# **San-Hussain eCommerce System Documentation**

## **1. Project Overview**

**San-Hussain** is an e-commerce platform designed to solve the problem of having to purchase products locally by allowing customers to shop online. Its main purpose is to enable users to browse and buy products from the comfort of their own home through a dedicated shop website.

## **2. Features**

### **For General Users**

* **Product Listing:** Browse a variety of products available for purchase.
* **Cart Functionality:** Add items to a cart, remove items, and view the cart’s content.
* **Checkout Process:** Proceed to checkout to confirm and finalize orders.
* **Order History & Tracking:** View previous orders and track the status of current orders.
* **Notifications:** Receive on-site notifications about order status and updates.
* **Receipt Printing:** Print a receipt after the order is approved.
* **Profile Settings:** Update and manage personal profile information.

### **For Admins**

* **User Management:** View and manage registered users.
* **Product Management:** Add, update, and remove products.
* **Category Management:** Create and organize product categories.
* **Order Management:** View orders, update order status (approval process), and manage order details.
* **Admin Management:** Add or manage admin accounts (with roles like Pro or Superior).
* **Profile Settings:** Both admins and users can adjust their profile settings.

## **3. Technology Stack**

* **Backend:** PHP
* **Frontend:** HTML, CSS, Bootstrap, JavaScript (minimal use)
* **Database:** MySQL

## **4. System Architecture**

* **Frontend Layer:**The user interface is built using HTML, CSS, and Bootstrap to ensure responsiveness and ease of use. Minimal JavaScript is used for dynamic interactions (like updating the cart).
* **Backend Layer:**PHP is used to handle server-side operations, such as user authentication, processing orders, and interacting with the database. PHP files are organized to separate business logic (e.g., session handling, file uploads) from the presentation.
* **Database Layer:**The MySQL database stores data for products, users, orders, carts, and admin details. Communication between PHP and MySQL is handled via scripts (using the db.php file for connection settings).

## **5. User Roles & Functionalities**

### **General User**

* **Sign Up / Login:** Create an account and sign in.
* **Product Interaction:** View product listings and detailed pages.
* **Cart Management:** Add products to the cart, remove products, and proceed to checkout.
* **Order Tracking:** View the history of orders and check current order status.
* **Receipt and Transactions:** Once an admin approves an order, the user can print a receipt and view transaction details.
* **Notifications:** Receive real-time notifications on the website regarding order updates.

### **Admin**

* **Role Variations:** There are different admin roles (e.g., Pro or Superior) with varying privileges.
* **Product & Category Management:** Add new categories, list and manage products.
* **User Oversight:** View and manage the list of users.
* **Order Approval:** Review and approve orders before users can print receipts or consider an order final.
* **Admin Account Management:** Ability to view existing admins and add new admin users.
* **Profile Settings:** Adjust personal account details.

## **6. Database Structure**

Key tables include:

* **admin:** Stores details of administrators (including role, credentials, etc.).
* **userslist:** Contains data of all registered users.
* **products:** Information about each product (name, price, description, image links, etc.).
* **carts:** Holds temporary data of products added to user carts.
* **orders:** Records orders placed by users including order status, transaction details, etc.

## **7. Implementation Details**

### **Project Setup**

* **Folder Structure:**The main project folder contains an includes directory. This directory holds:
  + **db.php:** The file for database connection settings.
  + **db.sql:** Contains MySQL code compatible with both XAMPP and MySQL Workbench. Users should select and copy the code that suits their DBMS.

### **Installation Steps**

1. **Database Setup:**
   * Import the appropriate db.sql file into your MySQL server (compatible with either XAMPP or Workbench).
   * Adjust db.php with your specific database credentials and settings.
2. **Server Setup:**
   * Place the entire project folder in a directory accessible by your Apache server (e.g., in XAMPP, place it in the htdocs folder).
3. **Run Locally:**
   * Access the project via localhost on your local machine to start using the platform.

**DEFAULT ADMIN LOGIN:** user: proh password: 00000

## **8. Challenges & Solutions**

* **Cart Functionality:**
  + **Challenge:** Implementing the add-to-cart feature required creating a separate file to fetch and handle a specific product upon clicking.
  + **Solution:** Separated the functionality into an independent script to isolate the cart logic from the home page.
* **Session Management:**
  + **Challenge:** Managing user sessions and tracking user states across pages.
  + **Solution:** Refined session handling practices and ensured consistent session start and usage across the project.
* **Folder Structure Issues:**
  + **Challenge:** Incorrect folder organization and file misnaming causing broken links or functionality.
  + **Solution:** Standardized the folder structure and naming conventions.
* **File Uploads:**
  + **Challenge:** Handling file uploads required a temporary file location due to initial misconfigurations.
  + **Solution:** Created a designated temporary file location and adjusted the upload process accordingly.

## **9. Project History & Future Improvements**

### **Project History**

* **Initial Development:**
  + Began with basic product listing and user management.
  + Faced early challenges with session handling and dynamic cart functionality.
* **Feature Enhancements:**
  + Introduced notifications, order tracking, and multi-role admin functionality.
  + Iterative improvements resolved issues with file uploads and folder structure inconsistencies.
* **Current State:**
  + The system now effectively allows users to shop online, manage their orders, and enables admins to oversee and approve orders before finalizing transactions.

### **Future Improvements**

* **Driver/Delivery Section:**
  + Add a module for managing delivery drivers, including real-time tracking and driver assignment.
* **Enhanced Notifications:**
  + Implement mobile or email notifications for order updates.
* **Payment Gateway Integration:**
  + Integrate secure online payment systems for smoother transactions.
* **User Experience (UX):**
  + Revamp the UI for improved accessibility and a more modern look.
* **Scalability:**
  + Consider refactoring the codebase for scalability and maintainability, possibly adopting an MVC framework.
* **Security Enhancements:**
  + Introduce more robust security measures for user data and transaction handling.
* **Analytics & Reporting:**
  + Build reporting features to provide insights into sales, user behavior, and product popularity.