**ESHOP**

**A PROJECT REPORT**

**for**

**Project (KCA451)**

**Session (2024-25)**

**Submitted by  
  
Rajat Agarwal**

**2300290140132**

**Sanchi Singh**

**2300290140157**

**Sarthak Semwal**

**2300290140160**

**Submitted in partial fulfilment of the**

**Requirements for the Degree of**

**MASTER OF COMPUTER APPLICATION**

**Under the Supervision of**

**Mr. Apoorv Jain  
Assistant Professor**



**Submitted to**

**Department Of Computer Applications**

**KIET Group of Institutions, Ghaziabad**

**Uttar Pradesh-201206**

**(MAY 2025)**

**CERTIFICATE**

Certified that Sarthak Semwal 2300290140160, Rajat Aggarwal 2300290140132, Sanchi Singh 2300290140157 has carried out the project work having “Eshop” (Project-KCA451) for Master of Computer Application from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student themself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

Ms. Neha Tyagi Dr. Akash Rajak

Teaching Assistant Dean

Department of Computer Applications Department of Computer Applications

KIET Group of Institutions, Ghaziabad KIET Group of Institutions, Ghaziabad

# ABSTRACT

This project report presents the development of a dynamic and responsive eCommerce website built using HTML, CSS, JavaScript, Laravel, and MySQL. The primary objective of the project was to design and implement a fully functional online shopping platform that facilitates seamless interaction between buyers and sellers, offering features such as product browsing, cart management, secure checkout, and user authentication.

The frontend of the application was developed using HTML, CSS, and JavaScript to ensure a user-friendly and visually appealing interface. The backend was powered by Laravel, a robust PHP framework, which provided structured routing, MVC architecture, and built-in security features. MySQL was utilized as the relational database management system to handle and store data related to users, products, orders, and transactions efficiently.

Key functionalities implemented in the system include user registration and login, product categorization, search functionality, real-time cart updates, order history tracking, and admin panel controls for managing inventory and orders. Special attention was given to responsiveness and cross-browser compatibility to enhance user experience across various devices.

The project highlights the integration of frontend and backend technologies to build a scalable and secure eCommerce solution. It serves as a practical demonstration of applying full-stack web development skills to solve real-world problems in the digital retail space.

## ACKNOWLEDGEMENT

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor **Ms. Neha Tyagi**, for her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Akash Rajak**, Professor and Dean, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

**Sarthak Semwal   
  
  
Rajat Agarwal   
  
  
Sanchi Singh**

# TABLE OF CONTENTS

Certificate ii

Abstract iii

Acknowledgements iv

Table of Contents

1. Introduction 7-11  
   1. Overview
   2. Motivation
   3. Problem Statement
   4. Expected Outcome
2. Literature Survey 12

1. Feasibility Study 15-18  
   1. Technical Feasibility
   2. Economic Feasibility

1. Design 21-25
   1. Data Flow Diagram
   2. ER Diagram
   3. Use Case Diagram
2. Proposed Work 26-32   
   1. Technology Description
   2. Approach Used
   3. Implementation Details
   4. Challenges Faced
   5. Future Enhancements
3. Coding 33-57

1. Results 58-69   
   1. Screens and explanation

8 Discussions 70-71

* 1. Performance
  2. Future Research Directions

1. Conclusion 72
2. References 73  
      
   11 Bibliography 75

**CHAPTER 1**

## INTRODUCTION

### 1.1 Overview In the rapidly evolving digital era, eCommerce has emerged as a dominant force, reshaping the way businesses and consumers interact. With the increasing demand for online shopping platforms, building a robust, user-friendly, and secure eCommerce website has become a significant area of interest in web development. This project aims to create a fully functional eCommerce website that enables users to browse products, add items to a shopping cart, place orders, and manage their accounts, all within a seamless and responsive interface.

### The project leverages a combination of modern web development technologies. The frontend is developed using HTML, CSS, and JavaScript, which ensures an interactive and responsive user experience. The backend is powered by Laravel, a powerful PHP framework known for its elegant syntax, security features, and MVC architecture. For data storage and management, MySQL is used as the relational database management system, offering reliable performance and scalability.

### The website includes essential features such as user authentication, product listings with categories, shopping cart functionality, order management, and an admin dashboard for managing products and tracking sales. These features are designed to simulate a real-world online store environment, providing practical experience in full-stack development.

### This project not only demonstrates the application of technical skills in building a web application but also emphasizes the importance of design, usability, and security in delivering a quality user experience. It serves as a comprehensive learning experience in developing modern web-based solutions aligned with industry standards..

### `

### 1.2 Motivation

The rapid growth of internet usage and digital transformation across industries has significantly influenced consumer behavior, shifting traditional shopping practices toward online platforms. Observing this trend, the motivation behind developing an eCommerce website stems from the desire to understand and contribute to the evolving landscape of digital commerce.

This project was undertaken to bridge the gap between theoretical knowledge and practical application in full-stack web development. Building an eCommerce platform provided the opportunity to work with real-world use cases such as product listings, user management, secure transactions, and inventory handling — all of which are core components of modern web applications.

Another key motivation was to gain hands-on experience with Laravel, a popular PHP framework, and MySQL for backend development, as well as to enhance frontend skills using HTML, CSS, and JavaScript. This technology stack was chosen to develop a scalable, maintainable, and secure web solution while strengthening the understanding of MVC architecture and database-driven design.

From a user perspective, creating a platform that simplifies shopping and improves accessibility to products reflects a genuine need in today's fast-paced lifestyle. The project, therefore, aims not just to fulfill technical learning objectives, but also to create a solution that could be adapted for real-world deployment.

Ultimately, this project is motivated by the passion to innovate, solve real-world problems, and gain valuable experience in web application development — laying a strong foundation for a future career in software and web development.

### 1.3 Problem Statement

In today’s digital age, consumers increasingly prefer the convenience of online shopping over traditional retail methods. However, many small and medium-sized businesses lack the technical resources or expertise to establish a strong online presence. Existing eCommerce platforms can be costly, complex, or limited in customization, making it difficult for emerging businesses to compete in the online marketplace.

The core problem addressed by this project is the absence of an affordable, user-friendly, and customizable eCommerce solution that meets the needs of both customers and administrators. Customers require an intuitive interface to browse products, manage carts, and place secure orders, while administrators need a reliable backend to manage inventory, process orders, and view sales data.

This project aims to solve these issues by developing a fully functional eCommerce website using open-source technologies such as HTML, CSS, JavaScript, Laravel, and MySQL. The solution focuses on delivering a seamless shopping experience, robust backend management, and scalable architecture, all while maintaining security, responsiveness, and performance.

### 1.4 Expected Outcome

The primary expected outcome of this project is the successful development and deployment of a fully functional, responsive, and secure eCommerce website that meets the essential requirements of both end-users (customers) and administrators (store owners). The platform will serve as a practical demonstration of full-stack web development and is expected to include the following features and deliverables:

1. **User-Friendly Frontend Interface**
   * A clean, modern, and responsive user interface built with HTML, CSS, and JavaScript.
   * Easy navigation for browsing products by categories, brands, or price.
   * Search functionality for quick product discovery.
   * Detailed product pages displaying images, descriptions, prices, and stock information.
2. **User Authentication and Profile Management**
   * Secure registration and login system using Laravel’s built-in authentication features.
   * Password hashing and session management to ensure user data privacy.
   * User dashboard to manage personal information, view order history, and track orders.
3. **Shopping Cart and Checkout Functionality**
   * Ability to add, update, or remove products in the shopping cart.
   * Dynamic cart updates using JavaScript for a seamless user experience.
   * Checkout system with order summary, shipping information, and order confirmation.
4. **Backend Admin Panel**
   * Admin login and secure access control.
   * Dashboard to manage products (add/edit/delete), view inventory, and monitor stock levels.
   * View, manage, and update order statuses.
   * Basic analytics on sales, customers, and product performance.
5. **Database Management with MySQL**
   * Properly structured and normalized database to store and manage user data, products, orders, and transactions.
   * Relationships between tables (e.g., one-to-many between users and orders) implemented using Laravel's Eloquent ORM.
   * Data integrity, consistency, and efficiency maintained across CRUD operations.
6. **Security and Performance Considerations**
   * Input validation and sanitization to prevent SQL injection, XSS, and CSRF attacks.
   * Use of Laravel’s middleware for authentication and route protection.
   * Optimized asset loading (CSS/JS minification, image compression) to ensure fast load times and responsiveness.
7. **Scalability and Extendability**
   * Codebase designed with modularity and reusability in mind to allow future enhancements such as payment gateway integration, product reviews, coupons, wishlists, or a blog section.
   * MVC architecture ensures that the project can be scaled or modified without disrupting the existing functionality.
8. **Learning and Skill Development**
   * Improved proficiency in full-stack web development using a real-world use case.
   * Practical experience with Laravel, MySQL, and version control using Git.
   * Insight into best practices in UI/UX design, backend logic, and secure data handling.

By the end of the project, a live prototype of the eCommerce platform will be ready to demonstrate its features and functionality. This outcome will not only showcase technical competencies but also offer a solid base for real-world deployment or further academic exploration.

**CHAPTER 2**

## LITERATURE SURVEY

The development of eCommerce platforms has been a significant area of research and technological advancement over the past two decades. Numerous studies and real-world implementations have demonstrated the importance of integrating both frontend and backend technologies to build scalable, secure, and user-friendly web applications. This literature survey explores various tools, technologies, frameworks, and methodologies relevant to the development of an eCommerce platform using **HTML**, **CSS**, **JavaScript**, **Laravel**, and **MySQL**.

**1. Evolution of eCommerce Systems**

Early eCommerce platforms were static, limited in functionality, and lacked dynamic user interaction. Over time, advancements in web technologies enabled the creation of dynamic websites that could handle complex functionalities like user authentication, cart systems, real-time inventory tracking, and secure transactions. Platforms such as Amazon, eBay, and Flipkart have set benchmarks in terms of user experience, system performance, and security.

Researchers and developers have studied various eCommerce models such as **B2B (Business-to-Business)**, **B2C (Business-to-Consumer)**, **C2C (Consumer-to-Consumer)**, and **C2B (Consumer-to-Business)** to understand different requirements and technological solutions.

**2. Frontend Technologies**

**a. HTML (HyperText Markup Language)**

HTML is the foundational markup language used for structuring content on the web. Studies have shown that proper semantic use of HTML improves accessibility, SEO, and maintainability of websites. It defines the skeleton of eCommerce pages, including headers, product listings, forms, and buttons.

**b. CSS (Cascading Style Sheets)**

CSS is used to style and layout HTML elements. Literature supports the use of modern CSS features such as Flexbox and Grid for responsive design. Frameworks like Bootstrap and Tailwind CSS have also been widely studied for rapid UI development and mobile-first design strategies.

**c. JavaScript**

JavaScript brings interactivity to eCommerce platforms. It enables dynamic cart updates, form validation, and real-time UI changes without reloading pages. Libraries like jQuery and frameworks like Vue.js or React (although not used in this project) have shown great efficiency in improving frontend performance and user experience.

**3. Backend Technologies**

**a. Laravel Framework**

Laravel is a PHP-based backend framework that follows the **Model-View-Controller (MVC)** architectural pattern. Academic and developer literature widely recognize Laravel for its:

* **Eloquent ORM**, which simplifies database operations
* **Blade templating engine**, which separates logic from presentation
* **Built-in authentication and security features**, such as CSRF protection and route guarding
* **Artisan CLI**, which boosts developer productivity

Studies emphasize Laravel’s modular structure and extensive documentation, making it ideal for rapid application development and long-term maintainability of web applications.

**b. MySQL**

MySQL is a popular open-source **Relational Database Management System (RDBMS)**. It is extensively used in eCommerce solutions due to its performance, reliability, and ease of integration with web applications. Literature highlights its strengths in handling structured data, enforcing data integrity, and managing relationships between tables using foreign keys.

**4. Security in eCommerce**

Security is a crucial aspect of eCommerce systems. Studies and guidelines by OWASP (Open Web Application Security Project) underline the importance of securing web applications from threats such as **SQL injection**, **Cross-Site Scripting (XSS)**, and **Cross-Site Request Forgery (CSRF)**. Laravel’s built-in protections against these threats have been positively reviewed in various research articles and developer forums.

Furthermore, secure authentication systems, encrypted sessions, and form validation practices have been studied and recommended as best practices for building secure and trustworthy eCommerce platforms.

**5. User Experience (UX) and Design Trends**

Research in HCI (Human-Computer Interaction) and web usability indicates that **ease of navigation**, **fast loading times**, **mobile responsiveness**, and **clean layouts** are key to retaining customers on eCommerce platforms. Studies also suggest that visual elements, such as product images, quick filters, and feedback messages, significantly improve user engagement and conversion rates.

**6. Comparative Studies**

Comparative studies between Laravel and other backend frameworks like Django (Python), Express (Node.js), and Ruby on Rails show that Laravel offers a balanced mix of simplicity, flexibility, and built-in features for medium-scale eCommerce projects. Similarly, MySQL is often compared to PostgreSQL and SQLite, with MySQL being favored for its ease of use in web-based systems.

**Conclusion of Literature Survey**  
This literature review affirms the suitability of the chosen technologies — HTML, CSS, JavaScript, Laravel, and MySQL — for building a secure, scalable, and feature-rich eCommerce platform. It draws from existing research and development practices to guide the design and implementation of the project. Leveraging insights from the academic and industrial studies ensures that the resulting system aligns with modern web development standards and user expectations.

#### Chapter 3 Feasibility Study 3.1. Technical Feasibility

Technical feasibility refers to evaluating whether the proposed project can be successfully developed, implemented, and maintained with the available technical resources, tools, and skills. This section assesses the practicality of building an eCommerce website using the selected technology stack: **HTML, CSS, JavaScript, Laravel (PHP Framework), and MySQL**.

**1. Technology Stack Availability and Compatibility**

**a. Frontend Technologies**

* **HTML (HyperText Markup Language)** is the standard markup language for creating web pages and is universally supported by all browsers. It is suitable for structuring all visual content on the site such as product listings, navigation bars, and checkout forms.
* **CSS (Cascading Style Sheets)** enables the styling of web pages. With modern techniques like Flexbox and Grid, it allows responsive and mobile-friendly design.
* **JavaScript** enhances the interactivity of the site, such as enabling live cart updates, form validations, and dynamic content without reloading pages.

These frontend technologies are lightweight, platform-independent, and require no special hardware or proprietary software, making them highly feasible for any system with a modern web browser.

**b. Backend Framework: Laravel (PHP)**

* Laravel is a free, open-source PHP framework that supports rapid web application development.
* It includes built-in features like routing, authentication, session handling, and middleware, reducing the development time.
* Laravel supports MVC architecture, enhancing the separation of concerns and maintainability of the codebase.
* It also integrates easily with front-end tools, supports RESTful API development, and has Artisan CLI for automation tasks.
* The system requirements for running Laravel (PHP ≥ 8.1, Composer, and a web server like Apache/Nginx) are easily met on any modern development machine or web hosting environment.

**c. Database Management System: MySQL**

* MySQL is an open-source relational database system widely supported and easy to set up.
* It integrates seamlessly with Laravel via Eloquent ORM, enabling smooth CRUD operations.
* MySQL is highly scalable and can manage a large volume of transactions efficiently, which is crucial for an eCommerce platform.

All selected technologies are well-documented and supported by large communities, ensuring help is available for troubleshooting and development challenges.

**2. Hardware Requirements**

The development and deployment of the eCommerce website do not demand high-end hardware. A standard development setup includes:

* **Development Machine**
  + OS: Windows, macOS, or Linux (Ubuntu preferred for Laravel)
  + RAM: Minimum 4 GB (8 GB recommended)
  + Storage: Minimum 10 GB free space
  + Software: Apache/Nginx, PHP, MySQL, Composer, Laravel, a code editor (e.g., VS Code)
* **Server Deployment (for live hosting)**
  + A shared or VPS server with LAMP/LEMP stack
  + PHP 8.1 or higher
  + MySQL 5.7+ or MariaDB
  + SSL certificate for HTTPS security
  + Optional: cPanel for easier deployment and management

These hardware requirements are minimal and affordable, making the project technically feasible for students or small business deployment.

**3. Software Tools and Libraries**

* **Code Editor**: VS Code or Sublime Text for efficient coding with extensions for Laravel and PHP.
* **Version Control**: Git and GitHub for source code management and collaboration.
* **Browser Developer Tools**: For debugging HTML, CSS, and JavaScript issues.
* **Composer**: Dependency manager for PHP and Laravel packages.
* **XAMPP/Laragon** (for local development): Simple packages to set up Apache, MySQL, and PHP environments.

All these tools are either free or open-source, making them accessible and cost-effective.

**4. Developer Skill Set**

* Basic to intermediate knowledge of HTML, CSS, and JavaScript is sufficient for designing the frontend.
* Laravel simplifies backend development with its expressive syntax and built-in features, making it suitable even for developers with limited PHP experience.
* Eloquent ORM reduces the complexity of writing SQL queries manually.
* Ample documentation, tutorials, and community support make the learning curve manageable.

Hence, the skill set required to develop this project is well within reach of undergraduate students or entry-level developers.

**5. Integration and Scalability**

* Laravel supports API integrations (e.g., payment gateways like Razorpay, Stripe, or PayPal) for future extensions.
* The modular structure and MVC architecture ensure that additional features such as wishlists, reviews, email notifications, or mobile app integration can be added without redesigning the entire system.
* MySQL databases can be scaled vertically or replicated horizontally, making the system technically scalable for a larger user base.

**6. Security Considerations**

Laravel comes with built-in protection against:

* **CSRF (Cross-Site Request Forgery)**
* **XSS (Cross-Site Scripting)**
* **SQL Injection**
* **Password encryption** using Bcrypt or Argon2

This ensures the platform is secure enough for handling sensitive user data and transactions.

**Conclusion of Technical Feasibility**

Based on the above analysis, the development of an eCommerce website using HTML, CSS, JavaScript, Laravel, and MySQL is technically feasible. The technologies are accessible, open-source, well-supported, and compatible with one another. The hardware and software requirements are minimal, and the developer skill requirements are attainable. Additionally, the system offers flexibility for future enhancements and is secure enough to be deployed in real-world environments.

#### 3.2. Economic Feasibility

Economic feasibility assesses whether the proposed project is financially viable — that is, whether the expected benefits outweigh the costs. In the context of this eCommerce website built using **HTML, CSS, JavaScript, Laravel, and MySQL**, the goal is to evaluate if the development, deployment, and maintenance of the system are cost-effective, especially for small businesses, startups, or student-led initiatives.

**1. Cost Analysis**

**a. Development Costs**

* **Human Resources**:  
  Since this project is likely undertaken by students or in-house developers, there is **no direct labor cost**. However, in a professional setting, development costs may include payments for frontend and backend developers, designers, and testers.
* **Software Costs**:  
  All the tools and frameworks used are **free and open-source**, eliminating the need for purchasing software licenses:

| **Tool/Technology** | **Cost** |
| --- | --- |
| HTML/CSS/JS | Free |
| Laravel Framework | Free (Open-source) |
| PHP & MySQL | Free (LAMP stack) |
| Composer, Git | Free |
| Code Editors (VS Code, Sublime) | Free |

* **Hardware Costs**:  
    
  No special hardware is required beyond a standard personal computer. Most development can be done using existing devices:
  + Development PC/Laptop (already available in most academic/professional environments)
  + No additional investment needed for local development

**b. Hosting and Deployment Costs**

If deployed publicly, the following minimal recurring costs apply:

* **Domain Name**: ₹800–₹1,200/year (~$10–$15/year)
* **Web Hosting** (Shared/VPS): ₹2,000–₹5,000/year (~$25–$60/year)
* **SSL Certificate**: Free (Let’s Encrypt) or Paid (~₹1,000/year)

These costs are relatively low and are often manageable for startups or students showcasing their work online.

**2. Benefit Analysis**

**a. Direct Benefits**

* **Online Sales Enablement**: Allows businesses to sell products online 24/7, expanding their customer reach beyond geographical boundaries.
* **Cost Savings**: Eliminates the need for physical storefronts, reducing rental, staffing, and operational costs.
* **Inventory Efficiency**: Real-time product tracking reduces overstocking or stockouts.
* **Customer Convenience**: Offers users a seamless experience in browsing, selecting, and purchasing products from any device.

**b. Indirect Benefits**

* **Brand Visibility**: Online presence improves brand awareness and trust.
* **Data Collection**: Order and user data can be used for marketing and business analytics.
* **Scalability**: The system can be enhanced over time with features like payment gateways, customer reviews, and personalized recommendations without heavy reinvestment.
* **Skill Development**: Developers involved in the project gain valuable full-stack experience, which has long-term career and academic value.

**3. Return on Investment (ROI)**

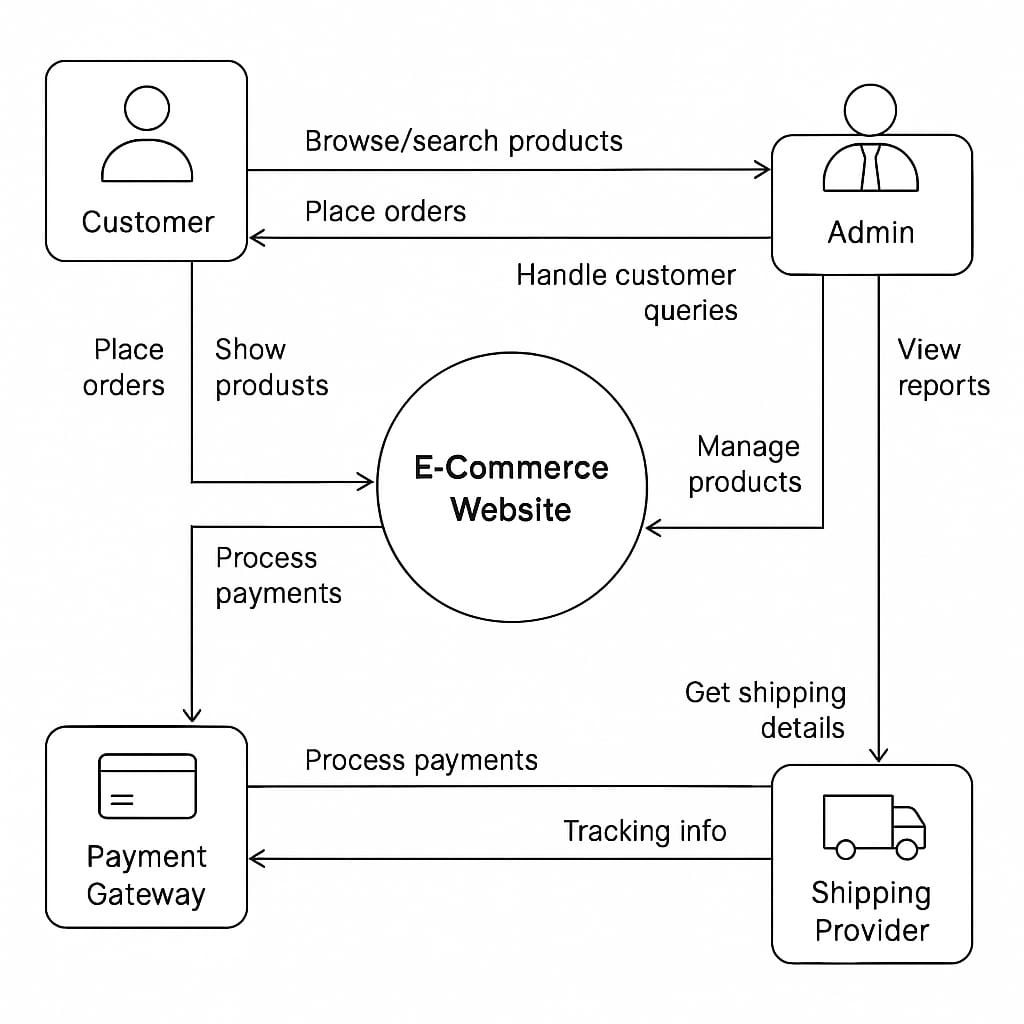
Given the low development and maintenance cost and high potential benefits, especially for small businesses or startups, the return on investment is **high**. The initial investment mainly includes domain/hosting costs, which can be recovered quickly through even a small volume of online transactions.

**CHAPTER 4**

**Design**

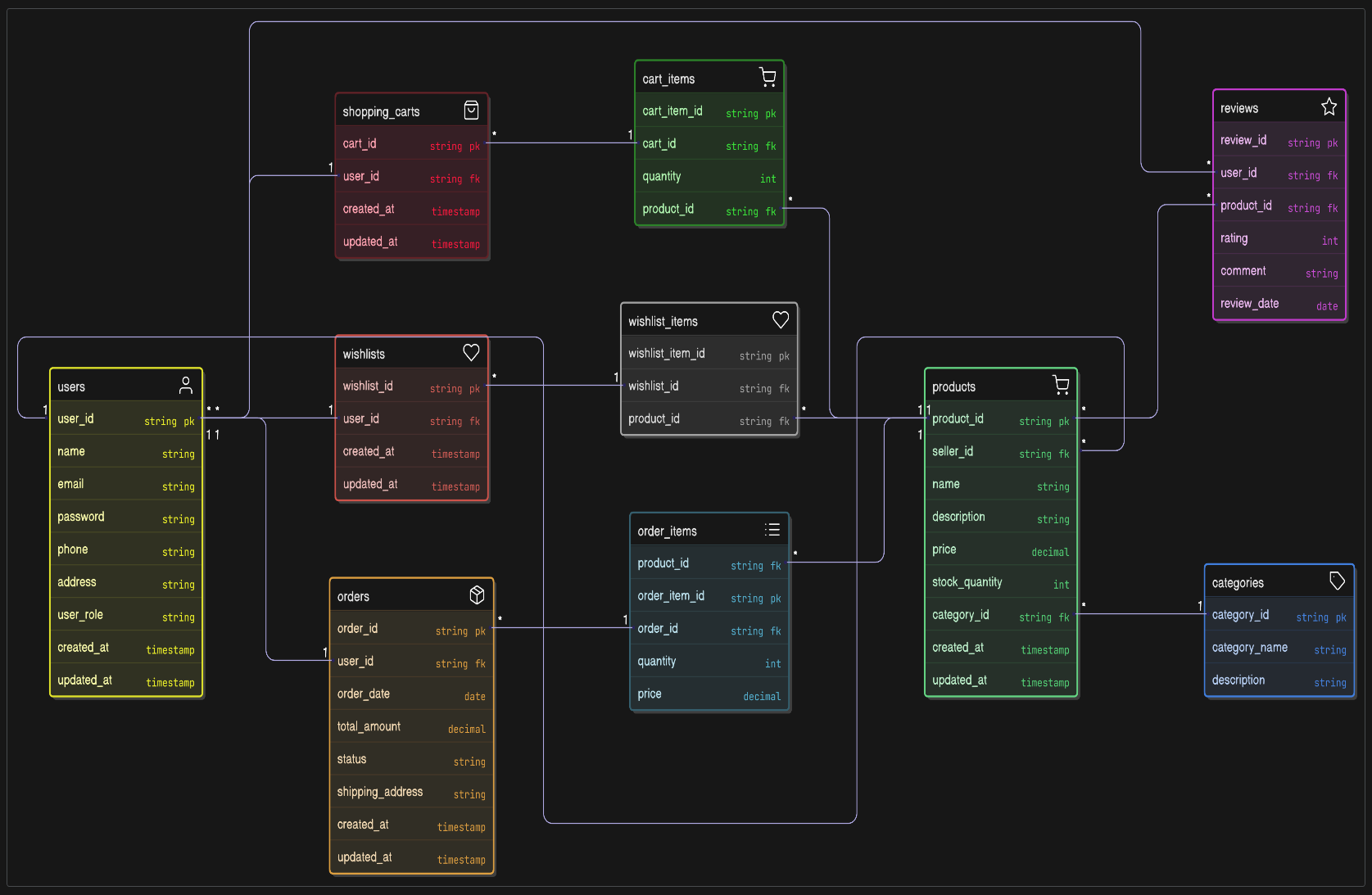
## 4.1 Data Flow Diagram

Data Flow Diagram will explain the basic flow of data in a system which shows how the new or old user will interact with the system.

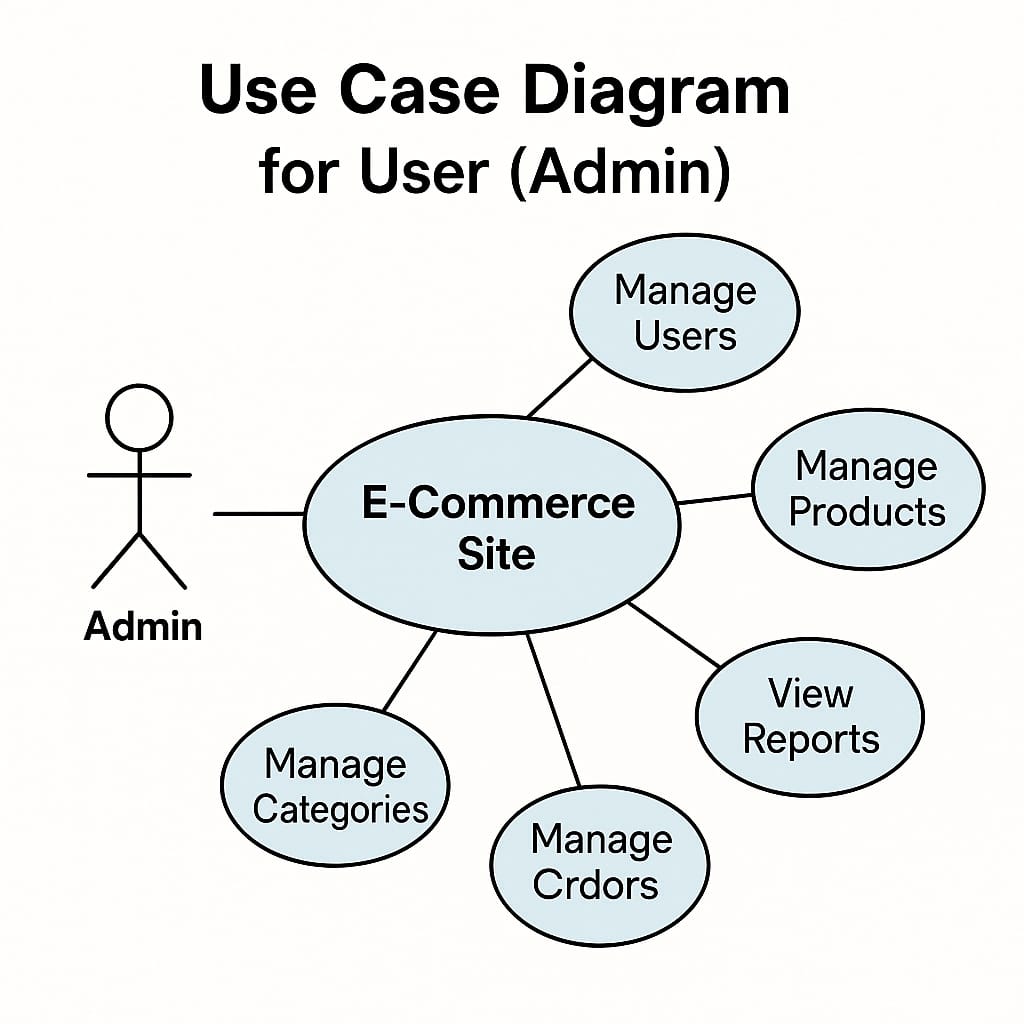


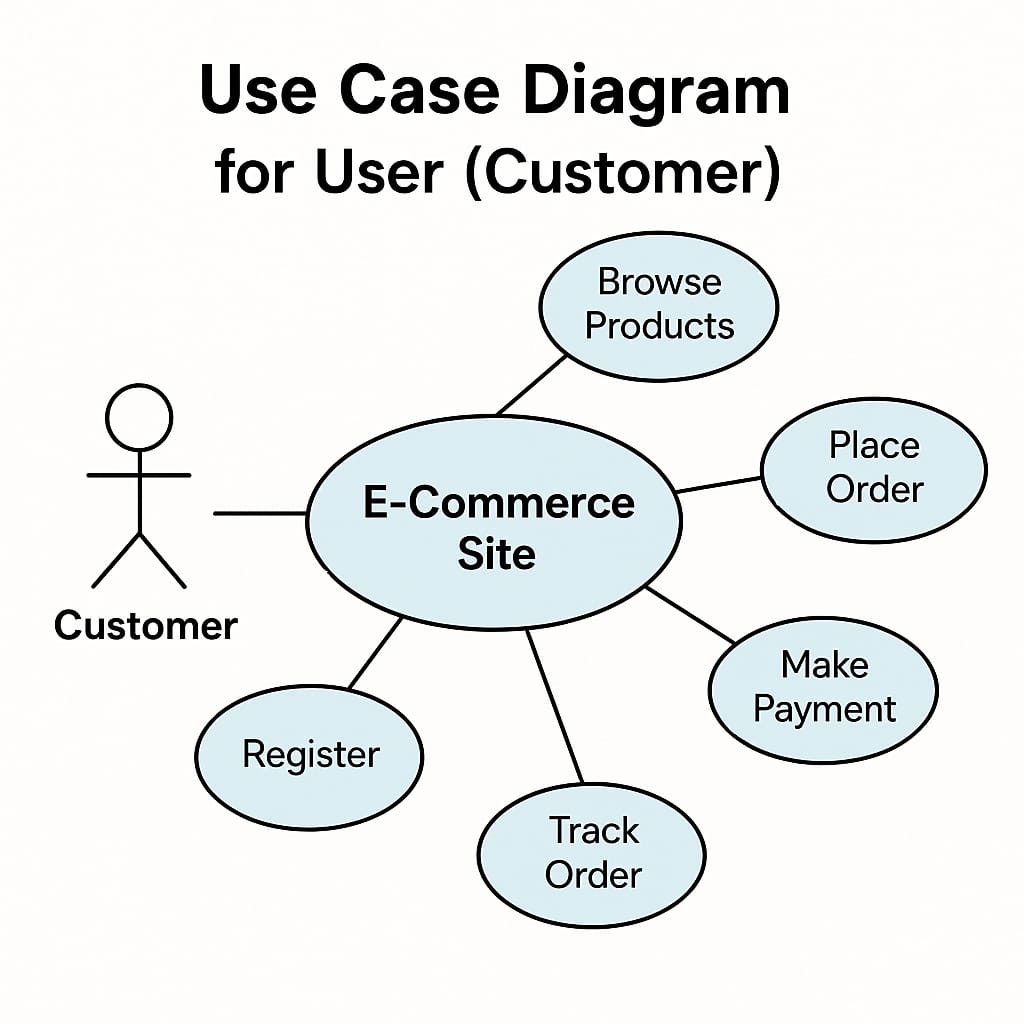
## 

**4.2 ER Diagram**Er diagram is used to visually display the structure of the database They represent the entity their attribute and the relationship between the entities.



## 4.3 Use Case Diagram

In Use Case Diagram we elaborate about the purpose, actor, pre-condition, post- condition, basic flow, and alternate flow of all the use cases.



**CHAPTER 5**

## PROPOSED WORK

### 5.1 Technology Description

This project utilizes a modern and robust web development stack consisting of **HTML**, **CSS**, **JavaScript**, **Laravel (PHP Framework)**, and **MySQL**. Each technology plays a critical role in building a fully functional, responsive, secure, and scalable eCommerce website. Below is a detailed description of the technologies used and their specific purposes in the development of the platform.

**1. HTML (HyperText Markup Language)**

**Purpose:** Structuring the content of web pages  
HTML is the backbone of all web development. It provides the structure and layout of the website by defining elements such as headings, paragraphs, forms, product listings, navigation menus, buttons, and other interface components. In this project, HTML was used to:

* Create the basic structure of all web pages (home, product, cart, checkout, etc.)
* Define input fields for user registration and login
* Organize content for optimal readability and accessibility

**2. CSS (Cascading Style Sheets)**

**Purpose:** Styling and layout of web pages  
CSS is used to enhance the visual presentation of the HTML structure. It allows for the customization of colors, fonts, margins, paddings, and responsiveness. In this project, CSS was used to:

* Create a visually appealing, professional design for the eCommerce platform
* Implement responsive layouts using media queries, Flexbox, and CSS Grid
* Style buttons, cards, product listings, headers, and footers
* Improve user experience across devices (mobile, tablet, desktop)

Frameworks or libraries such as **Bootstrap** (if used) may have been integrated to speed up UI development and ensure cross-browser compatibility.

**3. JavaScript**

**Purpose:** Client-side interactivity and dynamic behavior  
JavaScript adds life to static HTML/CSS pages by enabling real-time interactivity and dynamic content. In this project, JavaScript was used to:

* Validate forms on the client side before submission
* Handle events such as adding/removing items from the cart
* Update UI elements dynamically without page reloads
* Enable modal popups, sliders, and dropdown interactions
* Enhance responsiveness and user engagement

If jQuery or vanilla JavaScript was used, it helped simplify DOM manipulation and event handling.

**4. Laravel (PHP Framework)**

**Purpose:** Backend development and server-side logic  
Laravel is a powerful, open-source PHP framework designed for building modern web applications with elegant syntax and a strong architectural foundation. It follows the **Model-View-Controller (MVC)** pattern, promoting separation of concerns and clean code practices. In this project, Laravel was used to:

* Manage user authentication (login, register, logout)
* Handle product data CRUD operations (Create, Read, Update, Delete)
* Implement shopping cart and order management systems
* Manage secure routing, sessions, and middleware
* Use Blade templating for rendering dynamic HTML views
* Process and validate HTTP requests via Laravel’s form request classes
* Run background tasks and scheduled jobs via Artisan CLI

Laravel’s ecosystem also provides tools such as:

* **Eloquent ORM** for interacting with the database
* **Laravel Mix** for asset compilation (CSS, JS)
* **Artisan** command-line tool for scaffolding and automation

**5. MySQL**

**Purpose:** Database management system  
MySQL is a widely-used open-source relational database. It was selected for this project due to its reliability, ease of use, and seamless integration with Laravel. In this project, MySQL was used to:

* Store product details (name, price, category, stock, etc.)
* Manage user data (registration details, order history, addresses)
* Record transaction details such as cart items, payments, and order status
* Enforce data integrity using relational tables and foreign keys
* Perform efficient queries using Laravel’s Eloquent ORM

Database tables were designed with normalization principles to reduce redundancy and improve data consistency.

**6. Additional Tools and Technologies**

* **XAMPP/Laragon**: Local development environments that bundle Apache, MySQL, and PHP for testing the application locally.
* **Composer**: PHP dependency manager used to install and manage Laravel packages.
* **Git and GitHub**: Version control system for code management and collaboration.
* **VS Code/Sublime Text**: Source code editors with extensions for PHP, HTML, and Laravel development.

### 5.2 Approach Used The eCommerce platform is designed to deliver a seamless online shopping experience, including product browsing, user registration, cart management, and secure checkout. The frontend is developed using HTML, CSS, and JavaScript to create responsive and interactive user interfaces. Laravel, a PHP framework, handles the backend operations such as user authentication, product management, and order processing. MySQL serves as the database to store user data, product details, and transaction records. The system follows the MVC architecture to ensure clean separation of concerns and maintainability

### 

#### 5.2.1 Objective

The objective of this eCommerce platform is to create a user-friendly and efficient online shopping experience that benefits customers, sellers, and administrators. The application aims to:

* Simplify the product browsing and purchasing process with an intuitive interface and seamless navigation.
* Enable users to search, filter, and discover products based on categories, price, and ratings.
* Provide customers with a secure and easy-to-use shopping cart and checkout system.
* Allow sellers to manage their product listings, inventory, and orders through an admin dashboard.
* Equip administrators with tools to oversee user activity, manage product data, and maintain the overall health of the platform.
* Enhance accessibility and scalability to support a growing user base with secure and reliable transaction processing.

#### 

#### 5.2.2 Technologies Used

5.2.2.1 **Frontend:** HTML , CSS, JS.

5.2.2.2 **Backend:** LARAVEL, MYSQL

#### 5.2.3 Features

The eCommerce platform offers dedicated features for Customers, Sellers, and Admins, ensuring a smooth and efficient online shopping and management experience.  
  
**Customer Features**

* **Register & Login**  
  Secure sign-up and login using email and password.
* **Browse Products**  
  Easily explore products categorized by type, brand, and price.
* **Search & Filter**  
  Find products using search and apply filters based on price, ratings, and categories.
* **Add to Cart & Wishlist**  
  Add desired products to the shopping cart or wishlist for future purchases.
* **Secure Checkout & Payments**  
  Complete purchases with secure payment integration supporting multiple methods.
* **Order Tracking**  
  View order history and track current order status in real-time.
* **Edit Profile**  
  Update personal details such as name, address, and contact information.  
    
    
  **Admin Features**
* **Dashboard Overview**  
  Monitor total number of customers, sellers, products, and orders on the platform.
* **User & Seller Management**  
  Approve, verify, or deactivate customer and seller accounts.
* **Product Approval**  
  Review and approve new product listings before they go live.
* **Order Monitoring**  
  Track order processing and resolve disputes or issues.

### 5.3 Implementation Details

The eCommerce platform is implemented using a combination of frontend and backend technologies to deliver a seamless, secure, and scalable shopping experience. The development followed the Model-View-Controller (MVC) architecture, ensuring a clean separation of concerns and maintainable code.

**1. Frontend Implementation**

* **Technologies Used:** HTML, CSS, JavaScript
* The frontend focuses on creating an intuitive and responsive user interface.
* **HTML** provides the basic page structure, including forms for user registration, login, product listings, and checkout.
* **CSS** is utilized for styling the pages with a consistent theme, responsive layouts, and mobile-friendly designs using media queries and Flexbox/Grid.
* **JavaScript** enhances interactivity by handling form validations, dynamic updates such as adding/removing items from the cart, and real-time feedback to the user without full page reloads.

**2. Backend Implementation**

* **Technology Used:** Laravel Framework (PHP)
* The backend handles business logic, database interactions, authentication, and API routing.
* **Routing:** Laravel’s routing system manages URL endpoints, directing requests to appropriate controllers.
* **Controllers:** Controllers process incoming requests, apply business logic, and prepare data to be displayed by views  
  .
* **Models:** Eloquent ORM models represent database tables and handle queries, allowing seamless CRUD operations for users, products, orders, and carts.
* **Authentication:** Laravel’s built-in authentication scaffolding is implemented to secure user login, registration, password reset, and session management.
* **Security:** Middleware is used to restrict access to certain routes based on user roles (customer, seller, admin) and ensure data validation and sanitization.

**3. Database Implementation**

* **Technology Used:** MySQL
* The database schema is designed to support all eCommerce functionalities efficiently.
* Tables include Users, Products, Categories, Orders, Order\_Items, Cart\_Items, and Transactions.
* Relationships such as one-to-many (one user can place many orders) and many-to-many (products in multiple categories) are defined using foreign keys.
* Data integrity is enforced using constraints and indexes to improve query performance.

**4. Additional Features**

* **Shopping Cart:**  
  Implemented using session management to temporarily store user-selected items before purchase.
* **Order Processing:**  
  Once the user confirms the purchase, order details are saved in the database, and inventory is updated accordingly.
* **Admin Panel:**  
  Provides CRUD operations for managing users, products, and orders, with role-based access control.
* **Responsive Design:**  
  Ensures the website adapts to various screen sizes and devices for optimal usability.

**5. Development Tools**

* **Local Development:** XAMPP or Laragon used to simulate Apache, MySQL, and PHP environments.
* **Version Control:** Git and GitHub used for source code management and collaboration.
* **Package Management:** Composer used to manage PHP dependencies and Laravel packages.

**Summary**

The implementation integrates frontend responsiveness with backend robustness, allowing secure user management, product cataloging, and order processing. Leveraging Laravel’s MVC framework and MySQL’s relational database ensures scalability and ease of future enhancements.

### 5.4 Challenges Faced

During the development of the eCommerce platform, several challenges were encountered, which required careful consideration and problem-solving:

1. **Integrating Frontend and Backend Seamlessly**  
   Ensuring smooth communication between the HTML/CSS/JavaScript frontend and the Laravel backend was initially complex. Managing asynchronous requests and dynamically updating the UI without page reloads required careful handling of AJAX and API endpoints.
2. **User Authentication and Security**  
   Implementing secure user authentication with role-based access control (for customers, sellers, and admins) was challenging. Protecting sensitive data, preventing unauthorized access, and managing sessions securely required leveraging Laravel’s built-in security features and additional middleware.
3. **Database Design and Relationships**  
   Designing a normalized database schema that supports various entities like users, products, orders, and carts while maintaining data integrity and efficient querying was complex. Handling many-to-many relationships (e.g., products and categories) and ensuring smooth CRUD operations posed initial difficulties.
4. **Shopping Cart and Order Management**  
   Managing the shopping cart’s state across sessions and ensuring accurate inventory updates upon order placement required synchronization between frontend state and backend data. Handling edge cases such as stock unavailability and concurrent order requests added complexity.
5. **Responsive and Cross-Browser Compatibility**  
   Creating a consistent user experience across different devices and browsers was time-consuming. Ensuring that the layout and interactive elements work well on mobile, tablet, and desktop required extensive testing and CSS adjustments.
6. **Payment Gateway Integration**  
   Although not fully implemented in this version, integrating secure payment gateways (e.g., Stripe, PayPal) involves handling sensitive financial data and ensuring PCI compliance, which presents future challenges.
7. **Performance Optimization**  
   As product data grew, optimizing database queries and frontend loading times became necessary to maintain responsiveness and avoid slowdown

### 5.5 Future Enhancements

To further improve the functionality, usability, and scalability of the eCommerce platform, several enhancements are planned for future development:

1. **Payment Gateway Integration**  
   Implementing secure and versatile payment gateways such as **Stripe**, **PayPal**, or **Razorpay** will enable users to make smooth, reliable, and diverse payment transactions online. This will enhance the checkout experience by supporting multiple payment options, ensuring PCI compliance, and providing secure processing of credit/debit card payments, net banking, and digital wallets.
2. **Advanced Search and Filtering**  
   Adding more sophisticated search capabilities with filters like brand, price range, customer reviews, and discounts will help users find products more efficiently.
3. **User Reviews and Ratings**  
   Allowing customers to leave product reviews and ratings will build trust and assist other buyers in making informed decisions.
4. **Personalized Recommendations**  
   Integrating machine learning algorithms to offer personalized product recommendations based on user behavior and purchase history.
5. **Mobile Application**  
   Developing native mobile apps for Android and iOS platforms to provide a seamless shopping experience on smartphones and tablets.
6. **Multi-language and Multi-currency Support**  
   Expanding accessibility by supporting multiple languages and currencies, catering to a broader international audience.
7. **Enhanced Admin Dashboard**  
   Providing admin with advanced analytics, inventory alerts, and promotional tools to manage their storefronts more effectively.
8. **Order Tracking and Notifications**  
   Implementing real-time order tracking and automated notifications via email or SMS to keep customers informed about their purchase status.
9. **Improved Security Measures**  
   Adding features like two-factor authentication (2FA), CAPTCHA on forms, and enhanced data encryption to strengthen platform security.
10. **Scalability and Cloud Deployment**  
    Migrating the platform to cloud services such as AWS or Azure for better scalability, performance, and uptime.

**CHAPTER 6**

**CODING**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Banner extends Model

{

    protected $fillable=['title','slug','description','photo','status'];

}

**Banner page**

<?php

namespace App\Models;

use App\Models\Product;

use Illuminate\Database\Eloquent\Model;

class Brand extends Model

{

    protected $fillable=['title','slug','status'];

    // public static function getProductByBrand($id){

    //     return Product::where('brand\_id',$id)->paginate(10);

    // }

    public function products(){

        return $this->hasMany('App\Models\Product','brand\_id','id')->where('status','active');

    }

    public static function getProductByBrand($slug){

        // dd($slug);

        return Brand::with('products')->where('slug',$slug)->first();

        // return Product::where('cat\_id',$id)->where('child\_cat\_id',null)->paginate(10);

    }

}

**Brand Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Cart extends Model

{

    protected $fillable=['user\_id','product\_id','order\_id','quantity','amount','price','status'];

    // public function product(){

    //     return $this->hasOne('App\Models\Product','id','product\_id');

    // }

    // public static function getAllProductFromCart(){

    //     return Cart::with('product')->where('user\_id',auth()->user()->id)->get();

    // }

    public function product()

    {

        return $this->belongsTo(Product::class, 'product\_id');

    }

    public function order(){

        return $this->belongsTo(Order::class,'order\_id');

    }

}

**Cart Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Category extends Model

{

    protected $fillable=['title','slug','summary','photo','status','is\_parent','parent\_id','added\_by'];

    public function parent\_info(){

        return $this->hasOne('App\Models\Category','id','parent\_id');

    }

    public static function getAllCategory(){

        return  Category::orderBy('id','DESC')->with('parent\_info')->paginate(10);

    }

    public static function shiftChild($cat\_id){

        return Category::whereIn('id',$cat\_id)->update(['is\_parent'=>1]);

    }

    public static function getChildByParentID($id){

        return Category::where('parent\_id',$id)->orderBy('id','ASC')->pluck('title','id');

    }

    public function child\_cat(){

        return $this->hasMany('App\Models\Category','parent\_id','id')->where('status','active');

    }

    public static function getAllParentWithChild(){

        return Category::with('child\_cat')->where('is\_parent',1)->where('status','active')->orderBy('title','ASC')->get();

    }

    public function products(){

        return $this->hasMany('App\Models\Product','cat\_id','id')->where('status','active');

    }

    public function sub\_products(){

        return $this->hasMany('App\Models\Product','child\_cat\_id','id')->where('status','active');

    }

    public static function getProductByCat($slug){

        // dd($slug);

        return Category::with('products')->where('slug',$slug)->first();

        // return Product::where('cat\_id',$id)->where('child\_cat\_id',null)->paginate(10);

    }

    public static function getProductBySubCat($slug){

        // return $slug;

        return Category::with('sub\_products')->where('slug',$slug)->first();

    }

    public static function countActiveCategory(){

        $data=Category::where('status','active')->count();

        if($data){

            return $data;

        }

        return 0;

    }

}

**Category Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Coupon extends Model

{

    protected $fillable=['code','type','value','status'];

    public static function findByCode($code){

        return self::where('code',$code)->first();

    }

    public function discount($total){

        if($this->type=="fixed"){

            return $this->value;

        }

        elseif($this->type=="percent"){

            return ($this->value /100)\*$total;

        }

        else{

            return 0;

        }

    }

}

**Coupon Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Message extends Model

{

    public $fillable=['name','message','email','phone','read\_at','photo','subject'];

}

**Message Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Notification extends Model

{

    protected $fillable=['data','type','notifiable','read\_at'];

}

**Notification Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Order extends Model

{

    protected $fillable=['user\_id','order\_number','sub\_total','quantity','delivery\_charge','status','total\_amount','first\_name','last\_name','country','post\_code','address1','address2','phone','email','payment\_method','payment\_status','shipping\_id','coupon'];

    public function cart\_info(){

        return $this->hasMany('App\Models\Cart','order\_id','id');

    }

    public static function getAllOrder($id){

        return Order::with('cart\_info')->find($id);

    }

    public static function countActiveOrder(){

        $data=Order::count();

        if($data){

            return $data;

        }

        return 0;

    }

    public function cart(){

        return $this->hasMany(Cart::class);

    }

    public function shipping(){

        return $this->belongsTo(Shipping::class,'shipping\_id');

    }

    public function user()

    {

        return $this->belongsTo('App\User', 'user\_id');

    }

}

**Order Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Post extends Model

{

    protected $fillable=['title','tags','summary','slug','description','photo','quote','post\_cat\_id','post\_tag\_id','added\_by','status'];

    public function cat\_info(){

        return $this->hasOne('App\Models\PostCategory','id','post\_cat\_id');

    }

    public function tag\_info(){

        return $this->hasOne('App\Models\PostTag','id','post\_tag\_id');

    }

    public function author\_info(){

        return $this->hasOne('App\User','id','added\_by');

    }

    public static function getAllPost(){

        return Post::with(['cat\_info','author\_info'])->orderBy('id','DESC')->paginate(10);

    }

    // public function get\_comments(){

    //     return $this->hasMany('App\Models\PostComment','post\_id','id');

    // }

    public static function getPostBySlug($slug){

        return Post::with(['tag\_info','author\_info'])->where('slug',$slug)->where('status','active')->first();

    }

    public function comments(){

        return $this->hasMany(PostComment::class)->whereNull('parent\_id')->where('status','active')->with('user\_info')->orderBy('id','DESC');

    }

    public function allComments(){

        return $this->hasMany(PostComment::class)->where('status','active');

    }

    // public static function getProductByCat($slug){

    //     // dd($slug);

    //     return Category::with('products')->where('slug',$slug)->first();

    //     // return Product::where('cat\_id',$id)->where('child\_cat\_id',null)->paginate(10);

    // }

    // public static function getBlogByCategory($id){

    //     return Post::where('post\_cat\_id',$id)->paginate(8);

    // }

    public static function getBlogByTag($slug){

        // dd($slug);

        return Post::where('tags',$slug)->paginate(8);

    }

    public static function countActivePost(){

        $data=Post::where('status','active')->count();

        if($data){

            return $data;

        }

        return 0;

    }

}

**Post Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

use App\Models\Post;

class PostCategory extends Model

{

    protected $fillable=['title','slug','status'];

    public function post(){

        return $this->hasMany('App\Models\Post','post\_cat\_id','id')->where('status','active');

    }

    public static function getBlogByCategory($slug){

        return PostCategory::with('post')->where('slug',$slug)->first();

    }

}

**Post Category Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class PostComment extends Model

{

    protected $fillable=['user\_id','post\_id','comment','replied\_comment','parent\_id','status'];

    public function user\_info(){

        return $this->hasOne('App\User','id','user\_id');

    }

    public static function getAllComments(){

        return PostComment::with('user\_info')->paginate(10);

    }

    public static function getAllUserComments(){

        return PostComment::where('user\_id',auth()->user()->id)->with('user\_info')->paginate(10);

    }

    public function post(){

        return $this->belongsTo(Post::class,'post\_id','id');

    }

    public function replies(){

        return $this->hasMany(PostComment::class,'parent\_id')->where('status','active');

    }

}

**Post Comment Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

use App\Models\Cart;

class Product extends Model

{

    protected $fillable=['title','slug','summary','description','cat\_id','child\_cat\_id','price','brand\_id','discount','status','photo','size','stock','is\_featured','condition'];

    public function cat\_info(){

        return $this->hasOne('App\Models\Category','id','cat\_id');

    }

    public function sub\_cat\_info(){

        return $this->hasOne('App\Models\Category','id','child\_cat\_id');

    }

    public static function getAllProduct(){

        return Product::with(['cat\_info','sub\_cat\_info'])->orderBy('id','desc')->paginate(10);

    }

    public function rel\_prods(){

        return $this->hasMany('App\Models\Product','cat\_id','cat\_id')->where('status','active')->orderBy('id','DESC')->limit(8);

    }

    public function getReview(){

        return $this->hasMany('App\Models\ProductReview','product\_id','id')->with('user\_info')->where('status','active')->orderBy('id','DESC');

    }

    public static function getProductBySlug($slug){

        return Product::with(['cat\_info','rel\_prods','getReview'])->where('slug',$slug)->first();

    }

    public static function countActiveProduct(){

        $data=Product::where('status','active')->count();

        if($data){

            return $data;

        }

        return 0;

    }

    public function carts(){

        return $this->hasMany(Cart::class)->whereNotNull('order\_id');

    }

    public function wishlists(){

        return $this->hasMany(Wishlist::class)->whereNotNull('cart\_id');

    }

    public function brand(){

        return $this->hasOne(Brand::class,'id','brand\_id');

    }

}

**Products Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class ProductReview extends Model

{

    protected $fillable=['user\_id','product\_id','rate','review','status'];

    public function user\_info(){

        return $this->hasOne('App\User','id','user\_id');

    }

    public static function getAllReview(){

        return ProductReview::with('user\_info')->paginate(10);

    }

    public static function getAllUserReview(){

        return ProductReview::where('user\_id',auth()->user()->id)->with('user\_info')->paginate(10);

    }

    public function product(){

        return $this->hasOne(Product::class,'id','product\_id');

    }

}

**Product Review Page**

<?php

namespace App\Events;

use App\Models\Message;

use Illuminate\Broadcasting\Channel;

use Illuminate\Broadcasting\InteractsWithSockets;

use Illuminate\Broadcasting\PresenceChannel;

use Illuminate\Broadcasting\PrivateChannel;

use Illuminate\Contracts\Broadcasting\ShouldBroadcast;

use Illuminate\Foundation\Events\Dispatchable;

use Illuminate\Queue\SerializesModels;

class MessageSent implements ShouldBroadcast

{

    use Dispatchable, InteractsWithSockets, SerializesModels;

    /\*\*

     \* Create a new event instance.

     \*

     \* @return void

     \*/

    public $message;

    public function \_\_construct($message)

    {

        // return $message;

        // dd($message);

        $this->message=$message;

    }

    /\*\*

     \* Get the channels the event should broadcast on.

     \*

     \* @return \Illuminate\Broadcasting\Channel|array

     \*/

    public function broadcastOn()

    {

        return new Channel('message');

    }

}

**Message Sent Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Settings extends Model

{

    protected $fillable=['short\_des','description','photo','address','phone','email','logo'];

}

Settings page

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Shipping extends Model

{

    protected $fillable=['type','price','status'];

}

**Shipping Page**

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Wishlist extends Model

{

    protected $fillable=['user\_id','product\_id','cart\_id','price','amount','quantity'];

    public function product(){

        return $this->belongsTo(Product::class,'product\_id');

    }

}

**Wishlist Page**

<?php

namespace App\Notifications;

use Illuminate\Bus\Queueable;

use Illuminate\Contracts\Queue\ShouldQueue;

use Illuminate\Notifications\Messages\MailMessage;

use Illuminate\Notifications\Notification;

use Illuminate\Notifications\Messages\BroadcastMessage;

class StatusNotification extends Notification

{

    use Queueable;

    private $details;

    /\*\*

     \* Create a new notification instance.

     \*

     \* @return void

     \*/

    public function \_\_construct($details)

    {

        $this->details=$details;

    }

    /\*\*

     \* Get the notification's delivery channels.

     \*

     \* @param  mixed  $notifiable

     \* @return array

     \*/

    public function via($notifiable)

    {

        return ['database','broadcast'];

    }

    /\*\*

     \* Get the mail representation of the notification.

     \*

     \* @param  mixed  $notifiable

     \* @return \Illuminate\Notifications\Messages\MailMessage

     \*/

    // public function toMail($notifiable)

    // {

    //     return (new MailMessage)

    //                 ->subject('Status Notification')

    //                 ->from(env('MAIL\_USERNAME','test@gmail.com'),'E-shop')

    //                 ->line($this->details['title'])

    //                 ->action('View Order', $this->details['actionURL'])

    //                 ->line('Thank you!');

    // }

    /\*\*

     \* Get the array representation of the notification.

     \*

     \* @param  mixed  $notifiable

     \* @return array

     \*/

    // public function toArray($notifiable)

    // {

    //     return [

    //         'title'=>$this->details['title'],

    //         'actionURL'=>$this->details['actionURL'],

    //         'fas'=>$this->details['fas']

    //     ];

    // }

    public function toArray($notifiable)

    {

        return [

            'title'=>$this->details['title'],

            'actionURL'=>$this->details['actionURL'],

            'fas'=>$this->details['fas']

        ];

    }

    /\*\*

     \* Get the broadcastable representation of the notification.

     \*

     \* @param  mixed  $notifiable

     \* @return BroadcastMessage

     \*/

    public function toBroadcast($notifiable)

    {

        return new BroadcastMessage([

            'title' => $this->details['title'],

            'actionURL' => $this->details['actionURL'],

            'url' => route('admin.notification', $this->id),

            'fas' => $this->details['fas'],

            'time' => date('F d, Y h:i A')

        ]);

    }

}

**Status Notification Page**

<?php

namespace App;

use Illuminate\Contracts\Auth\MustVerifyEmail;

use Illuminate\Foundation\Auth\User as Authenticatable;

use Illuminate\Notifications\Notifiable;

class User extends Authenticatable

{

    use Notifiable;

    /\*\*

     \* The attributes that are mass assignable.

     \*

     \* @var array

     \*/

    protected $fillable = [

        'name', 'email', 'password','role','photo','status','provider','provider\_id',

    ];

    /\*\*

     \* The attributes that should be hidden for arrays.

     \*

     \* @var array

     \*/

    protected $hidden = [

        'password', 'remember\_token',

    ];

    /\*\*

     \* The attributes that should be cast to native types.

     \*

     \* @var array

     \*/

    protected $casts = [

        'email\_verified\_at' => 'datetime',

    ];

    public function orders(){

        return $this->hasMany('App\Models\Order');

    }

}

**User Page**

<?php

use Illuminate\Support\Str;

return [

    /\*

    |--------------------------------------------------------------------------

    | Default Database Connection Name

    |--------------------------------------------------------------------------

    |

    | Here you may specify which of the database connections below you wish

    | to use as your default connection for all database work. Of course

    | you may use many connections at once using the Database library.

    |

    \*/

    'default' => env('DB\_CONNECTION', 'mysql'),

    /\*

    |--------------------------------------------------------------------------

    | Database Connections

    |--------------------------------------------------------------------------

    |

    | Here are each of the database connections setup for your application.

    | Of course, examples of configuring each database platform that is

    | supported by Laravel is shown below to make development simple.

    |

    |

    | All database work in Laravel is done through the PHP PDO facilities

    | so make sure you have the driver for your particular database of

    | choice installed on your machine before you begin development.

    |

    \*/

    'connections' => [

        'sqlite' => [

            'driver' => 'sqlite',

            'url' => env('DATABASE\_URL'),

            'database' => env('DB\_DATABASE', database\_path('database.sqlite')),

            'prefix' => '',

            'foreign\_key\_constraints' => env('DB\_FOREIGN\_KEYS', true),

        ],

        'mysql' => [

            'driver' => 'mysql',

            'url' => env('DATABASE\_URL'),

            'host' => env('DB\_HOST', '127.0.0.1'),

            'port' => env('DB\_PORT', '3306'),

            'database' => env('DB\_DATABASE', 'forge'),

            'username' => env('DB\_USERNAME', 'forge'),

            'password' => env('DB\_PASSWORD', ''),

            'unix\_socket' => env('DB\_SOCKET', ''),

            'charset' => 'utf8mb4',

            'collation' => 'utf8mb4\_unicode\_ci',

            'prefix' => '',

            'prefix\_indexes' => true,

            'strict' => true,

            'engine' => null,

            'options' => extension\_loaded('pdo\_mysql') ? array\_filter([

                PDO::MYSQL\_ATTR\_SSL\_CA => env('MYSQL\_ATTR\_SSL\_CA'),

            ]) : [],

        ],

        'pgsql' => [

            'driver' => 'pgsql',

            'url' => env('DATABASE\_URL'),

            'host' => env('DB\_HOST', '127.0.0.1'),

            'port' => env('DB\_PORT', '5432'),

            'database' => env('DB\_DATABASE', 'forge'),

            'username' => env('DB\_USERNAME', 'forge'),

            'password' => env('DB\_PASSWORD', ''),

            'charset' => 'utf8',

            'prefix' => '',

            'prefix\_indexes' => true,

            'schema' => 'public',

            'sslmode' => 'prefer',

        ],

        'sqlsrv' => [

            'driver' => 'sqlsrv',

            'url' => env('DATABASE\_URL'),

            'host' => env('DB\_HOST', 'localhost'),

            'port' => env('DB\_PORT', '1433'),

            'database' => env('DB\_DATABASE', 'forge'),

            'username' => env('DB\_USERNAME', 'forge'),

            'password' => env('DB\_PASSWORD', ''),

            'charset' => 'utf8',

            'prefix' => '',

            'prefix\_indexes' => true,

        ],

    ],

    /\*

    |--------------------------------------------------------------------------

    | Migration Repository Table

    |--------------------------------------------------------------------------

    |

    | This table keeps track of all the migrations that have already run for

    | your application. Using this information, we can determine which of

    | the migrations on disk haven't actually been run in the database.

    |

    \*/

    'migrations' => 'migrations',

    /\*

    |--------------------------------------------------------------------------

    | Redis Databases

    |--------------------------------------------------------------------------

    |

    | Redis is an open source, fast, and advanced key-value store that also

    | provides a richer body of commands than a typical key-value system

    | such as APC or Memcached. Laravel makes it easy to dig right in.

    |

    \*/

    'redis' => [

        'client' => env('REDIS\_CLIENT', 'phpredis'),

        'options' => [

            'cluster' => env('REDIS\_CLUSTER', 'redis'),

            'prefix' => env('REDIS\_PREFIX', Str::slug(env('APP\_NAME', 'laravel'), '\_').'\_database\_'),

        ],

        'default' => [

            'url' => env('REDIS\_URL'),

            'host' => env('REDIS\_HOST', '127.0.0.1'),

            'password' => env('REDIS\_PASSWORD', null),

            'port' => env('REDIS\_PORT', '6379'),

            'database' => env('REDIS\_DB', '0'),

        ],

        'cache' => [

            'url' => env('REDIS\_URL'),

            'host' => env('REDIS\_HOST', '127.0.0.1'),

            'password' => env('REDIS\_PASSWORD', null),

            'port' => env('REDIS\_PORT', '6379'),

            'database' => env('REDIS\_CACHE\_DB', '1'),

        ],

    ],

];

**Database Page**

<?php

/\*\*

 \* PayPal Setting & API Credentials

 \* Created by Raza Mehdi <srmk@outlook.com>.

 \*/

return [

    'mode'    => env('PAYPAL\_MODE', 'sandbox'), // Can only be 'sandbox' Or 'live'. If empty or invalid, 'live' will be used.

    'sandbox' => [

        'username'    => env('PAYPAL\_SANDBOX\_API\_USERNAME', ''),

        'password'    => env('PAYPAL\_SANDBOX\_API\_PASSWORD', ''),

        'secret'      => env('PAYPAL\_SANDBOX\_API\_SECRET', ''),

        'certificate' => env('PAYPAL\_SANDBOX\_API\_CERTIFICATE', ''),

        'app\_id'      => 'APP-80W284485P519543T', // Used for testing Adaptive Payments API in sandbox mode

    ],

    'live' => [

        'username'    => env('PAYPAL\_LIVE\_API\_USERNAME', ''),

        'password'    => env('PAYPAL\_LIVE\_API\_PASSWORD', ''),

        'secret'      => env('PAYPAL\_LIVE\_API\_SECRET', ''),

        'certificate' => env('PAYPAL\_LIVE\_API\_CERTIFICATE', ''),

        'app\_id'      => '', // Used for Adaptive Payments API

    ],

    'payment\_action' => 'Sale', // Can only be 'Sale', 'Authorization' or 'Order'

    'currency'       => env('PAYPAL\_CURRENCY', 'USD'),

    'billing\_type'   => 'MerchantInitiatedBilling',

    'notify\_url'     => '', // Change this accordingly for your application.

    'locale'         => '', // force gateway language  i.e. it\_IT, es\_ES, en\_US ... (for express checkout only)

    'validate\_ssl'   => true, // Validate SSL when creating api client.

];

**Paypal Page**

<?php

return [

    'mailgun' => [

        'domain' => env('MAILGUN\_DOMAIN'),

        'secret' => env('MAILGUN\_SECRET'),

        'endpoint' => env('MAILGUN\_ENDPOINT', 'api.mailgun.net'),

    ],

    'postmark' => [

        'token' => env('POSTMARK\_TOKEN'),

    ],

    'ses' => [

        'key' => env('AWS\_ACCESS\_KEY\_ID'),

        'secret' => env('AWS\_SECRET\_ACCESS\_KEY'),

        'region' => env('AWS\_DEFAULT\_REGION', 'us-east-1'),

    ],

    'github' => [

        'client\_id' => 'YOUR\_GITHUB\_API', //Github API

        'client\_secret' => 'YOUR\_GITHUB\_SECRET', //Github Secret

        'redirect' => 'http://localhost:8000/login/github/callback',

     ],

     'google' => [

        'client\_id' => 'YOUR\_GOOGLE\_API', //Google API

        'client\_secret' => 'YOUR\_GOOGLE\_SECRET', //Google Secret

        'redirect' => 'http://localhost:8000/login/google/callback',

     ],

     'facebook' => [

        'client\_id' => 'YOUR\_FACEBOOK\_API', //Facebook API

        'client\_secret' => 'YOUR\_FACEBOK\_SECRET', //Facebook Secret

        'redirect' => 'http://localhost:8000/login/facebook/callback',

     ],

];

**Services Page**

CREATE TABLE `banners` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `title` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `slug` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `photo` varchar(191) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `description` text COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `status` enum('active','inactive') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'inactive',

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `banners`

--

INSERT INTO `banners` (`id`, `title`, `slug`, `photo`, `description`, `status`, `created\_at`, `updated\_at`) VALUES

(1, 'Lorem Ipsum is', 'lorem-ipsum-is', '/storage/photos/1/Banner/banner-01.jpg', '<h2><span style=\"font-weight: bold; color: rgb(99, 99, 99);\">Up to 10%</span></h2>', 'active', '2020-08-14 01:47:38', '2020-08-14 01:48:21'),

(2, 'Lorem Ipsum', 'lorem-ipsum', '/storage/photos/1/Banner/banner-07.jpg', '<p>Up to 90%</p>', 'active', '2020-08-14 01:50:23', '2020-08-14 01:50:23'),

(4, 'Banner', 'banner', '/storage/photos/1/Banner/banner-06.jpg', '<h2><span style=\"color: rgb(156, 0, 255); font-size: 2rem; font-weight: bold;\">Up to 40%</span><br></h2><h2><span style=\"color: rgb(156, 0, 255);\"></span></h2>', 'active', '2020-08-17 20:46:59', '2020-08-17 20:46:59');

-- --------------------------------------------------------

--

-- Table structure for table `brands`

--

CREATE TABLE `brands` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `title` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `slug` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `status` enum('active','inactive') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'active',

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `brands`

--

INSERT INTO `brands` (`id`, `title`, `slug`, `status`, `created\_at`, `updated\_at`) VALUES

(1, 'Adidas', 'adidas', 'active', '2020-08-14 04:23:00', '2020-08-14 04:23:00'),

(2, 'Nike', 'nike', 'active', '2020-08-14 04:23:08', '2020-08-14 04:23:08'),

(3, 'Kappa', 'kappa', 'active', '2020-08-14 04:23:48', '2020-08-14 04:23:48'),

(4, 'Prada', 'prada', 'active', '2020-08-14 04:24:08', '2020-08-14 04:24:08'),

(6, 'Brand', 'brand', 'active', '2020-08-17 20:50:31', '2020-08-17 20:50:31');

-- --------------------------------------------------------

--

-- Table structure for table `carts`

--

CREATE TABLE `carts` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `product\_id` bigint(20) UNSIGNED NOT NULL,

  `order\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `user\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `price` double(8,2) NOT NULL,

  `status` enum('new','progress','delivered','cancel') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'new',

  `quantity` int(11) NOT NULL,

  `amount` double(8,2) NOT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `carts`

--

INSERT INTO `carts` (`id`, `product\_id`, `order\_id`, `user\_id`, `price`, `status`, `quantity`, `amount`, `created\_at`, `updated\_at`) VALUES

(1, 8, 1, 3, 200.00, 'new', 2, 400.00, '2020-08-14 07:15:45', '2020-08-14 07:20:45'),

(2, 7, 1, 3, 1939.03, 'new', 1, 1999.00, '2020-08-14 07:15:59', '2020-08-14 07:20:45'),

(3, 5, 1, 3, 3600.00, 'new', 3, 12000.00, '2020-08-14 07:16:12', '2020-08-14 07:20:45'),

(4, 7, 2, 2, 1939.03, 'new', 1, 1939.03, '2020-08-14 22:13:51', '2020-08-14 22:14:59'),

(5, 8, 3, 3, 200.00, 'new', 1, 200.00, '2020-08-15 06:39:59', '2020-08-15 06:41:00'),

(8, 9, 4, 3, 190.00, 'new', 2, 380.00, '2020-08-15 07:44:53', '2020-08-15 07:54:53'),

(9, 6, 4, 3, 5820.00, 'new', 4, 23280.00, '2020-08-15 07:45:29', '2020-08-15 07:54:53'),

(10, 10, NULL, 2, 270.00, 'new', 1, 270.00, '2020-08-17 21:07:33', '2020-08-17 21:17:03'),

(11, 9, NULL, 2, 190.00, 'new', 2, 380.00, '2020-08-17 21:08:35', '2020-08-17 21:17:03');

-- --------------------------------------------------------

--

-- Table structure for table `categories`

--

CREATE TABLE `categories` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `title` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `slug` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `summary` text COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `photo` varchar(191) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `is\_parent` tinyint(1) NOT NULL DEFAULT 1,

  `parent\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `added\_by` bigint(20) UNSIGNED DEFAULT NULL,

  `status` enum('active','inactive') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'inactive',

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `categories`

--

INSERT INTO `categories` (`id`, `title`, `slug`, `summary`, `photo`, `is\_parent`, `parent\_id`, `added\_by`, `status`, `created\_at`, `updated\_at`) VALUES

(1, 'Men\'s Fashion', 'mens-fashion', NULL, '/storage/photos/1/Category/mini-banner1.jpg', 1, NULL, NULL, 'active', '2020-08-14 04:26:15', '2020-08-14 04:26:15'),

(2, 'Women\'s Fashion', 'womens-fashion', NULL, '/storage/photos/1/Category/mini-banner2.jpg', 1, NULL, NULL, 'active', '2020-08-14 04:26:40', '2020-08-14 04:26:40'),

(3, 'Kid\'s', 'kids', NULL, '/storage/photos/1/Category/mini-banner3.jpg', 1, NULL, NULL, 'active', '2020-08-14 04:27:10', '2020-08-14 04:27:42'),

(4, 'T-shirt\'s', 't-shirts', NULL, NULL, 0, 1, NULL, 'active', '2020-08-14 04:32:14', '2020-08-14 04:32:14'),

(5, 'Jeans pants', 'jeans-pants', NULL, NULL, 0, 1, NULL, 'active', '2020-08-14 04:32:49', '2020-08-14 04:32:49'),

(6, 'Sweater & Jackets', 'sweater-jackets', NULL, NULL, 0, 1, NULL, 'active', '2020-08-14 04:33:37', '2020-08-14 04:33:37'),

(7, 'Rain Coats & Trenches', 'rain-coats-trenches', NULL, NULL, 0, 1, NULL, 'active', '2020-08-14 04:34:04', '2020-08-14 04:34:04');

-- --------------------------------------------------------

--

-- Table structure for table `coupons`

--

CREATE TABLE `coupons` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `code` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `type` enum('fixed','percent') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'fixed',

  `value` decimal(20,2) NOT NULL,

  `status` enum('active','inactive') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'inactive',

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `coupons`

--

INSERT INTO `coupons` (`id`, `code`, `type`, `value`, `status`, `created\_at`, `updated\_at`) VALUES

(1, 'abc123', 'fixed', '300.00', 'active', NULL, NULL),

(2, '111111', 'percent', '10.00', 'active', NULL, NULL),

(5, 'abcd', 'fixed', '250.00', 'active', '2020-08-17 20:54:58', '2020-08-17 20:54:58');

-- --------------------------------------------------------

--

-- Table structure for table `failed\_jobs`

--

CREATE TABLE `failed\_jobs` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `connection` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `queue` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `payload` longtext COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `exception` longtext COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `failed\_at` timestamp NOT NULL DEFAULT current\_timestamp()

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Table structure for table `messages`

--

CREATE TABLE `messages` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `name` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `subject` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `email` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `photo` varchar(191) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `phone` varchar(191) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `message` longtext COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `read\_at` timestamp NULL DEFAULT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `messages`

--

INSERT INTO `messages` (`id`, `name`, `subject`, `email`, `photo`, `phone`, `message`, `read\_at`, `created\_at`, `updated\_at`) VALUES

(1, 'Prajwal Rai', 'About price', 'prajwal.iar@gmail.com', NULL, '9807009999', 'Hello sir i am from kathmandu nepal.', '2020-08-14 08:25:46', '2020-08-14 08:00:01', '2020-08-14 08:25:46'),

(2, 'Prajwal Rai', 'About Price', 'prajwal.iar@gmail.com', NULL, '9800099000', 'Hello i am Prajwal Rai', '2020-08-18 03:04:15', '2020-08-15 07:52:39', '2020-08-18 03:04:16'),

(3, 'Prajwal Rai', 'lorem ipsum', 'prajwal.iar@gmail.com', NULL, '1200990009', 'hello sir sdfdfd dfdjf ;dfjd fd ldkfd', NULL, '2020-08-17 21:15:12', '2020-08-17 21:15:12');

-- --------------------------------------------------------

--

-- Table structure for table `migrations`

--

CREATE TABLE `migrations` (

  `id` int(10) UNSIGNED NOT NULL,

  `migration` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `batch` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `migrations`

--

INSERT INTO `migrations` (`id`, `migration`, `batch`) VALUES

(1, '2014\_10\_12\_000000\_create\_users\_table', 1),

(2, '2014\_10\_12\_100000\_create\_password\_resets\_table', 1),

(3, '2019\_08\_19\_000000\_create\_failed\_jobs\_table', 1),

(4, '2020\_07\_10\_021010\_create\_brands\_table', 1),

(5, '2020\_07\_10\_025334\_create\_banners\_table', 1),

(6, '2020\_07\_10\_112147\_create\_categories\_table', 1),

(7, '2020\_07\_11\_063857\_create\_products\_table', 1),

(8, '2020\_07\_12\_073132\_create\_post\_categories\_table', 1),

(9, '2020\_07\_12\_073701\_create\_post\_tags\_table', 1),

(10, '2020\_07\_12\_083638\_create\_posts\_table', 1),

(11, '2020\_07\_13\_151329\_create\_messages\_table', 1),

(12, '2020\_07\_14\_023748\_create\_shippings\_table', 1),

(13, '2020\_07\_15\_054356\_create\_orders\_table', 1),

(14, '2020\_07\_15\_102626\_create\_carts\_table', 1),

(15, '2020\_07\_16\_041623\_create\_notifications\_table', 1),

(16, '2020\_07\_16\_053240\_create\_coupons\_table', 1),

(17, '2020\_07\_23\_143757\_create\_wishlists\_table', 1),

(18, '2020\_07\_24\_074930\_create\_product\_reviews\_table', 1),

(19, '2020\_07\_24\_131727\_create\_post\_comments\_table', 1),

(20, '2020\_08\_01\_143408\_create\_settings\_table', 1);

-- --------------------------------------------------------

--

-- Table structure for table `notifications`

--

CREATE TABLE `notifications` (

  `id` char(36) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `type` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `notifiable\_type` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `notifiable\_id` bigint(20) UNSIGNED NOT NULL,

  `data` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `read\_at` timestamp NULL DEFAULT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `notifications`

--

INSERT INTO `notifications` (`id`, `type`, `notifiable\_type`, `notifiable\_id`, `data`, `read\_at`, `created\_at`, `updated\_at`) VALUES

('2145a8e3-687d-444a-8873-b3b2fb77a342', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New Comment created\",\"actionURL\":\"http:\\/\\/e-shop.loc\\/blog-detail\\/where-can-i-get-some\",\"fas\":\"fas fa-comment\"}', NULL, '2020-08-15 07:31:21', '2020-08-15 07:31:21'),

('3af39f84-cab4-4152-9202-d448435c67de', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New order created\",\"actionURL\":\"http:\\/\\/localhost:8000\\/admin\\/order\\/4\",\"fas\":\"fa-file-alt\"}', NULL, '2020-08-15 07:54:52', '2020-08-15 07:54:52'),

('4a0afdb0-71ad-4ce6-bc70-c92ef491a525', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New Comment created\",\"actionURL\":\"http:\\/\\/localhost:8000\\/blog-detail\\/the-standard-lorem-ipsum-passage-used-since-the-1500s\",\"fas\":\"fas fa-comment\"}', NULL, '2020-08-17 21:13:51', '2020-08-17 21:13:51'),

('540ca3e9-0ff9-4e2e-9db3-6b5abc823422', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New Comment created\",\"actionURL\":\"http:\\/\\/e-shop.loc\\/blog-detail\\/where-can-i-get-some\",\"fas\":\"fas fa-comment\"}', '2020-08-15 07:30:44', '2020-08-14 07:12:28', '2020-08-15 07:30:44'),

('5da09dd1-3ffc-43b0-aba2-a4260ba4cc76', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New Comment created\",\"actionURL\":\"http:\\/\\/localhost:8000\\/blog-detail\\/the-standard-lorem-ipsum-passage\",\"fas\":\"fas fa-comment\"}', NULL, '2020-08-15 07:51:02', '2020-08-15 07:51:02'),

('5e91e603-024e-45c5-b22f-36931fef0d90', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New Product Rating!\",\"actionURL\":\"http:\\/\\/localhost:8000\\/product-detail\\/white-sports-casual-t\",\"fas\":\"fa-star\"}', NULL, '2020-08-15 07:44:07', '2020-08-15 07:44:07'),

('73a3b51a-416a-4e7d-8ca2-53b216d9ad8e', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New Comment created\",\"actionURL\":\"http:\\/\\/e-shop.loc\\/blog-detail\\/where-can-i-get-some\",\"fas\":\"fas fa-comment\"}', NULL, '2020-08-14 07:11:03', '2020-08-14 07:11:03'),

('8605db5d-1462-496e-8b5f-8b923d88912c', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New order created\",\"actionURL\":\"http:\\/\\/e-shop.loc\\/admin\\/order\\/1\",\"fas\":\"fa-file-alt\"}', NULL, '2020-08-14 07:20:44', '2020-08-14 07:20:44'),

('a6ec5643-748c-4128-92e2-9a9f293f53b5', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New order created\",\"actionURL\":\"http:\\/\\/localhost:8000\\/admin\\/order\\/5\",\"fas\":\"fa-file-alt\"}', NULL, '2020-08-17 21:17:03', '2020-08-17 21:17:03'),

('b186a883-42f2-4a05-8fc5-f0d3e10309ff', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New order created\",\"actionURL\":\"http:\\/\\/e-shop.loc\\/admin\\/order\\/2\",\"fas\":\"fa-file-alt\"}', '2020-08-15 04:17:24', '2020-08-14 22:14:55', '2020-08-15 04:17:24'),

('d2fd7c33-b0fe-47d6-8bc6-f377d404080d', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New Comment created\",\"actionURL\":\"http:\\/\\/e-shop.loc\\/blog-detail\\/where-can-i-get-some\",\"fas\":\"fas fa-comment\"}', NULL, '2020-08-14 07:08:50', '2020-08-14 07:08:50'),

('dff78b90-85c8-42ee-a5b1-de8ad0b21be4', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New order created\",\"actionURL\":\"http:\\/\\/e-shop.loc\\/admin\\/order\\/3\",\"fas\":\"fa-file-alt\"}', NULL, '2020-08-15 06:40:54', '2020-08-15 06:40:54'),

('e28b0a73-4819-4016-b915-0e525d4148f5', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New Product Rating!\",\"actionURL\":\"http:\\/\\/localhost:8000\\/product-detail\\/lorem-ipsum-is-simply\",\"fas\":\"fa-star\"}', NULL, '2020-08-17 21:08:16', '2020-08-17 21:08:16'),

('ffffa177-c54e-4dfe-ba43-27c466ff1f4b', 'App\\Notifications\\StatusNotification', 'App\\User', 1, '{\"title\":\"New Comment created\",\"actionURL\":\"http:\\/\\/localhost:8000\\/blog-detail\\/the-standard-lorem-ipsum-passage-used-since-the-1500s\",\"fas\":\"fas fa-comment\"}', NULL, '2020-08-17 21:13:29', '2020-08-17 21:13:29');

-- --------------------------------------------------------

--

-- Table structure for table `orders`

--

CREATE TABLE `orders` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `order\_number` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `user\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `sub\_total` double(8,2) NOT NULL,

  `shipping\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `coupon` double(8,2) DEFAULT NULL,

  `total\_amount` double(8,2) NOT NULL,

  `quantity` int(11) NOT NULL,

  `payment\_method` enum('cod','paypal') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'cod',

  `payment\_status` enum('paid','unpaid') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'unpaid',

  `status` enum('new','process','delivered','cancel') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'new',

  `first\_name` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `last\_name` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `email` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `phone` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `country` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `post\_code` varchar(191) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `address1` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `address2` text COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `orders`

--

INSERT INTO `orders` (`id`, `order\_number`, `user\_id`, `sub\_total`, `shipping\_id`, `coupon`, `total\_amount`, `quantity`, `payment\_method`, `payment\_status`, `status`, `first\_name`, `last\_name`, `email`, `phone`, `country`, `post\_code`, `address1`, `address2`, `created\_at`, `updated\_at`) VALUES

(1, 'ORD-PMIQF5MYPK', 3, 14399.00, 1, 573.90, 13925.10, 6, 'cod', 'unpaid', 'delivered', 'Prajwal', 'Rai', 'prajwal.iar@gmail.com', '9800887778', 'NP', '44600', 'Koteshwor', 'Kathmandu', '2020-08-14 07:20:44', '2020-08-14 09:37:37'),

(2, 'ORD-YFF8BF0YBK', 2, 1939.03, 1, NULL, 2039.03, 1, 'cod', 'unpaid', 'delivered', 'Sandhya', 'Rai', 'user@gmail.com', '90000000990', 'NP', NULL, 'Lalitpur', NULL, '2020-08-14 22:14:49', '2020-08-14 22:15:19'),

(3, 'ORD-1CKWRWTTIK', 3, 200.00, 1, NULL, 300.00, 1, 'paypal', 'paid', 'process', 'Prajwal', 'Rai', 'prajwal.iar@gmail.com', '9807009999', 'NP', '44600', 'Kathmandu', 'Kadaghari', '2020-08-15 06:40:49', '2020-08-17 20:52:40'),

(4, 'ORD-HVO0KX0YHW', 3, 23660.00, 3, 150.00, 23910.00, 6, 'paypal', 'paid', 'new', 'Prajwal', 'Rai', 'prajwal.iar@gmail.com', '9800098878', 'NP', '44600', 'Pokhara', NULL, '2020-08-15 07:54:52', '2020-08-15 07:54:52');

-- --------------------------------------------------------

--

-- Table structure for table `password\_resets`

--

CREATE TABLE `password\_resets` (

  `email` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `token` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `created\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Table structure for table `posts`

--

CREATE TABLE `posts` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `title` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `slug` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `summary` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `description` longtext COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `quote` text COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `photo` varchar(191) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `tags` varchar(191) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `post\_cat\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `post\_tag\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `added\_by` bigint(20) UNSIGNED DEFAULT NULL,

  `status` enum('active','inactive') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'active',

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

CREATE TABLE `post\_categories` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `title` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `slug` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `status` enum('active','inactive') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'active',

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `post\_categories`

--

INSERT INTO `post\_categories` (`id`, `title`, `slug`, `status`, `created\_at`, `updated\_at`) VALUES

(1, 'Travel', 'contrary', 'active', '2020-08-14 01:51:03', '2020-08-14 01:51:39'),

(2, 'Electronics', 'richard', 'active', '2020-08-14 01:51:22', '2020-08-14 01:52:00'),

(3, 'Cloths', 'cloths', 'active', '2020-08-14 01:52:22', '2020-08-14 01:52:22'),

(4, 'enjoy', 'enjoy', 'active', '2020-08-14 03:16:10', '2020-08-14 03:16:10'),

(5, 'Post Category', 'post-category', 'active', '2020-08-15 06:59:04', '2020-08-15 06:59:04');

-- --------------------------------------------------------

--

-- Table structure for table `post\_comments`

--

CREATE TABLE `post\_comments` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `user\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `post\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `comment` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `status` enum('active','inactive') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'active',

  `replied\_comment` text COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `parent\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `post\_comments`

--

INSERT INTO `post\_comments` (`id`, `user\_id`, `post\_id`, `comment`, `status`, `replied\_comment`, `parent\_id`, `created\_at`, `updated\_at`) VALUES

(1, 1, 2, 'Testing comment edited', 'active', NULL, NULL, '2020-08-14 07:08:42', '2020-08-15 06:59:58'),

(2, 3, 2, 'testing 2', 'active', NULL, 1, '2020-08-14 07:11:03', '2020-08-14 07:11:03'),

(3, 2, 2, 'That\'s cool', 'active', NULL, 2, '2020-08-14 07:12:27', '2020-08-14 07:12:27'),

(4, 1, 2, 'nice', 'active', NULL, NULL, '2020-08-15 07:31:19', '2020-08-15 07:31:19'),

(5, 3, 5, 'nice blog', 'active', NULL, NULL, '2020-08-15 07:51:01', '2020-08-15 07:51:01'),

(6, 2, 3, 'nice', 'active', NULL, NULL, '2020-08-17 21:13:29', '2020-08-17 21:13:29'),

(7, 2, 3, 'really', 'active', NULL, 6, '2020-08-17 21:13:51', '2020-08-17 21:13:51');

-- --------------------------------------------------------

--

-- Table structure for table `post\_tags`

--

CREATE TABLE `post\_tags` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `title` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `slug` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `status` enum('active','inactive') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'active',

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `post\_tags`

--

INSERT INTO `post\_tags` (`id`, `title`, `slug`, `status`, `created\_at`, `updated\_at`) VALUES

(1, 'Enjoy', 'enjoy', 'active', '2020-08-14 01:53:52', '2020-08-14 01:53:52'),

(2, '2020', '2020', 'active', '2020-08-14 01:54:09', '2020-08-14 01:54:09'),

(3, 'Visit nepal 2020', 'visit-nepal-2020', 'active', '2020-08-14 01:54:33', '2020-08-14 01:54:33'),

(4, 'Tag', 'tag', 'active', '2020-08-15 06:59:31', '2020-08-15 06:59:31');

-- --------------------------------------------------------

--

-- Table structure for table `products`

--

CREATE TABLE `products` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `title` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `slug` varchar(191) COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `summary` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `description` longtext COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

  `photo` text COLLATE utf8mb4\_unicode\_ci NOT NULL,

  `stock` int(11) NOT NULL DEFAULT 1,

  `size` varchar(191) COLLATE utf8mb4\_unicode\_ci DEFAULT 'M',

  `condition` enum('default','new','hot') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'default',

  `status` enum('active','inactive') COLLATE utf8mb4\_unicode\_ci NOT NULL DEFAULT 'inactive',

  `price` double(8,2) NOT NULL,

  `discount` double(8,2) NOT NULL,

  `is\_featured` tinyint(1) NOT NULL,

  `cat\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `child\_cat\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `brand\_id` bigint(20) UNSIGNED DEFAULT NULL,

  `created\_at` timestamp NULL DEFAULT NULL,

  `updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

**SQL Page**

<?php

/\*\*

 \* Laravel - A PHP Framework For Web Artisans

 \*

 \* @package  Laravel

 \* @author   Taylor Otwell <taylor@laravel.com>

 \*/

$uri = urldecode(

    parse\_url($\_SERVER['REQUEST\_URI'], PHP\_URL\_PATH)

);

// This file allows us to emulate Apache's "mod\_rewrite" functionality from the

// built-in PHP web server. This provides a convenient way to test a Laravel

// application without having installed a "real" web server software here.

if ($uri !== '/' && file\_exists(\_\_DIR\_\_.'/public'.$uri)) {

    return false;

}

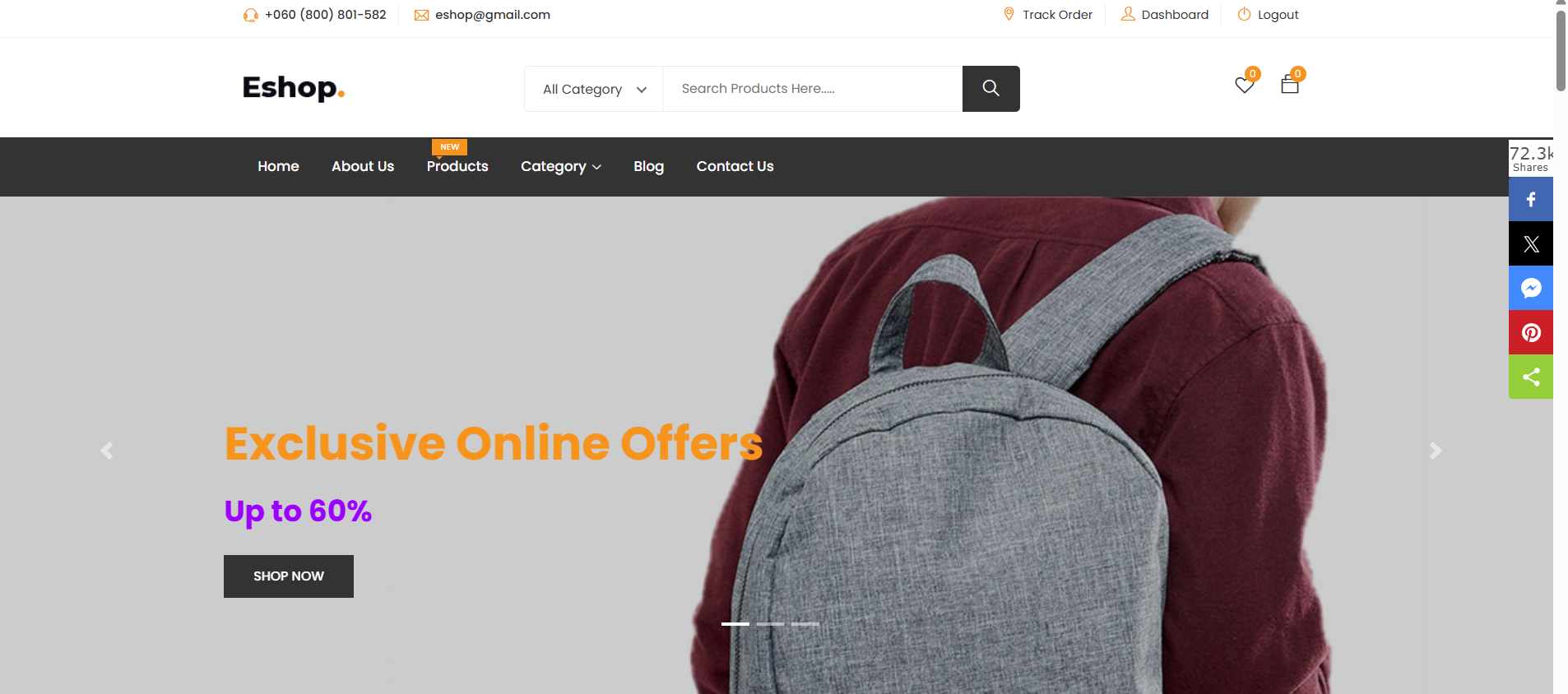
require\_once \_\_DIR\_\_.'/public/index.php';

**Server Page**

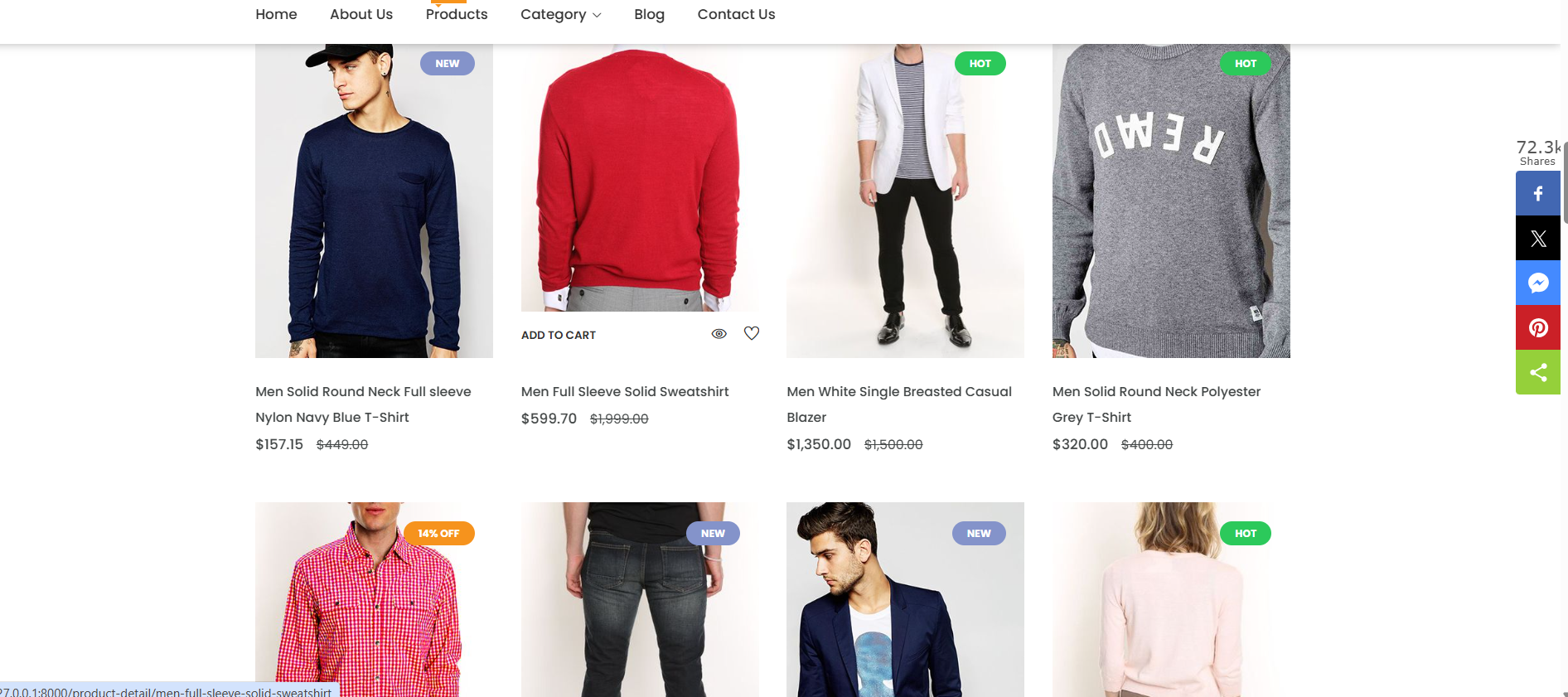
**CHAPTER 7   
  
  
RESULTS**

#### 7.1 Screens and Explanations

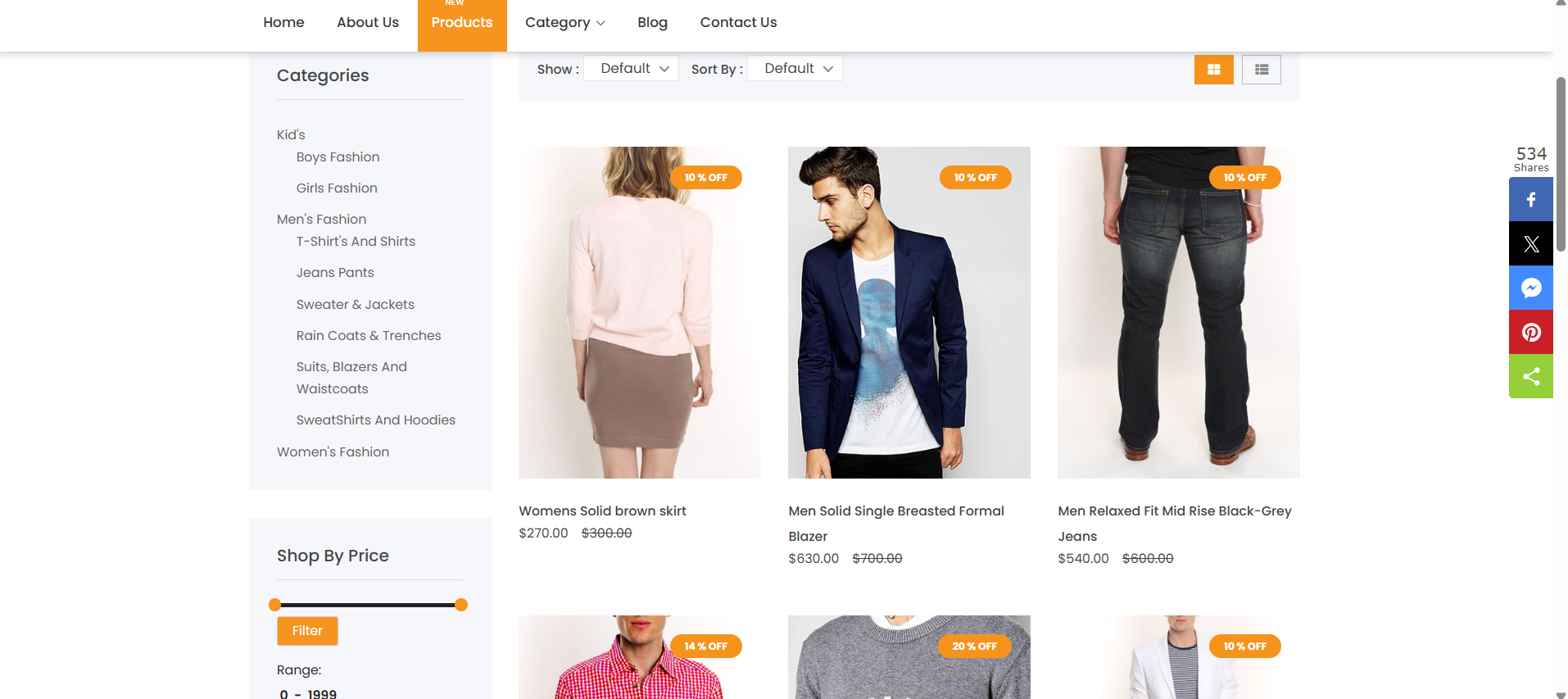
This chapter will include all the screens available in the project



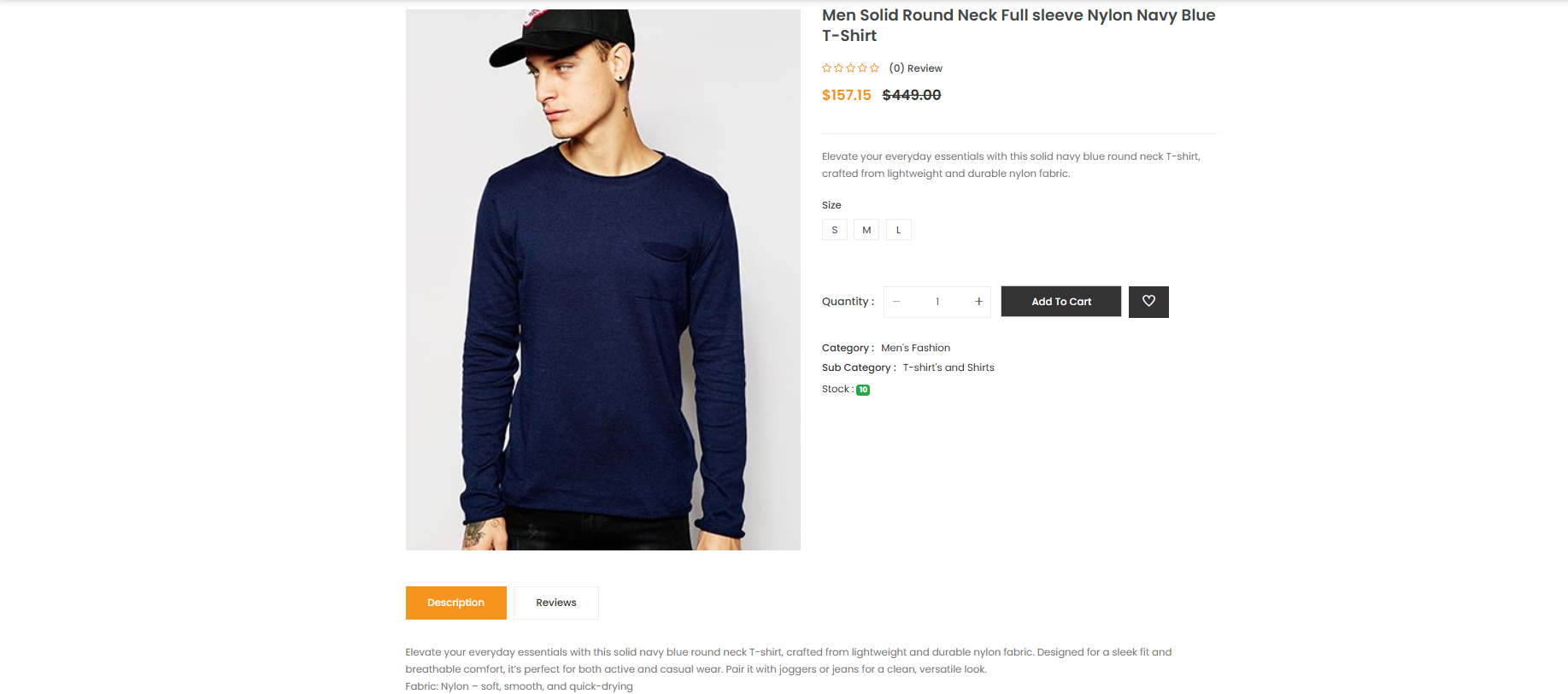
**Home page**

****

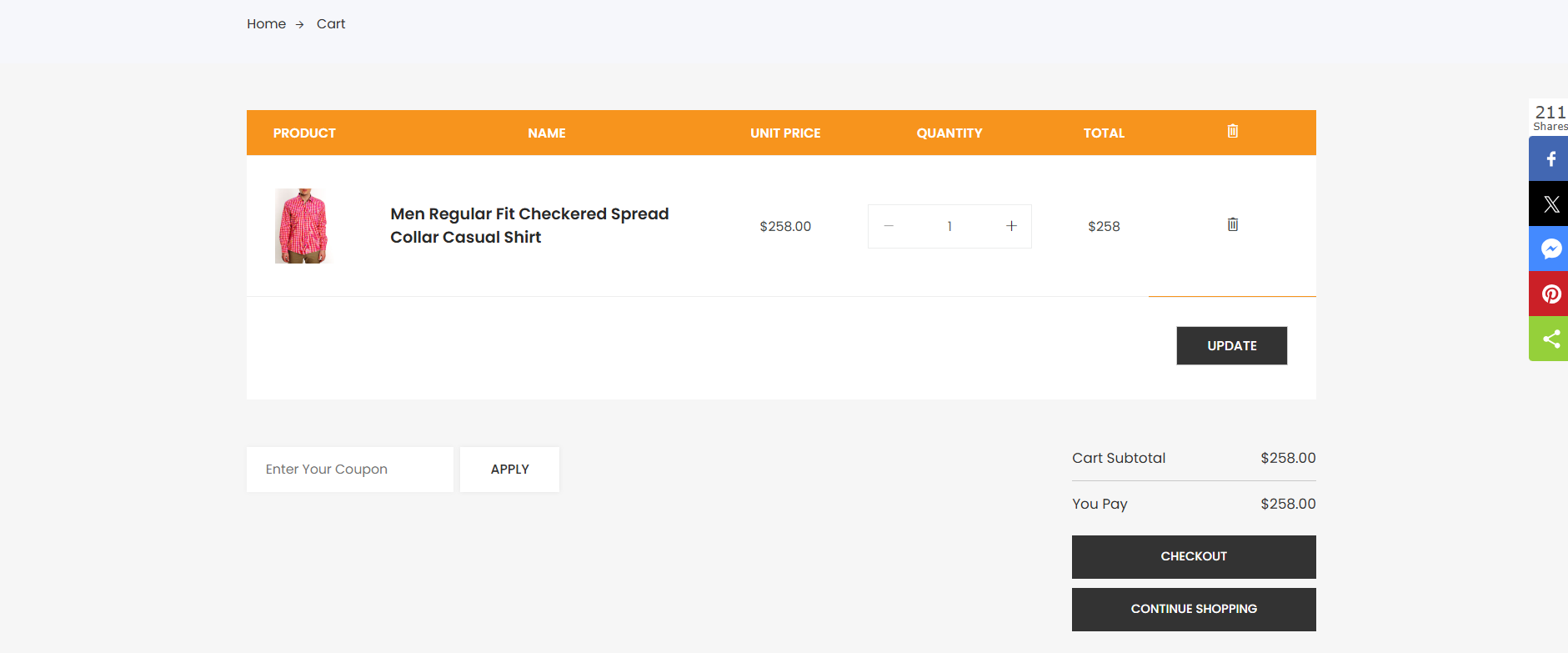
**Trending Item Page**



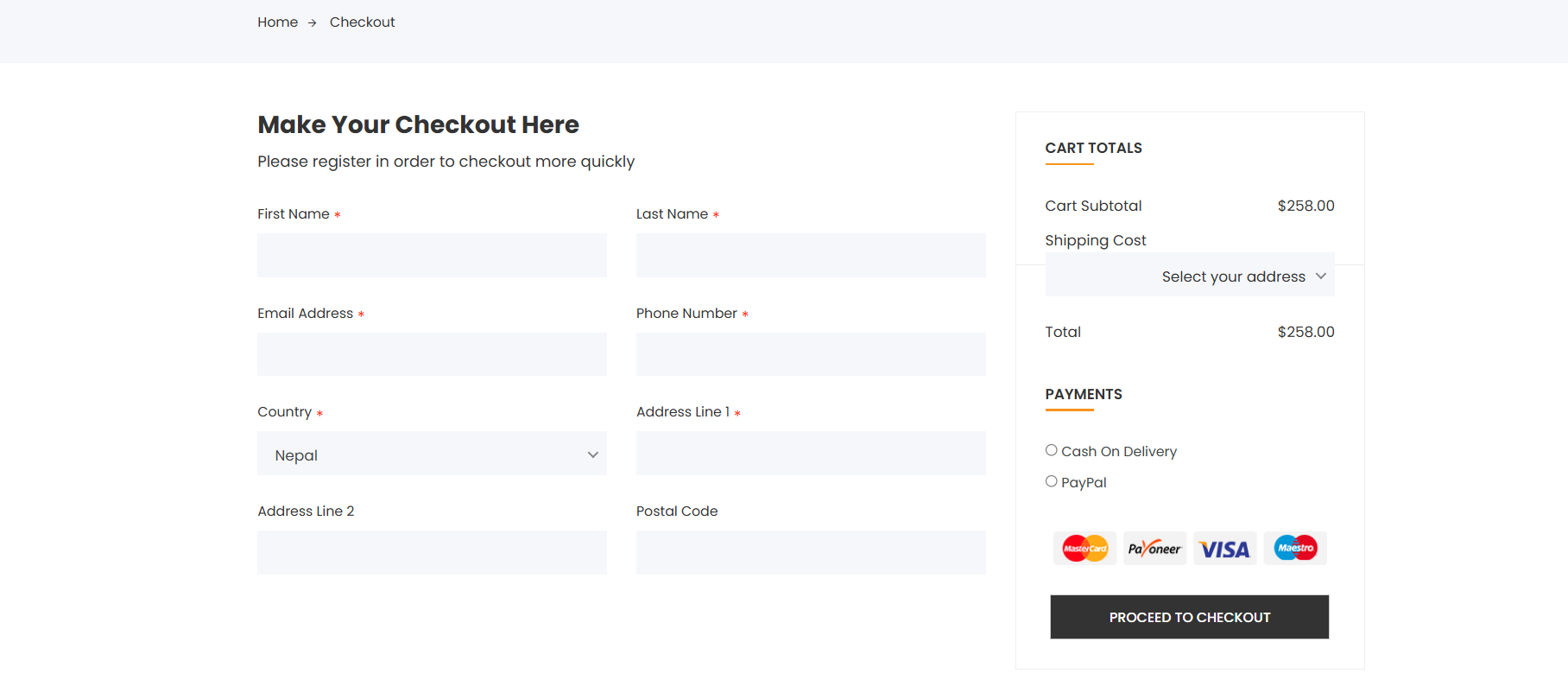
**Products Page**



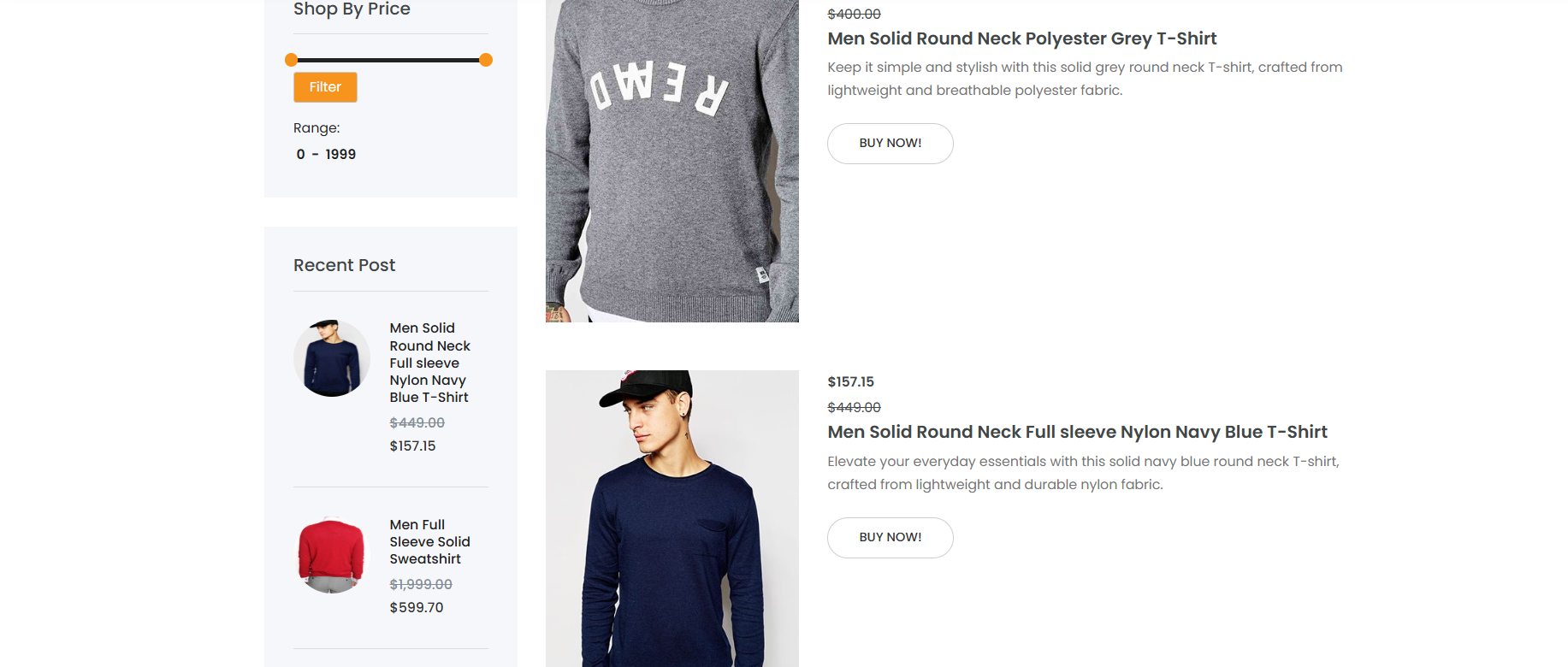
**Product Description Page**



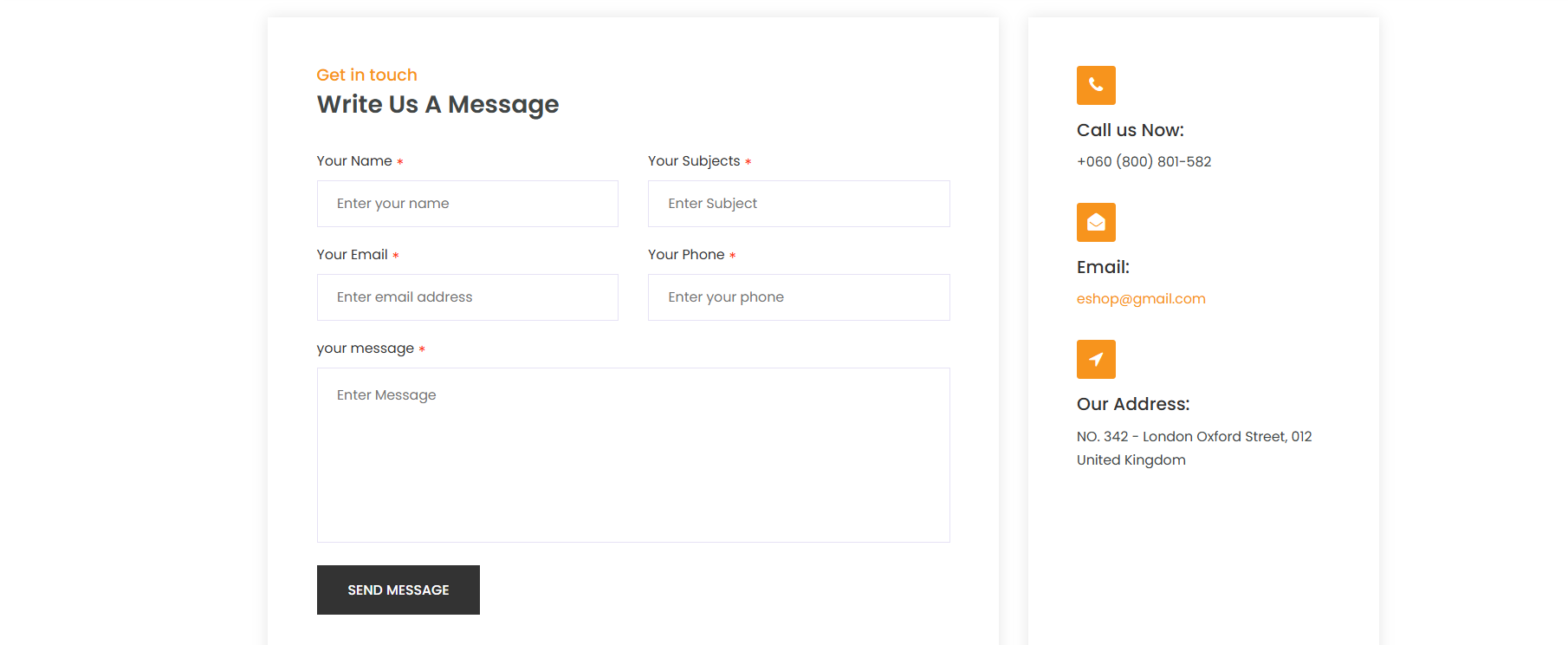
**Add to Cart Page**



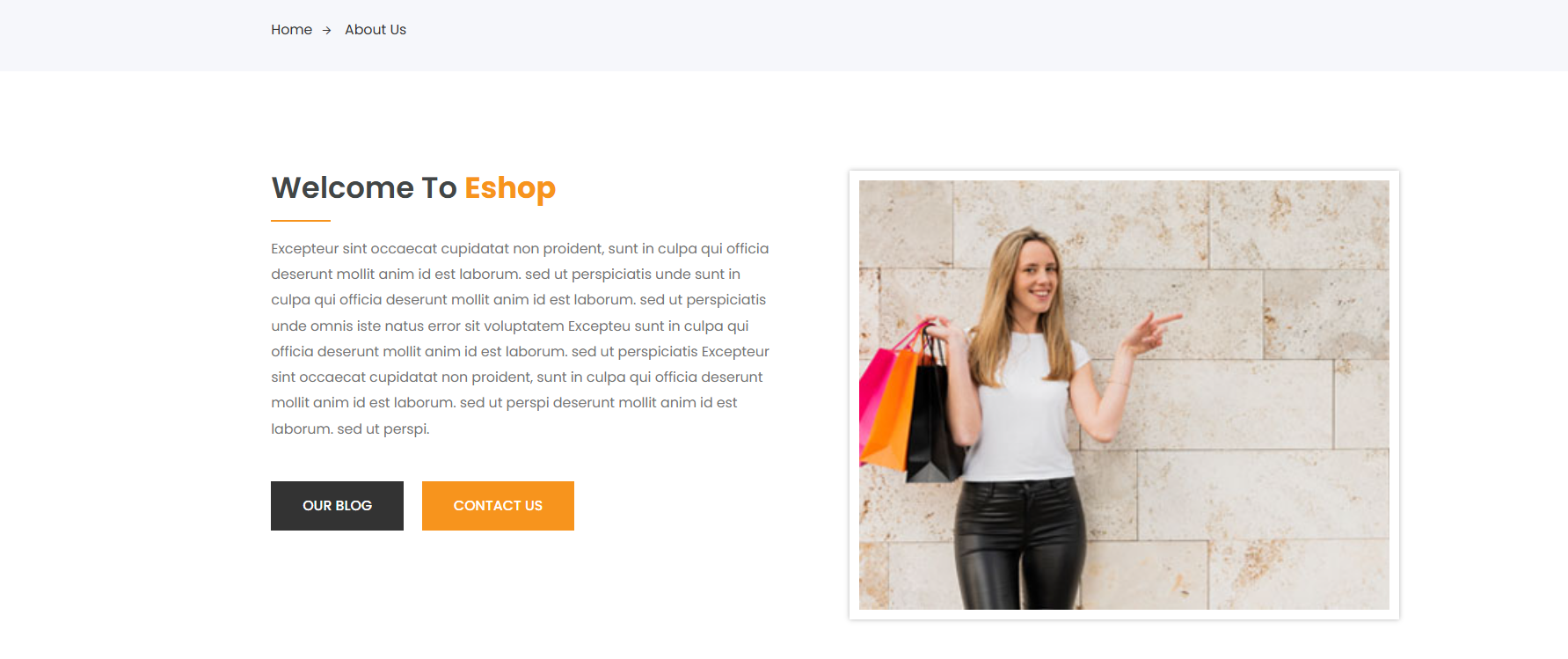
**Check Out Page**



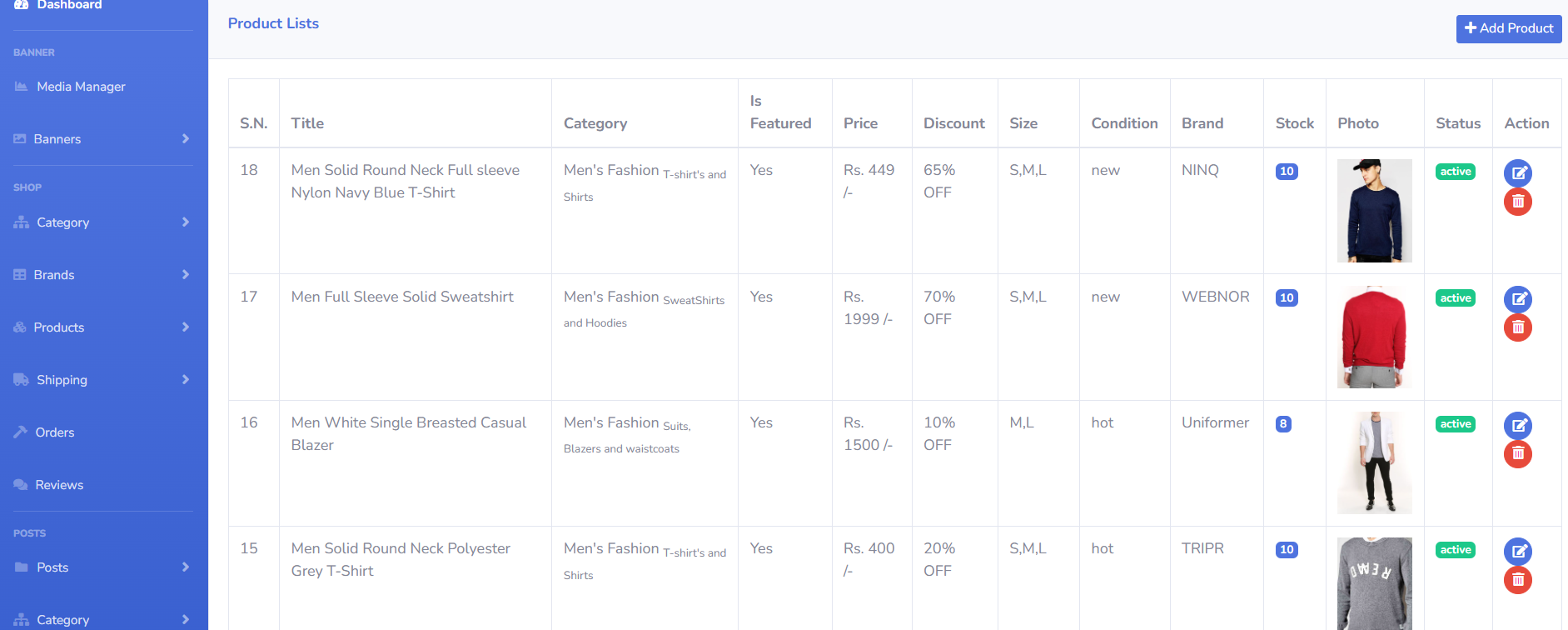
**Category Page**



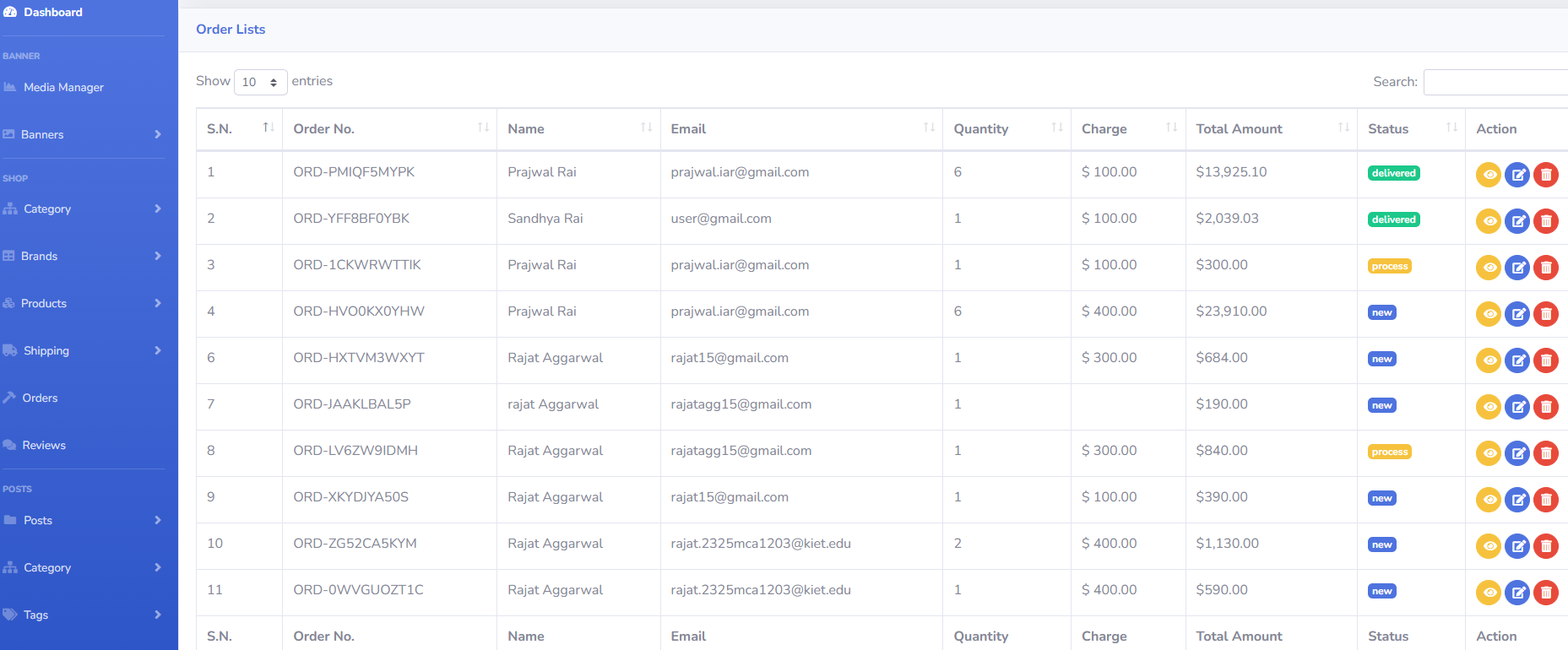
**Contact Page**



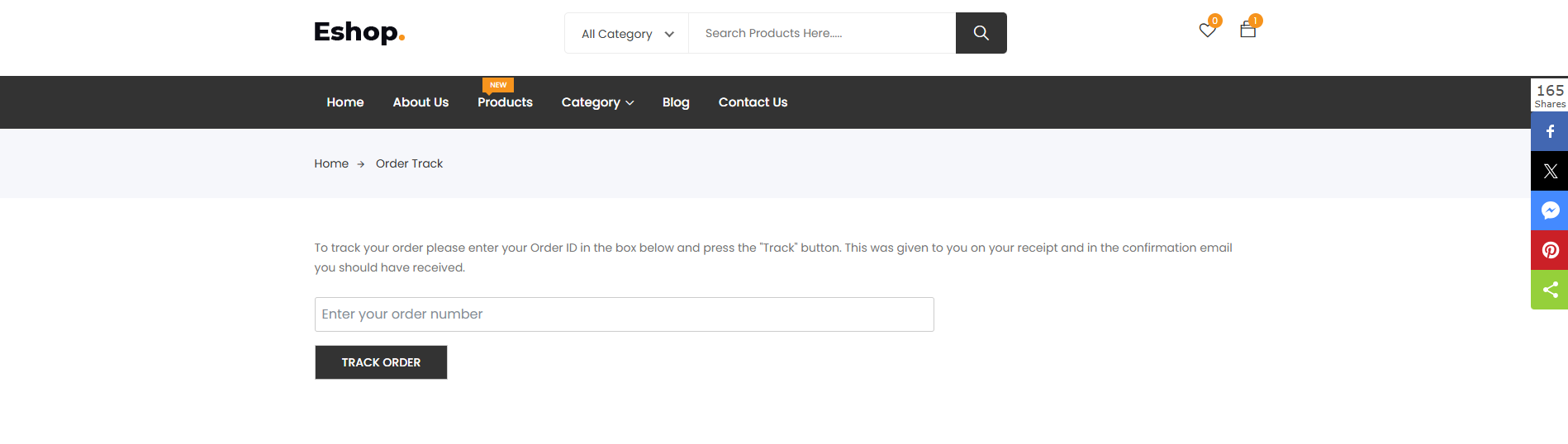
**About Us Page**



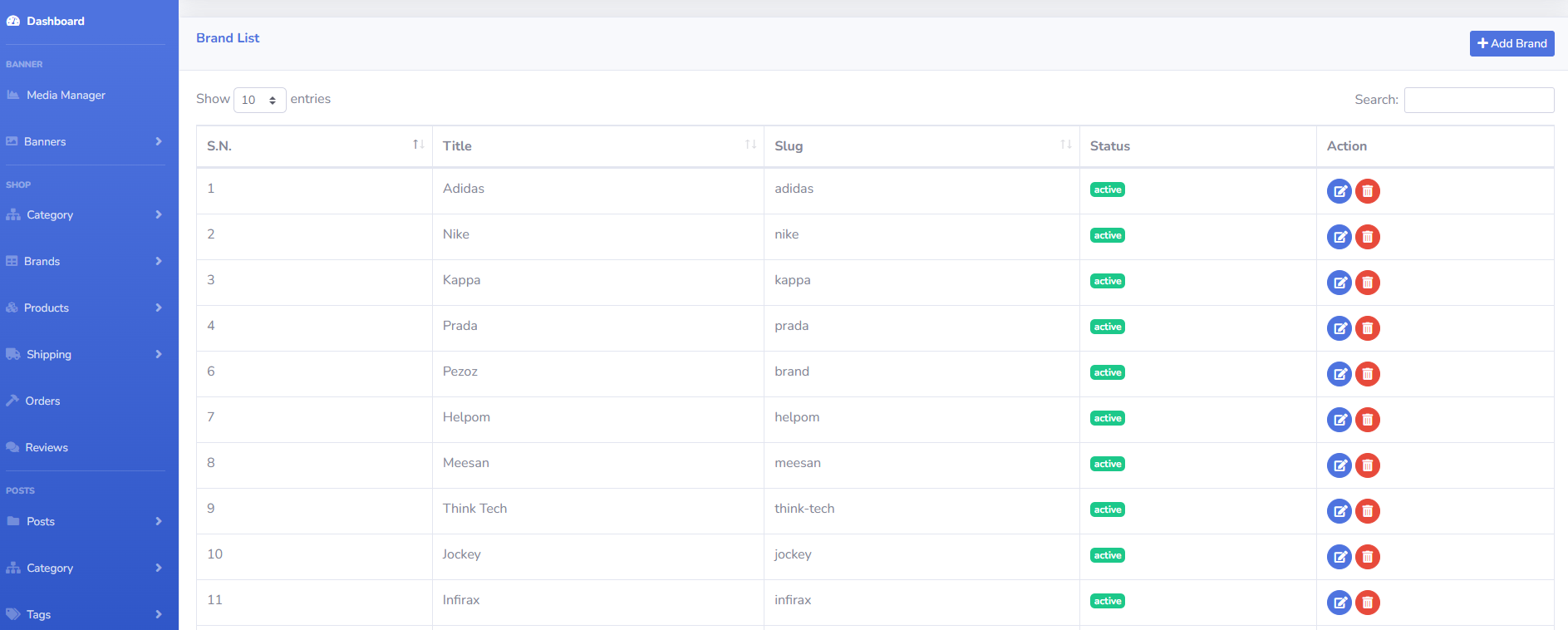
**Admin panel page**



**Admin Panel order Page**



**Tracking Order Page**



**Add Brands Admin Panel**

**CHAPTER 8   
  
DISCUSSION**

The eCommerce platform represents a significant advancement in the online retail sector by providing a streamlined and efficient shopping experience for customers, sellers, and administrators. Built using HTML, CSS, JavaScript, Laravel, and MySQL, the platform ensures scalability, security, and ease of use. This discussion covers the platform’s performance, its impact on online commerce, and potential areas for future enhancement.

**8.1 Performance**

The eCommerce platform has demonstrated reliability and efficiency through the following key factors:

* **Scalability**  
  The use of Laravel’s MVC framework combined with MySQL database enables the platform to scale effectively as the number of users, products, and transactions increase. Optimized database queries and modular code structure allow for smooth handling of growing traffic and data.
* **User-Friendly Interface**  
  The frontend built with HTML, CSS, and JavaScript provides a clean, responsive, and intuitive interface. Customers can effortlessly browse products, apply filters, manage their cart, and complete purchases across various devices and screen sizes.
* **Secure Authentication and Role Management**  
  Laravel’s built-in authentication system ensures secure login and registration for customers, sellers, and admins. Role-based access control restricts sensitive actions and data visibility, maintaining user privacy and data integrity.
* **Efficient Order and Inventory Management**  
  The backend efficiently manages product inventories and order processing, ensuring real-time updates to stock availability and timely order fulfillment, which enhances customer satisfaction.
* **Fast and Responsive Backend**  
  Laravel and MySQL together provide a robust backend capable of handling multiple concurrent requests with minimal latency, ensuring a smooth shopping experience without delays during product searches or checkout.
* **Data Integrity and Security**  
  Secure data handling is maintained through validation, sanitization, and use of middleware to prevent unauthorized access, ensuring customer and transaction data remains protected.

**8.2 Future Research Directions**

Although the platform is fully functional, several future enhancements can further improve its usability and features:

1. **Payment Gateway Integration**  
   Integrating popular payment gateways like Stripe, PayPal, or Razorpay will enable secure, seamless, and diverse payment options, enhancing customer trust and checkout convenience.
2. **User Reviews and Ratings**  
   Allowing customers to provide feedback and rate products will help improve product quality, assist other buyers, and foster transparency.
3. **Personalized Recommendations**  
   Implementing machine learning algorithms for personalized product suggestions based on browsing and purchase history will enhance user engagement and increase sales.
4. **Mobile Application Development**  
   Creating native mobile apps for iOS and Android will make the platform accessible on the go, improving customer convenience and expanding market reach.
5. **Multilingual and Multi-Currency Support**  
   Expanding the platform to support multiple languages and currencies will attract a global user base and facilitate international transactions.
6. **Advanced Analytics for Sellers and Admins**  
   Providing comprehensive dashboards with sales trends, customer insights, and inventory alerts will empower sellers and administrators to make data-driven decisions.

**CHAPTER 9**

### CONCLUSION

The eCommerce platform developed through this project successfully meets the essential requirements of a modern online shopping system by providing a secure, scalable, and user-friendly environment for customers, sellers, and administrators. The integration of frontend technologies like HTML, CSS, and JavaScript with the powerful backend framework Laravel and the relational database MySQL has resulted in a robust and efficient solution that handles critical eCommerce functionalities including product catalog management, user authentication, shopping cart operations, and order processing.

This platform effectively simplifies the online shopping experience by offering intuitive navigation, advanced search and filtering capabilities, and real-time updates on product availability. Customers can seamlessly browse and purchase products, while sellers have dedicated tools for managing inventory and tracking orders. The administrative interface empowers platform management by enabling user verification, product approval, and analytics monitoring, thereby maintaining the integrity and quality of the marketplace.

Security considerations were thoroughly incorporated by implementing role-based access control, input validation, and session management to safeguard sensitive user and transaction data. The modular MVC architecture not only enhances code maintainability and scalability but also facilitates future feature additions.

Despite the successful implementation, the platform has room for improvement. Future enhancements such as integrating secure payment gateways, enabling customer reviews and ratings, developing native mobile applications, and introducing multilingual and multi-currency support will significantly elevate the user experience and broaden the platform’s market reach. Additionally, leveraging machine learning for personalized product recommendations and advanced analytics will empower sellers and administrators with valuable insights to optimize their operations.

In summary, this eCommerce platform represents a comprehensive and adaptable digital solution that addresses current market needs and is well-positioned to evolve with emerging technologies and consumer trends. It lays a strong foundation for building a competitive and scalable online retail ecosystem that can deliver convenience, security, and efficiency to all stakeholders involved.

**CHAPTER 10**

### REFERENCES

**Technical Documentation & Development Resources**

**Laravel Documentation**  
Laravel. (2024). *Laravel 10.x Documentation*. Retrieved from <https://laravel.com/docs>

**MySQL Official Documentation**  
Oracle Corporation. (2024). *MySQL 8.0 Reference Manual*. Retrieved from <https://dev.mysql.com/doc/>

**PHP Manual**  
The PHP Group. (2024). *PHP Documentation*. Retrieved from <https://www.php.net/manual/en/>

**MDN Web Docs – JavaScript**  
Mozilla Contributors. (2024). *JavaScript Guide*. Retrieved from <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide>

**MDN Web Docs – HTML**  
Mozilla Contributors. (2024). *HTML Standard*. Retrieved from <https://developer.mozilla.org/en-US/docs/Web/HTML>

**MDN Web Docs – CSS**  
Mozilla Contributors. (2024). *CSS Reference*. Retrieved from <https://developer.mozilla.org/en-US/docs/Web/CSS>

**OWASP Foundation**  
OWASP. (2024). *OWASP Top Ten Security Risks*. Retrieved from https://owasp.org/www-project-top-ten/

**Stripe Payment Gateway API Documentation**  
Stripe Inc. (2024). *Stripe API Reference*. Retrieved from <https://stripe.com/docs/api>

**Bootstrap Documentation** (if you used Bootstrap)  
Bootstrap Team. (2024). *Bootstrap 5 Documentation*. Retrieved from https://getbootstrap.com/docs/5.0/getting-started/introduction/

**Google Developers – Web Performance Optimization**  
Google. (2024). *Web Performance Best Practices*. Retrieved from https://developers.google.com/web/fundamentals/performance

**Additional Research & Concepts**

1. Laudon, K. C., & Traver, C. G. (2020). *E-Commerce 2020: Business, Technology, Society*. Pearson.
2. Laudon, K. C., & Laudon, J. P. (2016). *Management Information Systems: Managing the Digital Firm*. Pearson.
3. Krug, S. (2014). *Don’t Make Me Think, Revisited: A Common Sense Approach to Web Usability*. New Riders.
4. Flanagan, D. (2020). *JavaScript: The Definitive Guide*. O’Reilly Media.
5. Ullman, L. (2019). *PHP for the Web: Visual QuickStart Guide*. Peachpit Press.
6. Welling, L., & Thomson, L. (2017). *PHP and MySQL Web Development*. Addison-Wesley Professional.
7. Richardson, L. (2007). *RESTful Web Services*. O’Reilly Media.
8. MacDonald, A. (2021). *Modern Web Development: Understanding Domains, Technologies, and User Experience*. Apress.
9. Stauffer, M. (2021). *Laravel: Up and Running: A Framework for Building Modern PHP Apps*. O’Reilly Media.
10. Nielsen, J. (2012). *Usability Engineering*. Morgan Kaufmann.

**CHAPTER 11**

### BIBLIOGRAPHY

**1. Books**

Ullman, L. (2011). *PHP and MySQL for Dynamic Web Sites: Visual QuickPro Guide*. Peachpit Press.

Nixon, R. (2018). *Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5*. O'Reilly Media.

Deitel, P. J., & Deitel, H. M. (2012). *Internet & World Wide Web: How to Program*. Pearson Education.

**2. Websites**

Laravel Documentation: <https://laravel.com/docs>

PHP Manual: <https://www.php.net/manual/en/>

MySQL Documentation: <https://dev.mysql.com/doc/>

W3Schools (HTML/CSS/JS Reference): <https://www.w3schools.com>

Mozilla Developer Network (HTML/CSS/JS): <https://developer.mozilla.org/en-US/>

**3. Reports and Industry Insight**

Statista. (2023). *Global E-commerce Market Size 2020–2027*. Retrieved from <https://www.statista.com>

Deloitte Insights. (2022). *Digital Commerce Trends: How Online Retail is Evolving*.

McKinsey & Company. (2023). *The Future of E-commerce: Omnichannel Strategies and Growth*.