**ONLINE BOTIQUE STORE MANAGEMENT**

**ABSTRACT**

In the era of digital transformation, the retail landscape has seen a significant shift towards online platforms. Online boutique stores have emerged as a popular choice for consumers seeking unique and personalized shopping experiences. This project aims to develop an online boutique store using a combination of HTML, CSS, JavaScript, Bootstrap, PHP, and MySQL technologies.

The proposed online boutique store will offer a seamless and user-friendly interface for browsing and purchasing a variety of fashion products, including clothing, accessories, and footwear. Leveraging HTML and CSS, the design will focus on aesthetics, ensuring visually appealing layouts and responsive design to enhance the user experience across different devices.

JavaScript will be utilized to implement dynamic features such as interactive product galleries, real-time search functionality, and dynamic content updates without page refresh. Bootstrap will be integrated to streamline the development process and ensure consistency in design elements and responsiveness.

PHP will serve as the server-side scripting language, facilitating the handling of user authentication, product management, and order processing. MySQL will be employed as the database management system to store product information, user data, and order details securely.

Key features of the online boutique store will include:

1. User registration and authentication.
2. Browse and search functionality for products based on categories, sizes, colors, and prices.
3. Product details pages with images, descriptions, pricing, and reviews.
4. Shopping cart functionality for adding, removing, and updating selected items.
5. Secure checkout process.
6. Order tracking and management for both customers and administrators.
7. Admin panel for managing product inventory, user accounts, orders, and site content.

By integrating these technologies, the proposed online boutique store will provide a comprehensive solution for entrepreneurs looking to establish or expand their presence in the e-commerce market. The project will demonstrate the effective utilization of modern web development tools and techniques to create a robust and scalable platform for online retail operations.

**INTRODUCTION**

\*\*Introduction:\*\*

In recent years, the landscape of retail has undergone a significant transformation with the advent of online shopping platforms. Among these, online boutique stores have emerged as a prominent choice for consumers seeking personalized and unique shopping experiences in the fashion industry. With the convenience of browsing and purchasing from the comfort of one's own home, coupled with an extensive array of curated products, these online boutiques cater to the evolving demands of modern shoppers.

This project endeavors to explore and develop an online boutique store utilizing a blend of cutting-edge technologies including HTML, CSS, JavaScript, Bootstrap, PHP, and MySQL. Through the integration of these tools, we aim to create a sophisticated and user-friendly platform that not only showcases a diverse range of fashion products but also provides seamless navigation and functionality to enhance the overall shopping experience.

By leveraging HTML and CSS, we will focus on crafting visually captivating and responsive layouts, ensuring optimal viewing across various devices and screen sizes. JavaScript will play a pivotal role in adding dynamic elements to the website, such as interactive product galleries, real-time search functionality, and dynamic content updates, thereby enriching user engagement and satisfaction.

Furthermore, the utilization of Bootstrap will streamline the development process, facilitating the creation of consistent design elements and responsive components. PHP will serve as the backbone of the server-side scripting, enabling essential functionalities such as user authentication, product management, and order processing. Meanwhile, MySQL will be employed as the database management system, ensuring secure storage and retrieval of product information, user data, and transaction details.

Through the seamless integration of these technologies, our online boutique store will offer an immersive and feature-rich shopping environment, empowering both customers and administrators alike. From user registration and authentication to browsing, purchasing, and order management, our platform will cater to the diverse needs of online shoppers while providing robust tools for efficient site administration.

This project not only aims to showcase the technical prowess in web development but also underscores our commitment to delivering innovative solutions in the ever-evolving e-commerce landscape. By harnessing the power of modern technologies, we endeavor to create a compelling and scalable platform that sets new standards for online boutique shopping, providing a seamless blend of style, convenience, and functionality for fashion enthusiasts worldwide.

**OBJECTIVES**

1. \*\*Create an Engaging User Experience:\*\* Design and develop a visually appealing and intuitive user interface for the online boutique store, ensuring ease of navigation and seamless browsing of fashion products.

2. \*\*Implement Dynamic Features:\*\* Utilize JavaScript to incorporate dynamic elements such as interactive product galleries, real-time search functionality, and dynamic content updates to enhance user engagement and satisfaction.

3. \*\*Ensure Responsiveness:\*\* Employ responsive design techniques using HTML, CSS, and Bootstrap to ensure optimal viewing and functionality across various devices and screen sizes, including desktops, tablets, and smartphones.

4. \*\*Enable User Authentication and Authorization:\*\* Implement user registration, login, and authentication functionalities using PHP to ensure secure access to user accounts and personalized shopping experiences.

5. \*\*Facilitate Product Management:\*\* Develop administrative capabilities for managing product inventory, including adding new products, updating existing listings, and categorizing items for easy navigation.

6. \*\*Streamline Checkout Process:\*\* Create a secure and user-friendly checkout process with multiple payment options, enabling customers to complete transactions efficiently and securely.

7. \*\*Integrate Order Management System:\*\* Implement an order tracking and management system for both customers and administrators, allowing for easy monitoring of order status, processing, and fulfillment.

8. \*\*Ensure Data Security:\*\* Utilize MySQL as the database management system to securely store and manage sensitive information, including product details, user profiles, and transaction records, adhering to best practices for data protection.

9. \*\*Optimize Performance:\*\* Conduct thorough testing and optimization of the online boutique store to ensure fast loading times, smooth functionality, and overall performance efficiency.

10. \*\*Provide Documentation and Support:\*\* Create comprehensive documentation outlining the project's architecture, design, implementation details, and deployment instructions. Additionally, offer ongoing support and maintenance to address any issues or enhancements post-launch.

By achieving these objectives, the online boutique store will offer a compelling and seamless shopping experience for customers while providing robust tools for efficient management and administration, ultimately positioning it as a leading destination in the competitive e-commerce landscape.

**EXISTING SYSTEM**

The current landscape of online boutique stores presents several challenges and inefficiencies due to outdated methods and technologies. Highlighting these shortcomings is crucial to understanding the necessity and benefits of modernizing the system. Here are some key aspects of the existing system that illustrate outdated methods:

1. **Static Website Design:** Many online boutique stores still rely on static website designs built with basic HTML and CSS. While these websites may serve their purpose initially, they lack dynamic features and responsiveness, leading to a subpar user experience, especially on mobile devices.
2. **Limited Interactivity:** Traditional online boutiques often lack interactive features such as real-time search functionality, dynamic content updates, and personalized recommendations. This static approach fails to engage users effectively and may result in lower conversion rates and customer retention.
3. **Manual Inventory Management:** Inefficient manual methods for managing product inventory, including updating product listings, tracking stock levels, and categorizing items, can lead to errors, inconsistencies, and delays in updating the online store.
4. **Fragmented User Experience:** Without a centralized system for user authentication and authorization, customers may encounter fragmented experiences across different platforms and devices, leading to confusion and frustration during the shopping process.
5. **Complex Checkout Process:** Cumbersome and lengthy checkout processes, often lacking multiple payment options and security measures, can deter customers from completing their purchases, resulting in abandoned shopping carts and lost sales opportunities.
6. **Lack of Order Management Tools:** The absence of integrated order management systems makes it challenging for administrators to track, process, and fulfill orders efficiently. This manual approach can lead to delays, errors, and poor customer service.
7. **Data Security Risks:** Storing sensitive customer information using outdated or insecure methods poses significant data security risks, including unauthorized access, data breaches, and compliance violations, which can damage the reputation and credibility of the online boutique store.
8. **Performance Limitations:** Static websites with outdated technologies may suffer from performance limitations such as slow loading times, poor scalability, and unreliable uptime, impacting user satisfaction and search engine rankings.
9. **Limited Scalability:** Without scalable architectures and flexible technologies, online boutique stores may struggle to adapt to changing market demands, accommodate growth, and integrate new features and functionalities effectively.
10. **Lack of Documentation and Support:** Inadequate documentation and support resources further exacerbate the challenges of maintaining and updating the existing system, hindering continuous improvement and innovation.

Addressing these outdated methods and challenges through the adoption of modern technologies and best practices is essential to revitalizing the online boutique store, enhancing its competitiveness, and delivering superior experiences for customers and administrators alike.

**PROPOSED SYSTEM**

The proposed system for the online boutique store aims to address the limitations of the existing system while leveraging modern technologies to deliver a superior shopping experience. Here are the key advantages of the proposed system:

1. **Dynamic and Responsive Design:** The proposed system will feature a dynamic and responsive website design built with HTML, CSS, and Bootstrap. This approach ensures optimal viewing and functionality across various devices, enhancing user engagement and satisfaction.
2. **Enhanced Interactivity:** Leveraging JavaScript, the proposed system will incorporate interactive features such as real-time search functionality, dynamic product galleries, and personalized recommendations. These interactive elements will elevate the user experience and drive higher levels of engagement and conversion.
3. **Automated Inventory Management:** The proposed system will implement automated inventory management tools powered by PHP and MySQL. This includes real-time updates of product listings, automatic stock level tracking, and seamless categorization of items, reducing errors and streamlining the management process.
4. **Centralized User Authentication:** A centralized user authentication and authorization system will be implemented, ensuring a seamless and secure login experience across all platforms and devices. This centralized approach enhances security, simplifies user management, and provides a consistent user experience.
5. **Streamlined Checkout Process:** The proposed system will feature a streamlined and user-friendly checkout process with multiple payment options, secure transaction handling, and simplified order confirmation. This reduces cart abandonment rates and enhances customer satisfaction and retention.
6. **Integrated Order Management:** An integrated order management system will be implemented, allowing administrators to track, process, and fulfill orders efficiently. This includes order tracking, automated notifications, and centralized order processing tools, improving operational efficiency and customer service.
7. **Robust Data Security Measures:** The proposed system will prioritize data security with robust encryption techniques, secure server configurations, and regular security audits. This ensures the protection of sensitive customer information, minimizes the risk of data breaches, and maintains compliance with regulatory requirements.
8. **Optimized Performance:** Through optimized code, caching mechanisms, and scalable architectures, the proposed system will deliver superior performance with fast loading times, high reliability, and efficient resource utilization. This enhances user satisfaction, improves search engine rankings, and supports scalability and growth.
9. **Scalable and Flexible Architecture:** The proposed system will be built on a scalable and flexible architecture, allowing for easy integration of new features and functionalities, accommodation of increased traffic and transactions, and adaptation to evolving market demands. This ensures long-term viability and competitiveness in the ever-changing e-commerce landscape.
10. **Comprehensive Documentation and Support:** Extensive documentation and support resources will be provided to facilitate system maintenance, updates, and troubleshooting. This includes user guides, developer documentation, and ongoing support channels, empowering administrators to maximize the system's potential and drive continuous improvement.

By leveraging these advantages, the proposed system for the online boutique store will revolutionize the shopping experience, drive business growth, and establish a competitive edge in the dynamic e-commerce market.

Top of Form

**MODULE DESCRIPTION**

1. **User Authentication Module:**
   * Description: This module facilitates user registration, login, and authentication processes.
   * Features:
     + User registration form with validation.
     + Secure login functionality with authentication.
     + Password reset option for users.
2. **Product Management Module:**
   * Description: This module enables administrators to manage product inventory and listings.
   * Features:
     + Add, edit, and delete product listings.
     + Categorize products into different categories and subcategories.
     + Upload product images and descriptions.
3. **Search and Browse Module:**
   * Description: This module provides users with tools to search and browse products efficiently.
   * Features:
     + Real-time search functionality with autocomplete suggestions.
     + Filter products by category, size, color, price range, etc.
     + Sort products by relevance, price, popularity, etc.
4. **Product Details Module:**
   * Description: This module displays detailed information about individual products.
   * Features:
     + Product images gallery with zoom functionality.
     + Product description, specifications, and reviews.
     + Add to cart button for easy purchasing.
5. **Shopping Cart Module:**
   * Description: This module allows users to add, remove, and manage items in their shopping cart.
   * Features:
     + View and edit items in the shopping cart.
     + Update quantity, remove items, or proceed to checkout.
     + Calculate total price and display subtotal.
6. **Checkout Module:**
   * Description: This module facilitates the secure and seamless checkout process.
   * Features:
     + Secure checkout page with SSL encryption.
     + Multiple payment options (credit card, PayPal, etc.).
     + Shipping address and payment information forms.
7. **Order Management Module:**
   * Description: This module enables administrators to manage customer orders efficiently.
   * Features:
     + View list of orders with details (order ID, date, status, etc.).
     + Update order status (processing, shipped, delivered, etc.).
     + Generate invoices and packing slips.
8. **User Profile Module:**
   * Description: This module allows users to manage their personal information and preferences.
   * Features:
     + View and edit user profile details (name, email, address, etc.).
     + Change password and update account settings.
     + View order history and track order status.
9. **Admin Dashboard Module:**
   * Description: This module provides administrators with an overview of site activity and performance.
   * Features:
     + Dashboard with key metrics (total sales, number of orders, etc.).
     + Charts and graphs for visualizing data.
     + Quick access to important sections (product management, order management, etc.).
10. **Reporting Module:**
    * Description: This module generates reports and analytics to track business performance.
    * Features:
      + Generate sales reports, inventory reports, and customer analytics.
      + Filter and export data for further analysis.
      + Identify trends and insights to inform decision-making.

Each module plays a crucial role in the overall functionality and success of the online boutique store, providing users with a seamless shopping experience while empowering administrators to efficiently manage operations and drive business growth.

**SYSTEM SPECIFICATION**

**HARDWARE SPECIFICATION**

|  |  |
| --- | --- |
| System | HP 15s |
| Processor | Ryzen 5 2.1 GHz |
| Storage | 512 GB SSD |
| RAM | 16 GB |
| Monitor | Integrated Monitor |
| Mouse | Integrated Trackpad |
| Keyboard | Integrated Keyboard |

**OPERATING SYSTEM**

|  |  |
| --- | --- |
| Operating System | Windows 11 |
| Front End | HTML, Bootstrap |
| Back End | PHP Version 8, MySQL Version 8 |
| Server | XAMPP |

**SOFTWARE SPECIFICATION**

**XAMPP:**

XAMPP is an [open-source](https://en.wikipedia.org/wiki/Free_software) [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [web server](https://en.wikipedia.org/wiki/Web_server) [solution stack](https://en.wikipedia.org/wiki/Solution_stack) package developed by Apache Friends, consisting mainly of the [Apache HTTP Server](https://en.wikipedia.org/wiki/Apache_HTTP_Server), [Maria DB](https://en.wikipedia.org/wiki/MariaDB) [database](https://en.wikipedia.org/wiki/Database), and [interpreters](https://en.wikipedia.org/wiki/Interpreter_%28computing%29) for scripts written in the [PHP](https://en.wikipedia.org/wiki/PHP) and [Perl](https://en.wikipedia.org/wiki/Perl) [programming languages](https://en.wikipedia.org/wiki/Programming_language). XAMPP stands for Cross-Platform (X), Apache (A), Maria DB (M), PHP (P), and Perl (P). It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing and deployment purposes.

Everything needed to set up a web server – server application (Apache), database (Maria DB), and scripting language (PHP) – is included in an extractable file. XAMPP is also cross-platform, which means it works equally well on Linux, Mac, and Windows.

XAMPP's designers intended it for use only as a development tool, to allow website designers and programmers to test their work on their computers without any access to the Internet.

**CROSS-PLATFORM**

Cross-platform software is a type of software application that works on multiple operating systems or devices, which are often referred to as platforms. A platform means an operating system such as Windows, Mac OS, Android, or iOS. When a software application works on more than one platform, the user can utilize the software on a wider choice of devices and computers.

**BENEFITS OF CROSS-PLATFORM**

The benefit of a cross-platform software app or program is that you can use the same program whether you’re on a Windows PC or whether you’re logging in from your laptop or smartphone. The Microsoft Office suite of applications, which includes Word, Excel, and PowerPoint, is available on Windows, Mac OS, iOS (iPhone/iPad), and Android. While there are differences based on how the platforms work, you’ll have a similar experience within the application between all of your devices.

Having a similar experience across any platform means there’s a much smaller learning curve if one even exists at all, so you’ll be more productive and be able to use a software product you’re familiar with regardless of the operating system or device you choose. In addition, your files can be moved much more easily between your devices so you can use the software with whatever device you have with you at the time. And there’s a way to keep all of your work in sync across all of your devices, by using the cloud.

**EXAMPLES OF CROSS-PLATFORM**

**Unity 3D**

First, let’s talk about Unity3D. I think the game engine should be preferred by people who want to write mobile games.  
You can develop games on 17 platforms using multiple languages, including Linux. Of course, iOS, Android, and Windows Phone is also the most ideal game engine to develop games.

You can develop your application using C #, JS, and C ++.

Link to: [https://unity3d.com](https://unity3d.com/)

# Xamarin

Xamarin Some time ago, it was purchased by Microsoft and is a perfect fit for developers using C #.

Because it is a C # language, it has a lot of documentation, and because of Microsoft support, Xamarin is the choice for C # developers.

In addition, you can do everything you can do in Objective-C, Swift, and Java with the Xamarin library.

Link to: [https://xamarin.com](https://xamarin.com/)

# React Native

React Native is an open-source JavaScript library developed by the new generation of React — Facebook, which was open to Github in 2013. Native application creation means writing applications only for a specific operating system. React Native helps developers reuse their code over the web and on mobile. Developers will not have to create the same app from scratch for iOS and Android. They will be able to reuse the code in each operating system. The great thing about React Native is that there is little difference between a finished application in Objective-C or Java and an application built using React Native. Android and iOS code development environments are very different. So it takes time to remove the application to two different platforms. However, with React Native, only one developer can write on different mobile operating systems.

**APACHE:**

The Apache HTTP Server, colloquially called Apache is a [free and open-source](https://en.wikipedia.org/wiki/Free_and_open-source) [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [web server](https://en.wikipedia.org/wiki/Web_server) software, released under the terms of [Apache License](https://en.wikipedia.org/wiki/Apache_License) 2.0. Apache is developed and maintained by an open community of developers under the auspices of the [Apache Software Foundation](https://en.wikipedia.org/wiki/Apache_Software_Foundation).

The vast majority of Apache HTTP Server instances run on a [Linux distribution](https://en.wikipedia.org/wiki/Linux_distribution), but current versions also run on [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows), [OpenVMS](https://en.wikipedia.org/wiki/OpenVMS),  and a wide variety of [Unix-like](https://en.wikipedia.org/wiki/Unix-like) systems. Past versions also ran on [NetWare](https://en.wikipedia.org/wiki/NetWare), [OS/2](https://en.wikipedia.org/wiki/OS/2), and other operating systems,  including ports to mainframes.

Originally based on the HTTP server, the development of Apache began in early 1995 after work on the NCSA code stalled. Apache played a key role in the initial growth of the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web), quickly overtaking NCSA HTTP as the dominant [HTTP](https://en.wikipedia.org/wiki/HTTP) server. In 2009, it became the first web server software to serve more than 100 million [websites](https://en.wikipedia.org/wiki/Website). As of January 2021, [Netcraft](https://en.wikipedia.org/wiki/Netcraft" \o "Netcraft) estimated that Apache served 24.63% of the million busiest websites, while [Nginx](https://en.wikipedia.org/wiki/Nginx) served 23.21% and Microsoft is in third place at 6.85% (for some of Netcraft's other stats Nginx is ahead of Apache), while according to W3Techs, Apache is ranked first at 35.0% and Nginx second at 33.0% and Cloudflare Server third at 17.3%.

**LANGUAGE SPECIFICATION**

**PHP**

**INTRODUCTION OF PHP**

PHP started as a small open-source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

* PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
* PHP is a server-side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, and even build entire e-commerce sites.
* It is integrated with several popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
* PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
* PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
* PHP is forgiving: PHP language tries to be as forgiving as possible.
* PHP Syntax is the same as C language.

**What is a PHP File?**

* PHP files can contain text, HTML, CSS, JavaScript, and PHP code.
* PHP code is executed on the server, and the result is returned to the browser as plain HTML.
* PHP files have the extension ".php".

**What Can PHP Do?**

* PHP can generate dynamic page content and it can create, open, read, write, delete, and close files on the server and it can collect form data.
* PHP can send and receive cookies it can add, delete, and modify data in your database and it can be used to control user-access and encrypt data.

**Why PHP?**

* PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.).
* PHP is compatible with almost all servers used today (Apache, IIS, etc.).
* PHP supports a wide range of databases.
* PHP is free.
* PHP is easy to learn and runs efficiently on the server side.

## **What is a Database?**

* A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching, and replicating the data it holds.
* Other kinds of data stores can be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those types of systems.
* Nowadays, we use relational database management systems (RDBMS) to store and manage huge volumes of data. This is called a relational database because all the data is stored in different tables and relations are established using primary keys or other keys known as foreign keys.

**MySQL Database**

* MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
* MySQL uses a standard form of the well-known SQL data language. MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc. MySQL works very quickly and works well even with large data sets.
* MySQL is very friendly to PHP, the most appreciated language for web development. MySQL supports large databases, up to 50 million rows or more in a table.
* The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). MySQL is customizable.
* The open-source GPL license allows programmers to modify the MySQL software to fit their specific environments.

**TABLE CREATION**

* Name of the table
* Names of fields
* Definitions for each field
* Field Attribute **NOT NULL** is being used because we do not want this field to be NULL. So if the user tries to create a record with a NULL value, then MySQL will raise an error.
* Field Attribute **AUTO\_INCREMENT** tells MySQL to go ahead and add the next available number to the id field.
* Keyword **PRIMARY KEY** is used to define a column as the primary key. You can use multiple columns separated by a comma to define a primary key.

## **ADMINISTRATIVE MYSQL COMMAND**

* **USE DATABASE NAME**: This will be used to select a particular database in the MySQL work area.
* **SHOW DATABASES:** Lists the databases that are accessible by the MySQL DBMS.
* **SHOW TABLES:** Shows the tables in the database once a database has been selected with the use command.
* **SHOW COLUMNS FROM Table name:** Shows the attributes, types of attributes, key information, whether NULL is permitted, defaults, and other information for a table.
* **SHOW INDEX FROM Table name:** Presents the details of all indexes on the table, including the PRIMARY KEY

## **CREATING TABLES USING PHP SCRIPT:**

To create a new table in any existing database you would need to use PHP function **mysqli\_query()**.

## **Dropping Tables Using PHP Script:**

Drop an existing table in any database, you would need to use the PHP function **mysqli\_query()**.

## **INSERTING DATA USING PHP SCRIPT:**

**CREATE**

Create table statement is used to create a table in MySQL.

**SELECT**

The SELECT statement is used to select data from one or more tables.

**UPDATE**

The UPDATE statement is used to update existing records in a table:

## **DELETE**

The DELETE statement is used to delete records from a table:

**DATABASE DESIGN:**

The data in the system has to be stored and retrieved from the database. Designing the database is part of system design. Data elements and data structures to be stored have been identified at the analysis stage.

They are structured and put together to design the data storage and retrieval system. A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently.

The general objective is to make database access easy, quick, inexpensive, and flexible for the user. Relationships are established between the data items and unnecessary data items are removed.

Normalization is done to get an internal consistency of data and to have minimum redundancy and maximum stability. This ensures minimizing data storage required, minimizing chances of data inconsistencies, and optimizing for updates.

**INPUT DESIGN**

The Input design is the main feature of the system. Input design determines the format and validation criteria for data entering the system. Inputs originate with end-users; human factors play a significant role in input design. The input design is designed to control the input, avoid delay, and errors in data, avoid extra steps, to keep the process simple. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps, and keeping the process simple. The input is designed in such a way that it provides security and ease of use while retaining privacy.

The following are the general principles, that are considered in designing inputs,

* + - Enter only variable data
    - Do not input data that can be calculated
    - List of values
    - Sequence entry

**OUTPUT DESIGN**

Designing the output is more important than working up with a few layout charts and reports. The outputs are designed based on the issue encountered. It will also take care of who will receive the output, what for it is produced how many details are needed, when it is needed, and by what method.

The outputs designed in this system are easy to use and useful for their jobs. The outputs are simple to read and interpret. The outputs obtained from this system are designed by using a few guidelines, which are given below. The information should be clear and accurate, yet concise and restricted to relevant data. Reports should have titles, data, and descriptive headings for columns of data, numbered pages, and so on.

**SYSTEM TESTING**

System testing is the process of exercising software with the intent of finding and ultimately correcting errors. This fundamental philosophy does not change for web applications, because Web-based systems and applications reside on a network and interoperate with many different operating systems, browsers, hardware platforms, and communication protocols; the search for errors represents a significant challenge for web applications.

The distributed nature of client/server environments, the performance issues associated with transaction processing, the potential presence of several different hardware platforms, the complexities of network communication, the need to serve multiple clients from a centralized database, and the requirements imposed on the server all combine to make testing of client\server architectures.

Testing issues

* Client GUI considerations
* Target environment and platform diversity considerations
* Distributed database considerations
* Distributed processing considerations

**TYPES OF TESTING**

1. Unit Testing

2. Integration Testing

3. Validation Testing

4. User Acceptance Testing

5. System Testing

**Unit Testing**

All modules were tested and individually as soon as they were completed were checked for their correct functionality. Unit testing is carried out by verifying and recovering errors within the boundary of the smallest unit or a module. In this testing step, each module was found to be working satisfactorily per the expected output of the module. In the package development, each module is tested separately after it has been completed and checked with valid data.

**Integration Testing**

The entire project was split into small programs; each of these single programs gives a frame as an output. These programs were tested individually; at last, all these programs were combined by creating another program where all these constructions were used. It causes a lot of problems by not functioning in an integrated manner.

The user interface testing is important since the user has to declare that the arrangements made in the frames are convenient and it is satisfied. When the frames are tested, the end user gives suggestions. Since they were much exposed to do the work manually.

**Validation Testing**

At the culmination of the black box testing software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of tests i.e., validation succeeds when the software functions in a manner that can be reasonably accepted by the customer.

**User Acceptance Testing**

User acceptance testing of the system is the key factor in the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective systems at the time of development and making changes whenever required. This is done concerning the input screen design and output screen design.

**System Testing**

This is to verify that all the system elements have been properly integrated and perform allocated functions. Testing is executing a program to test the logic changes made in it to find errors. Tests are also conducted to find discrepancies between the system and its original objective, current specifications, and documents.

**SYSTEM IMPLEMENTATION**

Implementation is the stage in the project where the theoretical design is turned into a working system. The most crucial stage is achieving a successful new system & and giving the user confidence that the new system will work efficiently & and effectively in the implementation stage.

The stage consists of

* + - Testing the developed program with simple data.
    - Detections and correction of errors.
    - Creating whether the system meets user requirements.
    - Testing whether the system.
    - Making necessary changes as desired by the user.
    - Training user personnel.

**Implementation Procedures**

The implementation phase is less creative than the system design. A system project may be dropped at any time before implementation, although it becomes more difficult when it goes to the design phase.

The final report to the implementation phase includes procedural flowcharts, record layouts, report layouts, and a workable plan for implementing the candidate system design into an operational one. Conversion is one aspect of implementation.

**System Maintenance**

Maintenance is the implementation of the review plan. As important as it is, many programmers and analysts are to perform or identify themselves with the maintenance effort. There are psychological, personality, and professional reasons for this. Analysts and programmers spend far more time maintaining programs than they do writing them. Maintenance accounts for 50-80 percent of total system development.

Maintenance is expensive. One way to reduce maintenance costs is through maintenance management and software modification audits.

* Maintenance is not as rewarding or exciting as developing systems. It is perceived as requiring neither skill nor experience.
* Users are not fully cognizant of the maintenance problem or its high cost.
* Few tools and techniques are available for maintenance.
* A good test plan is lacking.
* Standards, procedures, and guidelines are poorly defined and enforced.
* Programs are often maintained without care for structure and documentation.
* There are minimal standards for maintenance.
* Programmers expect that they will not be in their current commitment by the time their programs go into the maintenance cycle.

**SYSTEM DESIGN**

System design is "the process of studying a procedure or business to identify its goals, purposes and create systems and procedures that will efficiently achieve them". Another view sees system analysis as a problem-solving technique that breaks down a system into its component pieces for the study of how well those parts work and interact to accomplish their purpose.

The field of system analysis relates closely to requirements analysis or operations research. It is also "an explicit formal inquiry carried out to help a decision maker identify a better course of action and make a better decision than they might otherwise have made."

* **DESIGN NOTATION**

Design notations are used when planning and should be able to communicate the purpose of a program without the need for formal code. Commonly used design notations are:

* DFD
* ERD
* **DFD (DATA FLOW DIAGRAM):**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an [information system](https://en.wikipedia.org/wiki/Information_system), modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the [visualization](https://en.wikipedia.org/wiki/Data_visualization) of [data processing](https://en.wikipedia.org/wiki/Data_processing) (structured design). A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about the timing of the process or information about whether processes will operate in sequence or parallel, unlike a [flowchart](https://en.wikipedia.org/wiki/Flowchart) which also shows this information.

Data flow diagrams were popularized in the late 1970s, arising from the book Structured Design, by computing pioneers Ed Yourdon and Larry Constantine. They based it on the “data flow graph” computation models by David Martin and Gerald Estrin. The structured design concept took off in the software engineering field, and the DFD method took off with it. It became more popular in business circles, as it was applied to business analysis than in academic circles.

**DFD SYMBOLS**

The process that transforms data flow

Source or Destination of Data

Data Flow

Data source

**ENTITY RELATIONSHIP DIAGRAM**

The relation upon the system is structured through a conceptual ER-Diagram, which not only specifies the existential entities but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue. The Entity Relationship Diagram (ERD) depicts the relationship between the data objects. The ERD is the notation that is used to conduct the data modeling activity The attributes of each data object noted in the ERD can be described resign a data object description.

The set of primary components that are identified by the ERD are

* + Data object
  + Relationships
  + Attributes
  + Various types of indicators

The primary purpose of the ERD is to represent data objects and their relationships.

**ER-DIAGRAM SYMBOL**

Entity

Relationship

Flow

**INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specifications and procedures for data preparation and those steps are necessary to put transaction data into a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps, and keeping the process simple. The input is designed in such a way that it provides security and ease of use while retaining privacy.

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system. It is achieved by creating user-friendly screens for the data entry to handle large volumes of data.

The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulations can be performed. It also provides record viewing facilities. When the data is entered it will check for its validity. Data can be entered with the help of screens.

**DATABASE DESIGN**

The database is designed to manage large bodies of information. The management of data involves both the definitions of structures for the storage of information. In addition, the database system must provide for the safety of the information solved, despite system crashes or attempts at unauthorized access. For developing an efficient database users have to fulfill certain conditions such as controlled redundancy.

* Defining the data
* Inputting the data
* Locating the data
* Accessing the data
* Communicating the data

Revising the data

**Objectives of Database Design**

For designing a database design several objectives have to be met as follows:

* Ease of use
* Control of data integrity
* Control of redundancy
* Control of security
* Data independence (logical & physical)
* Data storage protection
* System performance

**OUTPUT DESIGN**

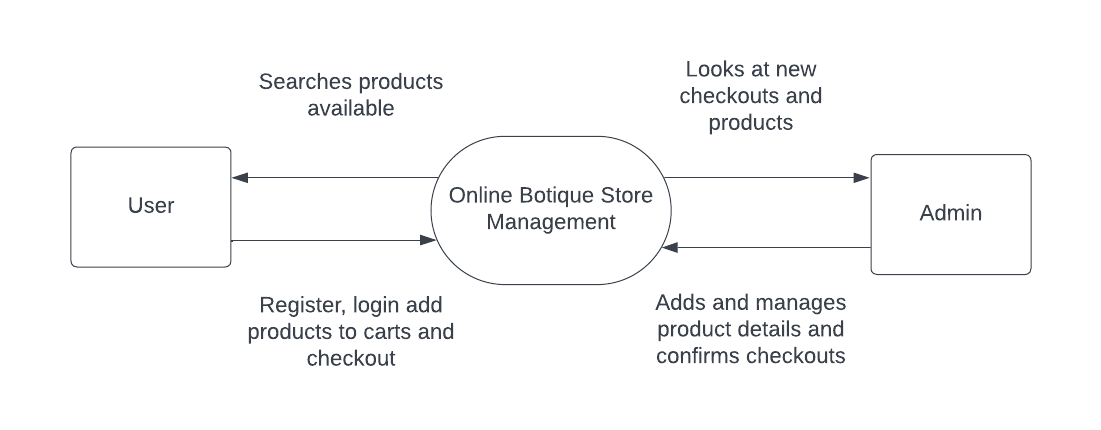
A quality output is one, which meets the requirements of the end user and presents the information. In any system results of processing are communicated to the users and other systems through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source of information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

Output design generally refers to the results and information that are generated by the system for many end-users; output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application.

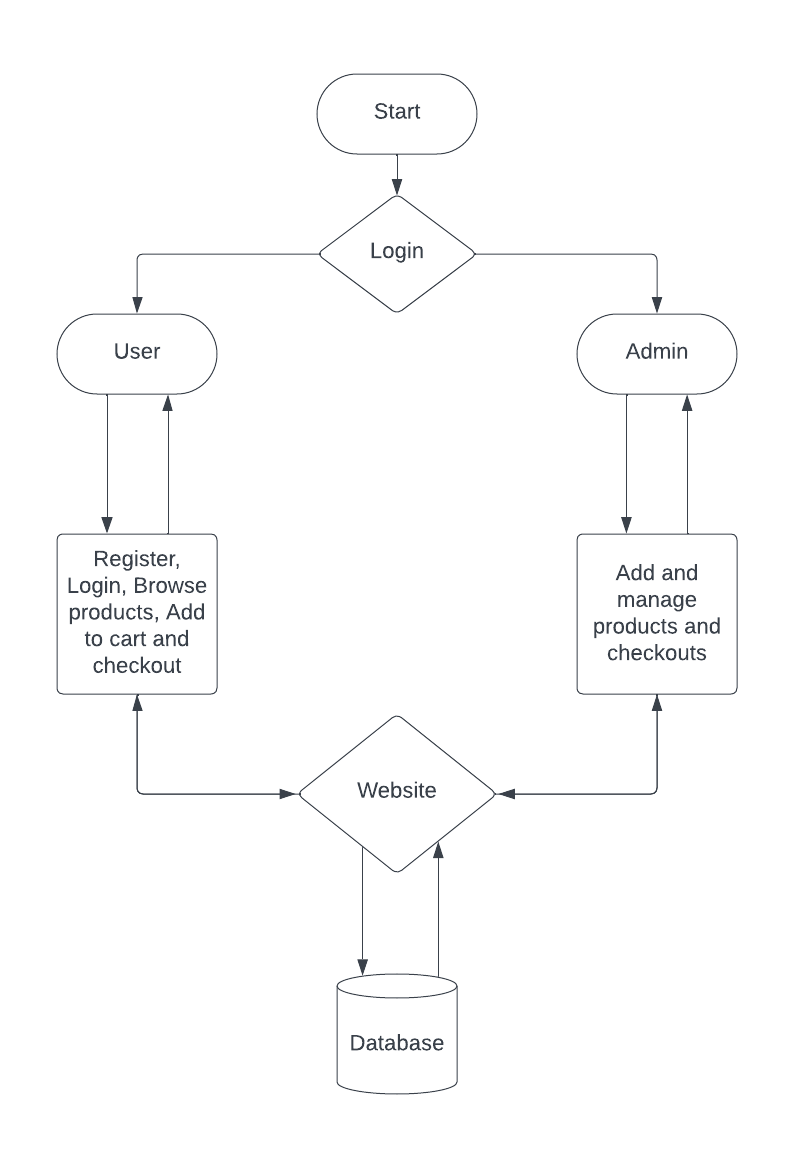
In this Online Repository System project output is to view customer details, employee lists, order tracking details, and attendance percentage results.

**SYSTEM FLOW DIAGRAM**

**DATA FLOW DIAGRAM**

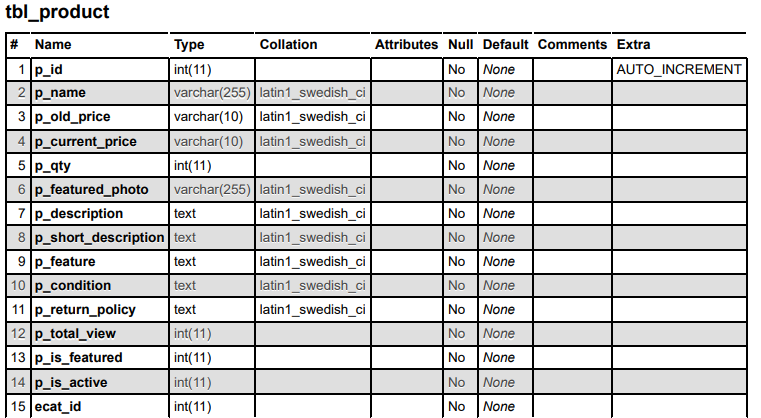


**ER DIAGRAM**

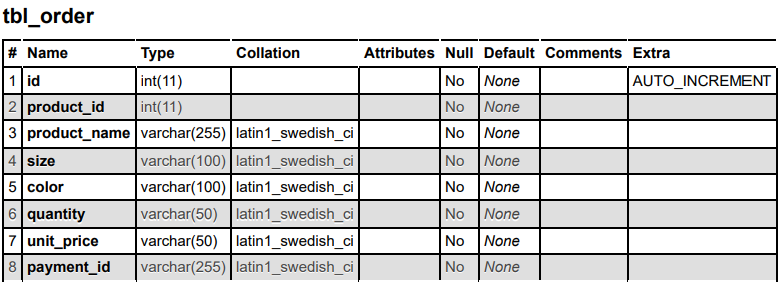


**DATABASE DESIGN**

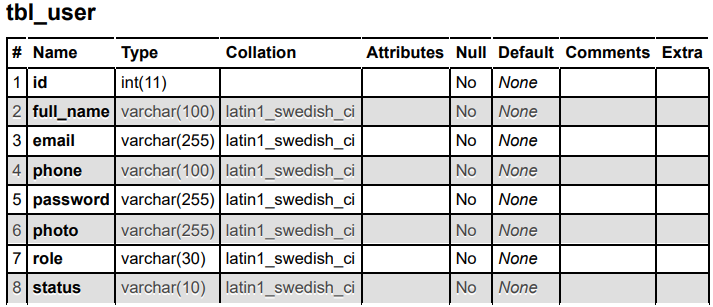
**Table name: tbl\_product**

****

**Table name: tbl\_order**

****

**Table name: tbl\_user**



**CONCLUSION**

In conclusion, the development of the proposed online boutique store represents a significant step towards modernizing the e-commerce landscape and delivering a superior shopping experience for customers. By leveraging a combination of cutting-edge technologies and best practices in web development, we have created a robust and feature-rich platform that addresses the limitations of traditional online boutiques while capitalizing on emerging trends and opportunities in the market.

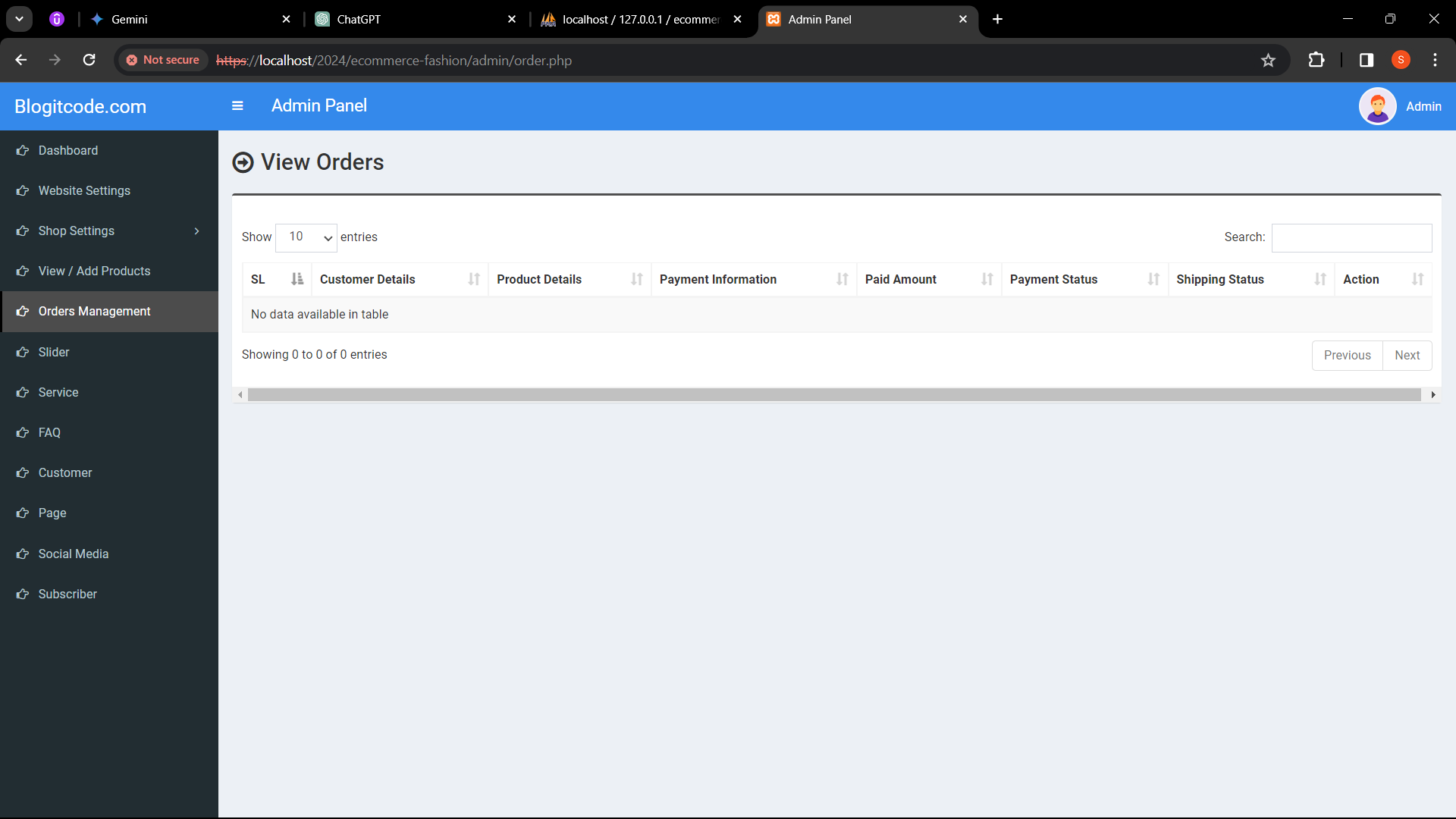
Through dynamic and responsive design, interactive features, and streamlined processes, the online boutique store offers users an engaging and intuitive interface for browsing, purchasing, and managing fashion products. From user authentication and product management to checkout and order fulfillment, each module is meticulously designed to optimize efficiency, security, and user satisfaction.

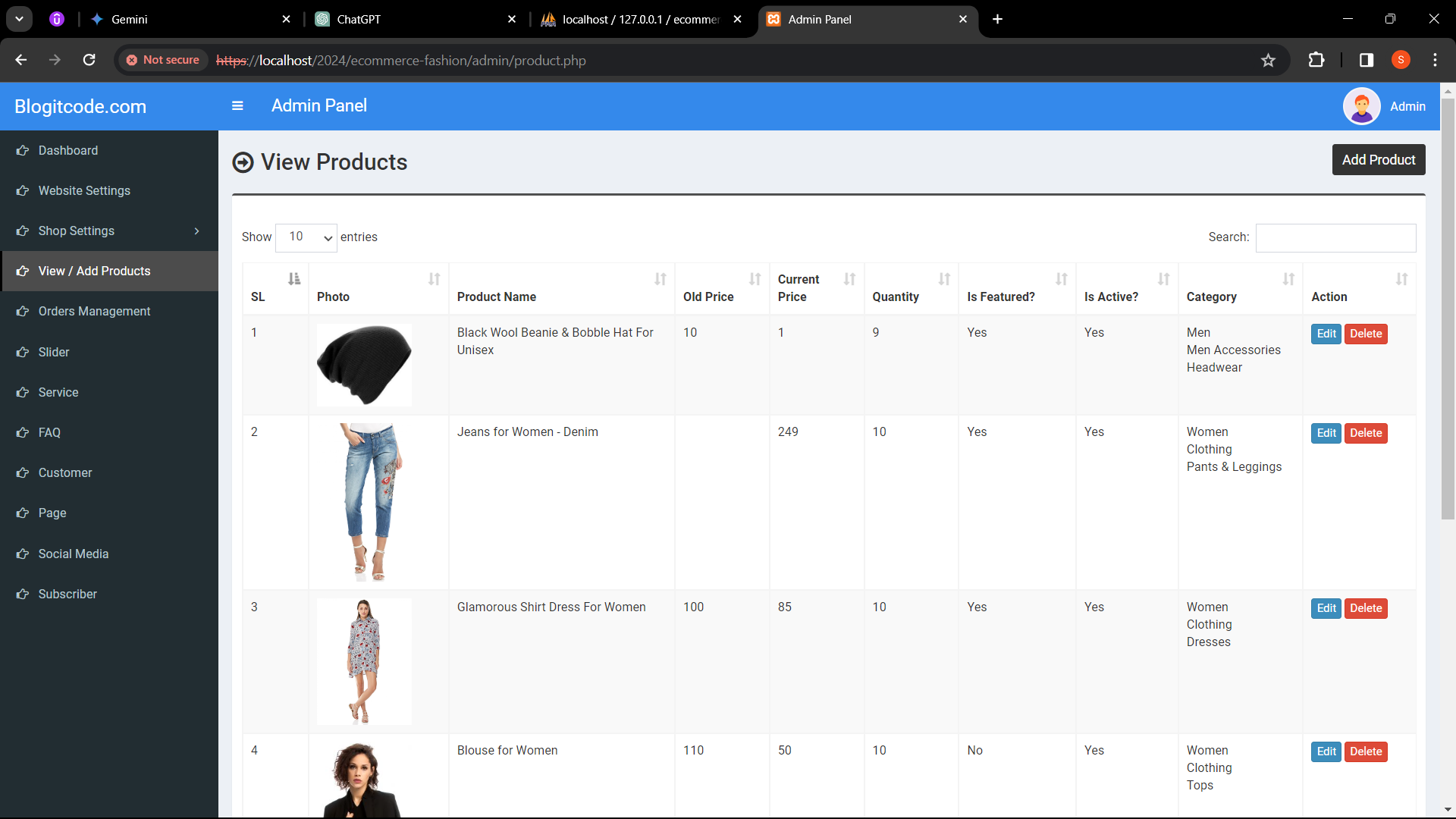
Furthermore, the scalability and flexibility of the architecture ensure that the online boutique store can adapt to evolving business needs, accommodate growth, and integrate new features seamlessly. With comprehensive documentation and ongoing support, administrators have the resources they need to maximize the system's potential and drive continuous improvement.

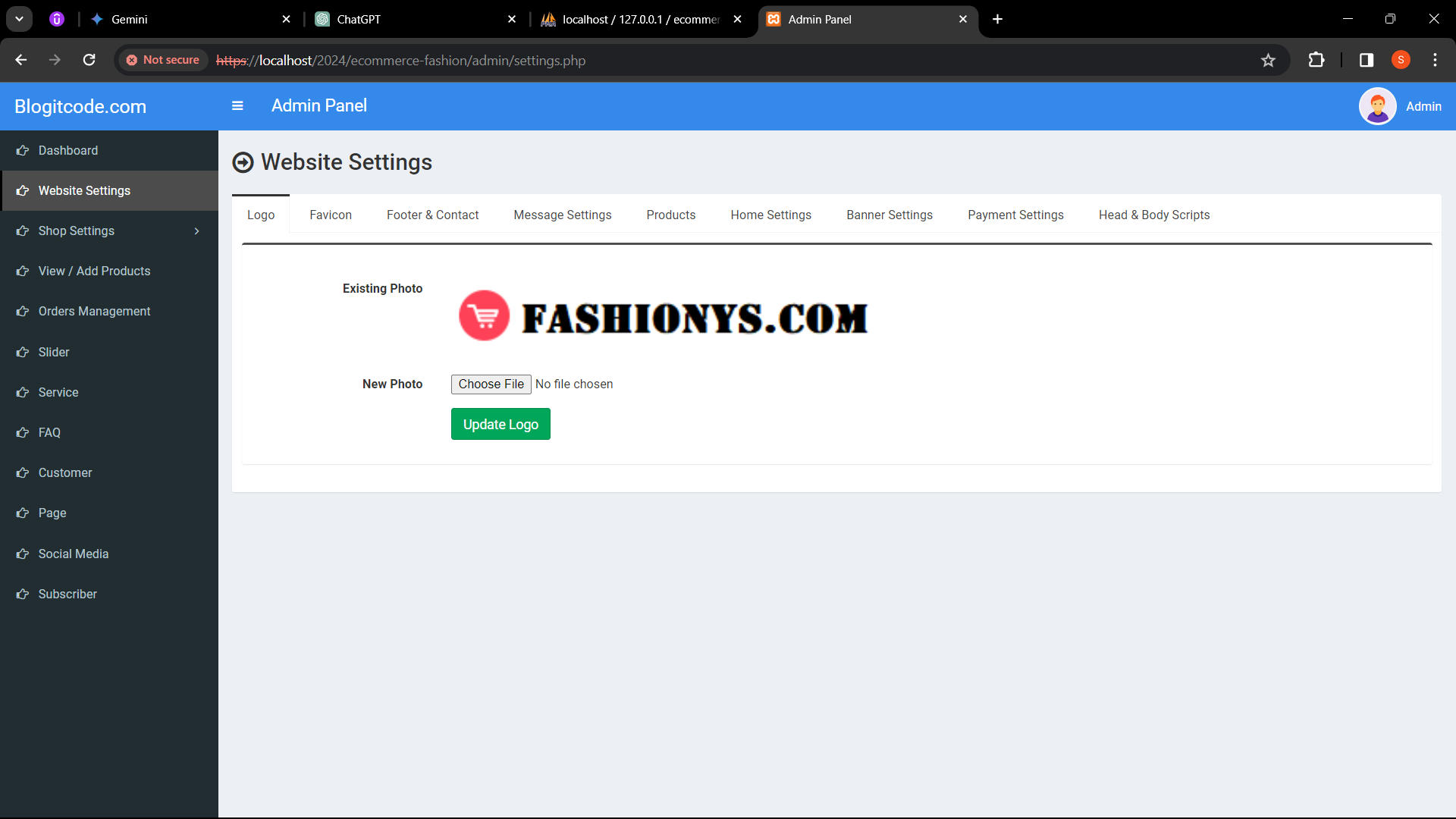
In summary, the proposed online boutique store not only represents a technological advancement in e-commerce but also embodies our commitment to innovation, customer-centricity, and excellence in digital commerce. As we continue to refine and enhance the platform, we are confident that it will establish itself as a leading destination for fashion enthusiasts, driving growth and success in the competitive online retail landscape.

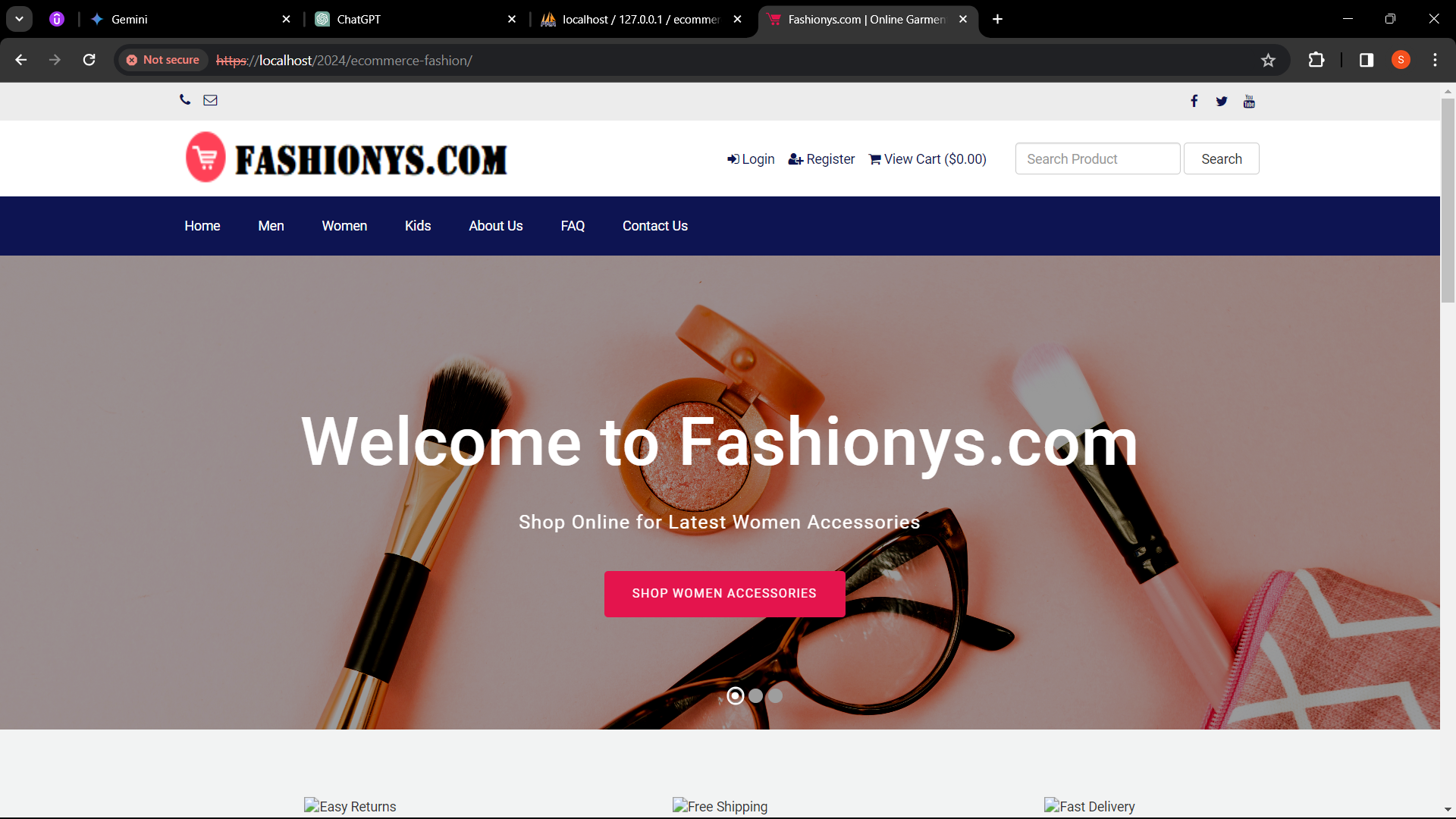
Top of Form

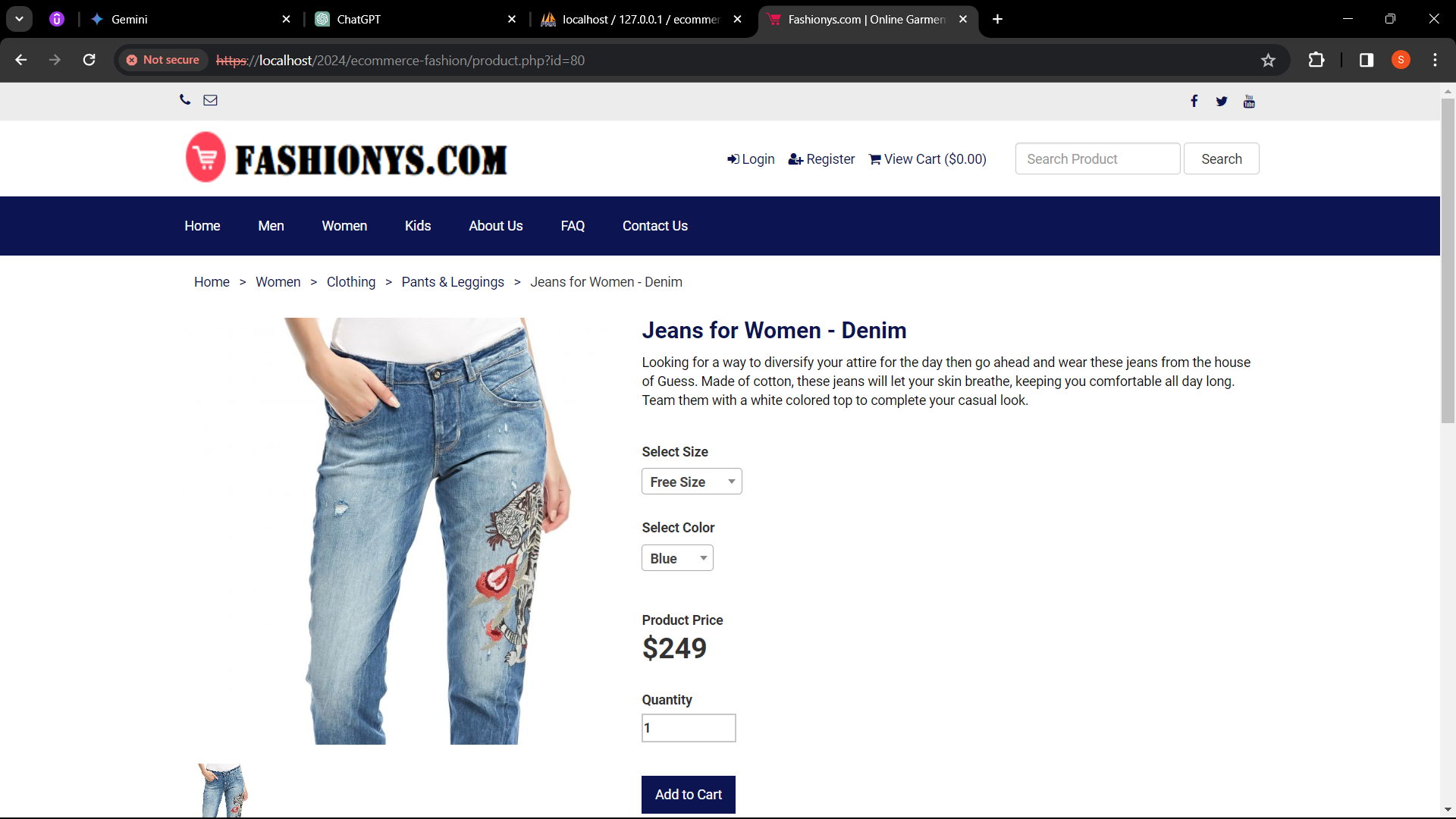
**SCREENSHOTS**

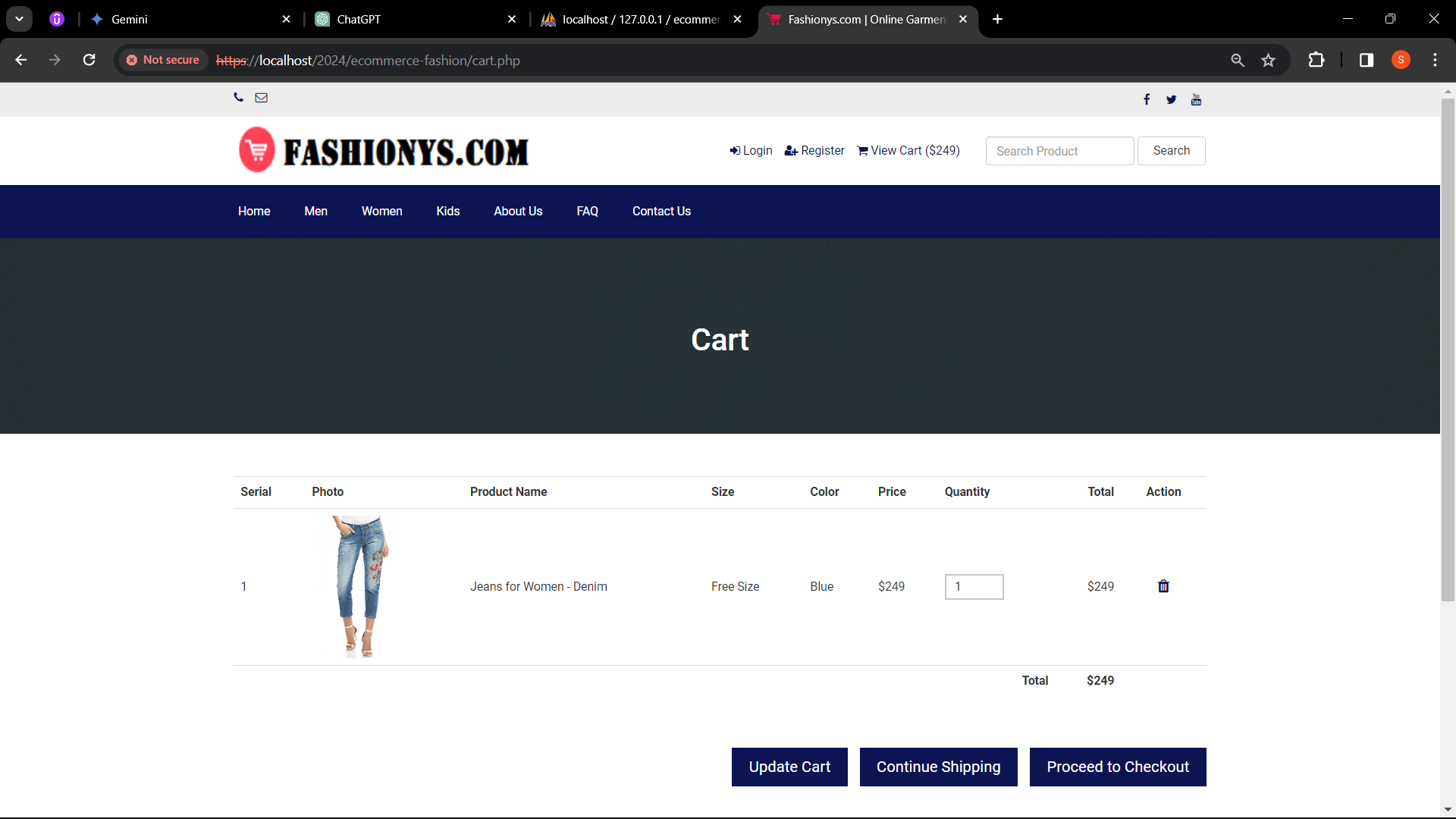
****

****

****

****

****

****

**SAMPLE CODE**

**<?php require\_once('header.php'); ?>**

**<?php**

**if(isset($\_POST['form\_about'])) {**

**$valid = 1;**

**if(empty($\_POST['about\_title'])) {**

**$valid = 0;**

**$error\_message .= 'Title can not be empty<br>';**

**}**

**if(empty($\_POST['about\_content'])) {**

**$valid = 0;**

**$error\_message .= 'Content can not be empty<br>';**

**}**

**$path = $\_FILES['about\_banner']['name'];**

**$path\_tmp = $\_FILES['about\_banner']['tmp\_name'];**

**if($path != '') {**

**$ext = pathinfo( $path, PATHINFO\_EXTENSION );**

**$file\_name = basename( $path, '.' . $ext );**

**if( $ext!='jpg' && $ext!='png' && $ext!='jpeg' && $ext!='gif' ) {**

**$valid = 0;**

**$error\_message .= 'You must have to upload jpg, jpeg, gif or png file<br>';**

**}**

**}**

**if($valid == 1) {**

**if($path != '') {**

**// removing the existing photo**

**$statement = $pdo->prepare("SELECT \* FROM tbl\_page WHERE id=1");**

**$statement->execute();**

**$result = $statement->fetchAll(PDO::FETCH\_ASSOC);**

**foreach ($result as $row) {**

**$about\_banner = $row['about\_banner'];**

**unlink('../assets/uploads/'.$about\_banner);**

**}**

**// updating the data**

**$final\_name = 'about-banner'.'.'.$ext;**

**move\_uploaded\_file( $path\_tmp, '../assets/uploads/'.$final\_name );**

**// updating the database**

**$statement = $pdo->prepare("UPDATE tbl\_page SET about\_title=?,about\_content=?,about\_banner=?,about\_meta\_title=?,about\_meta\_keyword=?,about\_meta\_description=? WHERE id=1");**

**$statement->execute(array($\_POST['about\_title'],$\_POST['about\_content'],$final\_name,$\_POST['about\_meta\_title'],$\_POST['about\_meta\_keyword'],$\_POST['about\_meta\_description']));**

**} else {**

**// updating the database**

**$statement = $pdo->prepare("UPDATE tbl\_page SET about\_title=?,about\_content=?,about\_meta\_title=?,about\_meta\_keyword=?,about\_meta\_description=? WHERE id=1");**

**$statement->execute(array($\_POST['about\_title'],$\_POST['about\_content'],$\_POST['about\_meta\_title'],$\_POST['about\_meta\_keyword'],$\_POST['about\_meta\_description']));**

**}**

**$success\_message = 'About Page Information is updated successfully.';**

**}**

**}**

**if(isset($\_POST['form\_faq'])) {**

**$valid = 1;**

**if(empty($\_POST['faq\_title'])) {**

**$valid = 0;**

**$error\_message .= 'Title can not be empty<br>';**

**}**

**$path = $\_FILES['faq\_banner']['name'];**

**$path\_tmp = $\_FILES['faq\_banner']['tmp\_name'];**

**if($path != '') {**

**$ext = pathinfo( $path, PATHINFO\_EXTENSION );**

**$file\_name = basename( $path, '.' . $ext );**

**if( $ext!='jpg' && $ext!='png' && $ext!='jpeg' && $ext!='gif' ) {**

**$valid = 0;**

**$error\_message .= 'You must have to upload jpg, jpeg, gif or png file<br>';**

**}**

**}**

**if($valid == 1) {**

**if($path != '') {**

**// removing the existing photo**

**$statement = $pdo->prepare("SELECT \* FROM tbl\_page WHERE id=1");**

**$statement->execute();**

**$result = $statement->fetchAll(PDO::FETCH\_ASSOC);**

**foreach ($result as $row) {**

**$faq\_banner = $row['faq\_banner'];**

**unlink('../assets/uploads/'.$faq\_banner);**

**}**

**// updating the data**

**$final\_name = 'faq-banner'.'.'.$ext;**

**move\_uploaded\_file( $path\_tmp, '../assets/uploads/'.$final\_name );**

**// updating the database**

**$statement = $pdo->prepare("UPDATE tbl\_page SET faq\_title=?,faq\_banner=?,faq\_meta\_title=?,faq\_meta\_keyword=?,faq\_meta\_description=? WHERE id=1");**

**$statement->execute(array($\_POST['faq\_title'],$final\_name,$\_POST['faq\_meta\_title'],$\_POST['faq\_meta\_keyword'],$\_POST['faq\_meta\_description']));**

**} else {**

**// updating the database**

**$statement = $pdo->prepare("UPDATE tbl\_page SET faq\_title=?,faq\_meta\_title=?,faq\_meta\_keyword=?,faq\_meta\_description=? WHERE id=1");**

**$statement->execute(array($\_POST['faq\_title'],$\_POST['faq\_meta\_title'],$\_POST['faq\_meta\_keyword'],$\_POST['faq\_meta\_description']));**

**}**

**$success\_message = 'FAQ Page Information is updated successfully.';**

**}**

**}**

**if(isset($\_POST['form\_contact'])) {**

**$valid = 1;**

**if(empty($\_POST['contact\_title'])) {**

**$valid = 0;**

**$error\_message .= 'Title can not be empty<br>';**

**}**

**$path = $\_FILES['contact\_banner']['name'];**

**$path\_tmp = $\_FILES['contact\_banner']['tmp\_name'];**

**if($path != '') {**

**$ext = pathinfo( $path, PATHINFO\_EXTENSION );**

**$file\_name = basename( $path, '.' . $ext );**

**if( $ext!='jpg' && $ext!='png' && $ext!='jpeg' && $ext!='gif' ) {**

**$valid = 0;**

**$error\_message .= 'You must have to upload jpg, jpeg, gif or png file<br>';**

**}**

**}**

**if($valid == 1) {**

**if($path != '') {**

**// removing the existing photo**

**$statement = $pdo->prepare("SELECT \* FROM tbl\_page WHERE id=1");**

**$statement->execute();**

**$result = $statement->fetchAll(PDO::FETCH\_ASSOC);**

**foreach ($result as $row) {**

**$contact\_banner = $row['contact\_banner'];**

**unlink('../assets/uploads/'.$contact\_banner);**

**}**

**// updating the data**

**$final\_name = 'contact-banner'.'.'.$ext;**

**move\_uploaded\_file( $path\_tmp, '../assets/uploads/'.$final\_name );**

**// updating the database**

**$statement = $pdo->prepare("UPDATE tbl\_page SET contact\_title=?,contact\_banner=?,contact\_meta\_title=?,contact\_meta\_keyword=?,contact\_meta\_description=? WHERE id=1");**

**$statement->execute(array($\_POST['contact\_title'],$final\_name,$\_POST['contact\_meta\_title'],$\_POST['contact\_meta\_keyword'],$\_POST['contact\_meta\_description']));**

**} else {**

**// updating the database**

**$statement = $pdo->prepare("UPDATE tbl\_page SET contact\_title=?,contact\_meta\_title=?,contact\_meta\_keyword=?,contact\_meta\_description=? WHERE id=1");**

**$statement->execute(array($\_POST['contact\_title'],$\_POST['contact\_meta\_title'],$\_POST['contact\_meta\_keyword'],$\_POST['contact\_meta\_description']));**

**}**

**$success\_message = 'Contact Page Information is updated successfully.';**

**}**

**}**

**?>**

**<section class="content-header">**

**<div class="content-header-left">**

**<h1>Page Settings</h1>**

**</div>**

**</section>**

**<?php**

**$statement = $pdo->prepare("SELECT \* FROM tbl\_page WHERE id=1");**

**$statement->execute();**

**$result = $statement->fetchAll(PDO::FETCH\_ASSOC);**

**foreach ($result as $row) {**

**$about\_title = $row['about\_title'];**

**$about\_content = $row['about\_content'];**

**$about\_banner = $row['about\_banner'];**

**$about\_meta\_title = $row['about\_meta\_title'];**

**$about\_meta\_keyword = $row['about\_meta\_keyword'];**

**$about\_meta\_description = $row['about\_meta\_description'];**

**$faq\_title = $row['faq\_title'];**

**$faq\_banner = $row['faq\_banner'];**

**$faq\_meta\_title = $row['faq\_meta\_title'];**

**$faq\_meta\_keyword = $row['faq\_meta\_keyword'];**

**$faq\_meta\_description = $row['faq\_meta\_description'];**

**$contact\_title = $row['contact\_title'];**

**$contact\_banner = $row['contact\_banner'];**

**$contact\_meta\_title = $row['contact\_meta\_title'];**

**$contact\_meta\_keyword = $row['contact\_meta\_keyword'];**

**$contact\_meta\_description = $row['contact\_meta\_description'];**

**}**

**?>**

**<section class="content" style="min-height:auto;margin-bottom: -30px;">**

**<div class="row">**

**<div class="col-md-12">**

**<?php if($error\_message): ?>**

**<div class="callout callout-danger">**

**<p>**

**<?php echo $error\_message; ?>**

**</p>**

**</div>**

**<?php endif; ?>**

**<?php if($success\_message): ?>**

**<div class="callout callout-success">**

**<p><?php echo $success\_message; ?></p>**

**</div>**

**<?php endif; ?>**

**</div>**

**</div>**

**</section>**

**<section class="content">**

**<div class="row">**

**<div class="col-md-12">**

**<div class="nav-tabs-custom">**

**<ul class="nav nav-tabs">**

**<li class="active"><a href="#tab\_1" data-toggle="tab">About Us</a></li>**

**<li><a href="#tab\_2" data-toggle="tab">FAQ</a></li>**

**<li><a href="#tab\_4" data-toggle="tab">Contact</a></li>**

**</ul>**

**<!-- About us Page Content -->**

**<div class="tab-content">**

**<div class="tab-pane active" id="tab\_1">**

**<form class="form-horizontal" action="" method="post" enctype="multipart/form-data">**

**<div class="box box-info">**

**<div class="box-body">**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Page Title \* </label>**

**<div class="col-sm-5">**

**<input class="form-control" type="text" name="about\_title" value="<?php echo $about\_title; ?>">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Page Content \* </label>**

**<div class="col-sm-8">**

**<textarea class="form-control" name="about\_content" id="editor1"><?php echo $about\_content; ?></textarea>**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Existing Banner Photo</label>**

**<div class="col-sm-6" style="padding-top:6px;">**

**<img src="../assets/uploads/<?php echo $about\_banner; ?>" class="existing-photo" style="height:80px;">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">New Banner Photo</label>**

**<div class="col-sm-6" style="padding-top:6px;">**

**<input type="file" name="about\_banner">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Meta Title</label>**

**<div class="col-sm-8">**

**<input class="form-control" type="text" name="about\_meta\_title" value="<?php echo $about\_meta\_title; ?>">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Meta Keyword </label>**

**<div class="col-sm-8">**

**<textarea class="form-control" name="about\_meta\_keyword" style="height:100px;"><?php echo $about\_meta\_keyword; ?></textarea>**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Meta Description </label>**

**<div class="col-sm-8">**

**<textarea class="form-control" name="about\_meta\_description" style="height:100px;"><?php echo $about\_meta\_description; ?></textarea>**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label"></label>**

**<div class="col-sm-6">**

**<button type="submit" class="btn btn-success pull-left" name="form\_about">Update</button>**

**</div>**

**</div>**

**</div>**

**</div>**

**</form>**

**</div>**

**<!-- FAQ Page Content -->**

**<div class="tab-pane" id="tab\_2">**

**<form class="form-horizontal" action="" method="post" enctype="multipart/form-data">**

**<div class="box box-info">**

**<div class="box-body">**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Page Title \* </label>**

**<div class="col-sm-5">**

**<input class="form-control" type="text" name="faq\_title" value="<?php echo $faq\_title; ?>">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Existing Banner Photo</label>**

**<div class="col-sm-6" style="padding-top:6px;">**

**<img src="../assets/uploads/<?php echo $faq\_banner; ?>" class="existing-photo" style="height:80px;">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">New Banner Photo</label>**

**<div class="col-sm-6" style="padding-top:6px;">**

**<input type="file" name="faq\_banner">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Meta Title</label>**

**<div class="col-sm-8">**

**<input class="form-control" type="text" name="faq\_meta\_title" value="<?php echo $faq\_meta\_title; ?>">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Meta Keyword </label>**

**<div class="col-sm-8">**

**<textarea class="form-control" name="faq\_meta\_keyword" style="height:100px;"><?php echo $faq\_meta\_keyword; ?></textarea>**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Meta Description </label>**

**<div class="col-sm-8">**

**<textarea class="form-control" name="faq\_meta\_description" style="height:100px;"><?php echo $faq\_meta\_description; ?></textarea>**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label"></label>**

**<div class="col-sm-6">**

**<button type="submit" class="btn btn-success pull-left" name="form\_faq">Update</button>**

**</div>**

**</div>**

**</div>**

**</div>**

**</form>**

**</div>**

**<!-- End of FAQ Page Content -->**

**<div class="tab-pane" id="tab\_4">**

**<form class="form-horizontal" action="" method="post" enctype="multipart/form-data">**

**<div class="box box-info">**

**<div class="box-body">**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Page Title \* </label>**

**<div class="col-sm-5">**

**<input class="form-control" type="text" name="contact\_title" value="<?php echo $contact\_title; ?>">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Existing Banner Photo</label>**

**<div class="col-sm-6" style="padding-top:6px;">**

**<img src="../assets/uploads/<?php echo $contact\_banner; ?>" class="existing-photo" style="height:80px;">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">New Banner Photo</label>**

**<div class="col-sm-6" style="padding-top:6px;">**

**<input type="file" name="contact\_banner">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Meta Title</label>**

**<div class="col-sm-8">**

**<input class="form-control" type="text" name="contact\_meta\_title" value="<?php echo $contact\_meta\_title; ?>">**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Meta Keyword </label>**

**<div class="col-sm-8">**

**<textarea class="form-control" name="contact\_meta\_keyword" style="height:100px;"><?php echo $contact\_meta\_keyword; ?></textarea>**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label">Meta Description </label>**

**<div class="col-sm-8">**

**<textarea class="form-control" name="contact\_meta\_description" style="height:100px;"><?php echo $contact\_meta\_description; ?></textarea>**

**</div>**

**</div>**

**<div class="form-group">**

**<label for="" class="col-sm-3 control-label"></label>**

**<div class="col-sm-6">**

**<button type="submit" class="btn btn-success pull-left" name="form\_contact">Update</button>**

**</div>**

**</div>**

**</div>**

**</div>**

**</form>**

**</div>**

**</form>**

**</div>**

**</div>**

**</section>**

**<?php require\_once('footer.php'); ?>**