International Journal of Science, Strategic Management and Technology
Volume **02** Issue **08** August-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Online Shopping System: A PHP–MySQL Based E-Commerce Platform with an Advanced Administrative Panel

Bhoomika B N
Department of Computer Science and Engineering
East West Institute of Technology (Autonomous, affiliated to VTU, Belagavi)
Bengaluru, Karnataka, India

https://doi.org/10.55041/ijsmt.v2i8.057

Cite this Article: N, B. B. (2026). Online Shopping System: A PHP–MySQL Based E-Commerce Platform with an Advanced Administrative Panel. International Journal of Science, Strategic Management and Technology, *02*(8), 1-9. https://doi.org/10.55041/ijsmt.v2i8.057

***Abstract*—** *The Online Shopping System with Advanced Admin Page is a web-based e-commerce application that automates the process of browsing, purchasing, and managing products over the internet. Traditional retail and manually maintained sales records are time-consuming, error-prone, and difficult to scale. This work uses PHP and MySQL, following a relational Database Management System (DBMS) approach, to build a two-sided platform consisting of a customer-facing storefront and an advanced administrator panel. Customers can register, log in, search and filter products, add items to a cart or wishlist, place orders, and post reviews, while administrators can manage products, categories, users, and orders through a centralized dashboard. The system ensures data consistency, secure authentication, and efficient retrieval of product and order information through well-structured relational tables and SQL queries.*

***Keywords*—** *Database Management System (DBMS); Online Shopping System; E-Commerce; PHP; MySQL; Admin Panel; Shopping Cart; Wishlist; Relational Database; Web Application; SQL; XAMPP.*

I. INTRODUCTION

Businesses are increasingly shifting from physical storefronts to online platforms to reach a wider customer base and streamline sales operations. The Online Shopping System with Advanced Admin Page is a web-based application that allows customers to browse products, add them to a cart or wishlist, and complete purchases entirely online, while giving administrators complete control over inventory, users, and orders through a dedicated dashboard. Unlike manually maintained sales registers or spreadsheet-based inventories, the system provides a centralized, searchable, and consistently updated record of products, customers, and transactions.

Traditional retail and record-keeping methods often suffer from duplicate entries, inconsistent stock counts, delayed order processing, and difficulty in generating sales reports. These limitations motivate a DBMS-driven application that can enforce data integrity, avoid redundancy, and support fast, reliable transactions between the front-end storefront and the back-end database.

A relational database lies at the heart of the system. Product details, category information, user accounts, cart items, orders, and reviews are stored in normalized MySQL tables linked through primary and foreign key relationships. PHP handles user requests, validates input, and executes SQL queries, while HTML, CSS, and JavaScript provide the interactive front-end experience for both customers and administrators. The system supports two distinct user roles — customers, who browse, search, and purchase — and administrators, who manage products, categories, users, and orders through a secured interface.

II. RELATED WORK

Online shopping systems are among the most widely studied applications of web development and database management. Early retail management relied on manual registers and physical billing; standalone desktop applications later introduced local inventory management but lacked remote access and multi-user support. The emergence of the internet and relational database systems transformed retail into e-commerce, enabling remote browsing, purchasing, and order tracking [1].

PHP remains a widely used server-side language for such systems owing to its integration with MySQL and ease of deployment on Apache servers, while HTML, CSS, and JavaScript structure and style the storefront and enable interactive client-side behaviour [2]. A relational schema for e-commerce typically comprises tables for products, categories, users, carts, orders, and reviews linked through foreign keys, with indexing on frequently queried columns required to sustain fast search and filtering as the catalogue scales [5].

Administrator panels are a critical component of such systems, allowing authorized personnel to manage the catalogue and monitor orders without direct database access, typically enforced through role-based access control [6]. Reported challenges include maintaining data consistency under concurrent stock updates, preserving search performance as the catalogue grows, and mitigating security risks such as SQL injection and session hijacking through prepared statements, hashed passwords, and disciplined session management [4]. Proposed future directions across the literature include payment-gateway integration, purchase-history-based recommendation engines, cloud deployment, and mobile-responsive interfaces.

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **08** August-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

TABLE I
Comparative Summary of Retail Data-Handling Approaches

Approach	Data Handling	Key Limitation
Manual registers	Paper ledgers	Slow, error-prone, no remote access
Spreadsheet-based	Local files	No multi-user support, no live sync
Desktop DBMS apps	Local relational DB	Not remote, limited concurrency
PHP/MySQL web system (this work)	Centralized relational DB via web	Needs careful security & scaling

III. SYSTEM ARCHITECTURE AND METHODOLOGY

A. Research Design

The system follows a design-and-development research methodology in which a database-driven online shopping platform is designed, implemented, and evaluated for functionality, data consistency, and usability. Development proceeded iteratively: database schema design, core storefront pages (product listing, product detail, cart), checkout and order processing, and finally the administrator dashboard, with each module tested against the database independently before integration.

B. Technology Stack

The application is built on a PHP–MySQL–HTML/CSS/JavaScript stack, commonly referred to as a LAMP/WAMP-style architecture, and is hosted on an Apache server through XAMPP/WAMP during development. PHP's mysqli extension connects to MySQL and executes SQL statements for operations such as inserting an order, updating stock after a purchase, or retrieving a filtered product list.

C. Entity–Relationship Design

The schema is organized around six core entities: users (customer and administrator credentials with a role field), categories, products (linked to categories by foreign key), cart (linking users and products with quantity), orders (created at checkout), and reviews (linking users and products). This structure ensures that every cart entry, order, and review can be traced back to a valid user and product, preventing orphaned records.

D. System Modules

- Product Catalogue Management — creation, categorization, and display of products.
- User and Session Management — registration, login, and role-based session state.
- Cart and Wishlist Management — temporary and saved-for-later product selections.
- Order Processing and Checkout — converts a cart into a confirmed, timestamped order.
- Review and Rating Module — customer feedback linked to product and user.
- Admin Dashboard Module — centralized management of products, categories, users, and orders.

IV. IMPLEMENTATION

A. Database Schema Overview

TABLE II
Core Tables of the onlineshop Database

Table	Key Columns
users	user_id (PK), name, email, password (hashed), role, address
categories	cat_id (PK), cat_name
products	product_id (PK), product_title, product_price, product_cat (FK), product_image, product_qty
cart	id (PK), user_id (FK), p_id (FK), qty
orders	order_id (PK), user_id (FK), product_id (FK), qty, total_price, order_date
reviews	review_id (PK), user_id (FK), product_id (FK), rating, comment

B. Sample SQL Operations

All queries are executed as parameterized (prepared) statements rather than by concatenating user input, to guard against SQL injection. Representative operations include:

```
Retrieve products by category: SELECT * FROM products WHERE product_cat = ? ORDER BY product_title;

Add an item to the cart: INSERT INTO cart (user_id, p_id, qty) VALUES (?, ?, ?);

Place an order from the cart: INSERT INTO orders (user_id, product_id, qty, total_price, order_date) SELECT user_id, p_id, qty, (qty*product_price), NOW() FROM cart JOIN products ON cart.p_id = products.product_id WHERE cart.user_id = ?;

Clear the cart after checkout: DELETE FROM cart WHERE user_id = ?;

Low-stock report: SELECT product_title, product_qty FROM products WHERE product_qty < 10;
```

C. Query Optimization and Reporting

Indexes are placed on columns frequently used in WHERE and JOIN clauses, such as category ID and product keywords; pages select only required columns and apply LIMIT/OFFSET pagination for large catalogues. The admin dashboard aggregates data directly from the orders and products tables using COUNT, SUM, and GROUP BY to surface best-selling products, order counts within a date range, and low-stock items, without requiring a separate reporting database.

D. Security Practices

The implementation combines hashed password storage, session-based authentication, role separation between customers and administrators, and prepared statements for every database interaction. Recommended production practices include enforcing HTTPS, periodic review of admin access logs, and regular updates of the PHP and MySQL versions in use.

V. EVALUATION THROUGH CASE STUDIES

A. Small Retail Deployment

A small retail business migrated from manual billing registers to the system. After onboarding its catalogue through the admin dashboard, stock levels updated automatically as orders were placed, and generating best-selling-product or inventory reports took seconds rather than the considerable manual effort previously required.

B. Multi-Category Product Management

A multi-category retailer used the admin dashboard to organize products by category and brand and to maintain a consolidated view of registered customers and order histories. Restricting management functions to authenticated administrators prevented unauthorized catalogue changes, and indexed queries kept category-filtered searches responsive.

C. Migration from Manual Billing

An electronics retailer previously reconciled sales and stock by hand at day's end, a process prone to arithmetic error. After migration, both walk-in and online orders wrote to the same orders and products tables, and end-of-day reconciliation was reduced to viewing a generated summary report, reducing stock-mismatch disputes with suppliers.

D. Observed Challenges

Across deployments, the principal challenges were maintaining accurate stock counts under concurrent access to limited-stock items, preventing SQL injection and session hijacking through input validation, and sustaining query performance as the catalogue and order volume grew — addressed respectively through transaction handling, prepared statements and hashed passwords, and appropriate indexing.

VI. FUTURE SCOPE

- Payment-gateway integration (e.g., Razorpay, Stripe, PayPal) for online transaction completion.
- Cloud deployment with a managed database for improved availability and concurrency.
- Mobile-responsive or dedicated mobile-application front ends.
- Recommendation and analytics features based on order history and browsing patterns.
- Order-status tracking with automated email/SMS notifications.
- Two-factor authentication and automated security audits for admin accounts.
- Migration toward modern frameworks (e.g., Laravel) or a REST-API-based architecture for maintainability.

VII. CONCLUSION

The Online Shopping System with Advanced Admin Page, implemented using PHP and MySQL, provides an efficient, centralized, and scalable solution for managing an online retail business. Its normalized relational database design, structured SQL queries, and clear separation between customer and administrator functionality yield accurate product management, reliable order processing, and reduced manual effort relative to traditional record-keeping. Features including product search and filtering, cart and wishlist management, customer reviews, and a comprehensive admin dashboard cover the core functional requirements of a modern e-commerce platform, while session-based authentication and role-based access control strengthen its security. With further enhancements such as payment-gateway integration, cloud deployment, and analytics, the system offers a strong foundation for modernizing small- to medium-scale retail operations through database-driven web technology.

REFERENCES

[1] A. Sharma and R. Verma, "Design and implementation of an e-commerce web application using PHP and MySQL," Int. J. Comput. Appl., vol. 176, no. 12, pp. 10–16, 2023.

[2] S. Gupta and P. Rao, "A relational database approach to online shopping cart systems," Int. J. Adv. Res. Comput. Sci., vol. 14, no. 2, pp. 45–52, 2023.

[3] M. Iqbal and T. Hasan, "Web-based inventory and order management system using PHP/MySQL," Int. J. Eng. Res. Technol. (IJERT), vol. 11, no. 7, pp. 210–217, 2022.

[4] K. Singh and N. Yadav, "Security practices in PHP-based web applications: Preventing SQL injection and session hijacking," J. Inf. Secur. Appl., vol. 66, Art. no. 103154, 2022.

[5] R. Patel and D. Mehta, "Database normalization techniques for e-commerce applications," Int. J. Innov. Res. Technol. (IJIRT), vol. 9, no. 5, pp. 512–518, 2022.

[6] J. Kim and H. Park, "A study on admin dashboard design for small-scale e-commerce platforms," Int. J. Comput. Sci. Mobile Comput., vol. 11, no. 9, pp. 66–73, 2022.