Source Codes:

```
package com.KitchenStory.controller;
import com.KitchenStory.entity.FoodItem;
import com.KitchenStory.service.FoodItemService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

import java.util.List;

@Controller
public class HomeController {

    @Autowired
    private FoodItemService foodItemService;

    @GetMapping("/")
    public String home(Model model) {
        List<FoodItem> foodItems = foodItemService.getAllFoodItems();
        model.addAttribute("foodItems", foodItems);
        return "index";
    }

    @GetMapping("/products")
    public String products(Model model) {
        List<FoodItem> foodItems = foodItemService.getAllFoodItems();
        model.addAttribute("foodItems", foodItems);
        return "products";
    }
}
```

Login Controller

```
package com.KitchenStory.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;

@Controller
public class LoginController {

    @GetMapping("/login")
    public String login(@RequestParam(required = false) String error, Model
model) {
```

```
        if (error != null) {
            model.addAttribute("loginError", "Your username or password is
invalid.");
        }
        return "login";
    }
}
```

User Controller

```
package com.KitchenStory.controller;

import com.KitchenStory.entity.FoodItem;
import com.KitchenStory.entity.User;
import com.KitchenStory.service.FoodItemService;
import com.KitchenStory.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

import java.util.List;

@Controller
public class UserController {

    @Autowired
    private UserService userService;

    @Autowired
    private FoodItemService foodItemService;

    @Autowired
    private PasswordEncoder passwordEncoder;

    @GetMapping("/register")
    public String showRegistrationForm(Model model) {
        model.addAttribute("user", new User());
        return "register";
    }
```

```
    @PostMapping("/register")
    public String registerUser(@ModelAttribute("user") User user,
@ModelAttribute("confirmPassword") String confirmPassword, Model model) {
        if (!user.getPassword().equals(confirmPassword)) {
            model.addAttribute("error", "Passwords do not match");
            return "register";
        }
        user.setPassword(passwordEncoder.encode(user.getPassword()));
        user.setRole("USER"); // Set default role to USER
        userService.saveUser(user);
        return "redirect:/register?success";
    }

    @GetMapping("/user/dashboard")
    public String userDashboard(Model model) {
        Authentication authentication =
SecurityContextHolder.getContext().getAuthentication();
        String username = null;
        if (authentication != null && authentication.getPrincipal() instanceof
UserDetails) {
            username = ((UserDetails)
authentication.getPrincipal()).getUsername();
        }
        List<FoodItem> foodItems = foodItemService.getAllFoodItems();
        model.addAttribute("foodItems", foodItems);
        model.addAttribute("username", username);
        return "user-dashboard";
    }
}
```

AdminController

```
package com.KitchenStory.controller;

import com.KitchenStory.entity.FoodItem;
import com.KitchenStory.entity.User;
import com.KitchenStory.service.FoodItemService;
import com.KitchenStory.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
```

```java
import org.springframework.web.bind.annotation.*;
import org.springframework.web.multipart.MultipartFile;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.List;

@Controller
@RequestMapping("/admin")
public class AdminController {

    @Autowired
    private PasswordEncoder passwordEncoder;

    @Autowired
    private UserService userService;

    @Autowired
    private FoodItemService foodItemService;

    private static final String UPLOADED_FOLDER =
"src/main/resources/static/images/";

    @GetMapping("/dashboard")
    public String adminDashboard(Model model) {
        Authentication authentication =
SecurityContextHolder.getContext().getAuthentication();
        String username = null;
        if (authentication != null && authentication.getPrincipal() instanceof
UserDetails) {
            username = ((UserDetails)
authentication.getPrincipal()).getUsername();
        }
        model.addAttribute("username", username);

        List<FoodItem> foodItems = foodItemService.getAllFoodItems();
        model.addAttribute("foodItems", foodItems);

        List<User> users = userService.getAllUsers();
        model.addAttribute("users", users);

        return "admin-dashboard";
    }

    @PostMapping("/change-password")
```

```
    public String changePassword(@RequestParam("currentPassword") String
currentPassword,
                                 @RequestParam("newPassword") String
newPassword, Model model) {
        Authentication authentication =
SecurityContextHolder.getContext().getAuthentication();
        String username = null;
        if (authentication != null && authentication.getPrincipal() instanceof
UserDetails) {
            username = ((UserDetails)
authentication.getPrincipal()).getUsername();
        }
        User admin = userService.findByUsername(username);
        if (admin != null && passwordEncoder.matches(currentPassword,
admin.getPassword())) {
            admin.setPassword(passwordEncoder.encode(newPassword));
            userService.saveUser(admin);
            model.addAttribute("message", "Password changed successfully!");
        } else {
            model.addAttribute("message", "Current password is incorrect!");
        }
        return "redirect:/admin/dashboard";
    }

    @PostMapping("/add-item")
    public String addItem(@RequestParam("itemName") String itemName,
                          @RequestParam("itemPrice") Double itemPrice,
                          @RequestParam("itemImage") MultipartFile itemImage,
Model model) {
        if (itemImage.isEmpty()) {
            model.addAttribute("message", "Please select an image file to
upload.");
            return "redirect:/admin/dashboard";
        }

        try {
            // Save the file to the static directory
            byte[] bytes = itemImage.getBytes();
            Path path = Paths.get(UPLOADED_FOLDER +
itemImage.getOriginalFilename());
            Files.write(path, bytes);

            // Store the relative URL to the image in the database
            String imageUrl = "/images/" + itemImage.getOriginalFilename();
            FoodItem newItem = new FoodItem(itemName, itemPrice, imageUrl);
            foodItemService.saveFoodItem(newItem);

            model.addAttribute("message", "Item added successfully!");
```

```
        } catch (IOException e) {
            e.printStackTrace();
            model.addAttribute("message", "Failed to upload image.");
            return "redirect:/admin/dashboard";
        }

        return "redirect:/admin/dashboard";
    }

    @PostMapping("/delete-item")
    public String deleteItem(@RequestParam("itemId") Long itemId, Model model)
{
        foodItemService.deleteFoodItemById(itemId);
        model.addAttribute("message", "Item deleted successfully!");
        return "redirect:/admin/dashboard";
    }
}
```

Cart Controller

```
package com.KitchenStory.controller;

import com.KitchenStory.entity.Cart;
import com.KitchenStory.entity.CartItem;
import com.KitchenStory.entity.FoodItem;
import com.KitchenStory.entity.User;
import com.KitchenStory.service.CartService;
import com.KitchenStory.service.FoodItemService;
import com.KitchenStory.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;

import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.util.List;

@Controller
public class CartController {

    @Autowired
```

```
    private CartService cartService;

    @Autowired
    private FoodItemService foodItemService;

    @Autowired
    private UserService userService;

    @GetMapping("/cart")
    public String viewCart(Model model) {
        User user = getCurrentUser();
        Cart cart = cartService.getCartByUser(user);
        List<CartItem> items = cartService.getCartItems(cart);
        double totalValue = items.stream().mapToDouble(item ->
item.getQuantity() * item.getFoodItem().getPrice()).sum();
        model.addAttribute("cart", cart);
        model.addAttribute("items", items);
        model.addAttribute("totalValue", totalValue);
        return "cart";
    }

    @PostMapping("/cart/add")
    @ResponseBody
    public String addItemToCart(@RequestParam("itemId") Long itemId,
@RequestParam("quantity") int quantity) {
        User user = getCurrentUser();
        FoodItem foodItem = foodItemService.getFoodItemById(itemId);
        cartService.addItemToCart(user, foodItem, quantity);
        return "{\"status\": \"success\"}";
    }

    @PostMapping("/cart/remove")
    public String removeItemFromCart(@RequestParam("cartItemId") Long
cartItemId) {
        cartService.removeItemFromCart(cartItemId);
        return "redirect:/cart";
    }

    @GetMapping("/checkout")
    public String checkout(Model model) {
        User user = getCurrentUser();
        Cart cart = cartService.getCartByUser(user);
        List<CartItem> cartItems = cartService.getCartItems(cart);
        double totalValue = cartItems.stream().mapToDouble(item ->
item.getQuantity() * item.getFoodItem().getPrice()).sum();
        model.addAttribute("cartItems", cartItems);
        model.addAttribute("totalValue", totalValue);
        return "checkout";
```

```
    }

    @PostMapping("/order/confirm")
    public String confirmOrder(
            @RequestParam("fullName") String fullName,
            @RequestParam("email") String email,
            @RequestParam("phone") String phone,
            @RequestParam("address") String address,
            @RequestParam("city") String city,
            @RequestParam("state") String state,
            @RequestParam("zip") String zip,
            @RequestParam("total") double total,
            @RequestParam("paymentMethod") String paymentMethod,
            @RequestParam(value = "cardNumber", required = false) String
cardNumber,
            @RequestParam(value = "cardName", required = false) String
cardName,
            @RequestParam(value = "expiryDate", required = false) String
expiryDate,
            @RequestParam(value = "cvv", required = false) String cvv,
            Model model) {

        if ("cashOnDelivery".equals(paymentMethod)) {
            // Calculate expected delivery date
            LocalDate deliveryDate = LocalDate.now().plusDays(5);
            DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd MMMM
yyyy");

            // Add relevant data to the model for the success page
            model.addAttribute("fullName", fullName);
            model.addAttribute("address", address);
            model.addAttribute("total", total);
            model.addAttribute("deliveryDate",
deliveryDate.format(formatter));

            return "orderSuccess";
        }

        // Handle other payment methods
        // Add relevant data to the model for the payment page
        model.addAttribute("fullName", fullName);
        model.addAttribute("address", address);
        model.addAttribute("total", total);
        model.addAttribute("paymentMethod", paymentMethod);
        model.addAttribute("cardNumber", cardNumber);
        model.addAttribute("cardName", cardName);
        model.addAttribute("expiryDate", expiryDate);
        model.addAttribute("cvv", cvv);
```

```
        return "payment";
    }

    private User getCurrentUser() {
        Authentication authentication =
SecurityContextHolder.getContext().getAuthentication();
        String username = null;
        if (authentication != null && authentication.getPrincipal() instanceof
UserDetails) {
            username = ((UserDetails)
authentication.getPrincipal()).getUsername();
        }
        return userService.findByUsername(username);
    }
}
```

Cart Service

```
package com.KitchenStory.service;

import com.KitchenStory.entity.Cart;
import com.KitchenStory.entity.CartItem;
import com.KitchenStory.entity.FoodItem;
import com.KitchenStory.entity.User;
import com.KitchenStory.repository.CartItemRepository;
import com.KitchenStory.repository.CartRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import java.util.List;

@Service
public class CartService {

    @Autowired
    private CartRepository cartRepository;

    @Autowired
    private CartItemRepository cartItemRepository;

    public Cart getCartByUser(User user) {
        return cartRepository.findByUser(user).orElseGet(() -> {
            Cart cart = new Cart();
            cart.setUser(user);
```

```
            return cartRepository.save(cart);
        });
    }

    @Transactional
    public void addItemToCart(User user, FoodItem foodItem, int quantity) {
        Cart cart = getCartByUser(user);
        CartItem cartItem = cart.getItems().stream()
                .filter(item ->
item.getFoodItem().getId().equals(foodItem.getId()))
                .findFirst()
                .orElseGet(() -> {
                    CartItem newItem = new CartItem();
                    newItem.setCart(cart);
                    newItem.setFoodItem(foodItem);
                    newItem.setQuantity(0);
                    cart.getItems().add(newItem);
                    return newItem;
                });
        cartItem.setQuantity(cartItem.getQuantity() + quantity);
        cartItemRepository.save(cartItem);
    }

    @Transactional
    public void removeItemFromCart(Long cartItemId) {
        cartItemRepository.deleteById(cartItemId);
    }

    public List<CartItem> getCartItems(Cart cart) {
        return cartItemRepository.findByCart(cart);
    }
}
```

FoodItem Service

```
package com.KitchenStory.service;

import com.KitchenStory.entity.FoodItem;
import com.KitchenStory.repository.FoodItemRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.Optional;

@Service
```

```java
public class FoodItemService {

    @Autowired
    private FoodItemRepository foodItemRepository;

    public List<FoodItem> getAllFoodItems() {
        return foodItemRepository.findAll();
    }

    public void saveFoodItem(FoodItem foodItem) {
        foodItemRepository.save(foodItem);
    }

    public void deleteFoodItemById(Long id) {
        foodItemRepository.deleteById(id);
    }

    public FoodItem getFoodItemById(Long itemId) {
        Optional<FoodItem> optionalFoodItem =
foodItemRepository.findById(itemId);
        return optionalFoodItem.orElse(null);
    }
}
```

UserService

```java
package com.KitchenStory.service;

import com.KitchenStory.entity.User;
import com.KitchenStory.repository.UserRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import java.util.List;

@Service
public class UserService {

    @Autowired
    private UserRepository userRepository;

    @Autowired
    private PasswordEncoder passwordEncoder;
```

```java
    @Transactional
    public void createUser(User user) {
        user.setPassword(passwordEncoder.encode(user.getPassword()));
        userRepository.save(user);
    }

    public void saveUser(User user) {
        userRepository.save(user);
    }

    public User findByUsername(String username) {
        return userRepository.findByUsername(username);
    }

    public List<User> getAllUsers() {
        return userRepository.findAll();
    }
}
```

Thanks

Priyanshu Ranjan

Java Developer