService.java
                 DishServiceImpl.java
                 FeedbackService.iava
                FeedbackServiceImpl.java

OnlineorderService.java
                OnlineorderServiceImpl.ja
RestuarantService.java
                 RestuarantServiceImpl.jav 56
57
                                                                                                                                                                                 ♦ 25°C Light rain ヘ 📴 🗈 🖟 ♦) ENG 01:37 AM
  Type here to search
                                                                          O 🛱 🧑 🙋 🥫 😭
```

8.6 LoginController.java

```
🏀 workspace-spring-tool-suite-4-4.11.0.RELEASE - FoodieAppLocal/src/main/java/com/foodieApp/food/entity/Login.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
Q 🔡 🐯
  ₽ Package Explorer ⊠
                                                                                                                                                                  public Login(String username, String password) {
   this.username = username;
   this.password = password;
}
                                           | UserRestuarantController | 159 | 159 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 150 | 1

y ∠ AuthResponsejava

) SecUserjava

) UserRespResponsejava

y Bonnodiappfood-entity

) Dishes,java

) I tegnijava

) I tegnijava

) OnlineOrdersjava

} Restuarnatjava

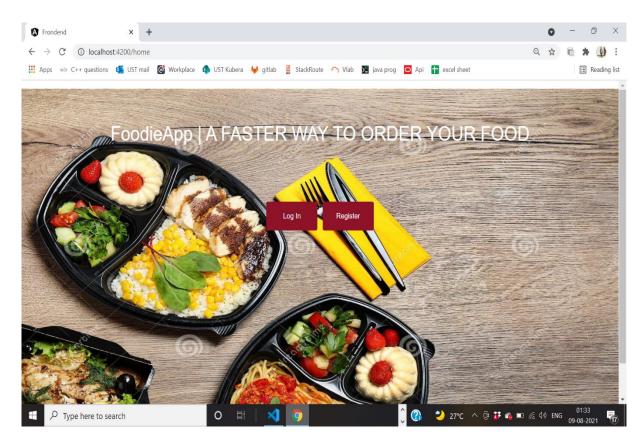
2 Userjava

2 Userjava
                                                                                                                                                                         public void setUsername(String username) {
   this.username = username;
                                                                                                                                                                            public String getPassword() {
    return password;
                                                                                                                                                                           public void setPassword(String password) {
   this.password = password;

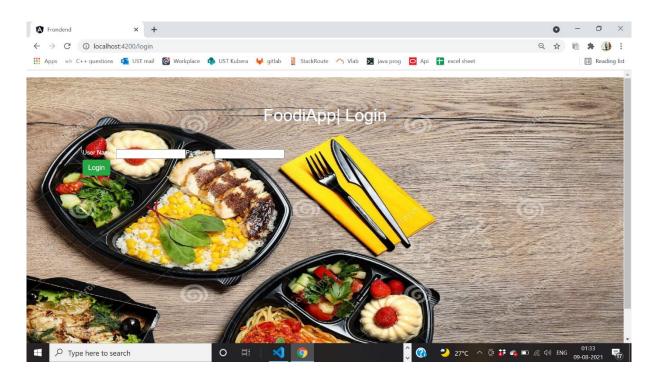
→ ∰ com.foodieApp.food.repo

                                                                                                                                                                          Smart Insert
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                Type here to search
```

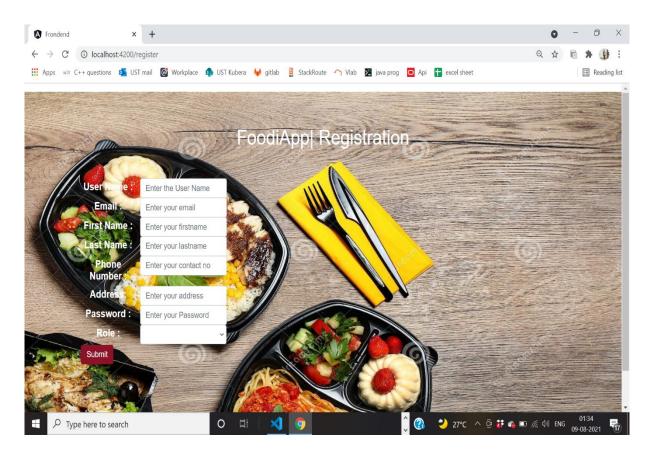
8.7 Login.java



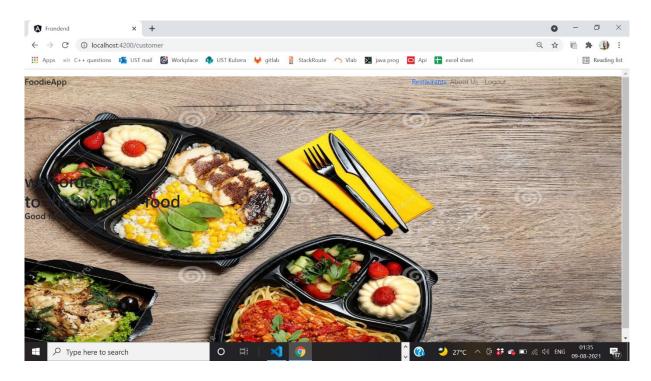
8.8 Homepage



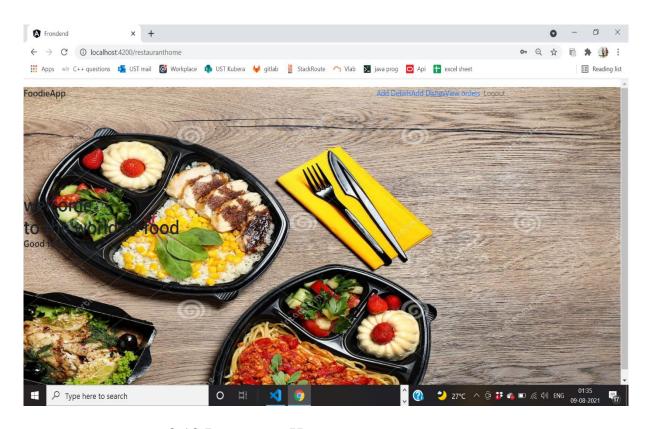
8.9 Login



8.10 Registration



8.11 Customer Homepage



8.12 Restaurant Homepage

9. REFERENCES

- https://www.youtube.com/watch?v=USQvn3O2mBY&list=PLJ-lrQx0LAbxJB6B7go8Tj39xD33fYDP6
- https://www.bezkoder.com/angular-11-spring-boot-jwt-auth/