
جامعة الدول العربية
المنظمة العربية للعلوم والتكنولوجيا والنقل البحري

📚 Online Shopping System - Project Documentation

Team

Mohsmed Tarek Fayesz 231004433

Youssef samaha

1. Introduction

The Online Shopping System is a web-based e-commerce application designed to allow users to browse products, add items to a shopping cart, and place orders. The system is built using a Java EE technology stack, demonstrating core principles of web application development and database interaction.

1.1 Project Goal

To develop a functional, user-friendly platform that simulates the core functionality of a modern e-commerce website.

1.2 Technology Stack

Category	Technology	Purpose
Backend Logic	Java (Servlets)	Handles all business logic, request processing, and data persistence.
Presentation (View)	JSP (JavaServer Pages)	Generates dynamic HTML content for the user interface.
Database	MySQL	Stores all system data (Users, Products, Orders, etc.).
Web Server	Apache Tomcat	Runtime environment for the Java web application.

Category	Technology	Purpose
Frontend	HTML, CSS, JavaScript, Bootstrap, jQuery	Provides a responsive, interactive, and visually appealing user interface.

2. System Requirements

2.1 Software Prerequisites

To run and develop this project, the following software must be installed:

- **Java Development Kit (JDK):** Version 8 or higher.
- **Integrated Development Environment (IDE):** Eclipse IDE for Java EE Developers or NetBeans.
- **Web Server:** Apache Tomcat (Version 8.0 or later recommended).
- **Database:** MySQL Server (and MySQL Workbench or similar client).

3. Installation and Setup Guide

3.1 Database Setup

1. **Create Database:** Open your MySQL client (e.g., Workbench) and create a new database.

CREATE DATABASE shopping_db;

 Schema and Data: The repository contains a database.sql file. Execute the commands within this file against the newly created shopping_db to create all necessary tables (like product, users, orders, etc.) and populate them with initial data.

 Configure Connection: Ensure the database connection details (username and password) in the project configuration files match your local MySQL credentials.

System Functionality

The application supports two primary roles: **Customer** and **Administrator**.

4.1 Customer Features

Feature	Description	Relevant Files
User Authentication	Registration, Login, and Logout functionality.	LoginServlet, RegisterServlet, login.jsp, register.jsp

Feature	Description	Relevant Files
Product Browsing	View all available products in a catalog format.	HomeServlet, index.jsp
Shopping Cart	Add, remove, and update the quantity of items before purchase.	CartServlet, cart.jsp
Checkout & Ordering	Finalize the purchase and place an order.	OrderServlet, checkout.jsp
Search	Ability to search for products by name or keywords.	(Handled via server-side processing)

4.2 Administrator Features (Inferred)

The system structure implies an admin module for management tasks:

- **Product Management:** Adding new products, editing existing details, and removing outdated items.
- **Order Tracking:** Viewing and updating the status of placed orders.
- **User Management:** Viewing or managing registered users.

5. Architectural Overview

The project follows the **Model-View-Controller (MVC)** architectural pattern, which is standard for Java Servlets and JSP applications.

1. **Model (Data/Business Logic):** Pure Java classes that interact with the database (via JDBC) and hold the business rules.
2. **View (Presentation):** JSP files and associated HTML/CSS/JS/Bootstrap code. This is what the user sees.
3. **Controller (Routing/Handling):** Servlets that receive user requests (e.g., button clicks, form submissions), execute the necessary Model logic, and then forward the result to the correct View (JSP).

6. Future Scope

The following features could be implemented to enhance the system:

1. **Integrated Payment Gateway:** Implement a real-world payment solution (e.g., Stripe or PayPal API) instead of simulated payment.
2. **Product Reviews and Ratings:** Allow users to leave feedback and display average ratings on product pages.
3. **Advanced Filtering and Sorting:** Improve product searching with more robust filters based on price, category, and brand.
4. **Security Enhancement:** Implement more robust input validation and modern session management practices.

7. Troubleshooting

- **HTTP Status 404:** Check the deployment context root in your Tomcat server settings, and ensure the Servlet mappings in web.xml are correct.
- **Database Connection Errors:** Verify the database name, username, and password in your Java code against your MySQL server configuration. Ensure the MySQL JDBC connector JAR file is correctly placed in your project's build path or WEB-INF/lib folder.