**WORKSHOP 2**

Xây dựng hệ thống quản lý sàn thương mại điện tử mini

Mục tiêu:

Xây dựng một ứng dụng web quản lý sàn TMĐT theo mô hình MVC2 sử dụng:

* Servlet (Jakarta)
* JSP (có JSTL)
* JDBC kết nối SQL Server
* Phân quyền người dùng (Admin, Seller, Buyer,...)
* Chức năng CRUD + Tìm kiếm + Phân trang

1. Các vai trò người dùng

| Mã vai trò | Vai trò |
| --- | --- |
| AD | Quản trị hệ thống |
| SE | Người bán |
| BU | Người mua |
| MK | Marketing |
| AC | Kế toán |
| DL | Giao hàng |
| CS | Chăm sóc khách hàng |

2. Các bảng dữ liệu chính

Tên cơ sở dữ liệu: ECommerceDB

| Bảng | Mô tả |
| --- | --- |
| tblUsers(userID, fullName, roleID, password, phone) | Người dùng |
| tblCategories(categoryID, categoryName, description) | Ngành hàng |
| tblProducts(productID, name, categoryID, price, quantity, sellerID, status) | Sản phẩm |
| tblPromotions(promoID, name, discountPercent, startDate, endDate, status) | Khuyến mãi |
| tblCarts(cartID, userID, createdDate) | Giỏ hàng |
| tblCartDetails(cartID, productID, quantity) | Chi tiết giỏ hàng |
| tblInvoices(invoiceID, userID, totalAmount, status, createdDate) | Hóa đơn |
| tblInvoiceDetails(invoiceID, productID, quantity, price) | Chi tiết hóa đơn |
| tblDeliveries(deliveryID, invoiceID, address, deliveryDate, status) | Giao hàng |
| tblReturns(returnID, invoiceID, reason, status) | Trả hàng |
| tblCustomerCares(ticketID, userID, subject, content, status, reply) | Phản hồi khách hàng |

3. Yêu cầu chức năng

3. Các chức năng yêu cầu

3.1. Quản lý người dùng

* Đăng nhập / Đăng xuất.
* Thêm mới, chỉnh sửa, xóa người dùng (Admin).
* Tìm kiếm theo userID, fullName, roleID.
* Session & kiểm tra quyền.

3.2. Quản lý ngành hàng (Admin)

* CRUD danh mục ngành hàng (categoryName, description).
* Tìm kiếm theo tên.

3.3. Quản lý sản phẩm

* Seller: thêm/sửa/xóa sản phẩm của mình.
* Admin: xem tất cả sản phẩm.
* Tìm kiếm theo tên, lọc theo category, status, giá.
* Kiểm tra số lượng và trạng thái (active/inactive).

3.4. Quản lý khuyến mãi (Marketing)

* CRUD chương trình khuyến mãi.
* Gắn sản phẩm với chương trình.
* Tự động hiển thị sản phẩm đang giảm giá.

3.5. Mua hàng và hóa đơn (Buyer)

* Thêm vào giỏ hàng.
* Tạo đơn từ giỏ hàng → tạo hóa đơn + chi tiết.
* Xem danh sách hóa đơn đã mua.
* Hủy đơn nếu chưa giao.

3.6. Giao hàng (Delivery)

* Cập nhật trạng thái giao hàng (pending, delivering, delivered).
* Tìm theo invoiceID hoặc trạng thái.

3.7. Trả hàng (Buyer + Admin)

* Người mua gửi yêu cầu trả hàng (reason, status).
* Admin phê duyệt hoặc từ chối.

3.8. Chăm sóc khách hàng (CS)

* Buyer tạo phản hồi (subject, content).
* CS phản hồi lại (reply) và cập nhật trạng thái (new, answered, closed).

Thư viện sử dụng

Chỉ dùng 3 thư viện sau, đặt vào thư mục lib/ và add vào project (chuột phải > Properties > Libraries > Add JAR/Folder)

1. jakarta.servlet.jsp.jstl-api-2.0.0.jar
2. jakarta.servlet.jsp.jstl-2.0.0.jar
3. sqljdbc4.jar

**4. Kiến trúc triển khai**

4.1. MVC2

* MainController: xử lý login, logout, điều hướng.
* UserController, ProductController, CategoryController, InvoiceController, v.v.: xử lý theo module.
* DAO/DTO cho từng bảng.

4.2. Thư mục

ECommerceManagement/

├── src/

│ ├── java/

│ │ ├── controller/ (Main + phụ)

│ │ ├── dao/

│ │ ├── dto/

│ │ └── util/ (DBUtil.java)

│ └── webapp/

│ ├── login.jsp, welcome.jsp

│ ├── userList.jsp, addUser.jsp, ...

│ └── WEB-INF/web.xml

4.3. web.xml

<servlet>

<servlet-name>MainController</servlet-name>

<servlet-class>controller.MainController</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>MainController</servlet-name>

<url-pattern>/MainController</url-pattern>

</servlet-mapping>

TẠO FILE SQL – ECommerceDB.sql

CREATE DATABASE ECommerceDB;

USE ECommerceDB;

-- 1. Bảng người dùng

CREATE TABLE tblUsers (

userID VARCHAR(20) PRIMARY KEY,

fullName NVARCHAR(100),

roleID VARCHAR(5),

password VARCHAR(100),

phone VARCHAR(15)

);

-- 2. Ngành hàng

CREATE TABLE tblCategories (

categoryID INT PRIMARY KEY AUTO\_INCREMENT,

categoryName NVARCHAR(100),

description NVARCHAR(255)

);

-- 3. Sản phẩm

CREATE TABLE tblProducts (

productID INT PRIMARY KEY AUTO\_INCREMENT,

name NVARCHAR(100),

categoryID INT,

price FLOAT,

quantity INT,

sellerID VARCHAR(20),

status VARCHAR(20),

FOREIGN KEY (categoryID) REFERENCES tblCategories(categoryID),

FOREIGN KEY (sellerID) REFERENCES tblUsers(userID)

);

-- 4. Chương trình khuyến mãi

CREATE TABLE tblPromotions (

promoID INT PRIMARY KEY AUTO\_INCREMENT,

name NVARCHAR(100),

discountPercent FLOAT,

startDate DATE,

endDate DATE,

status VARCHAR(20)

);

-- 5. Giỏ hàng

CREATE TABLE tblCarts (

cartID INT PRIMARY KEY AUTO\_INCREMENT,

userID VARCHAR(20),

createdDate DATE,

FOREIGN KEY (userID) REFERENCES tblUsers(userID)

);

CREATE TABLE tblCartDetails (

cartID INT,

productID INT,

quantity INT,

PRIMARY KEY (cartID, productID),

FOREIGN KEY (cartID) REFERENCES tblCarts(cartID),

FOREIGN KEY (productID) REFERENCES tblProducts(productID)

);

-- 6. Hóa đơn

CREATE TABLE tblInvoices (

invoiceID INT PRIMARY KEY AUTO\_INCREMENT,

userID VARCHAR(20),

totalAmount FLOAT,

status VARCHAR(20),

createdDate DATE,

FOREIGN KEY (userID) REFERENCES tblUsers(userID)

);

CREATE TABLE tblInvoiceDetails (

invoiceID INT,

productID INT,

quantity INT,

price FLOAT,

PRIMARY KEY (invoiceID, productID),

FOREIGN KEY (invoiceID) REFERENCES tblInvoices(invoiceID),

FOREIGN KEY (productID) REFERENCES tblProducts(productID)

);

-- 7. Giao hàng

CREATE TABLE tblDeliveries (

deliveryID INT PRIMARY KEY AUTO\_INCREMENT,

invoiceID INT,

address NVARCHAR(255),

deliveryDate DATE,

status VARCHAR(50),

FOREIGN KEY (invoiceID) REFERENCES tblInvoices(invoiceID)

);

-- 8. Trả hàng

CREATE TABLE tblReturns (

returnID INT PRIMARY KEY AUTO\_INCREMENT,

invoiceID INT,

reason NVARCHAR(255),

status VARCHAR(50),

FOREIGN KEY (invoiceID) REFERENCES tblInvoices(invoiceID)

);

-- 9. Chăm sóc khách hàng

CREATE TABLE tblCustomerCares (

ticketID INT PRIMARY KEY AUTO\_INCREMENT,

userID VARCHAR(20),

subject NVARCHAR(100),

content TEXT,

status VARCHAR(50),

reply TEXT,

FOREIGN KEY (userID) REFERENCES tblUsers(userID)

);

**Cài đặt DBUtil.java để kết nối SQL Server**

package util;

import java.sql.Connection;

import java.sql.DriverManager;

public class DBUtil {

public static Connection getConnection() throws Exception {

String url = "jdbc:sqlserver://localhost:1433;databaseName=ECommerceDB;encrypt=true;trustServerCertificate=true";

String user = "sa";

String password = "12345";

Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");

return DriverManager.getConnection(url, user, password);

}

}

// MainController.java - Điều phối chính gọi các controller phụ

package controller;

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.\*;

import java.io.IOException;

@WebServlet("/MainController")

public class MainController extends HttpServlet {

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

private void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

String action = request.getParameter("action");

String controller = "login.jsp"; // default page

try {

if (action == null) {

controller = "login.jsp";

} else if (action.equals("Login")) {

controller = "UserController?action=Login";

} else if (action.equals("Logout")) {

controller = "UserController?action=Logout";

} else if (action.equals("AddUser")) {

controller = "UserController?action=AddUser";

} else if (action.equals("ListUsers")) {

controller = "UserController?action=ListUsers";

} else if (action.equals("UpdateUser")) {

controller = "UserController?action=UpdateUser";

} else if (action.equals("DeleteUser")) {

controller = "UserController?action=DeleteUser";

} else if (action.equals("AddProduct")) {

controller = "ProductController?action=AddProduct";

} else if (action.equals("ListProducts")) {

controller = "ProductController?action=ListProducts";

} else if (action.equals("UpdateProduct")) {

controller = "ProductController?action=UpdateProduct";

} else if (action.equals("DeleteProduct")) {

controller = "ProductController?action=DeleteProduct";

} else if (action.equals("AddCategory")) {

controller = "CategoryController?action=AddCategory";

} else if (action.equals("ListCategories")) {

controller = "CategoryController?action=ListCategories";

} else if (action.equals("AddPromotion")) {

controller = "PromotionController?action=AddPromotion";

} else if (action.equals("ViewCart")) {

controller = "CartController?action=ViewCart";

} else if (action.equals("Checkout")) {

controller = "InvoiceController?action=Checkout";

} else if (action.equals("TrackDelivery")) {

controller = "DeliveryController?action=TrackDelivery";

} else if (action.equals("SubmitReturn")) {

controller = "ReturnController?action=SubmitReturn";

} else if (action.equals("CustomerSupport")) {

controller = "CustomerCareController?action=CustomerSupport";

}

request.getRequestDispatcher(controller).forward(request, response);

} catch (Exception e) {

request.setAttribute("ERROR", "Something went wrong: " + e.getMessage());

request.getRequestDispatcher("error.jsp").forward(request, response);

}

}

}

**UserController.java**

Xử lý các action như: Login, Logout, AddUser, ListUsers, DeleteUser, UpdateUser.

@WebServlet("/UserController")

public class UserController extends HttpServlet {

protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {

String action = req.getParameter("action");

if ("Login".equals(action)) {

// Chuyển sang doPost hoặc redirect form

} else if ("Logout".equals(action)) {

req.getSession().invalidate();

res.sendRedirect("login.jsp");

} else if ("ListUsers".equals(action)) {

List<UserDTO> list = UserDAO.getAllUsers();

req.setAttribute("USER\_LIST", list);

req.getRequestDispatcher("userList.jsp").forward(req, res);

}

}

protected void doPost(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {

String action = req.getParameter("action");

if ("Login".equals(action)) {

String userID = req.getParameter("userID");

String password = req.getParameter("password");

UserDTO user = UserDAO.checkLogin(userID, password);

if (user != null) {

req.getSession().setAttribute("LOGIN\_USER", user);

res.sendRedirect("welcome.jsp");

} else {

req.setAttribute("ERROR", "Sai tài khoản hoặc mật khẩu.");

req.getRequestDispatcher("login.jsp").forward(req, res);

}

} else if ("AddUser".equals(action)) {

String userID = req.getParameter("userID");

String fullName = req.getParameter("fullName");

String roleID = req.getParameter("roleID");

String password = req.getParameter("password");

String phone = req.getParameter("phone");

UserDTO user = new UserDTO(userID, fullName, roleID, password, phone);

boolean ok = UserDAO.insert(user);

if (ok) res.sendRedirect("MainController?action=ListUsers");

else {

req.setAttribute("ERROR", "Thêm user thất bại");

req.getRequestDispatcher("addUser.jsp").forward(req, res);

}

}

}

}

**7. Viết ProductController.java tương tự**

* doGet để hiển thị danh sách sản phẩm, lọc theo danh mục, status
* doPost để thêm hoặc cập nhật sản phẩm

**8. Các file JSP chính**

* login.jsp: Gửi form login về MainController với action=Login
* welcome.jsp: Hiển thị thông tin người đăng nhập từ session
* addUser.jsp, userList.jsp: Quản lý người dùng

**9. Tạo các controller khác (PromotionController, InvoiceController, CartController...)**

Tùy vào chức năng, bạn chỉ cần theo mẫu của UserController và ProductController, kết hợp:

* doGet để đọc dữ liệu (view/list/search)
* doPost để xử lý form (create/update/delete)

**Cấu hình web.xml**

<web-app xmlns="https://jakarta.ee/xml/ns/jakartaee"

version="5.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee https://jakarta.ee/xml/ns/jakartaee/web-app\_5\_0.xsd">

<servlet>

<servlet-name>MainController</servlet-name>

<servlet-class>controller.MainController</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>MainController</servlet-name>

<url-pattern>/MainController</url-pattern>

</servlet-mapping>

<welcome-file-list>

<welcome-file>login.jsp</welcome-file>

</welcome-file-list>

</web-app>

// ProductController.java

package controller;

import dao.ProductDAO;

import dto.ProductDTO;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.\*;

import jakarta.servlet.ServletException;

import java.io.IOException;

import java.util.List;

@WebServlet("/ProductController")

public class ProductController extends HttpServlet {

protected void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("AddProduct".equals(action)) {

String name = req.getParameter("name");

int categoryID = Integer.parseInt(req.getParameter("categoryID"));

double price = Double.parseDouble(req.getParameter("price"));

int quantity = Integer.parseInt(req.getParameter("quantity"));

String sellerID = req.getParameter("sellerID");

String status = req.getParameter("status");

ProductDTO product = new ProductDTO(0, name, categoryID, price, quantity, sellerID, status);

boolean success = ProductDAO.insert(product);

if (success) {

res.sendRedirect("MainController?action=ListProducts");

} else {

req.setAttribute("ERROR", "Không thể thêm sản phẩm.");

req.getRequestDispatcher("addProduct.jsp").forward(req, res);

}

}

}

protected void doGet(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("ListProducts".equals(action)) {

List<ProductDTO> list = ProductDAO.getAll();

req.setAttribute("PRODUCT\_LIST", list);

req.getRequestDispatcher("productList.jsp").forward(req, res);

}

}

}

// PromotionController.java

package controller;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.\*;

import jakarta.servlet.ServletException;

import java.io.IOException;

@WebServlet("/PromotionController")

public class PromotionController extends HttpServlet {

protected void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("AddPromotion".equals(action)) {

// TODO: thêm khuyến mãi mới

}

}

}

// InvoiceController.java

package controller;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.\*;

import jakarta.servlet.ServletException;

import java.io.IOException;

@WebServlet("/InvoiceController")

public class InvoiceController extends HttpServlet {

protected void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("Checkout".equals(action)) {

// TODO: xử lý tạo hóa đơn từ giỏ hàng

}

}

}

// CartController.java

package controller;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.\*;

import jakarta.servlet.ServletException;

import java.io.IOException;

@WebServlet("/CartController")

public class CartController extends HttpServlet {

protected void doGet(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("ViewCart".equals(action)) {

// TODO: hiển thị giỏ hàng người dùng

}

}

}

// DeliveryController.java

package controller;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.\*;

import jakarta.servlet.ServletException;

import java.io.IOException;

@WebServlet("/DeliveryController")

public class DeliveryController extends HttpServlet {

protected void doGet(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("TrackDelivery".equals(action)) {

// TODO: theo dõi đơn giao hàng theo mã hóa đơn

}

}

}

// ReturnController.java

package controller;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.\*;

import jakarta.servlet.ServletException;

import java.io.IOException;

@WebServlet("/ReturnController")

public class ReturnController extends HttpServlet {

protected void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("SubmitReturn".equals(action)) {

// TODO: xử lý yêu cầu trả hàng từ người mua

}

}

}

**Thêm sản phẩm (AddProduct)**

**a. ProductDTO.java**

Vị trí: src/java/dto/ProductDTO.java

package dto;

public class ProductDTO {

private int productID;

private String name;

private int categoryID;

private double price;

private int quantity;

private String sellerID;

private String status;

public ProductDTO() {}

public ProductDTO(int productID, String name, int categoryID, double price, int quantity, String sellerID, String status) {

this.productID = productID;

this.name = name;

this.categoryID = categoryID;

this.price = price;

this.quantity = quantity;

this.sellerID = sellerID;

this.status = status;

}

// Getters & Setters

public int getProductID() { return productID; }

public void setProductID(int productID) { this.productID = productID; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

public int getCategoryID() { return categoryID; }

public void setCategoryID(int categoryID) { this.categoryID = categoryID; }

public double getPrice() { return price; }

public void setPrice(double price) { this.price = price; }

public int getQuantity() { return quantity; }

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

public String getSellerID() { return sellerID; }

public void setSellerID(String sellerID) { this.sellerID = sellerID; }

public String getStatus() { return status; }

public void setStatus(String status) { this.status = status; }

}

**b. ProductDAO.java**

Vị trí: src/java/dao/ProductDAO.java

package dao;

import dto.ProductDTO;

import util.DBUtil;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class ProductDAO {

public static boolean insert(ProductDTO product) {

String sql = "INSERT INTO tblProducts(name, categoryID, price, quantity, sellerID, status) VALUES (?, ?, ?, ?, ?, ?)";

try (Connection con = DBUtil.getConnection();

PreparedStatement ps = con.prepareStatement(sql)) {

ps.setString(1, product.getName());

ps.setInt(2, product.getCategoryID());

ps.setDouble(3, product.getPrice());

ps.setInt(4, product.getQuantity());

ps.setString(5, product.getSellerID());

ps.setString(6, product.getStatus());

return ps.executeUpdate() > 0;

} catch (Exception e) {

e.printStackTrace();

}

return false;

}

public static List<ProductDTO> getAll() {

List<ProductDTO> list = new ArrayList<>();

String sql = "SELECT \* FROM tblProducts";

try (Connection con = DBUtil.getConnection();

PreparedStatement ps = con.prepareStatement(sql);

ResultSet rs = ps.executeQuery()) {

while (rs.next()) {

ProductDTO p = new ProductDTO(

rs.getInt("productID"),

rs.getString("name"),

rs.getInt("categoryID"),

rs.getDouble("price"),

rs.getInt("quantity"),

rs.getString("sellerID"),

rs.getString("status")

);

list.add(p);

}

} catch (Exception e) {

e.printStackTrace();

}

return list;

}

}

**c. ProductController.java (action: AddProduct, ListProducts)**

Cập nhật phần xử lý doPost và doGet:

protected void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("AddProduct".equals(action)) {

String name = req.getParameter("name");

int categoryID = Integer.parseInt(req.getParameter("categoryID"));

double price = Double.parseDouble(req.getParameter("price"));

int quantity = Integer.parseInt(req.getParameter("quantity"));

String sellerID = req.getParameter("sellerID");

String status = req.getParameter("status");

ProductDTO product = new ProductDTO(0, name, categoryID, price, quantity, sellerID, status);

boolean success = ProductDAO.insert(product);

if (success) {

res.sendRedirect("MainController?action=ListProducts");

} else {

req.setAttribute("ERROR", "Không thể thêm sản phẩm.");

req.getRequestDispatcher("addProduct.jsp").forward(req, res);

}

}

}

protected void doGet(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("ListProducts".equals(action)) {

req.setAttribute("PRODUCT\_LIST", ProductDAO.getAll());

req.getRequestDispatcher("productList.jsp").forward(req, res);

}

}

**d. addProduct.jsp**

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<html>

<head><title>Thêm sản phẩm</title></head>

<body>

<h2>Thêm sản phẩm</h2>

<form action="MainController" method="post">

<input type="hidden" name="action" value="AddProduct"/>

Tên sản phẩm: <input type="text" name="name"/><br/>

Mã ngành hàng: <input type="number" name="categoryID"/><br/>

Giá: <input type="number" step="0.01" name="price"/><br/>

Số lượng: <input type="number" name="quantity"/><br/>

Mã người bán: <input type="text" name="sellerID"/><br/>

Trạng thái: <select name="status">

<option value="active">Active</option>

<option value="inactive">Inactive</option>

</select><br/>

<input type="submit" value="Thêm"/>

</form>

<p style="color:red">${requestScope.ERROR}</p>

</body>

</html>

**e. productList.jsp**

<%@ page import="java.util.\*, dto.ProductDTO" %>

<%

List<ProductDTO> list = (List<ProductDTO>) request.getAttribute("PRODUCT\_LIST");

%>

<html>

<head><title>Danh sách sản phẩm</title></head>

<body>

<h2>Danh sách sản phẩm</h2>

<table border="1">

<tr>

<th>ID</th><th>Tên</th><th>Giá</th><th>Số lượng</th><th>Người bán</th><th>Trạng thái</th>

</tr>

<%

for (ProductDTO p : list) {

%>

<tr>

<td><%= p.getProductID() %></td>

<td><%= p.getName() %></td>

<td><%= p.getPrice() %></td>

<td><%= p.getQuantity() %></td>

<td><%= p.getSellerID() %></td>

<td><%= p.getStatus() %></td>

</tr>

<% } %>

</table>

</body>

</html>

**Cart – CartDTO, CartDAO, viewCart.jsp**

**a. CartDTO.java**

package dto;

public class CartDTO {

private int cartID;

private String userID;

private String productName;

private int productID;

private int quantity;

private double price;

public CartDTO() {}

public CartDTO(int cartID, String userID, int productID, String productName, int quantity, double price) {

this.cartID = cartID;

this.userID = userID;

this.productID = productID;

this.productName = productName;

this.quantity = quantity;

this.price = price;

}

public int getCartID() { return cartID; }

public String getUserID() { return userID; }

public int getProductID() { return productID; }

public String getProductName() { return productName; }

public int getQuantity() { return quantity; }

public double getPrice() { return price; }

public double getTotal() {

return quantity \* price;

}

}

**b. CartDAO.java**

package dao;

import dto.CartDTO;

import util.DBUtil;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class CartDAO {

public static List<CartDTO> getCartByUser(String userID) {

List<CartDTO> list = new ArrayList<>();

String sql = "SELECT c.cartID, c.userID, cd.productID, p.name, cd.quantity, p.price " +

"FROM tblCarts c " +

"JOIN tblCartDetails cd ON c.cartID = cd.cartID " +

"JOIN tblProducts p ON cd.productID = p.productID " +

"WHERE c.userID = ?";

try (Connection con = DBUtil.getConnection();

PreparedStatement ps = con.prepareStatement(sql)) {

ps.setString(1, userID);

ResultSet rs = ps.executeQuery();

while (rs.next()) {

CartDTO dto = new CartDTO(

rs.getInt("cartID"),

rs.getString("userID"),

rs.getInt("productID"),

rs.getString("name"),

rs.getInt("quantity"),

rs.getDouble("price")

);

list.add(dto);

}

} catch (Exception e) {

e.printStackTrace();

}

return list;

}

}

**c. viewCart.jsp**

<%@ page import="java.util.\*, dto.CartDTO" %>

<%

List<CartDTO> cart = (List<CartDTO>) request.getAttribute("CART\_LIST");

double total = 0;

%>

<html>

<head><title>Giỏ hàng của bạn</title></head>

<body>

<h2>Giỏ hàng</h2>

<table border="1">

<tr>

<th>Sản phẩm</th><th>Giá</th><th>Số lượng</th><th>Thành tiền</th>

</tr>

<%

for (CartDTO item : cart) {

double itemTotal = item.getTotal();

total += itemTotal;

%>

<tr>

<td><%= item.getProductName() %></td>

<td><%= item.getPrice() %></td>

<td><%= item.getQuantity() %></td>

<td><%= itemTotal %></td>

</tr>

<% } %>

<tr>

<td colspan="3">Tổng cộng:</td>

<td><%= total %></td>

</tr>

</table>

<form action="MainController" method="post">

<input type="hidden" name="action" value="Checkout"/>

<input type="submit" value="Thanh toán"/>

</form>

</body>

</html>

**2. Invoice – InvoiceDTO, InvoiceDAO, xử lý Checkout**

**a. InvoiceDTO.java**

package dto;

public class InvoiceDTO {

private int invoiceID;

private String userID;

private double totalAmount;

private String status;

private String createdDate;

public InvoiceDTO() {}

public InvoiceDTO(int invoiceID, String userID, double totalAmount, String status, String createdDate) {

this.invoiceID = invoiceID;

this.userID = userID;

this.totalAmount = totalAmount;

this.status = status;

this.createdDate = createdDate;

}

public int getInvoiceID() { return invoiceID; }

public String getUserID() { return userID; }

public double getTotalAmount() { return totalAmount; }

public String getStatus() { return status; }

public String getCreatedDate() { return createdDate; }

}

**b. InvoiceDAO.java**

package dao;

import dto.CartDTO;

import util.DBUtil;

import java.sql.\*;

import java.util.List;

public class InvoiceDAO {

public static boolean checkout(String userID, List<CartDTO> cartItems) {

Connection con = null;

PreparedStatement ps1 = null, ps2 = null, ps3 = null;

ResultSet rs = null;

try {

con = DBUtil.getConnection();

con.setAutoCommit(false);

double total = 0;

for (CartDTO c : cartItems) total += c.getTotal();

String sql1 = "INSERT INTO tblInvoices(userID, totalAmount, status, createdDate) VALUES (?, ?, 'Pending', GETDATE())";

ps1 = con.prepareStatement(sql1, Statement.RETURN\_GENERATED\_KEYS);

ps1.setString(1, userID);

ps1.setDouble(2, total);

ps1.executeUpdate();

rs = ps1.getGeneratedKeys();

int invoiceID = 0;

if (rs.next()) {

invoiceID = rs.getInt(1);

}

String sql2 = "INSERT INTO tblInvoiceDetails(invoiceID, productID, quantity, price) VALUES (?, ?, ?, ?)";

ps2 = con.prepareStatement(sql2);

for (CartDTO item : cartItems) {

ps2.setInt(1, invoiceID);

ps2.setInt(2, item.getProductID());

ps2.setInt(3, item.getQuantity());

ps2.setDouble(4, item.getPrice());

ps2.addBatch();

}

ps2.executeBatch();

con.commit();

return true;

} catch (Exception e) {

try { if (con != null) con.rollback(); } catch (Exception ex) {}

e.printStackTrace();

} finally {

try { if (con != null) con.setAutoCommit(true); } catch (Exception ex) {}

}

return false;

}

}

**c. InvoiceController.java (Checkout logic)**

protected void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("Checkout".equals(action)) {

UserDTO user = (UserDTO) req.getSession().getAttribute("LOGIN\_USER");

List<CartDTO> cartItems = CartDAO.getCartByUser(user.getUserID());

boolean success = InvoiceDAO.checkout(user.getUserID(), cartItems);

if (success) {

res.sendRedirect("thankyou.jsp");

} else {

req.setAttribute("ERROR", "Không thể thanh toán!");

req.getRequestDispatcher("viewCart.jsp").forward(req, res);

}

}

}

**3. Promotion – Giao diện và DAO**

**a. addPromotion.jsp**

<html>

<head><title>Thêm khuyến mãi</title></head>

<body>

<h2>Thêm chương trình khuyến mãi</h2>

<form action="MainController" method="post">

<input type="hidden" name="action" value="AddPromotion"/>

Tên chương trình: <input type="text" name="name"/><br/>

% giảm giá: <input type="number" name="discountPercent" step="0.1"/><br/>

Ngày bắt đầu: <input type="date" name="startDate"/><br/>

Ngày kết thúc: <input type="date" name="endDate"/><br/>

Trạng thái: <select name="status">

<option value="active">Active</option>

<option value="inactive">Inactive</option>

</select><br/>

<input type="submit" value="Tạo khuyến mãi"/>

</form>

</body>

</html>

**b. PromotionDAO.java**

package dao;

import util.DBUtil;

import java.sql.\*;

public class PromotionDAO {

public static boolean insert(String name, double percent, String start, String end, String status) {

String sql = "INSERT INTO tblPromotions(name, discountPercent, startDate, endDate, status) VALUES (?, ?, ?, ?, ?)";

try (Connection con = DBUtil.getConnection();

PreparedStatement ps = con.prepareStatement(sql)) {

ps.setString(1, name);

ps.setDouble(2, percent);

ps.setString(3, start);

ps.setString(4, end);

ps.setString(5, status);

return ps.executeUpdate() > 0;

} catch (Exception e) {

e.printStackTrace();

}

return false;

}

}

**1. DeliveryDTO.java**

Vị trí: src/java/dto/DeliveryDTO.java

package dto;

public class DeliveryDTO {

private int deliveryID;

private int invoiceID;

private String address;

private String deliveryDate;

private String status;

public DeliveryDTO() {}

public DeliveryDTO(int deliveryID, int invoiceID, String address, String deliveryDate, String status) {

this.deliveryID = deliveryID;

this.invoiceID = invoiceID;

this.address = address;

this.deliveryDate = deliveryDate;

this.status = status;

}

public int getDeliveryID() {

return deliveryID;

}

public int getInvoiceID() {

return invoiceID;

}

public String getAddress() {

return address;

}

public String getDeliveryDate() {

return deliveryDate;

}

public String getStatus() {

return status;

}

}

**2. DeliveryDAO.java**

Vị trí: src/java/dao/DeliveryDAO.java

package dao;

import dto.DeliveryDTO;

import util.DBUtil;

import java.sql.\*;

public class DeliveryDAO {

public static DeliveryDTO getByInvoiceID(int invoiceID) {

String sql = "SELECT deliveryID, invoiceID, address, deliveryDate, status FROM tblDeliveries WHERE invoiceID = ?";

try (Connection con = DBUtil.getConnection();

PreparedStatement ps = con.prepareStatement(sql)) {

ps.setInt(1, invoiceID);

ResultSet rs = ps.executeQuery();

if (rs.next()) {

return new DeliveryDTO(

rs.getInt("deliveryID"),

rs.getInt("invoiceID"),

rs.getString("address"),

rs.getString("deliveryDate"),

rs.getString("status")

);

}

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

}

**3. ReturnDTO.java**

Vị trí: src/java/dto/ReturnDTO.java

package dto;

public class ReturnDTO {

private int returnID;

private int invoiceID;

private String reason;

private String status;

public ReturnDTO() {}

public ReturnDTO(int returnID, int invoiceID, String reason, String status) {

this.returnID = returnID;

this.invoiceID = invoiceID;

this.reason = reason;

this.status = status;

}

public int getReturnID() {

return returnID;

}

public int getInvoiceID() {

return invoiceID;

}

public String getReason() {

return reason;

}

public String getStatus() {

return status;

}

}

**4. ReturnDAO.java**

Vị trí: src/java/dao/ReturnDAO.java

package dao;

import util.DBUtil;

import java.sql.\*;

public class ReturnDAO {

public static boolean insert(int invoiceID, String reason) {

String sql = "INSERT INTO tblReturns(invoiceID, reason, status) VALUES (?, ?, 'Pending')";

try (Connection con = DBUtil.getConnection();

PreparedStatement ps = con.prepareStatement(sql)) {

ps.setInt(1, invoiceID);

ps.setString(2, reason);

return ps.executeUpdate() > 0;

} catch (Exception e) {

e.printStackTrace();

}

return false;

}

}

**Gợi ý sử dụng trong ReturnController**

protected void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

String action = req.getParameter("action");

if ("SubmitReturn".equals(action)) {

int invoiceID = Integer.parseInt(req.getParameter("invoiceID"));

String reason = req.getParameter("reason");

boolean success = ReturnDAO.insert(invoiceID, reason);

if (success) {

res.sendRedirect("thankyou.jsp");

} else {

req.setAttribute("ERROR", "Không thể gửi yêu cầu trả hàng.");

req.getRequestDispatcher("returnForm.jsp").forward(req, res);

}

}

}