SPORTY SHOES

*Phase-3 Assessment Project by Jeevanantham M*

*Email id: [jeevanantham.m3108@gmail.com](mailto:jeevanantham.m3108@gmail.com)*

**CODE:**

package com.ss;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Phase3ProjectApplication {

public static void main(String[] args) {

SpringApplication.run(Phase3ProjectApplication.class, args);

}

}

package com.ss.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.ss.model.Admin;

import com.ss.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 "adminDashboard";

}

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 "adminDashboard";

}

@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 "adminDashboard";

}else {

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

return "change\_password";

}

}

@GetMapping("/logout")

public String adminLogout(HttpSession session) {

session.invalidate();

return "redirect:/";

}

}

package com.ss.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.ss.model.Cart;

import com.ss.model.Customer;

import com.ss.model.Product;

import com.ss.model.Purchase;

import com.ss.service.CartService;

import com.ss.service.CustomerService;

import com.ss.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("size") float size,@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);

cart.setSize(size);

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";

}

}

}

package com.ss.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.ss.model.Customer;

import com.ss.model.Purchase;

import com.ss.service.CartService;

import com.ss.service.CustomerService;

import com.ss.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();

}

}

package com.ss.controller;

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.GetMapping;

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

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

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

import com.ss.model.Admin;

import com.ss.model.Cart;

import com.ss.model.Customer;

import com.ss.model.Product;

import com.ss.model.Purchase;

import com.ss.service.ProductService;

import com.ss.service.PurchaseService;

//import com.ss.service.PurchaseService;

@Controller

public class MainApp {

@Autowired

private ProductService productService;

@Autowired

private PurchaseService purchaseService;

@GetMapping("/")

public String viewHomePage(Model model,HttpSession session) {

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

session.setAttribute("action", null);

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

session.setAttribute("productList", productService.getAllProducts());

session.setAttribute("searchH", null);

}

return "home";

}

@GetMapping("/goHome")

public String goHome(Model model,HttpSession session) {

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

session.setAttribute("action", null);

session.setAttribute("productList", productService.getAllProducts());

session.setAttribute("searchH", null);

return "home";

}

@PostMapping("/searchHome")

public String searchHome(@RequestParam("keyword") String keyword,Model model,HttpSession session) {

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

session.setAttribute("action", null);

List<Product> productList = productService.homeSearch(keyword);

if(productList.isEmpty()) {

session.setAttribute("action", "Currently no products for searched");

session.setAttribute("productList", null);

return "redirect:/";

}

session.setAttribute("productList", productList);

session.setAttribute("searchH", "yes");

return "home";

}

@GetMapping("/register")

public String register(Model model) {

Customer customer = new Customer();

model.addAttribute("customer", customer);

return "new\_customer";

}

@GetMapping("/login")

public String customerLogin(Model model) {

Customer customer = new Customer();

model.addAttribute("customer", customer);

return "customer\_login";

}

@GetMapping("/adminLogin")

public String adminLogin(Model model) {

Admin admin = new Admin();

model.addAttribute("admin",admin);

return "admin\_login";

}

@GetMapping("/addCart/{id}")

public String selectProduct(@PathVariable("id") int id,HttpSession session,Model model) {

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

session.setAttribute("action", "Login or Register to start shopping");

return "redirect:/";

}else {

session.setAttribute("product", productService.getProductById(id));

Cart cart = new Cart();

model.addAttribute("cart", cart);

return "addCart";

}

}

@GetMapping("/viewOrders/{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 "ViewOrders";

}else {

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

return "redirect:/";

}

}

}

package com.ss.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.ss.model.Product;

import com.ss.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";

}

}

package com.ss.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.ss.model.Purchase;

import com.ss.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";

}

}

package com.ss.model;

import javax.persistence.Entity;

import javax.persistence.Id;

@Entity

public class Admin {

@Id

private String username;

private String password;

public Admin() {

super();

}

public Admin(String username, String password) {

super();

this.username = username;

this.password = password;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

package com.ss.model;

import javax.persistence.Entity;

import javax.persistence.Id;

@Entity

public class Cart {

@Id

private int id;

private int productId;

private int quantity;

private float size;

private float price;

public Cart() {

super();

}

public Cart(int id, int productId, int quantity, float size, float price) {

super();

this.id = id;

this.productId = productId;

this.quantity = quantity;

this.size = size;

this.price = price;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public int getProductId() {

return productId;

}

public void setProductId(int productId) {

this.productId = productId;

}

public int getQuantity() {

return quantity;

}

public void setQuantity(int quantity) {

this.quantity = quantity;

}

public float getSize() {

return size;

}

public void setSize(float size) {

this.size = size;

}

public float getPrice() {

return price;

}

public void setPrice(float price) {

this.price = price;

}

}

package com.ss.model;

import javax.persistence.Entity;

import javax.persistence.Id;

@Entity

public class Customer {

@Id

private String email;

private String name;

private String password;

private long contact;

public Customer() {

super();

}

public Customer(String email, String name, String password, long contact) {

super();

this.email = email;

this.name = name;

this.password = password;

this.contact = contact;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public long getContact() {

return contact;

}

public void setContact(long contact) {

this.contact = contact;

}

}

package com.ss.model;

import javax.persistence.Entity;

import javax.persistence.Id;

@Entity

public class Product {

@Id

private int id;

private String company;

private String name;

private float price;

private String category;

public Product(int id, String company, String name, float price, String category) {

super();

this.id = id;

this.company = company;

this.name = name;

this.price = price;

this.category = category;

}

public Product() {

super();

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getCompany() {

return company;

}

public void setCompany(String company) {

this.company = company;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public float getPrice() {

return price;

}

public void setPrice(float price) {

this.price = price;

}

public String getCategory() {

return category;

}

public void setCategory(String category) {

this.category = category;

}

}

package com.ss.model;

import java.sql.Date;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.OneToOne;

import com.ss.model.Customer;

@Entity

public class Purchase {

@Id

private int id;

private float size;

private Date dop;

private int quantity;

private float totalcost;

private int productid;

@OneToOne

private Customer customer;

public Purchase() {

super();

}

public Purchase(int id, float size, Date dop, int quantity, float totalcost, int productid, Customer customer) {

super();

this.id = id;

this.size = size;

this.dop = dop;

this.quantity = quantity;

this.totalcost = totalcost;

this.productid = productid;

this.customer = customer;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public float getSize() {

return size;

}

public void setSize(float size) {

this.size = size;

}

public Date getDop() {

return dop;

}

public void setDop(Date dop) {

this.dop = dop;

}

public int getQuantity() {

return quantity;

}

public void setQuantity(int quantity) {

this.quantity = quantity;

}

public float getTotalcost() {

return totalcost;

}

public void setTotalcost(float totalcost) {

this.totalcost = totalcost;

}

public int getProductid() {

return productid;

}

public void setProductid(int productid) {

this.productid = productid;

}

public Customer getCustomer() {

return customer;

}

public void setCustomer(Customer customer) {

this.customer = customer;

}

}

package com.ss.repository;

import java.util.List;

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

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

import com.ss.model.Admin;

public interface AdminRepository extends JpaRepository<Admin,String>{

Admin findByUsername(String username);

@Query(value="SELECT username FROM Admin",nativeQuery=true)

public List<String> findUsenames();

}

package com.ss.repository;

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

import com.ss.model.Cart;

public interface CartRepository extends JpaRepository<Cart, Integer>{

}

package com.ss.repository;

import java.util.List;

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

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

import com.ss.model.Customer;

public interface CustomerRepository extends JpaRepository<Customer, String> {

Customer findByEmail(String email);

@Query("SELECT c FROM Customer c WHERE c.email LIKE %?1%"

+" OR c.name LIKE %?1%"

+" OR c.contact LIKE %?1%")

public List<Customer> userSearch(String name);

@Query("SELECT c.email from Customer c")

public List<String> customerEmails();

}

package com.ss.repository;

import java.util.List;

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

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

import com.ss.model.Product;

public interface ProductRepository extends JpaRepository<Product, Integer> {

Product findById(int id);

@Query("Select id from Product where category Like %?1%")

List<Integer> getByCategory(String keyword);

@Query("SELECT p FROM Product p WHERE p.company LIKE %?1%"

+" OR p.name LIKE %?1%"

+" OR p.price LIKE %?1%"

+" OR p.category LIKE %?1%")

public List<Product> homeSearch(String keyword);

}

package com.ss.repository;

import java.sql.Date;

import java.util.List;

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

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

import com.ss.model.Purchase;

public interface PurchaseRepository extends JpaRepository<Purchase, Integer>{

@Query("SELECT p FROM Purchase p WHERE p.customer.email LIKE %?1%")

public List<Purchase> getByEmail(String email);

public List<Purchase> findByDop(Date dop);

public List<Purchase> findByproductid(int id);

}

package com.ss.service;

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

import org.springframework.stereotype.Service;

import com.ss.model.Admin;

import com.ss.repository.AdminRepository;

@Service

public class AdminService {

@Autowired

private AdminRepository adminRepository;

public Admin getAdmin(String username) {

return adminRepository.findByUsername(username);

}

public boolean loginVerify(String username, String password) {

Admin admin = adminRepository.findByUsername(username);

if (admin!= null && admin.getUsername().equals(username) && admin.getPassword().equals(password)) {

return true;

}

return false;

}

public void updatePassword(Admin admin) {

adminRepository.save(admin);

}

}

package com.ss.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.ss.model.Cart;

import com.ss.repository.CartRepository;

@Service

public class CartService {

@Autowired

private CartRepository cartRepository;

public void saveCart(Cart cart) {

cartRepository.save(cart);

}

public List<Cart> getAllCart() {

return cartRepository.findAll();

}

public void cartDeleteAll() {

cartRepository.deleteAll();

}

}

package com.ss.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.ss.model.Customer;

import com.ss.repository.CustomerRepository;

@Service

public class CustomerService {

@Autowired

private CustomerRepository customerRepository;

public void saveCustomer(Customer customer) {

this.customerRepository.save(customer);

}

public boolean loginVerify(String email, String password) {

Customer customer = customerRepository.findByEmail(email);

if (customer!= null && customer.getEmail().equals(email) && customer.getPassword().equals(password)) {

return true;

}

return false;

}

public Customer getCustomer(String email) {

return customerRepository.findByEmail(email);

}

public List<Customer> getAllCustomers() {

return customerRepository.findAll();

}

public void deleteCustomer(String email) {

customerRepository.deleteById(email);

}

public List<Customer> searchCustomer(String keyword) {

return customerRepository.userSearch(keyword);

}

public List<String> customerEmails(){

return customerRepository.customerEmails();

}

}

**package** com.ss.service;

**public** **class** MainAppService {

}

package com.ss.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.ss.model.Product;

import com.ss.repository.ProductRepository;

@Service

public class ProductService {

@Autowired

private ProductRepository productRepository;

public List<Product> getAllProducts(){

return productRepository.findAll();

}

public void addProduct(Product product) {

productRepository.save(product);

}

public Product getProductById(int id) {

return productRepository.findById(id);

}

public void deleteProduct(int id) {

productRepository.deleteById(id);

}

public List<Product> homeSearch(String keyword) {

return productRepository.homeSearch(keyword);

}

}

package com.ss.service;

import java.sql.Date;

import java.util.ArrayList;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.ss.model.Purchase;

import com.ss.repository.ProductRepository;

import com.ss.repository.PurchaseRepository;

@Service

public class PurchaseService {

@Autowired

private PurchaseRepository purchaseRepository;

@Autowired

private ProductRepository productRepository;

public List<Purchase> getAllPurchases(){

return purchaseRepository.findAll();

}

public List<Purchase> getByEmail(String email){

return purchaseRepository.getByEmail(email);

}

public List<Purchase> getPurchaseByDate(Date keyword) {

return purchaseRepository.findByDop(keyword);

}

public List<Purchase> getPurchaseByCategory(String keyword) {

List<Purchase> sPurchase = new ArrayList<>();

List<Integer> productIds = productRepository.getByCategory(keyword);

if (!productIds.isEmpty()) {

for (int id : productIds) {

List<Purchase> tempList = purchaseRepository.findByproductid(id);

if (!tempList.isEmpty()) {

for (Purchase p : tempList) {

sPurchase.add(p);

}

}

}

}

return sPurchase;

}

public void deletePurchase(int id) {

purchaseRepository.deleteById(id);

}

public void addPurchase(Purchase purchase) {

purchaseRepository.save(purchase);

}

}