Source Code

Frontend:

addFood.ts:

```
import { Component } from '@angular/core';
import { FormControl,FormGroup } from '@angular/forms';
import { RestControlService } from 'src/app/services/rest-control.service';
import { Router } from '@angular/router';


@Component({
  selector: 'app-add-food',
  templateUrl: './add-food.component.html',
  styleUrls: ['./add-food.component.css']
})
export class AddFoodComponent {
  msg:string="";

  foodRef=new FormGroup({
    fname:new FormControl(),
    category :new FormControl(),
    description :new FormControl(),
    price :new FormControl(),
    imageurl:new FormControl(),
    fid: new FormControl(null)
  });

  constructor(private restService:RestControlService,public router:Router){
  }

  addFood():void{
    console.log(this.foodRef.value);
    let result=this.restService.addFood(this.foodRef.value).subscribe({
      next:(value:any)=>{
        console.log(value);
        if(value=="Dish added succcessfully"){
          alert(value);
          this.msg=value;
        }
        else
          this.msg=value;
      },error:(error:any)=> {
        console.log(error);
    },
    complete:()=> {
      this.restService.findAllFoods();
      this.restService.findYourFood();
    }
    });
    this.foodRef.reset();
    this.msg="";
  }
}
```

Sign-in:

```
import { Component } from '@angular/core';
import { FormGroup,FormControl } from '@angular/forms';
import { RestAuthenticationService } from 'src/app/services/rest-
authentication.service';
import { Router } from '@angular/router';
import { RestControlService } from 'src/app/services/rest-control.service';

@Component({
  selector: 'app-rest-signin',
  templateUrl: './rest-signin.component.html',
  styleUrls: ['./rest-signin.component.css']
})
export class RestSigninComponent {
  msg:string="";
  restaurantname:string="";
  restRef=new FormGroup({
    restaurantname:new FormControl(),
    password:new FormControl(),
  });

  constructor(private authService:RestAuthenticationService,
    private restService:RestControlService, public router:Router){
  }

  signIn():void{
    this.restaurantname=this.restRef.value.restaurantname;
    let result=this.authService.signIn(this.restRef.value).subscribe({
      next:(value:any)=>{
        console.log(value);////////////////////
        if(value =="login successfull"){
          this.restService.getRestaurant(this.restaurantname);
          this.router.navigate(["restaurant-dashboard"]);
        }
        else
          this.msg=value;
      },error:(error:any)=> {
        console.log(error);
    },
    complete:()=> {
      console.log("---SignIn done!---");
    }
    });
    this.restRef.reset();
  }
}
```

Sign-up:

```
import { Component } from '@angular/core';
import { FormGroup,FormControl } from '@angular/forms';
import { RestAuthenticationService } from 'src/app/services/rest-
authentication.service';
import { Router } from '@angular/router';
import { RestControlService } from 'src/app/services/rest-control.service';

@Component({
  selector: 'app-rest-signup',
  templateUrl: './rest-signup.component.html',
  styleUrls: ['./rest-signup.component.css']
})
export class RestSignupComponent {
  msg:string="";
  restaurantname:string="";
  restRef=new FormGroup({
    restaurantname:new FormControl(),
    password:new FormControl(),
  });

  constructor(private authService:RestAuthenticationService,
    private restService:RestControlService,public router:Router){
  }

  signUp():void{
    this.restaurantname=this.restRef.value.restaurantname;
    let result=this.authService.signUp(this.restRef.value).subscribe({
      next:(value:any)=>{
        console.log(value);////////////////////
        if(value == "Restaurant account created successfully"){
          this.restService.getRestaurant(this.restaurantname);
          this.router.navigate(["restaurant-dashboard"]);
        }
        else
          this.msg=value;
      },error:(error:any)=> {
        console.log(error);
    },
    complete:()=> {
      console.log("---SignUp done!---");
    }
    });
    this.restRef.reset();
  }
}
```

updateFood:

```
import { Component,OnInit } from '@angular/core';
import { RestControlService } from 'src/app/services/rest-control.service';
import { Food } from 'src/app/models/food';

@Component({
  selector: 'app-update-food',
  templateUrl: './update-food.component.html',
  styleUrls: ['./update-food.component.css']
})
export class UpdateFoodComponent {
  products:Array<Food>=[];

  constructor(public restService:RestControlService){}

  ngOnInit(): void {
    this.loadProductInfo();
  }

  loadProductInfo(): void {
    this.restService.findYourFood().subscribe({
      next:(result:any)=> {
        this.products=result;
      },
      error:(error:any)=> {
          console.log(error)
      },
      complete:()=> {
        console.log("product loaded...");
      }
    })
  }

  updatePrice(fid:number,price:number): void{
    this.restService.updatePrice(fid,price).subscribe({
      next:(result:any)=> {
        alert(result);
      },
      error:(error:any)=> {
          console.log(error);
      },
      complete:()=> {
        console.log("price updated...");
        this.loadProductInfo();
      }
    })
  }
  deleteDish(fid:number): void{
    this.restService.deleteFood(fid).subscribe({
      next:(result:any)=> {
        alert(result);
      },
      error:(error:any)=> {
          console.log(error);
      },
      complete:()=> {
        console.log("food deleted...");
        this.loadProductInfo();
      }
    })
  }
}
```

RestaurantAuthenticationService.ts:

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class RestAuthenticationService {

  private url:String="http://localhost:7979/Restaurants/";

  constructor(private http: HttpClient) {
   }

   signIn(restaurant:any):Observable<string>{
    return this.http.post(this.url+"signIn",restaurant,{responseType:'text'});
   }

   signUp(restaurant:any):Observable<string>{
    return this.http.post(this.url+"signUp",restaurant,{responseType:'text'});
   }
}
```

RestaurantConrtrolService.ts:

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';
import { Food } from '../models/food';
import { Orders } from '../models/orders';

@Injectable({
  providedIn: 'root'
})
export class RestControlService {

  private url:String="http://localhost:7979/Restaurants/";
  restaurantname:any="";

  constructor(private http: HttpClient) {}

   getRestaurant(restaurantName:any){
    this.restaurantname = restaurantName;
   }
   setRestaurant():String{
      return this.restaurantname;
   }

   findAllFoods():Observable<Food[]>{
    return this.http.get<Food[]>(this.url+"findAllFoods");
   }

   findFood(food_id:any):Observable<Food>{
    return this.http.get<Food>(this.url+"/findFood/"+food_id)
  }

   findYourFood(){
    return
this.http.get<Food[]>(this.url+"findYourFood/"+this.restaurantname);
   }

   addFood(food:any):Observable<string>{
    return
this.http.post(this.url+"addFood/"+this.restaurantname,food,{responseType:'tex
t'});
   }
   deleteFood(food_id:any):Observable<string>{
    return
this.http.delete(this.url+"deleteFood/"+food_id,{responseType:'text'});
   }

   updatePrice(food_id:any,price:any):Observable<String>{
    return
this.http.put(this.url+"update/"+food_id,price,{responseType:'text'});
   }
   getOrdersOfRestaurant():Observable<Orders[]>{
    return
this.http.get<Orders[]>(this.url+"findYourOrders/"+this.restaurantname);
   }
   deliverFoods(order_id:any){
    return
this.http.get(this.url+"deliverOrders/"+order_id,{responseType:'text'});
   }

}
```

Food.ts:

```
import { Food } from "./food";

export class CartItems {
    constructor(
        public id:number,
        public quantity: number,
        public cart: {
            emailid:string
        },
        public food: Food
    ){}
}
```

Cart-items.ts:

```
import { Food } from "./food";

export class CartItems {
    constructor(
        public id:number,
        public quantity: number,
        public cart: {
            emailid:string
        },
        public food: Food
    ){}
}
```

cartComponent.ts:

```
import { Component } from '@angular/core';
import { CartItems } from 'src/app/models/cart-items';
import { UserControlService } from 'src/app/services/user-control.service';

@Component({
  selector: 'app-cart',
  templateUrl: './cart.component.html',
  styleUrls: ['./cart.component.css']
})
export class CartComponent {
  cartItems:Array<CartItems>=[];

  constructor(public userService:UserControlService){}
  ngOnInit(): void {
    this.loadCart();
  }

  loadCart(): void {
    this.userService.viewCart().subscribe({
      next:(result:any)=> {
        this.cartItems=result;
      },
      error:(error:any)=> {
          console.log(error)
      },
      complete:()=> {
        console.log("product loaded...");
      }
    })
  }

  updateQuantity(cartItem_id:any,qty:any): void {
    console.log(cartItem_id);
    this.userService.updateQuantity(qty,cartItem_id).subscribe({
      next:(result:any)=> {
        alert(result);
      },
      error:(error:any)=> {
          console.log(error)
      },
      complete:()=> {
        console.log("product loaded...");
        this.loadCart();
      }
    })
  }

  deleteFromCart(cartItem_id:any): void {
    console.log(cartItem_id);
    this.userService.deleteCart(cartItem_id).subscribe({
      next:(result:any)=> {
       alert(result);
      },
      error:(error:any)=> {
          console.log(error)
      },
      complete:()=> {
        console.log("product deleted...");
        this.loadCart();
      }
    })
  }

  placeOrder(cartItem_id:any,quantity:any): void {
    console.log(cartItem_id);
    this.userService.placeOrder(quantity,cartItem_id).subscribe({
      next:(result:any)=> {
        alert(result);
      },
      error:(error:any)=> {
          console.log(error)
      },
      complete:()=> {
        console.log("Order placed successfully...");
        this.loadCart();
      }
    })
  }
}
```

order.component.ts:

```
import { Component } from '@angular/core';
import { Orders } from 'src/app/models/orders';
import { UserControlService } from 'src/app/services/user-control.service';

@Component({
  selector: 'app-orders',
  templateUrl: './orders.component.html',
  styleUrls: ['./orders.component.css']
})
export class OrdersComponent {
  orders:Array<Orders>=[];
  pendingOrders:Array<Orders>=[];
  constructor(public userService:UserControlService){}
  ngOnInit(): void {
    this.loadProductInfo();
  }
  loadProductInfo(): void {
    this.userService.orderHistory().subscribe({
      next:(result:any)=> {
        this.orders=result;
        this.pendingOrders = this.orders.filter(order => order.state ===
'pending');
      },
      error:(error:any)=> {
          console.log(error)
      },
      complete:()=> {
        console.log("Pending orders loaded...");
      }
    })
  }

  cancelOrder(order_id:any){
    this.userService.cancelOrder(order_id).subscribe({
      next:(result:any)=> {
        alert(result);
      },
      error:(error:any)=> {
          console.log(error)
      },
      complete:()=> {
        console.log("Pending orders loaded...");
        this.loadProductInfo();
      }
    })
  }
}
```

Home.component.ts:

```
import { Component } from '@angular/core';
import { Food } from 'src/app/models/food';
import { UserControlService } from 'src/app/services/user-control.service';
```

```
@Component({
  selector: 'app-home',
  templateUrl: './home.component.html',
  styleUrls: ['./home.component.css']
})
export class HomeComponent {
  products:Array<Food>=[];

  constructor(public userService:UserControlService){}
  ngOnInit(): void {
    this.loadCart();
  }

  loadCart(): void {
    this.userService.findAllFoods().subscribe({
      next:(result:any)=> {
        this.products=result;
      },
      error:(error:any)=> {
          console.log(error)
      },
      complete:()=> {
        console.log("product loaded...");
      }
    })
  }

  addFoodToCart(food_id:any): void {
      this.userService.addFoodToCart(food_id).subscribe({
        next:(result:any)=> {
          console.log(result);
          alert(result);
        },
        error:(error:any)=> {
            console.log(error)
        },
        complete:()=> {
          console.log("product loaded...");
          this.loadCart();
        }
      })
    }
}
```

history.component.ts:

```
import { Component } from '@angular/core';
import { Orders } from 'src/app/models/orders';
import { UserControlService } from 'src/app/services/user-control.service';

@Component({
  selector: 'app-history',
  templateUrl: './history.component.html',
  styleUrls: ['./history.component.css']
})
export class HistoryComponent {
  orders:Array<Orders>=[];
  completedOrders:Array<Orders>=[];
  constructor(public userService:UserControlService){}
  ngOnInit(): void {
    this.loadProductInfo();
  }
  loadProductInfo(): void {
    this.userService.orderHistory().subscribe({
      next:(result:any)=> {
        this.orders=result;
        this.completedOrders = this.orders.filter(order => order.state ===
'delivered');
      },
      error:(error:any)=> {
          console.log(error)
      },
      complete:()=> {
        console.log("History of orders loaded...");
      }
    })
  }
}
```

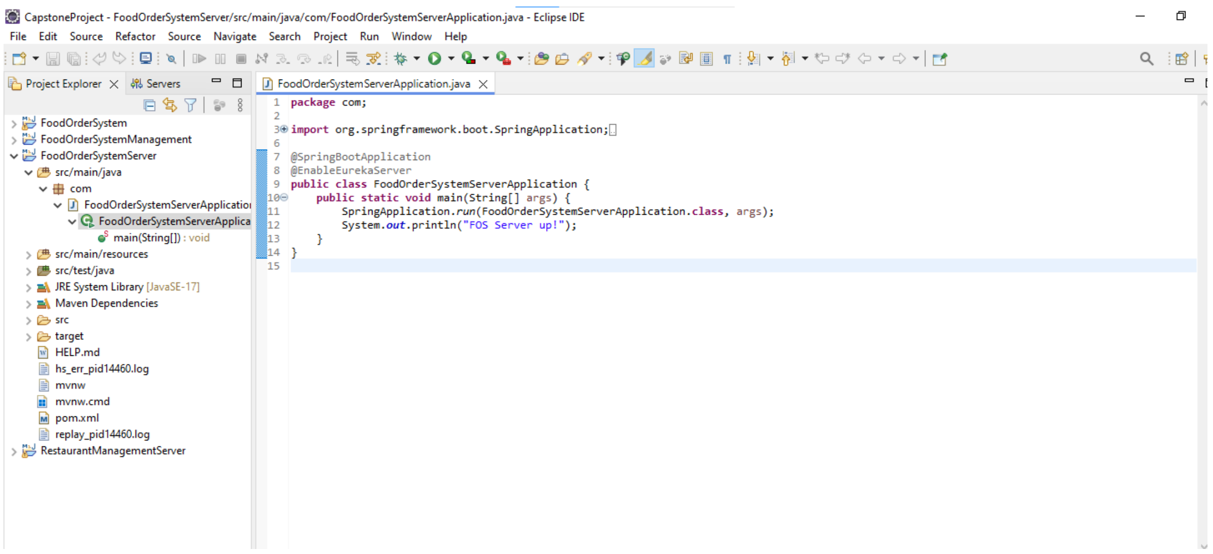
UserAuthenticationService.ts:

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class UserAuthenticationService {

  private url:String="http://localhost:7171/user/";

  constructor(private http: HttpClient) {
   }
   signIn(user:any):Observable<string>{
    return this.http.post(this.url+"signIn",user,{responseType:'text'});
   }

   signUp(user:any):Observable<string>{
    return this.http.post(this.url+"signUp",user,{responseType:'text'});
   }
}
```

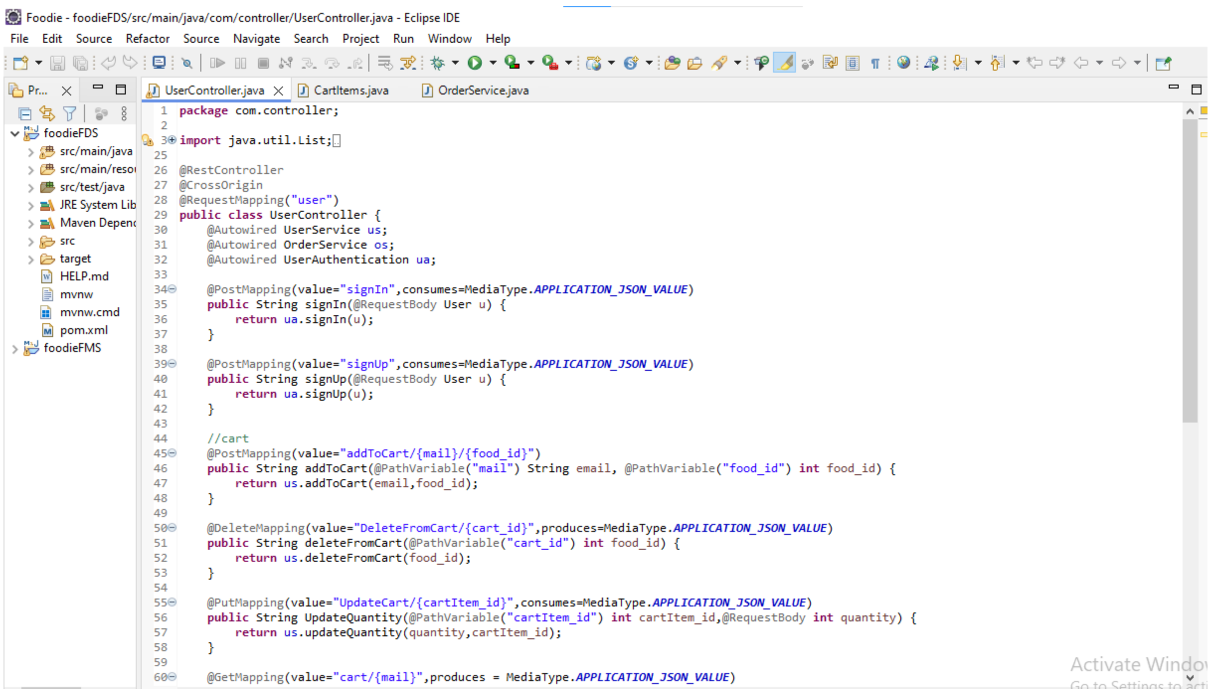
UserControlSrvice.ts:

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';
import { Food } from '../models/food';
import { Orders } from '../models/orders';
import { RestControlService } from './rest-control.service';
import { CartItems } from '../models/cart-items';

@Injectable({
  providedIn: 'root'
})
export class UserControlService {
  private url:String="http://localhost:7171/user/";
  private username:any="";

  constructor(private http: HttpClient,private restService:RestControlService)
{}

  setUser(username:any){
    this.username = username;
   }
  getUser():string{
      return this.username;
   }
  findAllFoods():Observable<Food[]>{
    return this.restService.findAllFoods();
   }
  addFoodToCart(food_id:any):Observable<String>{
    return
this.http.post(this.url+"addToCart/"+this.username+"/"+food_id,"",{responseTyp
e:'text'});
  }
  viewCart():Observable<CartItems[]>{
    return this.http.get<CartItems[]>(this.url+"cart/"+this.username);
  }
  updateQuantity(qty:any, cartItem_id:any):Observable<String>{
    return
this.http.put(this.url+"UpdateCart/"+cartItem_id,qty,{responseType:'text'});
  }
  deleteCart(cartItem_id:any):Observable<string>{
    return
this.http.delete(this.url+"DeleteFromCart/"+cartItem_id,{responseType:'text'})
;
  }
  orderHistory(){
    return this.http.get<Orders[]>(this.url+"orders/"+this.username);
  }
  placeOrder(quantity:any,cart_id:any){
    let orderRequest:any = {
      qty : quantity,
      user :  { emailid: this.username}
    }
    return
this.http.post(this.url+"placeOrder/"+cart_id,orderRequest,{responseType:'text
'});
  }
  cancelOrder(order_id:any){
    return
this.http.delete(this.url+"cancelOrders/"+order_id,{responseType:'text'});
  }
}
```

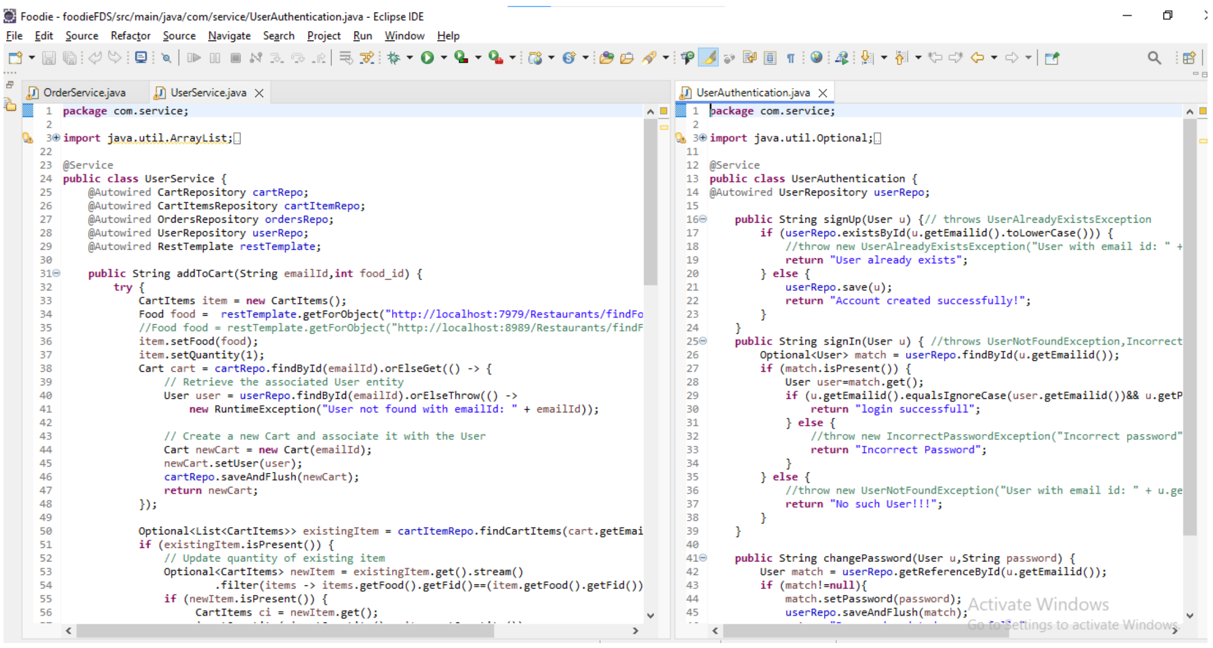
Backend:


CapstoneProject - FoodOrderSystemServer/src/main/java/com/FoodOrderSystemServerApplication.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Project Explorer
Servers
FoodOrderSystem
FoodOrderSystemManagement
FoodOrderSystemServer
src/main/java
com
FoodOrderSystemServerApplicatio
FoodOrderSystemServerApplica
main(String[]) : void
src/main/resources
src/test/java
JRE System Library [JavaSE-17]
Maven Dependencies
src
target
HELP.md
hs_err_pid14460.log
mvnw
mvnw.cmd
pom.xml
replay_pid14460.log
RestaurantManagementServer
FoodOrderSystemServerApplication.java

```
package com;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
@EnableEurekaServer
public class FoodOrderSystemServerApplication {
    public static void main(String[] args) {
        SpringApplication.run(FoodOrderSystemServerApplication.class, args);
        System.out.println("FOS Server up!");
    }
}
```


Foodie - foodieFDS/src/main/java/com/controller/UserController.java - Eclipse IDE

```java
package com.controller;

import java.util.List;

@RestController
@CrossOrigin
@RequestMapping("user")
public class UserController {
    @Autowired UserService us;
    @Autowired OrderService os;
    @Autowired UserAuthentication ua;

    @PostMapping(value="signIn",consumes=MediaType.APPLICATION_JSON_VALUE)
    public String signIn(@RequestBody User u) {
        return ua.signIn(u);
    }

    @PostMapping(value="signUp",consumes=MediaType.APPLICATION_JSON_VALUE)
    public String signUp(@RequestBody User u) {
        return ua.signUp(u);
    }

    //cart
    @PostMapping(value="addToCart/{mail}/{food_id}")
    public String addToCart(@PathVariable("mail") String email, @PathVariable("food_id") int food_id) {
        return us.addToCart(email,food_id);
    }

    @DeleteMapping(value="DeleteFromCart/{cart_id}",produces=MediaType.APPLICATION_JSON_VALUE)
    public String deleteFromCart(@PathVariable("cart_id") int food_id) {
        return us.deleteFromCart(food_id);
    }

    @PutMapping(value="UpdateCart/{cartItem_id}",consumes=MediaType.APPLICATION_JSON_VALUE)
    public String UpdateQuantity(@PathVariable("cartItem_id") int cartItem_id,@RequestBody int quantity) {
        return us.updateQuantity(quantity,cartItem_id);
    }

    @GetMapping(value="cart/{mail}",produces = MediaType.APPLICATION_JSON_VALUE)
```

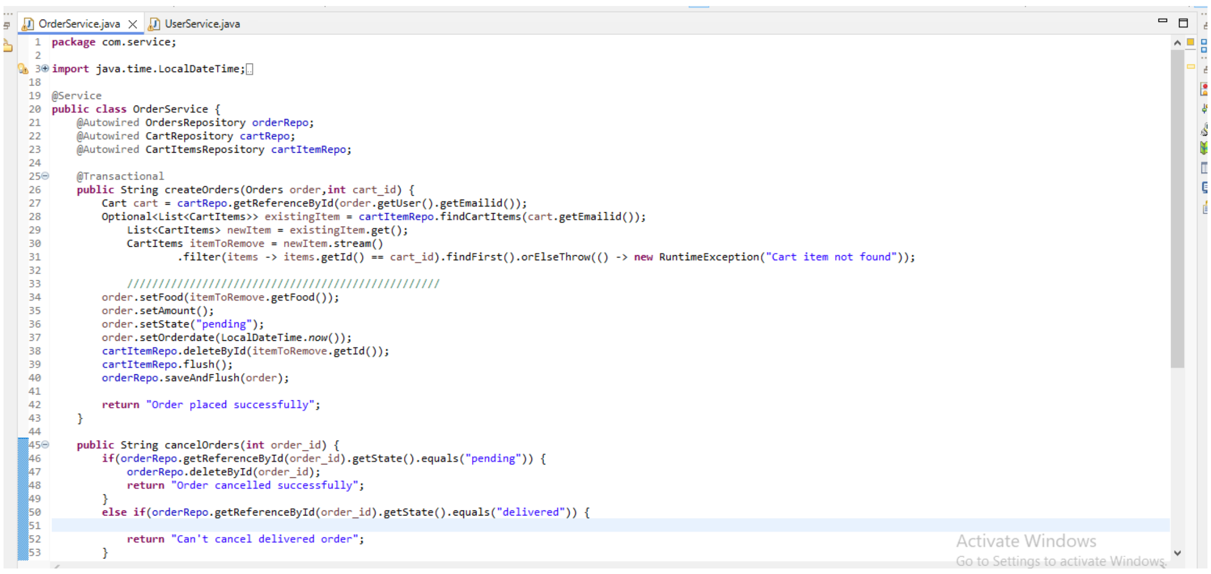

Foodie - foodieFDS/src/main/java/com/service/UserAuthentication.java - Eclipse IDE

UserService.java

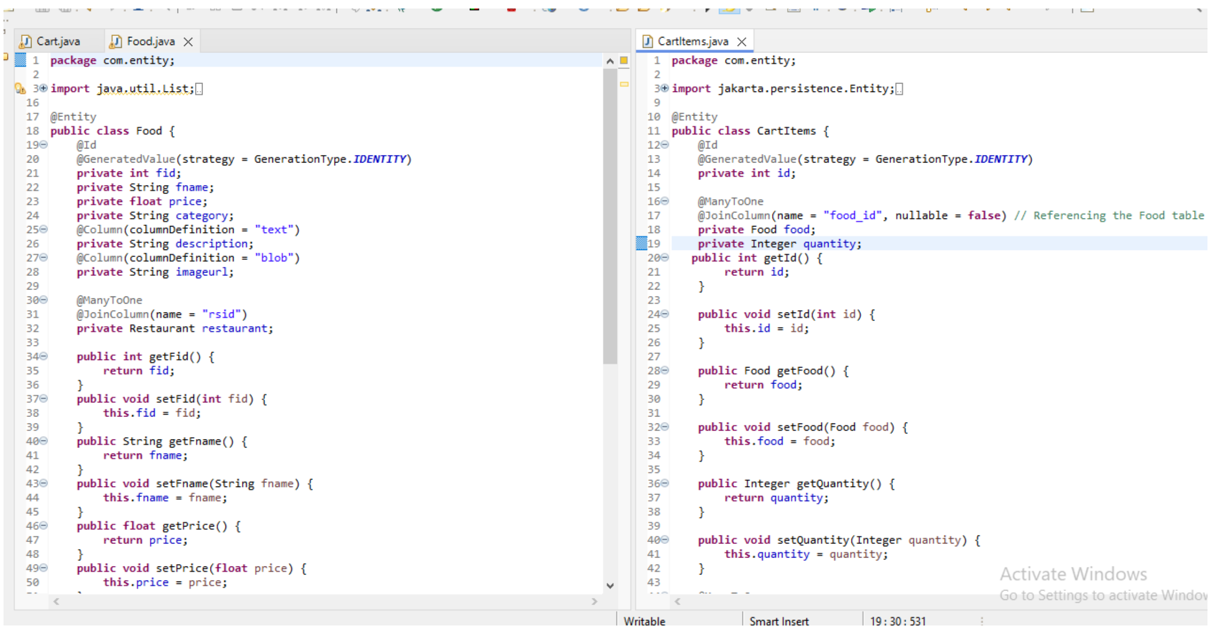
```java
package com.service;

import java.util.ArrayList;

@Service
public class UserService {
    @Autowired CartRepository cartRepo;
    @Autowired CartItemsRepository cartItemRepo;
    @Autowired OrdersRepository ordersRepo;
    @Autowired UserRepository userRepo;
    @Autowired RestTemplate restTemplate;

    public String addToCart(String emailId,int food_id) {
        try {
            CartItems item = new CartItems();
            Food food =  restTemplate.getForObject("http://localhost:7979/Restaurants/findFo
            //Food food = restTemplate.getForObject("http://localhost:8989/Restaurants/findF
            item.setFood(food);
            item.setQuantity(1);
            Cart cart = cartRepo.findById(emailId).orElseGet(() -> {
                // Retrieve the associated User entity
                User user = userRepo.findById(emailId).orElseThrow(() ->
                    new RuntimeException("User not found with emailId: " + emailId));

                // Create a new Cart and associate it with the User
                Cart newCart = new Cart(emailId);
                newCart.setUser(user);
                cartRepo.saveAndFlush(newCart);
                return newCart;
            });

            Optional<List<CartItems>> existingItem = cartItemRepo.findCartItems(cart.getEmai
            if (existingItem.isPresent()) {
                // Update quantity of existing item
                Optional<CartItems> newItem = existingItem.get().stream()
                        .filter(items -> items.getFood().getFid()==(item.getFood().getFid())
                if (newItem.isPresent()) {
                    CartItems ci = newItem.get();
```

UserAuthentication.java

```java
package com.service;

import java.util.Optional;

@Service
public class UserAuthentication {
@Autowired UserRepository userRepo;

    public String signUp(User u) {// throws UserAlreadyExistsException
        if (userRepo.existsById(u.getEmailid().toLowerCase())) {
            //throw new UserAlreadyExistsException("User with email id: " +
            return "User already exists";
        } else {
            userRepo.save(u);
            return "Account created successfully!";
        }
    }
    public String signIn(User u) { //throws UserNotFoundException,Incorrect
        Optional<User> match = userRepo.findById(u.getEmailid());
        if (match.isPresent()) {
            User user=match.get();
            if (u.getEmailid().equalsIgnoreCase(user.getEmailid())&& u.getP
                return "login successfull";
            } else {
                //throw new IncorrectPasswordException("Incorrect password"
                return "Incorrect Password";
            }
        } else {
            //throw new UserNotFoundException("User with email id: " + u.ge
            return "No such User!!!";
        }
    }

    public String changePassword(User u,String password) {
        User match = userRepo.getReferenceById(u.getEmailid());
        if (match!=null){
            match.setPassword(password);
            userRepo.saveAndFlush(match);
```

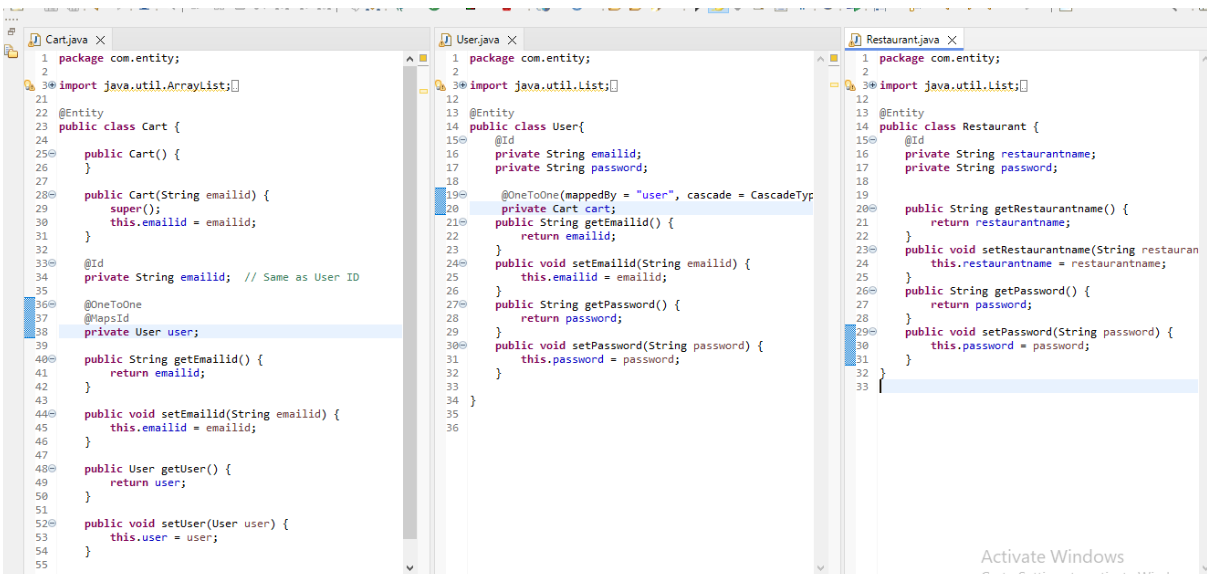


```java
package com.service;

import java.time.LocalDateTime;

@Service
public class OrderService {
    @Autowired OrdersRepository orderRepo;
    @Autowired CartRepository cartRepo;
    @Autowired CartItemsRepository cartItemRepo;

    @Transactional
    public String createOrders(Orders order,int cart_id) {
        Cart cart = cartRepo.getReferenceById(order.getUser().getEmailid());
        Optional<List<CartItems>> existingItem = cartItemRepo.findCartItems(cart.getEmailid());
            List<CartItems> newItem = existingItem.get();
            CartItems itemToRemove = newItem.stream()
                    .filter(items -> items.getId() == cart_id).findFirst().orElseThrow(() -> new RuntimeException("Cart item not found"));

            /////////////////////////////////////////////////
        order.setFood(itemToRemove.getFood());
        order.setAmount();
        order.setState("pending");
        order.setOrderdate(LocalDateTime.now());
        cartItemRepo.deleteById(itemToRemove.getId());
        cartItemRepo.flush();
        orderRepo.saveAndFlush(order);

        return "Order placed successfully";
    }

    public String cancelOrders(int order_id) {
        if(orderRepo.getReferenceById(order_id).getState().equals("pending")) {
            orderRepo.deleteById(order_id);
            return "Order cancelled successfully";
        }
        else if(orderRepo.getReferenceById(order_id).getState().equals("delivered")) {

            return "Can't cancel delivered order";
        }
```



```java
package com.entity;

import java.util.List;

@Entity
public class Food {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int fid;
    private String fname;
    private float price;
    private String category;
    @Column(columnDefinition = "text")
    private String description;
    @Column(columnDefinition = "blob")
    private String imageurl;

    @ManyToOne
    @JoinColumn(name = "rsid")
    private Restaurant restaurant;

    public int getFid() {
        return fid;
    }
    public void setFid(int fid) {
        this.fid = fid;
    }
    public String getFname() {
        return fname;
    }
    public void setFname(String fname) {
        this.fname = fname;
    }
    public float getPrice() {
        return price;
    }
    public void setPrice(float price) {
        this.price = price;
```

```java
package com.entity;

import jakarta.persistence.Entity;

@Entity
public class CartItems {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @ManyToOne
    @JoinColumn(name = "food_id", nullable = false) // Referencing the Food table
    private Food food;
    private Integer quantity;
   public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public Food getFood() {
        return food;
    }

    public void setFood(Food food) {
        this.food = food;
    }

    public Integer getQuantity() {
        return quantity;
    }

    public void setQuantity(Integer quantity) {
        this.quantity = quantity;
    }
```



Cart.java

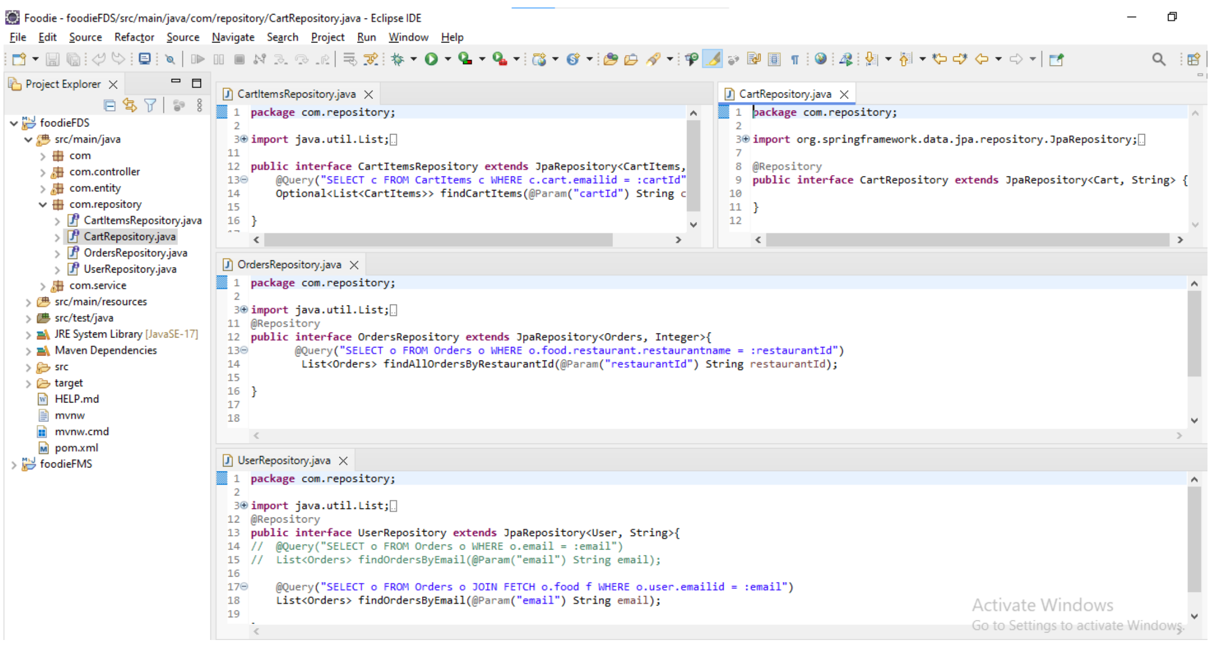
```java
package com.entity;

import java.util.ArrayList;

@Entity
public class Cart {

    public Cart() {
    }

    public Cart(String emailid) {
        super();
        this.emailid = emailid;
    }

    @Id
    private String emailid;  // Same as User ID

    @OneToOne
    @MapsId
    private User user;

    public String getEmailid() {
        return emailid;
    }

    public void setEmailid(String emailid) {
        this.emailid = emailid;
    }

    public User getUser() {
        return user;
    }

    public void setUser(User user) {
        this.user = user;
    }

```

User.java

```java
package com.entity;

import java.util.List;

@Entity
public class User{
    @Id
    private String emailid;
    private String password;

     @OneToOne(mappedBy = "user", cascade = CascadeTyp
     private Cart cart;
    public String getEmailid() {
        return emailid;
    }
    public void setEmailid(String emailid) {
        this.emailid = emailid;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }

}


```

Restaurant.java

```java
package com.entity;

import java.util.List;

@Entity
public class Restaurant {
    @Id
    private String restaurantname;
    private String password;


    public String getRestaurantname() {
        return restaurantname;
    }
    public void setRestaurantname(String restauran
        this.restaurantname = restaurantname;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
}

```



Foodie - foodieFDS/src/main/java/com/repository/CartRepository.java - Eclipse IDE

CartItemsRepository.java

```java
package com.repository;

import java.util.List;

public interface CartItemsRepository extends JpaRepository<CartItems,
    @Query("SELECT c FROM CartItems c WHERE c.cart.emailid = :cartId"
    Optional<List<CartItems>> findCartItems(@Param("cartId") String c

}
```

CartRepository.java

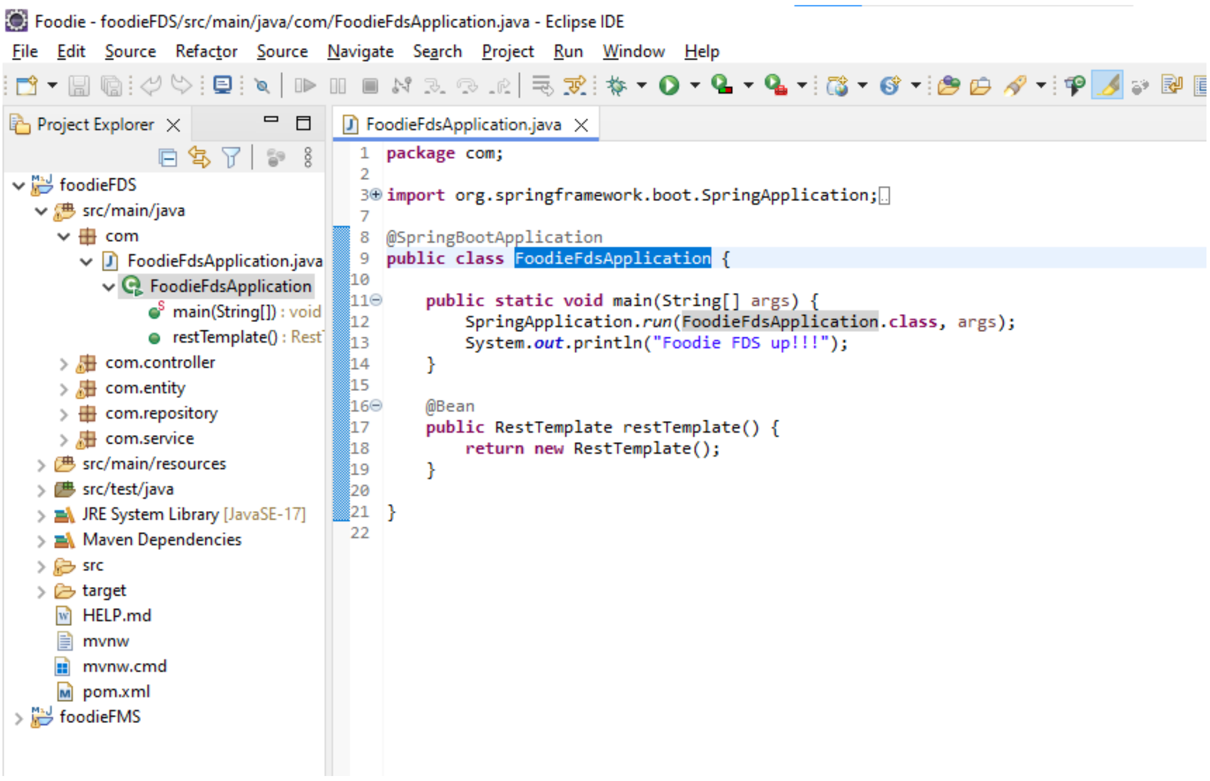
```java
package com.repository;

import org.springframework.data.jpa.repository.JpaRepository;

@Repository
public interface CartRepository extends JpaRepository<Cart, String> {

}

```

OrdersRepository.java

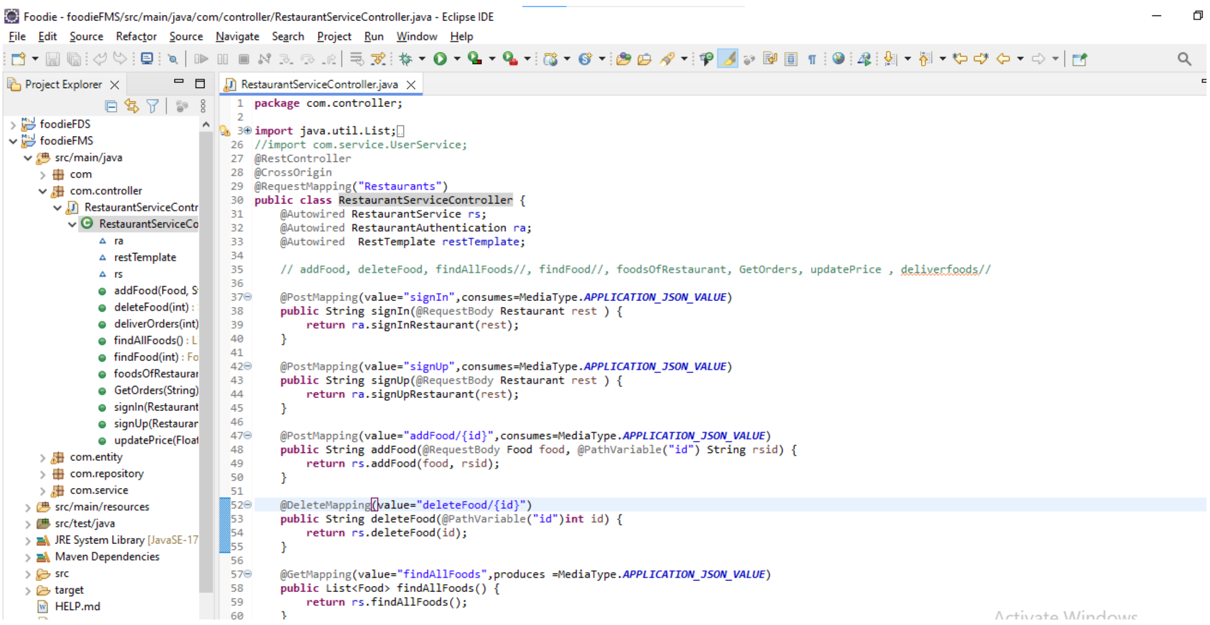
```java
package com.repository;

import java.util.List;
@Repository
public interface OrdersRepository extends JpaRepository<Orders, Integer>{
        @Query("SELECT o FROM Orders o WHERE o.food.restaurant.restaurantname = :restaurantId")
         List<Orders> findAllOrdersByRestaurantId(@Param("restaurantId") String restaurantId);

}


```

UserRepository.java

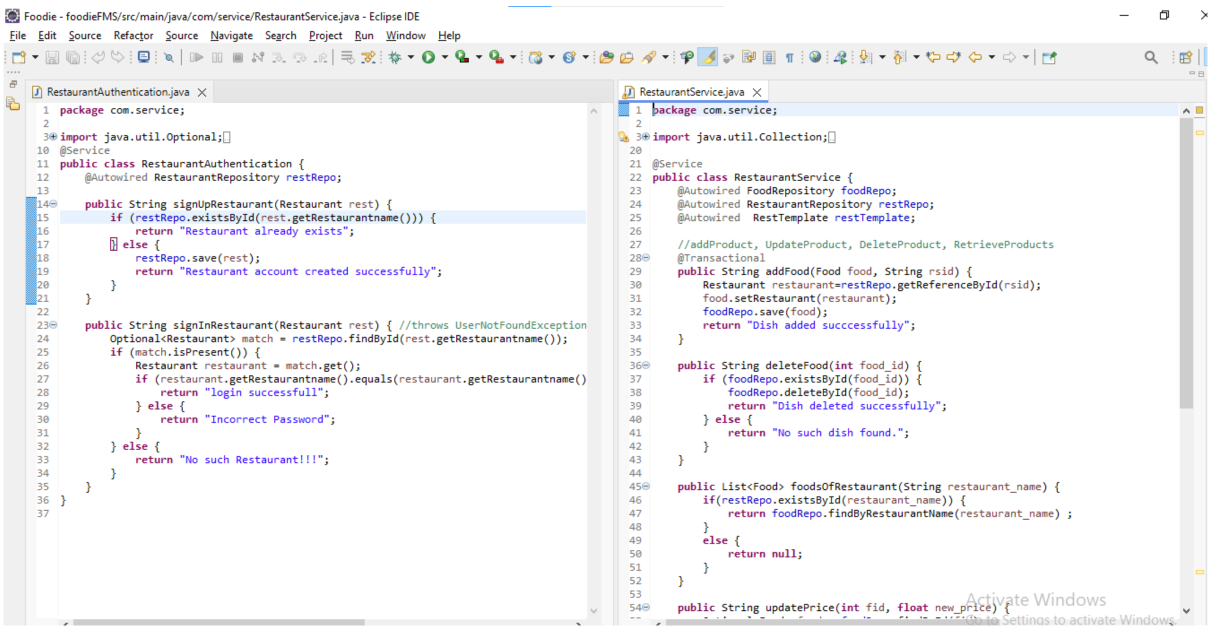
```java
package com.repository;

import java.util.List;
@Repository
public interface UserRepository extends JpaRepository<User, String>{
//  @Query("SELECT o FROM Orders o WHERE o.email = :email")
//  List<Orders> findOrdersByEmail(@Param("email") String email);

    @Query("SELECT o FROM Orders o JOIN FETCH o.food f WHERE o.user.emailid = :email")
    List<Orders> findOrdersByEmail(@Param("email") String email);

```



```java
package com;

import org.springframework.boot.SpringApplication;
@SpringBootApplication
public class FoodieFdsApplication {

    public static void main(String[] args) {
        SpringApplication.run(FoodieFdsApplication.class, args);
        System.out.println("Foodie FDS up!!!");
    }

    @Bean
    public RestTemplate restTemplate() {
        return new RestTemplate();
    }

}
```



```java
package com.controller;

import java.util.List;
//import com.service.UserService;
@RestController
@CrossOrigin
@RequestMapping("Restaurants")
public class RestaurantServiceController {
    @Autowired RestaurantService rs;
    @Autowired RestaurantAuthentication ra;
    @Autowired  RestTemplate restTemplate;

    // addFood, deleteFood, findAllFoods//, findFood//, foodsOfRestaurant, GetOrders, updatePrice , deliverfoods//

    @PostMapping(value="signIn",consumes=MediaType.APPLICATION_JSON_VALUE)
    public String signIn(@RequestBody Restaurant rest ) {
        return ra.signInRestaurant(rest);
    }

    @PostMapping(value="signUp",consumes=MediaType.APPLICATION_JSON_VALUE)
    public String signUp(@RequestBody Restaurant rest ) {
        return ra.signUpRestaurant(rest);
    }

    @PostMapping(value="addFood/{id}",consumes=MediaType.APPLICATION_JSON_VALUE)
    public String addFood(@RequestBody Food food, @PathVariable("id") String rsid) {
        return rs.addFood(food, rsid);
    }

    @DeleteMapping(value="deleteFood/{id}")
    public String deleteFood(@PathVariable("id")int id) {
        return rs.deleteFood(id);
    }

    @GetMapping(value="findAllFoods",produces =MediaType.APPLICATION_JSON_VALUE)
    public List<Food> findAllFoods() {
        return rs.findAllFoods();
    }
```



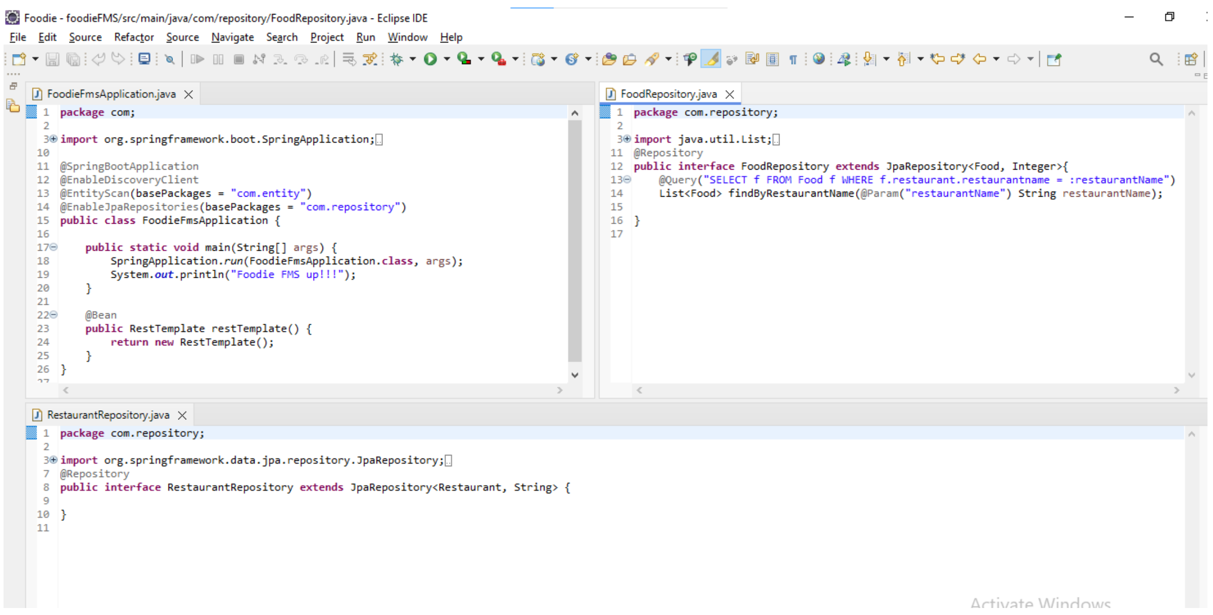
Foodie - foodieFMS/src/main/java/com/service/RestaurantService.java - Eclipse IDE

RestaurantAuthentication.java

```java
package com.service;

import java.util.Optional;
@Service
public class RestaurantAuthentication {
    @Autowired RestaurantRepository restRepo;

    public String signUpRestaurant(Restaurant rest) {
        if (restRepo.existsById(rest.getRestaurantname())) {
            return "Restaurant already exists";
        } else {
            restRepo.save(rest);
            return "Restaurant account created successfully";
        }
    }

    public String signInRestaurant(Restaurant rest) { //throws UserNotFoundException
        Optional<Restaurant> match = restRepo.findById(rest.getRestaurantname());
        if (match.isPresent()) {
            Restaurant restaurant = match.get();
            if (restaurant.getRestaurantname().equals(restaurant.getRestaurantname()
                return "login successfull";
            } else {
                return "Incorrect Password";
            }
        } else {
            return "No such Restaurant!!!";
        }
    }
}
```

RestaurantService.java

```java
package com.service;

import java.util.Collection;

@Service
public class RestaurantService {
    @Autowired FoodRepository foodRepo;
    @Autowired RestaurantRepository restRepo;
    @Autowired  RestTemplate restTemplate;

    //addProduct, UpdateProduct, DeleteProduct, RetrieveProducts
    @Transactional
    public String addFood(Food food, String rsid) {
        Restaurant restaurant=restRepo.getReferenceById(rsid);
        food.setRestaurant(restaurant);
        foodRepo.save(food);
        return "Dish added succcessfully";
    }

    public String deleteFood(int food_id) {
        if (foodRepo.existsById(food_id)) {
            foodRepo.deleteById(food_id);
            return "Dish deleted successfully";
        } else {
            return "No such dish found.";
        }
    }

    public List<Food> foodsOfRestaurant(String restaurant_name) {
        if(restRepo.existsById(restaurant_name)) {
            return foodRepo.findByRestaurantName(restaurant_name) ;
        }
        else {
            return null;
        }
    }

    public String updatePrice(int fid, float new_price) {
```



Foodie - foodieFMS/src/main/java/com/repository/FoodRepository.java - Eclipse IDE

FoodieFmsApplication.java

```java
package com;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
@EnableDiscoveryClient
@EntityScan(basePackages = "com.entity")
@EnableJpaRepositories(basePackages = "com.repository")
public class FoodieFmsApplication {

    public static void main(String[] args) {
        SpringApplication.run(FoodieFmsApplication.class, args);
        System.out.println("Foodie FMS up!!!");
    }

    @Bean
    public RestTemplate restTemplate() {
        return new RestTemplate();
    }
}
```

FoodRepository.java

```java
package com.repository;

import java.util.List;
@Repository
public interface FoodRepository extends JpaRepository<Food, Integer>{
    @Query("SELECT f FROM Food f WHERE f.restaurant.restaurantname = :restaurantName")
    List<Food> findByRestaurantName(@Param("restaurantName") String restaurantName);

}
```

RestaurantRepository.java

```java
package com.repository;

import org.springframework.data.jpa.repository.JpaRepository;
@Repository
public interface RestaurantRepository extends JpaRepository<Restaurant, String> {

}
```