**FOOD BOX**

**Application:**

package com.FoodBox.FoodBox;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@*SpringBootApplication*

public class FoodBoxApplication {

    public static *void* main(*String*[] *args*) {

        SpringApplication.run(FoodBoxApplication.class, args);

    }

}

**Admin Controller:**

package com.FoodBox.FoodBox.controller;

import javax.servlet.http.HttpSession;

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 org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestParam;

import com.FoodBox.FoodBox.model.Admin;

import com.FoodBox.FoodBox.service.AdminService;

@*Controller*

public class AdminController {

    @*Autowired*

    private *AdminService* adminService;

    @*PostMapping*("/verifyLogin")

    public *String* verifyLogin(@*RequestParam*(name="username") *String* *username*,@*RequestParam*(name="password") *String* *password*,*HttpSession* *session*,*Model* *model*) {

        if(!username.isEmpty() || !password.isEmpty()) {

            if(adminService.loginVerify(username,password)) {

                session.setAttribute("uname", username);

                return "admin\_Dashboard";

            }

            else {

                model.addAttribute("action","Username or password wrong");

                return "admin\_login";

            }

        }else {

            model.addAttribute("action", "Fields must not be empty");

            return "admin\_login";

        }

    }

    @*GetMapping*("/getDashboard")

    public *String* getDashboard() {

        return "admin\_Dashboard";

    }

    @*GetMapping*("/changePassword")

    public *String* changeAdminPassword(*HttpSession* *session*, *Model* *model*) {

*String* username=(String) session.getAttribute("uname");

*Admin* admin = adminService.getAdmin(username);

        model.addAttribute("admin", admin);

        return "change\_password";

    }

    @*PostMapping*("/updatePassword")

    public *String* updatePassword(@*RequestParam*(name="oldPassword") *String* *oldPassword*,@*RequestParam*(name="newPassword") *String* *newPassword*,*HttpSession* *session*,*Model* *model*) {

*String* username=(String) session.getAttribute("uname");

*Admin* admin = adminService.getAdmin(username);

        if(oldPassword.equals(admin.getPassword())) {

            admin.setPassword(newPassword);

            adminService.updatePassword(admin);

            model.addAttribute("action", "Password changed Successfully");

            return "admin\_Dashboard";

        }else {

            model.addAttribute("action", "Old Password not matching");

            return "change\_password";

        }

    }

    @*GetMapping*("/logout")

    public *String* adminLogout(*HttpSession* *session*) {

        session.invalidate();

        return "redirect:/";

    }

}

**Cart Controller:**

package com.FoodBox.FoodBox.controller;

//import java.sql.Date;

import java.util.List;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestParam;

import com.FoodBox.FoodBox.model.Cart;

import com.FoodBox.FoodBox.model.Customer;

import com.FoodBox.FoodBox.model.Product;

import com.FoodBox.FoodBox.model.Purchase;

import com.FoodBox.FoodBox.service.CartService;

import com.FoodBox.FoodBox.service.CustomerService;

import com.FoodBox.FoodBox.service.PurchaseService;

@*Controller*

public class CartController {

    @*Autowired*

    private *CartService* cartService;

    @*Autowired*

    private *CustomerService* customerService;

    @*Autowired*

    private *PurchaseService* purchaseService;

    @*ExceptionHandler*(Exception.class)

    public *String* handleSqlException(*Exception* *e*, *HttpSession* *session*) {

        session.setAttribute("action", "Choose Payment before Buying");

        return "redirect:/viewCart";

    }

    @*PostMapping*("/confirmCart")

    public *String* addToCart(@*RequestParam*("quantity") *int* *quantity*,*HttpSession* *session*) {

*Cart* cart = new Cart();

*Product* product = (Product) session.getAttribute("product");

*int* min=100;*int* max=999;*int* b = (*int*)(Math.random()\*(max-min+1)+min);

        cart.setId(b);

        cart.setProductId(product.getId());

        cart.setQuantity(quantity);

        cart.setPrice(product.getPrice()\*quantity);

        cartService.saveCart(cart);

        session.setAttribute("action", "Product added to cart");

*float* temp=0;

        if(session.getAttribute("sessionCost")==null) {

            temp=0;

        }else {

            temp=(*float*) session.getAttribute("sessionCost");

        }

*float* sessionCost=(cart.getPrice()+temp);

        session.setAttribute("sessionCost", sessionCost);

        return "redirect:/";

    }

    @*GetMapping*("/viewCart")

    public *String* viewCart(*Model* *model*,*HttpSession* *session*) {

*List*<*Cart*> cartList = cartService.getAllCart();

        if(!cartList.isEmpty()) {

        model.addAttribute("cartList", cartList);

        model.addAttribute("action", session.getAttribute("action"));

        session.setAttribute("action", null);

        return "viewCart";

        }else {

            session.setAttribute("action", "No products currently in Cart");

            return "redirect:/";

        }

    }

    @*PostMapping*("/buyNow")

    public *String* buyProducts(@*RequestParam*("pm") *String* *pm*, *HttpSession* *session*) {

        System.out.println(pm);

        if(pm.equals("yes")) {

*List*<*Cart*> cartList = cartService.getAllCart();

*Purchase* purchase = new Purchase();

*String* email = (String) session.getAttribute("customerLogin");

*Customer* customer = customerService.getCustomer(email);

            for(*Cart* cl:cartList) {

*java*.*sql*.*Date* date = new java.sql.Date(new java.util.Date().getTime());

*int* min=100000;*int* max=999999;*int* b = (*int*)(Math.random()\*(max-min+1)+min);

                purchase.setId(b);

                purchase.setDop(date);

                System.out.println(date);

                purchase.setCustomer(customer);

                purchase.setProductid(cl.getProductId());

                purchase.setQuantity(cl.getQuantity());

                purchase.setTotalcost(cl.getPrice());

                purchaseService.addPurchase(purchase);

            }

        session.setAttribute("action", "Products added to Customer Order List Sucessfully");

        return "redirect:/";

        }else {

            session.setAttribute("action", "Make Payment before to finilize orders");

            return "redirect:/viewCart";

        }

    }

}

**Customer Controller:**

package com.FoodBox.FoodBox.controller;

import java.sql.SQLException;

import java.util.List;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestParam;

import com.FoodBox.FoodBox.model.Customer;

import com.FoodBox.FoodBox.model.Purchase;

import com.FoodBox.FoodBox.service.CartService;

import com.FoodBox.FoodBox.service.CustomerService;

import com.FoodBox.FoodBox.service.PurchaseService;

@*Controller*

public class CustomerController {

    @*Autowired*

    private *CustomerService* customerService;

    @*Autowired*

    private *PurchaseService* purchaseService;

    @*Autowired*

    private *CartService* cartService;

    @*ExceptionHandler*(SQLException.class)

    public *String* handleSqlException(*SQLException* *e*, *HttpSession* *session*) {

        session.setAttribute("action", "User can't be deleted until their orders are deleted");

        return "redirect:/manageCustomer";

    }

    @*PostMapping*("/saveCustomer")

    public *String* saveCustomer(*Customer* *customer*, *Model* *model*, *HttpSession* *session*) {

*List*<*String*> cEmails = customerService.customerEmails();

*boolean* notExist = true;

        for(*String* e : cEmails) {

            if(customer.getEmail().equals(e))

                notExist=false;

        }

        if(notExist) {

            if (validate(customer.getEmail())) {

                customerService.saveCustomer(customer);

                model.addAttribute("action", "Added successfully, login to shop");

                session.setAttribute("customerLogin", customer.getEmail());

                session.setAttribute("custName", customer.getName());

                cartService.cartDeleteAll();

                return "redirect:/";

            } else {

                model.addAttribute("action", "Email pattern doesn't match");

                return "new\_customer";

            }

        }else {

            session.setAttribute("action", "Entered Email Already Exist please Login");

            return "redirect:/";

        }

    }

    @*PostMapping*("/verifyCustLogin")

    public *String* verifyLogin(@*RequestParam*(name = "email") *String* *email*,

            @*RequestParam*(name = "password") *String* *password*, *HttpSession* *session*, *Model* *model*) {

        if (!email.isEmpty() || !password.isEmpty()) {

            if (customerService.loginVerify(email, password)) {

                session.setAttribute("customerLogin", email);

*Customer* customer = customerService.getCustomer(email);

                session.setAttribute("custName", customer.getName());

                cartService.cartDeleteAll();

                return "redirect:/";

            } else {

                model.addAttribute("action", "email or password wrong");

                return "customer\_login";

            }

        } else {

            model.addAttribute("action", "Fields must not be empty");

            return "customer\_login";

        }

    }

    @*GetMapping*("/customerLogout")

    public *String* customerLogout(*HttpSession* *session*) {

        cartService.cartDeleteAll();

        session.invalidate();

        return "redirect:/";

    }

    @*GetMapping*("/manageCustomer")

    public *String* manageCustomer(*Model* *model*,*HttpSession* *session*) {

        model.addAttribute("action", session.getAttribute("action"));

        session.setAttribute("action", null);

        model.addAttribute("customers", customerService.getAllCustomers());

        return "manageCustomer";

    }

    @*GetMapping*("/deleteCustomer/{email}")

    public *String* deleteCustomer(@*PathVariable*(name = "email") *String* *email*, *Model* *model*) {

        customerService.deleteCustomer(email);

        model.addAttribute("action", "Customer Deleted Sucessfully");

        return "redirect:/manageCustomer";

    }

    @*GetMapping*("/customerOrders/{email}")

    public *String* customerOrders(@*PathVariable*(name = "email") *String* *email*, *Model* *model*,*HttpSession* *session*) {

*List*<*Purchase*> sPurchase = purchaseService.getByEmail(email);

        if(!sPurchase.isEmpty()) {

        model.addAttribute("sPurchase", sPurchase);

        return "customerPurchase";

        }else {

            session.setAttribute("action", "No Active Orders/Purchases by Customer");

            return "redirect:/manageCustomer";

        }

    }

    @*PostMapping*("/searchCustomer")

    public *String* searchCustomer(@*RequestParam*("keyword") *String* *keyword*,*Model* *model*) {

*List*<*Customer*> sCustomer = customerService.searchCustomer(keyword);

        if(sCustomer.isEmpty()) {

            model.addAttribute("action", "No Customer found");

            model.addAttribute("customers", customerService.getAllCustomers());

            return "manageCustomer";

        }else {

            model.addAttribute("searchHeading","Entered Catogery");

            model.addAttribute("sCustomer", sCustomer);

            return "searchCustomer";

        }

    }

    public static final *Pattern* VALID\_EMAIL\_ADDRESS\_REGEX = Pattern.compile("^[A-Z0-9.\_%+-]+@[A-Z0-9.-]+\\.[A-Z]{2,6}$",

            Pattern.CASE\_INSENSITIVE);

    public static *boolean* validate(*String* *emailStr*) {

*Matcher* matcher = VALID\_EMAIL\_ADDRESS\_REGEX.matcher(emailStr);

        return matcher.find();

    }

}

**Product Controller:**

package com.FoodBox.FoodBox.controller;

import javax.servlet.http.HttpSession;

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 org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import com.FoodBox.FoodBox.model.Product;

import com.FoodBox.FoodBox.service.ProductService;

@*Controller*

public class ProductController {

    @*Autowired*

    private *ProductService* productService;

    @*GetMapping*("/manageProduct")

    public *String* manageProduct(*Model* *model*) {

        model.addAttribute("products",productService.getAllProducts());

*Product* product = new Product();

        model.addAttribute("product", product);

        return "manageProduct";

    }

    @*PostMapping*("/addProduct")

    public *String* addProduct(@*ModelAttribute*("product") *Product* *product*, *Model* *model*, *HttpSession* *session*) {

*int* min=10000;*int* max=99999;*int* b = (*int*)(Math.random()\*(max-min+1)+min);

        product.setId(b);

        productService.addProduct(product);

        session.setAttribute("action","Product Added succesfully");

        model.addAttribute("product", product);

        return "redirect:/manageProduct";

    }

    @*GetMapping*("/showProductUpdate/{id}")

    public *String* showProductUpdate(@*PathVariable*(value="id") *int* *id*, *Model* *model*) {

*Product* product = productService.getProductById(id);

        model.addAttribute("product", product);

        return "update\_product";

    }

    @*PostMapping*("/updateProduct")

    public *String* updateProduct(@*ModelAttribute*("product") *Product* *product*, *Model* *model*,*HttpSession* *session*) {

        productService.addProduct(product);

        session.setAttribute("action","Product Updated succesfully");

        model.addAttribute("product", product);

        return "redirect:/manageProduct";

    }

    @*GetMapping*("/deleteProduct/{id}")

    public *String* deleteProduct(@*PathVariable*(value="id") *int* *id*,*Model* *model*,*HttpSession* *session*) {

        productService.deleteProduct(id);

        session.setAttribute("action", "Product Deleted Succesfully");

*Product* product = new Product();

        model.addAttribute("product", product);

        return "redirect:/manageProduct";

    }

}

**Purchase controller:**

package com.FoodBox.FoodBox.controller;

import java.text.SimpleDateFormat;

import java.sql.Date;

import java.util.List;

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 org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestParam;

import com.FoodBox.FoodBox.model.Purchase;

import com.FoodBox.FoodBox.service.PurchaseService;

@*Controller*

public class PurchaseController {

    @*Autowired*

    private *PurchaseService* purchaseService;

    @*GetMapping*("/managePurchase")

    public *String* managePurchase(*Model* *model*) {

        model.addAttribute("purchases", purchaseService.getAllPurchases());

        return "managePurchase";

    }

    @*PostMapping*("/searchPurchaseDate")

    public *String* searchPurchaseDate(@*RequestParam*("keyword") *String* *keyword*,*Model* *model*) {

*Date* date=null;

        try {

        //DateFormat parser = new SimpleDateFormat("yyyy-MM-dd");

        date = new Date(new SimpleDateFormat("yyyy-MM-dd").parse(keyword).getTime());

        }catch(*Exception* *e*) { System.out.println(e); }

*List*<*Purchase*> sPurchase = purchaseService.getPurchaseByDate(date);

        if(sPurchase.isEmpty()) {

            model.addAttribute("action", "No purchases on the selected date");

            model.addAttribute("purchases", purchaseService.getAllPurchases());

            return "managePurchase";

        }else {

            model.addAttribute("searchHeading","selected Date");

            model.addAttribute("sPurchase", sPurchase);

            return "searchPurchase";

        }

    }

    @*PostMapping*("/searchPurchaseCategory")

    public *String* searchPurchaseCategory(@*RequestParam*("keyword") *String* *keyword*,*Model* *model*) {

*List*<*Purchase*> sPurchase = purchaseService.getPurchaseByCategory(keyword);

        if(sPurchase.isEmpty()) {

            model.addAttribute("action", "No purchases on the Entered Category");

            model.addAttribute("purchases", purchaseService.getAllPurchases());

            return "managePurchase";

        }else {

            model.addAttribute("searchHeading","Entered Catogery");

            model.addAttribute("sPurchase", sPurchase);

            return "searchPurchase";

        }

    }

    @*GetMapping*("/deletePurchase/{id}")

    public *String* deletePurchase(@*PathVariable*("id") *int* *id*,*Model* *model*) {

        purchaseService.deletePurchase(id);

        model.addAttribute("action", "Purchase Deleted Succesfully");

        return "redirect:/managePurchase";

    }

}