# 



**<BronxLuggage>**

**Documentation**

|  |  |
| --- | --- |
| **Authors** | **C2406L – Group 2** |
| **NGUYEN HAI ANH**  **THAI DIEU PHUONG**  **NGUYEN TIEN NGOC LINH**  **VU HOANG NAM** |
| **Date** | **25/03/2025** |
| **Instructor** | **TO HOANG ANH** |

**Hanoi, 2025**

**Index**

[Executive Summary 6](#_Toc193644437)

[Acknowledgement 8](#_Toc193644438)

[Introduction 10](#_Toc193644439)

[Problem Definition 12](#_Toc193644440)

[Problem Abstraction 12](#_Toc193644441)

[The Current System 13](#_Toc193644442)

[The Proposed System 14](#_Toc193644443)

[Key Features of the Proposed System: 15](#_Toc193644444)

[I. Home pages Design 15](#_Toc193644445)

[II. Navigation & Product Categories 15](#_Toc193644446)

[III. Product Search & Filtering 15](#_Toc193644447)

[IV. Product Details & Ratings 15](#_Toc193644448)

[V. Brand Showcase 15](#_Toc193644449)

[VI. Image Gallery 16](#_Toc193644450)

[VII. Customer Feedback & Contact Information 16](#_Toc193644451)

[VIII. Additional Functionalities for an Enhanced User Experience 16](#_Toc193644452)

[**Boundaries of the System** 16](#_Toc193644453)

[I. Target Users 16](#_Toc193644454)

[II. Scope of the System 17](#_Toc193644455)

[Offered Keys Featured: 17](#_Toc193644456)

[Platform Limitations 18](#_Toc193644457)

[III. System Limitations 18](#_Toc193644458)

[Hardware and Software Requirements 19](#_Toc193644459)

[Hardware Requirements 19](#_Toc193644460)

[Server (Hosting Requirements): 19](#_Toc193644461)

[Client (User Devices): 19](#_Toc193644462)

[Software Requirements 20](#_Toc193644463)

[Technologies Used: 20](#_Toc193644464)

[Frontend: 20](#_Toc193644465)

[Backend: 20](#_Toc193644466)

[Database: 20](#_Toc193644467)

[Web Server: 21](#_Toc193644468)

[API: 21](#_Toc193644469)

[Development Tools: 21](#_Toc193644470)

[Testing & Debugging Tools: 21](#_Toc193644471)

[Device Simulation & Browser Testing: 21](#_Toc193644472)

[Other Requirements: 22](#_Toc193644473)

[Customer Requirements Specification 22](#_Toc193644474)

[Users Of the System 22](#_Toc193644475)

[System Function 23](#_Toc193644476)

[**User Authentication** 23](#_Toc193644477)

[**Product Management** 24](#_Toc193644478)

[Systems Designs 24](#_Toc193644479)

[Entity Relationship Diagram 24](#_Toc193644480)

[Database Design 25](#_Toc193644481)

[**Overview of the Database Design** 25](#_Toc193644482)

[**Detailed Table Descriptions** 25](#_Toc193644483)

[**1. users Table** 25](#_Toc193644484)

[**2. orders Table** 26](#_Toc193644485)

[**3. payments Table** 27](#_Toc193644486)

[**4. order\_items Table** 28](#_Toc193644487)

[**5. cart Table** 29](#_Toc193644488)

[**6. products Table** 30](#_Toc193644489)

[**7. product\_images Table** 31](#_Toc193644490)

[**8. sizes Table** 32](#_Toc193644491)

[**9. colors Table** 33](#_Toc193644492)

[**10. brands Table** 33](#_Toc193644493)

[**11. categories Table** 34](#_Toc193644494)

[**12. feedback Table** 34](#_Toc193644495)

[**13. contact Table** 35](#_Toc193644496)

[**Summary of Relationships** 36](#_Toc193644497)

[**Design Considerations** 37](#_Toc193644498)

[**Potential Improvements** 37](#_Toc193644499)

[Sitemap 38](#_Toc193644500)

[**Site Map Overview** 38](#_Toc193644501)

[**Textual Site Map** 38](#_Toc193644502)

[**Conceptual Site Map Diagram** 40](#_Toc193644503)

[**Explanation of the Diagram** 41](#_Toc193644504)

[**Detailed Explanation of the Site Map and Page Interactions** 41](#_Toc193644505)

[**1. Navigation Structure** 41](#_Toc193644506)

[**2. Page Interactions and Navigation Flow** 42](#_Toc193644507)

[**Home (index.php)** 42](#_Toc193644508)

[**Shop (shop.php)** 42](#_Toc193644509)

[**Pages** 44](#_Toc193644510)

[**About Us (about.php)** 45](#_Toc193644511)

[**Contact (contact.php)** 45](#_Toc193644512)

[**Cart Icon (cart.php)** 46](#_Toc193644513)

[**3. Dynamic Behavior and User Flow** 46](#_Toc193644514)

[**4. Typical User Journey** 47](#_Toc193644515)

[**5. Design Considerations** 47](#_Toc193644516)

[**6. Potential Improvements** 48](#_Toc193644517)

[**Conclusion** 49](#_Toc193644518)

[System Functions Design 49](#_Toc193644519)

[**Conceptual Diagram of the Updated Project Structure** 49](#_Toc193644520)

[**Conceptual Diagram Description** 49](#_Toc193644521)

[**Diagram Notes** 51](#_Toc193644522)

[**Detailed Explanation of the Updated Project Structure** 52](#_Toc193644523)

[**Project Root** 52](#_Toc193644524)

[**Folder: account/** 53](#_Toc193644525)

[**Folder: actions/** 54](#_Toc193644526)

[**Folder: css/** 54](#_Toc193644527)

[**Folder: database/** 55](#_Toc193644528)

[**Folder: img/** 55](#_Toc193644529)

[**Folder: includes/** 55](#_Toc193644530)

[**Folder: js/** 56](#_Toc193644531)

[**Folder: pages/** 56](#_Toc193644532)

[**Folder: scss/** 57](#_Toc193644533)

[**1. Overview of System Functions** 57](#_Toc193644534)

[**2. Detailed Description of Each Function** 58](#_Toc193644535)

[**2.1 User Management** 58](#_Toc193644536)

[**2.2 Product Browsing and Filtering** 60](#_Toc193644537)

[**2.3 Cart Management** 61](#_Toc193644538)

[**2.4 Order Processing** 63](#_Toc193644539)

[**2.5 Feedback and Reviews** 65](#_Toc193644540)

[**2.6 Contact Management** 65](#_Toc193644541)

[**2.7 Content Management** 66](#_Toc193644542)

[**3. Interactions Between Functions** 66](#_Toc193644543)

[**4. Design Considerations** 67](#_Toc193644544)

[**Conclusion** 67](#_Toc193644545)

[Task Sheet 68](#_Toc193644546)

[Feasibility study phase 68](#_Toc193644547)

[Requirement Analysis phase 68](#_Toc193644548)

[Design phase 68](#_Toc193644549)

[Development phase 69](#_Toc193644550)

[Testing phase 69](#_Toc193644551)

[Deployment phase 69](#_Toc193644552)

[Validation Checklists 69](#_Toc193644553)

[**Detailed Explanation of the Task Sheet** 69](#_Toc193644554)

[**1. Overview of the Task Sheet** 70](#_Toc193644555)

[**2. Phase-by-Phase Breakdown** 71](#_Toc193644556)

[**2.1 Feasibility Study Phase (25/02/2025 – 04/03/2025)** 71](#_Toc193644557)

[**2.2 Requirement Analysis Phase (01/03/2025 – 04/03/2025)** 72](#_Toc193644558)

[**2.3 Design Phase (01/03/2025 – 05/03/2025)** 73](#_Toc193644559)

[**2.4 Development Phase (05/03/2025 – 18/03/2025)** 74](#_Toc193644560)

[**2.5 Testing Phase (19/03/2025 – 24/03/2025)** 75](#_Toc193644561)

[**2.6 Deployment Phase (25/03/2025 – 29/03/2025)** 76](#_Toc193644562)

[**3. Analysis and Insights** 77](#_Toc193644563)

[Conclusion 79](#_Toc193644564)

# Executive Summary

The **BronxLuggage** project is an academic endeavor undertaken by a team of four students at Aptech—Vu Hoang Nam, Nguyen Tien Ngoc Linh, Thai Dieu Phuong, and Nguyen Hai Anh—as part of our curriculum to apply software development principles in a real-world context. Developed between February 25, 2025, and March 29, 2025, under the guidance of our mentor, Mr. To Hoang Anh, this project aims to create a modern e-commerce platform for Bronx Luggage, a specialized luggage store located in Festival Marketplace. The primary objective of **BronxLuggage** is to enhance the store’s business operations by providing an intuitive, visually appealing, and responsive online shopping experience, enabling customers to browse, filter, and purchase luggage products with ease.

In today’s digital age, the demand for online shopping has surged, driven by the convenience and accessibility it offers. For a traditional retailer like Bronx Luggage, which currently relies on in-person sales at its physical store, transitioning to an online platform is a crucial step to expand its customer reach, streamline operations, and remain competitive in the market. The **BronxLuggage** project addresses this need by developing a Single-Page Application (SPA) and responsive website that supports key e-commerce functionalities, such as product categorization, user account management, shopping cart operations, and customer feedback, while ensuring compatibility across modern web browsers (e.g., Chrome, Firefox) and devices (e.g., desktops, tablets, smartphones).

This project not only serves as a practical application of the knowledge and skills we have acquired at Aptech—ranging from web development technologies (PHP, MySQL, HTML5, CSS3, JavaScript) to project management and teamwork—but also provides us with valuable insights into the challenges and opportunities of building a real-world e-commerce system. Over the course of one month, our team collaborated closely, following a structured development process that included phases such as Feasibility Study, Requirement Analysis, Design, Development, Testing, and Deployment. Each phase was carefully planned and executed, with contributions from all team members, ensuring that the final system meets both functional and non-functional requirements while adhering to the project’s scope and constraints.

This report documents the entire journey of the **BronxLuggage** project, providing a comprehensive overview of its development and outcomes. It begins with the **Problem Definition**, which outlines the background, current challenges faced by Bronx Luggage, the proposed solution, and the system’s boundaries. The **Hardware and Software Requirements** section details the technical specifications needed to implement the system, followed by the **Customer Requirements Specification**, which describes the system’s users and functionalities. The **Systems Designs** section presents the database structure, site map, and system functions, while the **Task Sheet** and **Validation Checklists** highlight the team’s workflow and quality assurance processes. Finally, the **Conclusion** reflects on our achievements, challenges, lessons learned, and future potential for the system.

Through this project, we aim to demonstrate our ability to apply theoretical knowledge to practical scenarios, deliver a high-quality e-commerce platform, and contribute to the digital transformation of Bronx Luggage. We hope that this report provides a clear and detailed account of our efforts, showcasing the skills, creativity, and dedication we have invested in **BronxLuggage**.

# Acknowledgement

We, the team behind the **BronxLuggage** project, would like to express our heartfelt gratitude to everyone who has supported us throughout the journey of completing this e-commerce project. This project has been a significant milestone in our academic and professional growth, providing us with a valuable opportunity to apply the knowledge and skills we have acquired during our time at Aptech.

Throughout our studies, we have been fortunate to be guided and inspired by our teachers at Aptech, who have imparted essential and highly valuable professional knowledge. Their dedication has equipped us with a strong foundation, fostering a faithful, careful, and creative approach to learning that we will carry forward into our future endeavors. The project has allowed us to put into practice the concepts and subjects we have studied, giving us a deeper understanding of real-world applications in software development.

While working on the **BronxLuggage** project, we gained practical and meaningful experience that has enriched our skills and perspectives. Over the course of one term, we dedicated ourselves fully to this project, coming together as a team to brainstorm, collaborate, and labor diligently to bring our vision to life. The process of executing this project was both challenging and rewarding, and we are grateful for the support and guidance we received along the way.

We would like to extend our sincere thanks to our teachers and guider at Aptech for their continuous encouragement and assistance. Their insights and feedback were instrumental in helping us navigate the complexities of the project, from the feasibility study phase to the final deployment.

In particular, we wish to express our deepest gratitude to our teacher, **Mr. To Hoang Anh**, for his enthusiastic, thoughtful, and careful guidance. His mentorship played a pivotal role in shaping the direction of our project, ensuring that we stayed on track and met our objectives. Mr. To Hoang Anh’s dedication to our success, along with his constructive feedback and encouragement, motivated us to strive for excellence at every stage of the project.

Finally, we would like to acknowledge the contributions of each team member—Vu Hoang Nam, Nguyen Tien Ngoc Linh, Thai Dieu Phuong, and Nguyen Hai Anh. Our collective efforts, collaboration, and commitment to the project have been the driving force behind its success. Working together on **BronxLuggage** has not only strengthened our technical skills but also taught us the importance of teamwork, communication, and perseverance.

This project has been a remarkable learning experience, and we are deeply grateful for the support and opportunities provided by Aptech and our mentors. We hope that **BronxLuggage** reflects the knowledge, creativity, and hard work we have invested, and we look forward to applying these experiences to our future projects and careers.

# Introduction

The **BronxLuggage** project is an academic endeavor undertaken by a team of four students at Aptech—Vu Hoang Nam, Nguyen Tien Ngoc Linh, Thai Dieu Phuong, and Nguyen Hai Anh—as part of our curriculum to apply software development principles in a real-world context. Developed between February 25, 2025, and March 29, 2025, under the guidance of our mentor, Mr. To Hoang Anh, this project aims to create a modern e-commerce platform for Bronx Luggage, a specialized luggage store located in Festival Marketplace. The primary objective of **BronxLuggage** is to enhance the store’s business operations by providing an intuitive, visually appealing, and responsive online shopping experience, enabling customers to browse, filter, and purchase luggage products with ease.

In today’s digital age, the demand for online shopping has surged, driven by the convenience and accessibility it offers. For a traditional retailer like Bronx Luggage, which currently relies on in-person sales at its physical store, transitioning to an online platform is a crucial step to expand its customer reach, streamline operations, and remain competitive in the market. The **BronxLuggage** project addresses this need by developing a Single-Page Application (SPA) and responsive website that supports key e-commerce functionalities, such as product categorization, user account management, shopping cart operations, and customer feedback, while ensuring compatibility across modern web browsers (e.g., Chrome, Firefox) and devices (e.g., desktops, tablets, smartphones).

This project not only serves as a practical application of the knowledge and skills we have acquired at Aptech—ranging from web development technologies (PHP, MySQL, HTML5, CSS3, JavaScript) to project management and teamwork—but also provides us with valuable insights into the challenges and opportunities of building a real-world e-commerce system. Over the course of one month, our team collaborated closely, following a structured development process that included phases such as Feasibility Study, Requirement Analysis, Design, Development, Testing, and Deployment. Each phase was carefully planned and executed, with contributions from all team members, ensuring that the final system meets both functional and non-functional requirements while adhering to the project’s scope and constraints.

This report documents the entire journey of the **BronxLuggage** project, providing a comprehensive overview of its development and outcomes. It begins with the **Problem Definition**, which outlines the background, current challenges faced by Bronx Luggage, the proposed solution, and the system’s boundaries. The **Hardware and Software Requirements** section details the technical specifications needed to implement the system, followed by the **Customer Requirements Specification**, which describes the system’s users and functionalities. The **Systems Designs** section presents the database structure, site map, and system functions, while the **Task Sheet** and **Validation Checklists** highlight the team’s workflow and quality assurance processes. Finally, the **Conclusion** reflects on our achievements, challenges, lessons learned, and future potential for the system.

Through this project, we aim to demonstrate our ability to apply theoretical knowledge to practical scenarios, deliver a high-quality e-commerce platform, and contribute to the digital transformation of Bronx Luggage. We hope that this report provides a clear and detailed account of our efforts, showcasing the skills, creativity, and dedication we have invested in **BronxLuggage**.

# Problem Definition

The BronxLuggage project is an e-commerce platform developed as part of our academic curriculum at Aptech. This project aims to create a fully functional online store for selling luggage products, providing users with a seamless shopping experience. The following subsections provide an overview of the project’s details, the team behind its development, the background and rationale for the product, and a review of existing systems that inspired or informed our design.

## Problem Abstraction

In today's modern world, the increasing demand for travel has made choosing the right luggage more important than ever. A high-quality luggage set not only ensures convenience for every trip but also reflects the style and class of its user. These products can be passed down through generations, becoming a reliable companion for every journey.

Bronx Luggage, a specialized luggage store located in Festival Marketplace, is committed to providing customers with a wide range of options, from affordable luggage sets to the most premium brands on the market. Additionally, the store offers backpacks, travel accessories, briefcases, and many other related products.

To enhance the shopping experience and effectively reach more customers, this project aims to develop a **Single-Page Application (SPA) and a responsive website**. The website will not only allow users to easily browse and select products but also help Bronx Luggage expand its online presence.

This project is carried out to create an intuitive and visually appealing e-commerce platform while ensuring compatibility with popular browsers such as Chrome, Internet Explorer, and Firefox, as well as mobile devices. It is a crucial step in modernizing Bronx Luggage's business operations, enabling the company to maximize the potential of digital technology.

## The Current System

Currently, Bronx Luggage primarily operates through its physical store located in Festival Marketplace. Customers visit the store in person to browse and purchase luggage, backpacks, travel accessories, and other related products. The purchasing process heavily relies on face-to-face interactions with sales representatives, who provide recommendations and assist customers in selecting suitable products based on their travel needs and preferences.

Since there is no dedicated online platform, product information is mostly shared through in-store displays and word-of-mouth. Customers who are not physically present at the store have limited access to product details, pricing, and availability. Marketing efforts are also restricted to traditional methods such as flyers, posters, and occasional promotions within the marketplace.

Without an online system, the business faces several challenges, including:

* Limited customer reach, as purchases can only be made in-store.
* No digital catalog, making it difficult for potential customers to browse products remotely.
* Dependence on manual inventory management, which can lead to inefficiencies.
* Lack of an online ordering system, preventing customers from making purchases conveniently from home.

Due to these limitations, Bronx Luggage needs a modern e-commerce solution that allows customers to explore and purchase products online, improving accessibility, customer engagement, and overall business growth.

## The Proposed System

To address the limitations of the current business model, this project proposes the development of a Single-Page Application (SPA) and a responsive website for Bronx Luggage. The new system will provide an intuitive, visually appealing, and user-friendly online shopping experience while ensuring compatibility across various browsers (Chrome, Internet Explorer, Firefox) and mobile devices.

#### Key Features of the Proposed System:

##### Home pages Design

* The home page will include multiple sections with a **suitable logo and header** showcasing various luggage and bags
* A modern and visually appealing design with smooth animations and high-quality images.

##### Navigation & Product Categories

* A well-structured navigation menu with sections for different types of **luggage, bags, and backpacks**.
* Separate sections for **Men and Women**, ensuring products are categorized accordingly.

##### Product Search & Filtering

* Customers can filter products based on **size, color, price, and brand**.
* A search bar to help users quickly find specific products.

##### Product Details & Ratings

* Each product will have a dedicated page displaying **specifications, pricing, and images**.
* Users will be able to **rate and review** products, enhancing customer engagement.

##### Brand Showcase

* A dedicated section displaying all **brands** offered in the store

##### Image Gallery

* A **gallery section** showcasing different images of products from various angles.

##### Customer Feedback & Contact Information

* Users can submit **feedback and reviews** about products and their shopping experience.
* An "About Us" and "Contact Us" section displaying the **company’s email, address, and phone number**.

##### Additional Functionalities for an Enhanced User Experience

* A **continuous scrolling ticker** at the bottom of the page displaying the **current date, time, and location** using HTML5 geolocation.
* A **visitor count** displayed at the **top right corner** of the page beside the store's logo.
* Interactive **menu options that change color on hover and after clicking**.
* Smooth **fade-in and fade-out effects** for menu transitions.

By implementing this **modern e-commerce solution**, Bronx Luggage will significantly enhance its online presence, streamline operations, and provide a **seamless, visually engaging, and responsive** shopping experience for customers.

### **Boundaries of the System**

The proposed system has specific boundaries that define its **users, scope, and limitations**. These boundaries ensure a clear understanding of who can access the system and how it will be utilized.

#### Target Users

* Individuals looking to purchase luggage, backpacks, travel accessories, and briefcases
* Users who want to browse, filter, and compare different products before making a purchase.
* Customers who wish to provide product reviews and feedback.

#### Scope of the System

* The system will operate as an **online e-commerce platform** for Bronx Luggage, allowing customers to view and purchase products
* It will be accessible via **modern web browsers** such as Chrome, Internet Explorer, and Firefox.
* The website will be fully **responsive**, supporting **desktop, tablet, and mobile devices**.
* The system will handle **customer accounts, order management, and payment transactions**.

##### Offered Keys Featured:

* **User Management**: Users can register, log in, update their profile, and have different roles (admin, customer).
* **Product Management**: Admins can add, edit, and remove products, categories, brands, sizes, and colors.
* **Shopping Cart & Payment**: Users can add products to the shopping cart and proceed to checkout with available payment methods.
* **Order Management**: The system tracks orders and updates their status (processing, shipping, completed, canceled).
* **Customer Feedback**: Users can leave ratings and reviews for products.
* **Admin Panel**: Allows the administration of product data, user management, order tracking, and revenue reporting.
* **Security**: Ensures data protection with encryption for sensitive data, and includes safeguards against security vulnerabilities like SQL Injection and XSS attacks.

##### Platform Limitations

* **Web-Only Access**: The system is currently available only as a web application and does not have a mobile app.
* **Online Transactions Only**: It supports only online payments and does not integrate with physical store POS systems.

#### System Limitations

* The platform will be limited to **online browsing and ordering**; it will **not support in-store inventory synchronization in real-time** unless integrated with an external system
* The website **does not include a full-fledged AI chatbot or customer support system** but may provide a contact form for inquiries.
* Shipping and delivery tracking features are **not included** in the initial phase; customers will receive order updates via email.
* The platform will support **limited payment methods** initially (e.g., credit/debit cards, PayPal).
* **No Multi-Platform Support**: The system is designed to function only on web browsers (no native mobile apps for Android or iOS).
* **No AI Recommendations**: Product recommendations based on user behavior or history are not included.
* **Limited Payment Methods**: Only basic payment methods such as credit card, bank transfers, and cash on delivery are available. No support for third-party services like PayPal, Apple Pay, etc.
* **Single Language Support**: The system currently supports only Vietnamese, with no multi-language support.
* **No Live Chat**: Real-time customer support chat is not available in this version.
* **No Product Comparison or Wishlist Features**: There are no features to compare multiple products or save products to a wishlist for later.

By defining these boundaries, the system ensures a **well-structured and focused implementation**, catering to both customers and administrators while setting clear expectations regarding its functionalities and limitations.

### Hardware and Software Requirements

#### Hardware Requirements

##### Server (Hosting Requirements):

* **CPU**: AMD Ryzen 7 or Intel Core i7 (latest generation), or equivalent multi-core processors.
* **RAM**: Minimum 16GB, recommended 32GB+ for optimal performance under heavy traffic.
* **Storage**: SSD storage (512GB+ recommended), NVMe SSD preferred for higher read/write speeds.
* **Network Bandwidth**: 1Gbps network connection for optimal speed and reliability.
* **Operating System**: Ubuntu Server (latest LTS version), CentOS, or Windows Server (latest version).
* **Other**: Backup power supply (UPS) to ensure uptime in case of power failure

##### Client (User Devices):

* **CPU**: Intel Core i5/i7 or AMD Ryzen 5/7 or higher (latest generation).
* **RAM**: Minimum 8GB, recommended 16GB for smooth multi-tasking.
* **Storage**: SSD (256GB or higher recommended for faster access to websites).
* **Graphics Card**: Integrated graphics are sufficient, but dedicated graphics may be beneficial for users with high media consumption.
* **Display**: Minimum resolution 1920x1080p (Full HD) for a better browsing experience.
* **Operating System**: Windows 10/11, macOS (latest stable version), Linux (latest LTS version).
* **Web Browsers**: Chrome, Firefox, Safari, Edge (latest versions recommended for full functionality and security).
* **Other**: Keyboard, mouse, and optional speakers/headphones if needed for audio.

#### Software Requirements

##### Technologies Used:

###### Frontend:

* HTML5, CSS3, JavaScript (ES6+), Bootstrap (latest version).
* UI Design: Figma (for wireframing and UI/UX design).

###### Backend:

* PHP (Latest stable version, 8.x+ reccmmendation) with MySQL 8.x.

###### Database:

* Primary: MySQL 8.x or MariaDB for robust database management.
* Pr Secondary (for lightweight storage): JSON or SQLite for lightweight data storage.

###### Web Server:

* Apache HTTP Server or Nginx (latest stable versions recommended).

###### API:

* RESTful API (for potential third-party integrations and mobile apps in future versions).

###### Development Tools:

* **Code Editors**: Visual Studio Code (VSCode), Sublime Text, or JetBrains PhpStorm.
* **Version Control**: Git, with repositories hosted on GitHub, GitLab, or Bitbucket.
* **Development Environment**: Docker for containerization and local development setups.
* **Package Manager**: npm (for JavaScript packages), Composer (for PHP).

###### Testing & Debugging Tools:

* **API Testing**: Postman for testing APIs.
* **Unit Testing**: PHPUnit for PHP, Jest for JavaScript testing.
* **UI Testing**: Selenium, Cypress for end-to-end testing.
* **Performance Testing**: Apache JMeter or Google Lighthouse for load testing and performance checks.

###### Device Simulation & Browser Testing:

* **Chrome DevTools** for inspecting and simulating device layouts and responsiveness.
* **BrowserStack** for cross-browser testing to ensure compatibility across all platforms and devices.

###### Other Requirements:

* **SSL/TLS**: SSL certificates are required for encrypted HTTPS communication in the live environment to ensure data security.
* **Security Measures**: Implement security best practices, including two-factor authentication (2FA) for users, server-side input validation, and HTTPS enforcement.

# Customer Requirements Specification

## Users Of the System

In this section, you will describe all the users who will interact with the system. This includes their roles, privileges, and responsibilities. Break down the types of users and explain what each user is able to do within the system.

Example:

* **Admin User**: Admins are responsible for overseeing the entire system. They can manage users, monitor system performance, set permissions, and configure the system settings. They will have full access to all features of the system.
* **End Users**: These are the people who will use the system for the core functionality. They might be employees, customers, or other types of users. Their responsibilities include [insert tasks such as data entry, processing requests, accessing reports, etc.]. They have restricted access based on roles.
* **Guest Users**: These are users who are not logged in and can only access a limited amount of information. They may be able to browse some public pages or view some general content but not interact with restricted or personalized features.
* **System**: The system itself, which processes data, performs automated tasks, and communicates with other systems as necessary.

## System Function

This section breaks down the functional requirements of the system, analyzing each function that was identified in section 1.3 of the document. For each function, you will provide the following details:

* **Inputs**: The data or information that needs to be provided for the function to work.
* **Outputs**: The data or result that the system will produce once the function is executed.
* **Processing**: The steps the system will take to process the input and generate the output.
* **Data Storage**: Where and how the data related to the function will be stored (e.g., in a database).

**User Authentication**

* **Inputs**: Username, password.
* **Outputs**: Success/failure message, user profile.
* **Processing**: The system will compare the provided username and password with those stored in the database. If they match, the user is authenticated. If they don't, an error message is displayed.
* **Data Storage**: The system stores encrypted passwords, last login time, and login attempts.

**Product Management**

* **Inputs**: Product name, description, price, stock quantity.
* **Outputs**: Success/failure message, updated product list.
* **Processing**: When a new product is added, the system will validate the data, check for any existing products with the same name, and add the new product to the database.
* **Data Storage**: Product details such as name, description, price, quantity, and product image are stored in the **Products** table.

# Systems Designs

## Entity Relationship Diagram

The Entity Relationship Diagram (ERD) visually represents how entities (such as Users, Products, Orders, etc.) relate to one another within the database. You should use a tool like Visio 2010 to draw a clean and professional ERD. Make sure each entity is represented by a box, and relationships between entities are shown as lines. Primary and foreign keys should be indicated to highlight relationships.

## Database Design

The diagram provided represents a database design for an e-commerce system, featuring multiple tables with defined relationships, fields, and their respective functionalities. Below is a detailed explanation of the database design, including the table names, their relationships, field names, and the purpose of each field within the tables. This explanation is written in English and aims to be as comprehensive as possible.

**Overview of the Database Design**

The database is structured to support an e-commerce platform, managing users, products, orders, payments, and related entities. It consists of the following tables: users, orders, payments, order\_items, cart, products, product\_images, sizes, colors, brands, categories, feedback, and contact. These tables are interconnected through foreign key relationships, ensuring referential integrity and enabling efficient querying for various e-commerce operations.

**Detailed Table Descriptions**

**1. users Table**

This table stores information about the users of the e-commerce platform, such as customers or administrators.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each user. This field is auto-incremented and serves as the primary key.
  + name (VARCHAR(100)): Stores the full name of the user.
  + email (VARCHAR(100), Index): Stores the user's email address, which is indexed for faster lookup during login or search operations.
  + password (VARCHAR(255)): Stores the hashed password of the user for authentication purposes. The length of 255 accommodates secure hashing algorithms like bcrypt.
  + phone (VARCHAR(20)): Stores the user's phone number.
  + address (VARCHAR(255)): Stores the user's physical address, which may be used for shipping purposes.
  + role (ENUM(...)): Represents the role of the user (e.g., "customer," "admin"). The ENUM type ensures only predefined roles are allowed.
  + created\_at (TIMESTAMP): Records the date and time when the user account was created.
* **Indexes:**
  + An index on the email field to optimize searches and lookups.
* **Relationships:**
  + The id field is referenced by the user\_id field in the orders, cart, feedback, and contact tables, establishing a one-to-many relationship. One user can have multiple orders, cart entries, feedback entries, and contact messages.

**2. orders Table**

This table stores information about customer orders.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each order.
  + user\_id (INT, Foreign Key): References the id field in the users table, identifying the user who placed the order.
  + status (ENUM(...)): Indicates the current status of the order (e.g., "pending," "shipped," "delivered").
  + created\_at (TIMESTAMP): Records the date and time when the order was placed.
* **Indexes:**
  + An index on the user\_id field to optimize queries involving user-specific orders.
* **Relationships:**
  + The id field is referenced by the order\_id field in the order\_items and payments tables, establishing a one-to-many relationship. One order can have multiple order items and payments.

**3. payments Table**

This table stores payment details for orders.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each payment.
  + order\_id (INT, Foreign Key): References the id field in the orders table, linking the payment to a specific order.
  + amount (DECIMAL(10,2)): Stores the payment amount with two decimal places for precision (e.g., 99.99).
  + status (ENUM(...)): Indicates the payment status (e.g., "pending," "completed," "failed").
  + payment\_method (ENUM(...)): Specifies the payment method used (e.g., "credit\_card," "paypal").
  + created\_at (TIMESTAMP): Records the date and time when the payment was initiated.
* **Indexes:**
  + An index on the order\_id field to optimize queries related to order payments.
* **Relationships:**
  + The order\_id field establishes a many-to-one relationship with the orders table. One order can have multiple payment attempts (e.g., if a payment fails and the user retries).

**4. order\_items Table**

This table stores the individual items within an order.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each order item.
  + order\_id (INT, Foreign Key): References the id field in the orders table, linking the item to a specific order.
  + product\_id (INT, Foreign Key): References the id field in the products table, identifying the product being ordered.
  + quantity (INT): Specifies the quantity of the product in the order.
  + price (DECIMAL(10,2)): Stores the price of the product at the time of the order (to account for price changes over time).
* **Indexes:**
  + Indexes on order\_id and product\_id to optimize queries involving order details and product lookups.
* **Relationships:**
  + The order\_id field establishes a many-to-one relationship with the orders table.
  + The product\_id field establishes a many-to-one relationship with the products table. One order can contain multiple items, and each item corresponds to a specific product.

**5. cart Table**

This table stores items in a user's shopping cart before they place an order.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each cart entry.
  + user\_id (INT, Foreign Key): References the id field in the users table, identifying the user who owns the cart.
  + product\_id (INT, Foreign Key): References the id field in the products table, identifying the product in the cart.
  + quantity (INT): Specifies the quantity of the product in the cart.
  + created\_at (TIMESTAMP): Records the date and time when the item was added to the cart.
* **Indexes:**
  + Indexes on user\_id and product\_id to optimize cart retrieval and product lookups.
* **Relationships:**
  + The user\_id field establishes a many-to-one relationship with the users table.
  + The product\_id field establishes a many-to-one relationship with the products table. One user can have multiple cart entries, and each entry corresponds to a specific product.

**6. products Table**

This table stores information about the products available for purchase.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each product.
  + name (VARCHAR(255)): Stores the name of the product.
  + description (TEXT): Provides a detailed description of the product.
  + category\_id (INT, Foreign Key): References the id field in the categories table, linking the product to a category.
  + brand\_id (INT, Foreign Key): References the id field in the brands table, linking the product to a brand.
  + size\_id (INT, Foreign Key): References the id field in the sizes table, linking the product to a size.
  + color\_id (INT, Foreign Key): References the id field in the colors table, linking the product to a color.
  + price (DECIMAL(10,2)): Stores the price of the product.
  + gender (ENUM(...)): Specifies the target gender for the product (e.g., "male," "female," "unisex").
  + created\_at (TIMESTAMP): Records the date and time when the product was added to the system.
* **Indexes:**
  + Indexes on category\_id, brand\_id, size\_id, and color\_id to optimize filtering and searching by these attributes.
* **Relationships:**
  + The id field is referenced by the product\_id field in the order\_items, cart, feedback, and product\_images tables, establishing a one-to-many relationship.
  + The category\_id, brand\_id, size\_id, and color\_id fields establish many-to-one relationships with the categories, brands, sizes, and colors tables, respectively.

**7. product\_images Table**

This table stores images associated with products.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each product image.
  + product\_id (INT, Foreign Key): References the id field in the products table, linking the image to a specific product.
  + image\_url (VARCHAR(255)): Stores the URL or file path of the product image.
  + is\_primary (TINYINT(1)): Indicates whether this image is the primary image for the product (1 for true, 0 for false).
  + created\_at (TIMESTAMP): Records the date and time when the image was added.
* **Indexes:**
  + An index on product\_id to optimize image retrieval for a specific product.
* **Relationships:**
  + The product\_id field establishes a many-to-one relationship with the products table. One product can have multiple images.

**8. sizes Table**

This table stores available sizes for products.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each size.
  + name (VARCHAR(50)): Stores the name of the size (e.g., "S," "M," "L").
* **Relationships:**
  + The id field is referenced by the size\_id field in the products table, establishing a one-to-many relationship. One size can be associated with multiple products.

**9. colors Table**

This table stores available colors for products.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each color.
  + name (VARCHAR(50)): Stores the name of the color (e.g., "Red," "Blue").
* **Relationships:**
  + The id field is referenced by the color\_id field in the products table, establishing a one-to-many relationship. One color can be associated with multiple products.

**10. brands Table**

This table stores information about product brands.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each brand.
  + name (VARCHAR(100)): Stores the name of the brand.
* **Indexes:**
  + An index on the name field to optimize brand lookups.
* **Relationships:**
  + The id field is referenced by the brand\_id field in the products table, establishing a one-to-many relationship. One brand can be associated with multiple products.

**11. categories Table**

This table stores product categories.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each category.
  + name (VARCHAR(100)): Stores the name of the category (e.g., "Clothing," "Electronics").
* **Indexes:**
  + An index on the name field to optimize category lookups.
* **Relationships:**
  + The id field is referenced by the category\_id field in the products table, establishing a one-to-many relationship. One category can contain multiple products.

**12. feedback Table**

This table stores user feedback on products.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each feedback entry.
  + user\_id (INT, Foreign Key): References the id field in the users table, identifying the user who provided the feedback.
  + product\_id (INT, Foreign Key): References the id field in the products table, identifying the product being reviewed.
  + message (TEXT): Stores the feedback message or review text.
  + rating (INT): Stores the rating given by the user (e.g., 1 to 5).
  + created\_at (TIMESTAMP): Records the date and time when the feedback was submitted.
* **Indexes:**
  + Indexes on user\_id and product\_id to optimize feedback retrieval.
* **Relationships:**
  + The user\_id field establishes a many-to-one relationship with the users table.
  + The product\_id field establishes a many-to-one relationship with the products table. One user can provide feedback for multiple products, and one product can have feedback from multiple users.

**13. contact Table**

This table stores contact messages submitted by users.

* **Fields:**
  + id (INT, Primary Key): A unique identifier for each contact message.
  + user\_id (INT, Foreign Key): References the id field in the users table, identifying the user who submitted the message.
  + name (VARCHAR(100)): Stores the name of the user (may be redundant if the user is logged in).
  + email (VARCHAR(100)): Stores the email address of the user for follow-up communication.
  + subject (VARCHAR(100)): Stores the subject of the contact message.
  + message (TEXT): Stores the content of the contact message.
* **Indexes:**
  + An index on user\_id to optimize message retrieval for a specific user.
* **Relationships:**
  + The user\_id field establishes a many-to-one relationship with the users table. One user can submit multiple contact messages.

**Summary of Relationships**

* **One-to-Many Relationships:**
  + users to orders, cart, feedback, and contact (via user\_id).
  + orders to order\_items and payments (via order\_id).
  + products to order\_items, cart, feedback, and product\_images (via product\_id).
  + categories, brands, sizes, and colors to products (via category\_id, brand\_id, size\_id, and color\_id).
* **Many-to-One Relationships:**
  + order\_items, cart, feedback, and product\_images to products (via product\_id).
  + orders, cart, feedback, and contact to users (via user\_id).
  + order\_items and payments to orders (via order\_id).
  + products to categories, brands, sizes, and colors (via their respective IDs).

**Design Considerations**

1. **Normalization:** The database is normalized to at least the third normal form (3NF), reducing redundancy. For example, product attributes like size, color, and brand are stored in separate tables rather than duplicating them in the products table.
2. **Indexes:** Indexes are used on frequently queried fields (e.g., email in users, user\_id in orders) to improve performance.
3. **Data Types:** Appropriate data types are chosen for each field (e.g., DECIMAL(10,2) for prices, ENUM for predefined values like status).
4. **Timestamps:** The created\_at field in most tables allows for tracking the creation time of records, which is useful for auditing and analytics.
5. **Flexibility:** The use of TEXT for fields like description and message allows for variable-length content, while VARCHAR is used for fields with predictable lengths.

**Potential Improvements**

1. **Additional Fields:** The orders table could include a total\_amount field to store the computed total for the order, avoiding the need to calculate it on the fly.
2. **Soft Deletion:** Adding a deleted\_at (TIMESTAMP) field to tables like users and products would enable soft deletion, preserving data for auditing purposes.
3. **Composite Indexes:** For tables like order\_items, a composite index on (order\_id, product\_id) could further optimize queries that filter by both fields.
4. **Stock Management:** The products table could include a stock\_quantity (INT) field to track inventory levels, which would be updated when orders are placed.

## Sitemap

**Site Map Overview**

A site map is a visual or textual representation of a website's structure, showing the hierarchy of pages and how they are interconnected through navigation links. For this e-commerce website, the site map is derived from the navbar structure, which includes links to various pages and dropdown menus. The site map will illustrate the relationships between the main pages (Home, Shop, Pages, About Us, Contact) and their subpages (e.g., Shop dropdown options and Pages dropdown options), as well as the dynamic category links generated from the database.

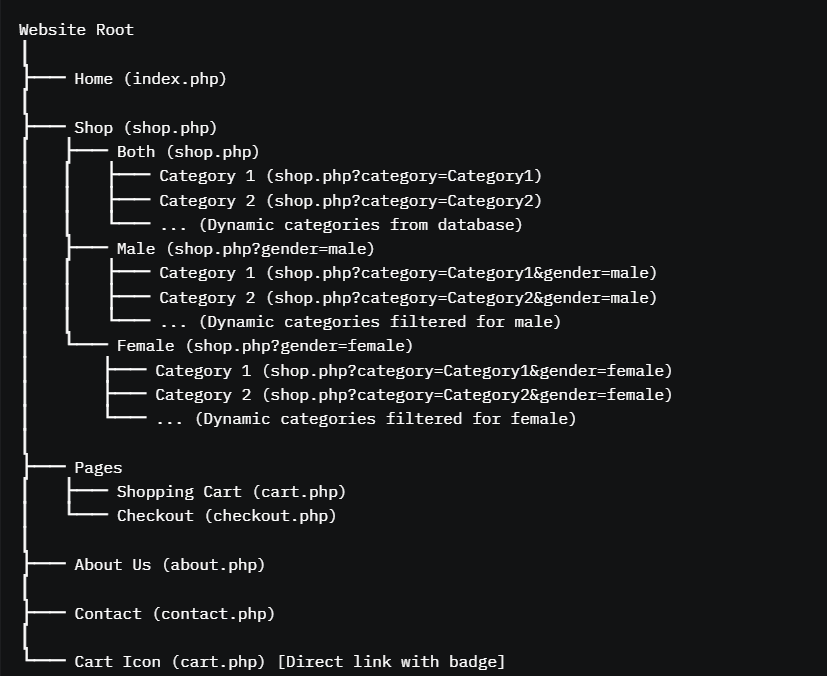
**Textual Site Map**

The site map is organized hierarchically, with the main navigation links at the top level and their subpages or dynamic links as child nodes. Here's the detailed structure:

* **Home** (index.php)
  + No subpages; this is the landing page of the website.
* **Shop** (shop.php)
  + **Both** (shop.php)
    - Displays products for all genders.
    - **Dynamic Categories** (shop.php?category=<category\_name>)
      * Each category (e.g., "Luggage", "Backpacks") links to shop.php?category=<category\_name>.
  + **Male** (shop.php?gender=male)
    - Displays products specifically for males.
    - **Dynamic Categories** (shop.php?category=<category\_name>&gender=male)
      * Filters male products by category.
  + **Female** (shop.php?gender=female)
    - Displays products specifically for females.
    - **Dynamic Categories** (shop.php?category=<category\_name>&gender=female)
      * Filters female products by category.
* **Pages**
  + **Shopping Cart** (cart.php)
    - Displays the user's cart and allows them to proceed to checkout.
  + **Checkout** (checkout.php)
    - Handles the checkout process for placing an order.
* **About Us** (about.php)
  + No subpages; provides information about the company.
* **Contact** (contact.php)
  + No subpages; allows users to submit contact messages.
* **Additional Navigation Element (Cart Icon)** (cart.php)
  + A direct link to the cart page, accessible from the navbar with a badge showing the total number of products in the cart.

**Conceptual Site Map Diagram**

Here’s a textual representation of the site map in a hierarchical format. .



**Explanation of the Diagram**

* **Root Level:** The website's root represents the starting point, with the main navigation links branching out.
* **Home:** A standalone page with no subpages, serving as the entry point.
* **Shop:** A dropdown menu with three options (Both, Male, Female). Each option links to shop.php with different query parameters. Additionally, the "Categories" section in the navbar dynamically generates links to filter products by category, which are nested under each gender option.
* **Pages:** A dropdown menu with two subpages (Shopping Cart and Checkout), representing the e-commerce workflow for purchasing products.
* **About Us and Contact:** Standalone pages with no subpages, providing static content.
* **Cart Icon:** A direct link in the navbar, visually distinct with a badge showing the number of products in the cart.

**Detailed Explanation of the Site Map and Page Interactions**

**1. Navigation Structure**

The navbar is the primary navigation tool for the website, providing access to all major sections. It is divided into two main parts:

* **Left Section (Categories):** A vertical dropdown menu visible on larger screens (col-lg-3 d-none d-lg-block), which dynamically lists categories fetched from the database. Each category links to shop.php?category=<category\_name>.
* **Right Section (Main Navbar):** A horizontal navbar (col-lg-9) with the following links:
  + **Home:** Links to index.php.
  + **Shop:** A dropdown with three options (Both, Male, Female), linking to shop.php with appropriate query parameters.
  + **Pages:** A dropdown with two options (Shopping Cart and Checkout), linking to cart.php and checkout.php.
  + **About Us:** Links to about.php.
  + **Contact:** Links to contact.php.
  + **Cart Icon:** A direct link to cart.php, with a badge showing the total number of products in the cart.

**2. Page Interactions and Navigation Flow**

**Home (index.php)**

* **Purpose:** The landing page of the website, likely showcasing featured products, promotions, or a general overview of the store.
* **Navigation Options:**
  + Users can navigate to any other page via the navbar: Shop, Pages (Cart/Checkout), About Us, or Contact.
  + The "Categories" dropdown allows users to jump directly to a category-specific shop page.
  + The cart icon provides quick access to the cart.
* **Interaction:** This page serves as the entry point. Users typically start here and then navigate to the Shop section to browse products.

**Shop (shop.php)**

* **Purpose:** The main product browsing page, with filtering options for gender and category.
* **Subpages and Filters:**
  + **Both (shop.php):** Displays all products without gender filtering.
  + **Male (shop.php?gender=male):** Filters products for males.
  + **Female (shop.php?gender=female):** Filters products for females.
  + **Categories (shop.php?category=<category\_name>):** Dynamically generated links from the database, allowing users to filter products by category. These can be combined with gender filters (e.g., shop.php?category=Luggage&gender=male).
* **Navigation Options:**
  + From any Shop page, users can switch between gender filters using the dropdown.
  + The "Categories" dropdown allows users to refine their search by category.
  + Users can add products to their cart, which updates the cart badge in the navbar.
  + Users can navigate to Cart or Checkout via the "Pages" dropdown or the cart icon.
  + Links to Home, About Us, and Contact are available in the navbar.
* **Interaction:** This is the core of the e-commerce experience. Users browse products, apply filters, and add items to their cart. The dynamic category links ensure flexibility in product discovery.

**Pages**

* **Shopping Cart (cart.php)**
  + **Purpose:** Displays the user's cart, showing the products they have added, their quantities, and the total price.
  + **Navigation Options:**
    - Users can proceed to Checkout to place their order.
    - They can return to Shop to continue browsing.
    - Links to Home, About Us, and Contact are available.
  + **Interaction:** This page is a stepping stone in the purchase process. Users can modify their cart (e.g., update quantities, remove items) before proceeding to checkout. The cart icon in the navbar provides a direct link to this page, making it easily accessible from anywhere on the site.
* **Checkout (checkout.php)**
  + **Purpose:** Handles the checkout process, where users enter shipping and payment information to place their order.
  + **Navigation Options:**
    - After completing the checkout, users are typically redirected to a confirmation page (not explicitly shown in the navbar but assumed to exist).
    - They can return to Cart if they need to modify their order.
    - Links to Home, Shop, About Us, and Contact are available.
  + **Interaction:** This page finalizes the purchase process. It is directly accessible from the "Pages" dropdown or after clicking "Proceed to Checkout" from the Cart page. The checkout process likely involves form submissions and payment processing, after which the user is redirected to a confirmation page or back to the Home page.

**About Us (about.php)**

* **Purpose:** Provides information about the company, such as its history, mission, or team.
* **Navigation Options:**
  + Users can navigate to any other page via the navbar: Home, Shop, Pages, or Contact.
  + The cart icon allows quick access to the cart.
* **Interaction:** This is a static informational page. Users typically visit this page to learn more about the company and then return to shopping or other activities.

**Contact (contact.php)**

* **Purpose:** Allows users to submit inquiries or feedback via a contact form.
* **Navigation Options:**
  + Users can navigate to any other page via the navbar: Home, Shop, Pages, or About Us.
  + The cart icon allows quick access to the cart.
* **Interaction:** This page is for user communication with the site administrators. After submitting a message, users are likely redirected to a confirmation page or back to the Home page.

**Cart Icon (cart.php)**

* **Purpose:** Provides a quick link to the cart page, with a badge showing the total number of products.
* **Interaction:** The cart icon is a convenience feature, allowing users to check their cart from any page without navigating through the "Pages" dropdown. The badge updates dynamically based on the $\_SESSION['cart'] and database queries, ensuring the user always sees the current cart status.

**3. Dynamic Behavior and User Flow**

* **Cart Updates:** The cart badge ($total\_products) is calculated dynamically using both session data ($\_SESSION['cart']) and database queries (SELECT COUNT(DISTINCT product\_id) FROM cart WHERE user\_id = ?). This ensures that the cart count is accurate whether the user is logged in or not. When a user adds a product to their cart from the Shop page, the badge updates immediately.
* **Category Filtering:** The "Categories" dropdown is populated dynamically using the getCategories($conn) function, which fetches categories from the database. Each category links to shop.php?category=<category\_name>, allowing users to filter products by category. This can be combined with gender filters (e.g., shop.php?category=Luggage&gender=male).
* **Active Page Highlighting:** The navbar uses PHP to highlight the active page by adding the active class to the corresponding nav link. For example, if the user is on shop.php?gender=male, the "Shop" dropdown and the "Male" dropdown item are both marked as active.
* **User Session:** The navbar checks for $\_SESSION['user\_id'] to determine if the user is logged in. If logged in, it fetches cart data from the database; otherwise, it relies on session data. This ensures a seamless experience for both logged-in and guest users.

**4. Typical User Journey**

1. **Landing on Home Page:** The user starts at index.php, where they see an overview of the store.
2. **Browsing Products:** They navigate to the Shop page, either by selecting "Both," "Male," or "Female" from the dropdown or by clicking a category from the "Categories" dropdown.
3. **Filtering Products:** On the Shop page, they apply filters (e.g., category or gender) to find products of interest.
4. **Adding to Cart:** They add products to their cart, which updates the cart badge in the navbar.
5. **Viewing Cart:** They click the cart icon or navigate to Cart via the "Pages" dropdown to review their cart.
6. **Proceeding to Checkout:** From the Cart page, they proceed to Checkout to place their order.
7. **Exploring Other Pages:** At any point, they can visit About Us or Contact to learn more about the company or submit a message.
8. **Returning to Home:** After completing their actions, they return to the Home page or continue shopping.

**5. Design Considerations**

* **User-Friendly Navigation:** The navbar is responsive, with a hamburger menu for mobile devices (d-block d-lg-none) and a full layout for larger screens. The "Categories" dropdown is hidden on smaller screens to save space.
* **Dynamic Content:** The use of dynamic categories ensures the site can scale as new categories are added to the database.
* **Cart Accessibility:** The cart icon with a badge provides a constant visual cue, encouraging users to review their cart and proceed to checkout.
* **Consistency:** The navbar is consistent across all pages, ensuring users can always navigate to any section of the site.

**6. Potential Improvements**

* **Search Functionality:** Adding a search bar to the navbar would allow users to search for products directly, enhancing the browsing experience.
* **Account Management:** Including links to user account pages (e.g., "My Account," "Order History") in the navbar would improve the experience for logged-in users.
* **Breadcrumb Navigation:** On the Shop page, adding breadcrumb navigation (e.g., "Home > Shop > Male > Luggage") would help users understand their current location within the site.
* **Confirmation Pages:** After checkout or contact form submission, redirecting users to dedicated confirmation pages (e.g., order-confirmation.php, contact-confirmation.php) would improve the user experience.

**Conclusion**

The site map for this e-commerce website is structured to provide a clear and intuitive navigation experience. The main pages (Home, Shop, Pages, About Us, Contact) are accessible from the navbar, with dropdown menus for Shop and Pages providing additional subpages. The dynamic "Categories" dropdown adds flexibility, allowing users to filter products by category. The cart icon ensures quick access to the cart, and the active page highlighting helps users understand their current location.

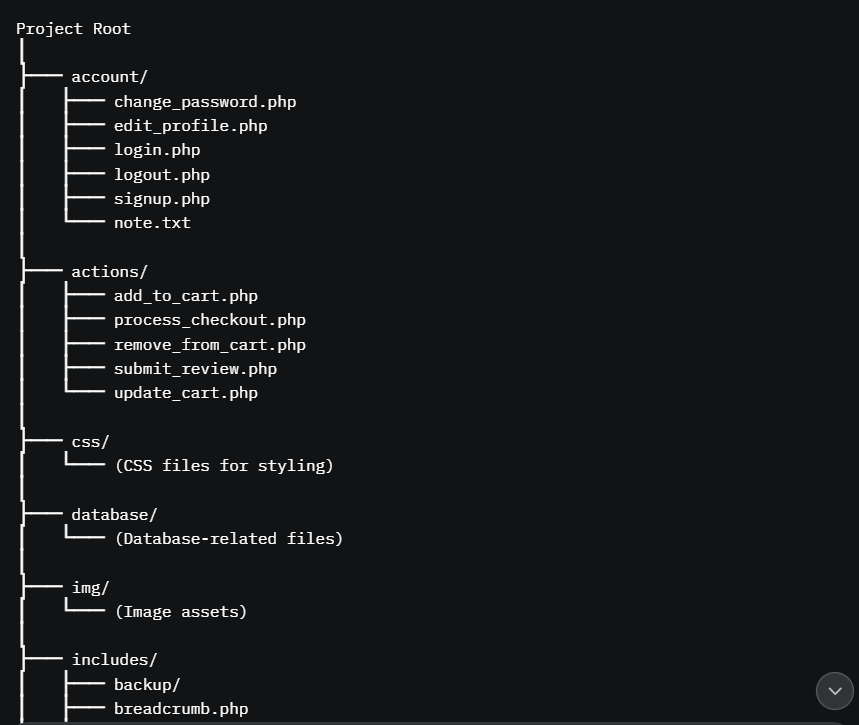
The navigation flow is designed to guide users through the e-commerce process: browsing products, adding them to the cart, and completing the purchase. Informational pages like About Us and Contact provide additional context, while the dynamic cart badge keeps users informed of their cart status. This structure balances functionality and usability, making the website easy to navigate for both new and returning users.

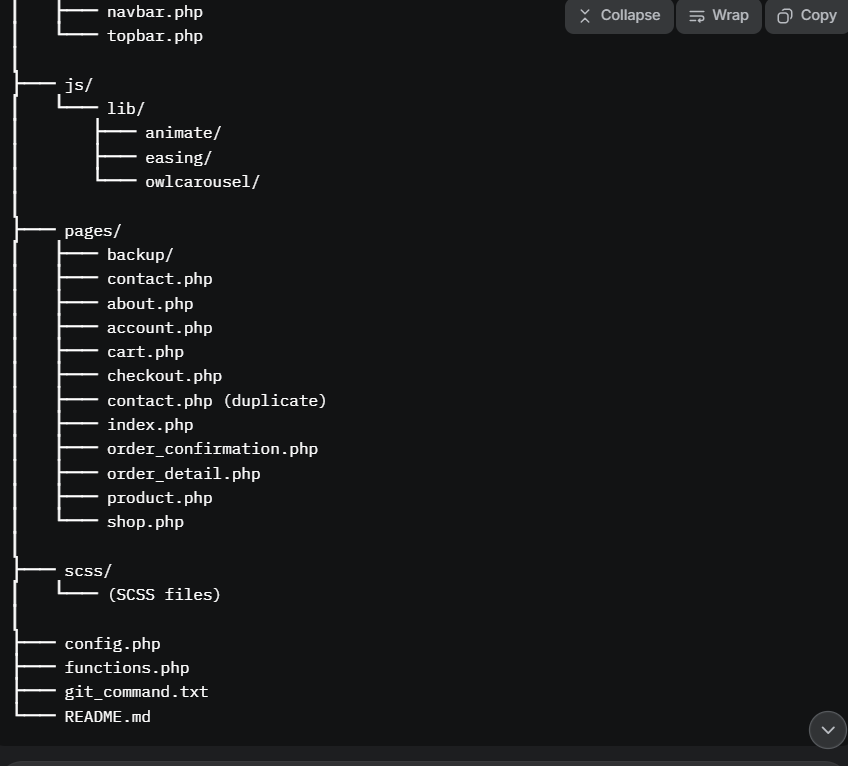
## System Functions Design

**Conceptual Diagram of the Updated Project Structure**

**Conceptual Diagram Description**

The conceptual diagram will be a tree-like structure representing the hierarchy of folders and files in the updated project structure. Each folder will be a parent node, and the files within it will be child nodes. Arrows or lines will connect related files to show their interactions (e.g., how navbar.php is included in multiple pages). Here’s the textual representation of the diagram:





**Diagram Notes**

* **Folders** are represented as boxes with a bold outline.
* **Files** are represented as smaller boxes or nodes within the folder boxes.
* **Interactions** are shown with arrows:
  + navbar.php, head.php, footer.php, topbar.php, and breadcrumb.php in the includes/ folder are connected to most files in the pages/ folder, indicating they are included in those pages.
  + Files in the actions/ folder (e.g., add\_to\_cart.php, process\_checkout.php) are connected to specific pages (e.g., cart.php, checkout.php) to show their role in handling form submissions or actions.
  + config.php and functions.php are connected to most files, as they provide global configuration and utility functions.
* **Dynamic Elements** (e.g., scss/ and css/) are noted as being compiled or used by the pages for styling.

**Detailed Explanation of the Updated Project Structure**

**Project Root**

The root directory contains the main structure of the project, including folders for different types of files and some configuration files.

* **config.php**: Contains configuration settings for the project, such as database connection details (e.g., host, username, password, database name). It likely defines constants or variables like DB\_HOST, DB\_USER, etc., and is included in most PHP scripts to establish a database connection.
* **functions.php**: Contains reusable utility functions used across the project, such as getCategories($conn) (seen in the navbar code), for sanitizing user input, or calculateCartTotal($cartItems) for computing the cart total. This file is included in most pages to provide common functionality.
* **git\_command.txt**: A text file containing Git commands for version control (e.g., git init, git add ., git commit -m "Initial commit"). This file is for development purposes and not part of the application’s runtime.
* **README.md**: A Markdown file providing documentation for the project, including setup instructions, project overview, or contribution guidelines.

**Folder: account/**

This folder contains PHP scripts for user account management.

* **change\_password.php**: Allows users to change their password. It includes a form for entering the current and new passwords, validates the input, and updates the password field in the users table.
* **edit\_profile.php**: Enables users to edit their profile information (e.g., name, email, phone, address). It retrieves the user’s data from the users table, displays it in a form, and updates the database upon submission.
* **login.php**: Manages user login by displaying a form, validating the email and password against the users table, and setting session variables (e.g., $\_SESSION['user\_id']) upon successful login.
* **logout.php**: Handles user logout by destroying the session (session\_destroy()) and redirecting to index.php.
* **signup.php**: Manages user registration with a form for entering details (e.g., name, email, password). It validates the input, hashes the password, and inserts a new record into the users table.
* **note.txt**: A text file for developer notes (e.g., "Add email verification for signup"). It is not part of the application’s runtime.

**Folder: actions/**

This folder contains PHP scripts that handle specific actions, typically form submissions or AJAX requests.

* **add\_to\_cart.php**: Adds a product to the user’s cart. It receives a product\_id and quantity, adds the product to $\_SESSION['cart'] (for guests) or the cart table (for logged-in users), and redirects to cart.php.
* **process\_checkout.php**: Processes the checkout form submission from checkout.php. It creates a new order in the orders table, adds items to the order\_items table, processes the payment (inserting into the payments table), clears the cart, and redirects to order\_confirmation.php.
* **remove\_from\_cart.php**: Removes a product from the cart. It receives a product\_id, removes the item from $\_SESSION['cart'] or the cart table, and redirects to cart.php.
* **submit\_review.php**: Handles product review submissions. It receives a product\_id, message, and rating, inserts a record into the feedback table, and redirects to product.php.
* **update\_cart.php**: Updates the quantity of a product in the cart. It receives a product\_id and new quantity, updates $\_SESSION['cart'] or the cart table, and redirects to cart.php.

**Folder: css/**

This folder contains CSS files for styling the website, likely compiled from SCSS files in the scss/ folder. These files style the navbar, product listings, forms, etc.

**Folder: database/**

This folder likely contains database-related files, such as SQL scripts for creating the database schema (e.g., users, orders, products) or a script for establishing a database connection.

**Folder: img/**

This folder contains image assets, such as product images (referenced in the product\_images table), logos, icons, and other static assets.

**Folder: includes/**

This folder contains reusable PHP scripts included in multiple pages for common functionality or layout components.

* **Subfolder: backup/**: Contains backups of the include files (e.g., older versions of navbar.php).
* **breadcrumb.php**: Generates a breadcrumb navigation trail (e.g., "Home > Shop > Male") based on the current page’s URL and query parameters.
* **footer.php**: Contains the HTML and PHP code for the website’s footer, included at the bottom of most pages. It might include links to about.php, contact.php, and social media icons.
* **head.php**: Contains the HTML <head> section for all pages, including meta tags, CSS links, and JavaScript includes. It is included at the top of each page in the pages/ folder.
* **navbar.php**: Contains the navigation bar code (as provided earlier), including links to Home, Shop, Pages, About Us, Contact, a dynamic "Categories" dropdown, and a cart icon with a badge.
* **topbar.php**: Contains the top bar of the website, which might include a search bar, user account links, or promotional messages.

**Folder: js/**

This folder contains JavaScript files for adding interactivity to the website.

**Folder: lib/**

* + **Subfolder: animate/**: Contains JavaScript libraries or scripts for animations (e.g., Animate.css).
  + **Subfolder: easing/**: Contains JavaScript for easing effects (e.g., jQuery Easing plugin).
  + **Subfolder: owlcarousel/**: Contains the Owl Carousel library for creating product sliders or carousels on pages like index.php or shop.php.

**Folder: pages/**

This folder contains the main PHP pages of the website.

* **Subfolder: backup/**: Contains backups of the page files (e.g., older versions of index.php).
* **contact.php**: Displays the contact page with a form for submitting inquiries. It inserts the form data into the contact table. (Note: There are two contact.php files, which is likely a mistake; one should be removed.)
* **about.php**: Displays the "About Us" page with static content about the company.
* **account.php**: Displays the user’s account dashboard, with options like "Edit Profile" (edit\_profile.php), "Change Password" (change\_password.php), and "Order History" (linking to order\_detail.php).
* **cart.php**: Displays the user’s shopping cart, showing products, quantities, and total price. It allows users to update quantities (update\_cart.php), remove items (remove\_from\_cart.php), or proceed to checkout (checkout.php).
* **checkout.php**: Displays the checkout page with forms for shipping and payment information. It submits the data to process\_checkout.php.
* **index.php**: The homepage, likely displaying featured products or a product carousel (using Owl Carousel).
* **order\_confirmation.php**: Displays a confirmation message after a successful checkout, showing the order details.
* **order\_detail.php**: Displays the details of a specific order, including items, total amount, and status.
* **product.php**: Displays the details of a specific product, including its name, description, price, images, and reviews. It includes forms to add to the cart (add\_to\_cart.php) and submit a review (submit\_review.php).
* **shop.php**: Displays the product listing page, allowing users to browse products with filters for gender and category.

**Folder: scss/**

This folder contains SCSS files that are compiled into CSS files in the css/ folder, defining the website’s styles in a modular way.

**1. Overview of System Functions**

The e-commerce system facilitates online shopping, user account management, and customer support. The key functional areas include:

* **User Management:** Handles user registration, login, profile editing, and password changes.
* **Product Browsing and Filtering:** Allows users to browse products by category and gender.
* **Cart Management:** Manages the user’s shopping cart, both in the session and database.
* **Order Processing:** Handles the checkout process, order creation, and payment.
* **Feedback and Reviews:** Allows users to submit feedback on products.
* **Contact Management:** Provides a contact form for user inquiries.
* **Content Management:** Displays static content (e.g., About Us) and dynamic content (e.g., product listings).

**2. Detailed Description of Each Function**

**2.1 User Management**

This module handles user-related operations.

* **Function 1: User Registration (signup.php)**
  + **Purpose:** Allows new users to create an account.
  + **Inputs:** User details (e.g., name, email, password, phone, address) submitted via a form.
  + **Process:**
    1. Validates the input (e.g., checks if the email is unique in the users table).
    2. Hashes the password using password\_hash().
    3. Inserts a new record into the users table.
    4. Sets session variables (e.g., $\_SESSION['user\_id']) and redirects to index.php.
  + **Outputs:** A new user record in the users table and a logged-in session.
  + **Interactions:** Uses config.php for database connection and functions.php for utility functions.

 **Function 2: User Login (login.php)**

* **Purpose:** Authenticates users and starts a session.
* **Inputs:** email and password submitted via a form.
* **Process:**
  1. Validates the input (e.g., checks if the email exists in the users table).
  2. Verifies the password using password\_verify().
  3. Sets session variables (e.g., $\_SESSION['user\_id']) and redirects to index.php or account.php.
* **Outputs:** A logged-in session or an error message.
* **Interactions:** Uses config.php and navbar.php (updates user-related links).

 **Function 3: User Logout (logout.php)**

* **Purpose:** Ends the user’s session.
* **Inputs:** None.
* **Process:**
  1. Destroys the session using session\_destroy().
  2. Redirects to index.php.
* **Outputs:** A cleared session and redirection to the homepage.
* **Interactions:** Affects navbar.php (updates user-related links).

 **Function 4: Edit Profile (edit\_profile.php)**

* **Purpose:** Allows users to update their profile information.
* **Inputs:** Updated user details submitted via a form.
* **Process:**
  1. Retrieves the user’s data from the users table using $\_SESSION['user\_id'].
  2. Displays the data in a form.
  3. Updates the users table upon submission and redirects to account.php.
* **Outputs:** An updated user record.
* **Interactions:** Uses config.php and navbar.php.

 **Function 5: Change Password (change\_password.php)**

* **Purpose:** Allows users to change their password.
* **Inputs:** Current and new passwords submitted via a form.
* **Process:**
  1. Validates the current password against the users table.
  2. Hashes the new password and updates the users table.
  3. Redirects to account.php.
* **Outputs:** An updated password in the users table.
* **Interactions:** Uses config.php and navbar.php.

**2.2 Product Browsing and Filtering**

This module allows users to browse products.

* **Function 6: Display Product Listings (shop.php)**
  + **Purpose:** Displays a list of products with filtering options.
  + **Inputs:** Query parameters (e.g., gender, category).
  + **Process:**
    1. Queries the products table with filters (e.g., SELECT \* FROM products WHERE gender = 'male' AND category\_id = ?).
    2. Retrieves categories from the categories table for the "Categories" dropdown.
    3. Displays the products with links to product.php.
  + **Outputs:** A webpage showing filtered products.
  + **Interactions:** Uses navbar.php, head.php, footer.php, and js/lib/owlcarousel/ for a product carousel.
* **Function 7: Display Product Details (product.php)**
  + **Purpose:** Shows detailed information about a product.
  + **Inputs:** product\_id from the URL.
  + **Process:**
    1. Queries the products table for product details.
    2. Queries the product\_images table for images.
    3. Queries the feedback table for reviews.
    4. Displays the product details, images, reviews, and forms for adding to the cart and submitting a review.
  + **Outputs:** A webpage showing the product details.
  + **Interactions:** Uses navbar.php, head.php, footer.php, and breadcrumb.php.

**2.3 Cart Management**

This module manages the user’s shopping cart.

* **Function 8: Add to Cart (add\_to\_cart.php)**
  + **Purpose:** Adds a product to the cart.
  + **Inputs:** product\_id and quantity.
  + **Process:**
    1. If logged in, inserts/updates a record in the cart table.
    2. If not logged in, adds to $\_SESSION['cart'].
    3. Redirects to cart.php.
  + **Outputs:** An updated cart.
  + **Interactions:** Affects navbar.php (cart badge) and cart.php.
* **Function 9: Display Cart (cart.php)**
  + **Purpose:** Shows the cart contents.
  + **Inputs:** None.
  + **Process:**
    1. Retrieves cart items from the cart table or $\_SESSION['cart'].
    2. Queries the products table for item details.
    3. Displays the cart with options to update, remove, or proceed to checkout.
  + **Outputs:** A webpage showing the cart.
  + **Interactions:** Uses navbar.php, head.php, footer.php, and breadcrumb.php.
* **Function 10: Update Cart (update\_cart.php)**
  + **Purpose:** Updates the quantity of a product in the cart.
  + **Inputs:** product\_id and new quantity.
  + **Process:**
    1. Updates the cart table or $\_SESSION['cart'].
    2. Redirects to cart.php.
  + **Outputs:** An updated cart.
  + **Interactions:** Affects cart.php and navbar.php.
* **Function 11: Remove from Cart (remove\_from\_cart.php)**
  + **Purpose:** Removes a product from the cart.
  + **Inputs:** product\_id.
  + **Process:**
    1. Deletes the item from the cart table or $\_SESSION['cart'].
    2. Redirects to cart.php.
  + **Outputs:** An updated cart.
  + **Interactions:** Affects cart.php and navbar.php.

**2.4 Order Processing**

This module handles the checkout process and order management.

* **Function 12: Checkout (checkout.php)**
  + **Purpose:** Allows users to place an order.
  + **Inputs:** None (displays a form).
  + **Process:**
    1. Displays the cart contents.
    2. Shows a form for shipping and payment information.
    3. Submits the form to process\_checkout.php.
  + **Outputs:** A webpage with a checkout form.
  + **Interactions:** Uses navbar.php, head.php, footer.php, and breadcrumb.php.
* **Function 13: Process Checkout (process\_checkout.php)**
  + **Purpose:** Creates an order and processes the payment.
  + **Inputs:** Shipping and payment details.
  + **Process:**
    1. Creates a new order in the orders table.
    2. Inserts cart items into the order\_items table.
    3. Inserts a payment record into the payments table.
    4. Clears the cart.
    5. Redirects to order\_confirmation.php.
  + **Outputs:** A new order and a cleared cart.
  + **Interactions:** Affects the orders, order\_items, payments, and cart tables.
* **Function 14: Order Confirmation (order\_confirmation.php)**
  + **Purpose:** Displays a confirmation message.
  + **Inputs:** order\_id.
  + **Process:**
    1. Retrieves the order details from the orders and order\_items tables.
    2. Displays the order summary.
  + **Outputs:** A webpage showing the order confirmation.
  + **Interactions:** Uses navbar.php, head.php, footer.php, and breadcrumb.php.
* **Function 15: Order Details (order\_detail.php)**
  + **Purpose:** Shows the details of a specific order.
  + **Inputs:** order\_id.
  + **Process:**
    1. Retrieves the order details from the orders and order\_items tables.
    2. Displays the order items, total amount, and status.
  + **Outputs:** A webpage showing the order details.
  + **Interactions:** Uses navbar.php, head.php, footer.php, and breadcrumb.php.

**2.5 Feedback and Reviews**

This module allows users to submit feedback on products.

* **Function 16: Submit Review (submit\_review.php)**
  + **Purpose:** Allows users to submit a product review.
  + **Inputs:** product\_id, message, and rating.
  + **Process:**
    1. Validates the input.
    2. Inserts a record into the feedback table.
    3. Redirects to product.php.
  + **Outputs:** A new feedback record.
  + **Interactions:** Affects the feedback table and product.php.

**2.6 Contact Management**

This module provides a contact form.

* **Function 17: Submit Contact Form (contact.php)**
  + **Purpose:** Allows users to submit inquiries.
  + **Inputs:** name, email, subject, and message.
  + **Process:**
    1. Validates the input.
    2. Inserts a record into the contact table.
    3. Redirects to contact.php with a success message.
  + **Outputs:** A new contact record.
  + **Interactions:** Affects the contact table and uses navbar.php, head.php, footer.php, and breadcrumb.php.

**2.7 Content Management**

This module displays static and dynamic content.

* **Function 18: Display Homepage (index.php)**
  + **Purpose:** Displays the homepage.
  + **Inputs:** None.
  + **Process:**
    1. Queries the products table for featured products.
    2. Displays the products in a carousel.
  + **Outputs:** A webpage showing the homepage.
  + **Interactions:** Uses navbar.php, head.php, footer.php, and js/lib/owlcarousel/.
* **Function 19: Display About Us (about.php)**
  + **Purpose:** Displays static content about the company.
  + **Inputs:** None.
  + **Process:** Renders static HTML content.
  + **Outputs:** A webpage showing the "About Us" content.
  + **Interactions:** Uses navbar.php, head.php, footer.php, and breadcrumb.php.

**3. Interactions Between Functions**

* **User Management and Cart/Order Processing:** The user\_id links cart items, orders, and feedback to a specific user.
* **Product Browsing and Cart Management:** add\_to\_cart.php links shop.php and product.php with cart.php.
* **Cart Management and Order Processing:** process\_checkout.php uses cart data to create an order.
* **Feedback and Product Browsing:** submit\_review.php adds feedback displayed on product.php.
* **Navbar and All Pages:** navbar.php provides consistent navigation and updates the cart badge.

**4. Design Considerations**

* **Session Management:** Uses both session data and database storage for the cart.
* **Dynamic Content:** The "Categories" dropdown in navbar.php scales with new categories.
* **Modular Design:** The includes/ folder promotes code reuse.
* **Security:** Assumes password hashing and input sanitization.

**Conclusion**

The updated project structure is streamlined, with clear separation of concerns across folders. The **System Functions Design** section details the system’s functionalities, their implementation, and interactions, providing a comprehensive overview of the e-commerce system.

# Task Sheet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | **Start time** | **Finish time** | **Implemented by** | **Evaluation(%)** |
| Feasibility study phase |  |  |  |  |
| Prepare Document | 25/02/2025 | 27/02/2025 | Vu Hoang Nam | 20 |
| Contribute ideas | 25/02/2025 | 27/02/2025 | Nguyen Tien Ngoc Linh | 15 |
| Contribute ideas | 25/02/2025 | 27/02/2025 | Thai Dieu Phuong | 15 |
| Contribute ideas | 25/02/2025 | 27/02/2025 | Nguyen Hai Anh | 20 |
| Revise and edit the content | 25/02/2025 | 04/03/2025 | Nguyen Hai Anh | 30 |
| Requirement Analysis phase |  |  |  |  |
| Contribute ideas | 01/03/2025 | 04/03/2025 | Nguyen Tien Ngoc Linh | 20 |
| Contribute ideas | 01/03/2025 | 04/03/2025 | Vu Hoang Nam | 20 |
| Contribute ideas | 01/03/2025 | 04/03/2025 | Thai Dieu Phuong | 20 |
| Revise and edit the content | 01/03/2025 | 04/03/2025 | Nguyen Hai Anh | 30 |
| Design phase |  |  |  |  |
| Contribute ideas | 01/03/2025 | 04/03/2025 | Nguyen Tien Ngoc Linh | 20 |
| Contribute ideas | 01/03/2025 | 04/03/2025 | Thai Dieu Phuong | 20 |
| Contribute ideas | 01/03/2025 | 04/03/2025 | Vu Hoang Nam | 20 |
| Revise and edit the content | 01/03/2025 | 05/03/2025 | Nguyen Hai Anh | 40 |
| Development phase |  |  |  |  |
| Contribute ideas | 05/03/2025 | 18/03/2025 | Nguyen Tien Ngoc Linh | 25 |
| Contribute ideas | 05/03/2025 | 18/03/2025 | Thai Dieu Phuong | 25 |
| Contribute ideas | 05/03/2025 | 18/03/2025 | Vu Hoang Nam | 15 |
| Revise and edit the content | 05/03/2025 | 18/03/2025 | Nguyen Hai Anh | 35 |
| Testing phase |  |  |  |  |
| Contribute ideas | 19/03/2025 | 24/03/2025 | Nguyen Tien Ngoc Linh | 15 |
| Contribute ideas | 19/03/2025 | 24/03/2025 | Thai Dieu Phuong | 35 |
| Contribute ideas | 19/03/2025 | 24/03/2025 | Vu Hoang Nam | 35 |
| Revise and edit the content | 19/03/2025 | 24/03/2025 | Nguyen Hai Anh | 15 |
| Deployment phase |  |  |  |  |
| Contribute ideas | 25/03/2025 | 29/03/2025 | Nguyen Tien Ngoc Linh | 25 |
| Contribute ideas | 25/03/2025 | 29/03/2025 | Thai Dieu Phuong | 25 |
| Contribute ideas | 25/03/2025 | 29/03/2025 | Vu Hoang Nam | 25 |
| Revise and edit the content | 25/03/2025 | 29/03/2025 | Nguyen Hai Anh | 25 |

# Validation Checklists

**Detailed Explanation of the Task Sheet**

The Task Sheet outlines the tasks, timelines, team member responsibilities, and evaluation percentages for an e-commerce project across six phases: **Feasibility Study**, **Requirement Analysis**, **Design**, **Development**, **Testing**, and **Deployment**. The project spans from February 25, 2025, to March 29, 2025, and involves four team members: Vu Hoang Nam, Nguyen Tien Ngoc Linh, Thai Dieu Phuong, and Nguyen Hai Anh. Each phase includes tasks such as contributing ideas and revising/editing content, with specific start and finish dates and evaluation percentages reflecting each member’s contribution.

**1. Overview of the Task Sheet**

The Task Sheet is structured as a table with the following columns:

* **Task:** The specific activity or responsibility (e.g., "Prepare Document," "Contribute ideas," "Revise and edit the content").
* **Start Time:** The date when the task begins.
* **Finish Time:** The date when the task is completed.
* **Implemented By:** The team member responsible for the task.
* **Evaluation (%):** The percentage of contribution or effort by the team member for that task, reflecting their involvement or responsibility.

The project is divided into six phases, each with a set of tasks assigned to the team members. The evaluation percentages indicate the relative effort or responsibility of each member for a given task, with the total percentage for each phase typically summing to 100% (or close to it, depending on the task distribution).

**2. Phase-by-Phase Breakdown**

**2.1 Feasibility Study Phase (25/02/2025 – 04/03/2025)**

This phase involves assessing the project’s feasibility, preparing documentation, and gathering initial ideas.

* **Tasks and Timelines:**
  + **Prepare Document (25/02/2025 – 27/02/2025):**
    - **Implemented By:** Vu Hoang Nam
    - **Evaluation:** 20%
    - **Description:** Vu Hoang Nam is responsible for preparing the initial feasibility study document, which likely includes an analysis of the project’s scope, resources, timeline, and potential risks. This task is completed within three days.
  + **Contribute Ideas (25/02/2025 – 27/02/2025):**
    - **Implemented By:** Nguyen Tien Ngoc Linh (15%), Thai Dieu Phuong (15%), Nguyen Hai Anh (20%)
    - **Total Evaluation:** 15% + 15% + 20% = 50%
    - **Description:** Three team members contribute ideas for the feasibility study, brainstorming aspects like market analysis, technical feasibility, and financial considerations. Nguyen Hai Anh has a slightly higher contribution (20%) compared to Nguyen Tien Ngoc Linh and Thai Dieu Phuong (15% each).
  + **Revise and Edit the Content (25/02/2025 – 04/03/2025):**
    - **Implemented By:** Nguyen Hai Anh
    - **Evaluation:** 30%
    - **Description:** Nguyen Hai Anh revises and edits the feasibility study document prepared by Vu Hoang Nam, incorporating the ideas contributed by the team. This task extends beyond the initial preparation and idea contribution, ending on March 4, 2025, indicating a thorough review process.
* **Total Evaluation for the Phase:** 20% (Prepare Document) + 50% (Contribute Ideas) + 30% (Revise and Edit) = 100%
* **Analysis:** The Feasibility Study phase is collaborative, with all team members contributing. Vu Hoang Nam takes the lead in preparing the document, while Nguyen Hai Anh plays a significant role in both contributing ideas and revising the content, ensuring the document is polished before moving to the next phase.

**2.2 Requirement Analysis Phase (01/03/2025 – 04/03/2025)**

This phase focuses on gathering and analyzing the project’s requirements.

* **Tasks and Timelines:**
  + **Contribute Ideas (01/03/2025 – 04/03/2025):**
    - **Implemented By:** Nguyen Tien Ngoc Linh (20%), Vu Hoang Nam (20%), Thai Dieu Phuong (20%)
    - **Total Evaluation:** 20% + 20% + 20% = 60%
    - **Description:** Three team members contribute ideas for the requirement analysis, identifying the functional and non-functional requirements of the e-commerce system (e.g., user management, product browsing, cart management).
  + **Revise and Edit the Content (01/03/2025 – 04/03/2025):**
    - **Implemented By:** Nguyen Hai Anh
    - **Evaluation:** 30%
    - **Description:** Nguyen Hai Anh revises and edits the requirement analysis document, ensuring clarity and completeness.
* **Total Evaluation for the Phase:** 60% (Contribute Ideas) + 30% (Revise and Edit) = 90%
* **Analysis:** The total evaluation is 90%, which suggests that an additional 10% might be missing or that the phase’s tasks are not fully weighted to 100%. Nguyen Hai Anh continues to play a key role in revising content, while the other three members contribute equally to idea generation.

**2.3 Design Phase (01/03/2025 – 05/03/2025)**

This phase involves designing the system’s architecture, database, and user interface.

* **Tasks and Timelines:**
  + **Contribute Ideas (01/03/2025 – 04/03/2025):**
    - **Implemented By:** Nguyen Tien Ngoc Linh (20%), Thai Dieu Phuong (20%), Vu Hoang Nam (20%)
    - **Total Evaluation:** 20% + 20% + 20% = 60%
    - **Description:** The team members contribute ideas for the system design, such as the database schema (as described earlier), site map, and UI/UX design.
  + **Revise and Edit the Content (01/03/2025 – 05/03/2025):**
    - **Implemented By:** Nguyen Hai Anh
    - **Evaluation:** 40%
    - **Description:** Nguyen Hai Anh revises and edits the design documentation, ensuring the design aligns with the requirements.
* **Total Evaluation for the Phase:** 60% (Contribute Ideas) + 40% (Revise and Edit) = 100%
* **Analysis:** This phase overlaps with the Requirement Analysis phase (March 1–4), indicating parallel work. Nguyen Hai Anh has a higher responsibility (40%) in revising the design, reflecting their role as the primary editor.

**2.4 Development Phase (05/03/2025 – 18/03/2025)**

This phase involves coding the e-commerce system based on the design.

* **Tasks and Timelines:**
  + **Contribute Ideas (05/03/2025 – 18/03/2025):**
    - **Implemented By:** Nguyen Tien Ngoc Linh (25%), Thai Dieu Phuong (25%), Vu Hoang Nam (15%)
    - **Total Evaluation:** 25% + 25% + 15% = 65%
    - **Description:** The team members contribute to the development by coding different parts of the system (e.g., Nguyen Tien Ngoc Linh and Thai Dieu Phuong might focus on front-end pages like shop.php and cart.php, while Vu Hoang Nam might handle back-end logic in actions/).
  + **Revise and Edit the Content (05/03/2025 – 18/03/2025):**
    - **Implemented By:** Nguyen Hai Anh
    - **Evaluation:** 35%
    - **Description:** Nguyen Hai Anh reviews and edits the code, ensuring it meets the design specifications and coding standards.
* **Total Evaluation for the Phase:** 65% (Contribute Ideas) + 35% (Revise and Edit) = 100%
* **Analysis:** This is the longest phase (14 days), reflecting the complexity of development. Nguyen Tien Ngoc Linh and Thai Dieu Phuong have the highest contributions (25% each), while Vu Hoang Nam’s contribution is lower (15%), possibly indicating a focus on specific components. Nguyen Hai Anh’s role in revising the code (35%) ensures quality control.

**2.5 Testing Phase (19/03/2025 – 24/03/2025)**

This phase involves testing the system to identify and fix bugs.

* **Tasks and Timelines:**
  + **Contribute Ideas (19/03/2025 – 24/03/2025):**
    - **Implemented By:** Nguyen Tien Ngoc Linh (15%), Thai Dieu Phuong (35%), Vu Hoang Nam (35%)
    - **Total Evaluation:** 15% + 35% + 35% = 85%
    - **Description:** The team members test the system, identifying bugs and suggesting fixes. Thai Dieu Phuong and Vu Hoang Nam have higher contributions (35% each), possibly indicating they are responsible for more critical testing tasks (e.g., functional testing, performance testing).
  + **Revise and Edit the Content (19/03/2025 – 24/03/2025):**
    - **Implemented By:** Nguyen Hai Anh
    - **Evaluation:** 15%
    - **Description:** Nguyen Hai Anh revises the testing reports and ensures all issues are addressed.
* **Total Evaluation for the Phase:** 85% (Contribute Ideas) + 15% (Revise and Edit) = 100%
* **Analysis:** Thai Dieu Phuong and Vu Hoang Nam take the lead in testing, while Nguyen Tien Ngoc Linh has a smaller role (15%). Nguyen Hai Anh’s role in revising is minimal (15%), possibly focusing on documentation rather than active testing.

**2.6 Deployment Phase (25/03/2025 – 29/03/2025)**

This phase involves deploying the system to a production environment.

* **Tasks and Timelines:**
  + **Contribute Ideas (25/03/2025 – 29/03/2025):**
    - **Implemented By:** Nguyen Tien Ngoc Linh (25%), Thai Dieu Phuong (25%), Vu Hoang Nam (25%)
    - **Total Evaluation:** 25% + 25% + 25% = 75%
    - **Description:** The team members contribute to the deployment process, such as setting up the server, configuring the database, and deploying the code.
  + **Revise and Edit the Content (25/03/2025 – 29/03/2025):**
    - **Implemented By:** Nguyen Hai Anh
    - **Evaluation:** 25%
    - **Description:** Nguyen Hai Anh ensures the deployment is successful, revising any documentation or configurations as needed.
* **Total Evaluation for the Phase:** 75% (Contribute Ideas) + 25% (Revise and Edit) = 100%
* **Analysis:** This phase is evenly distributed, with all team members contributing equally (25% each), indicating a collaborative effort to ensure a smooth deployment

**3. Analysis and Insights**

* **Team Roles:**
  + **Nguyen Hai Anh** plays a critical role across all phases, primarily in revising and editing content. Their contribution is highest in the Design phase (40%) and Development phase (35%), indicating a focus on quality control and documentation.
  + **Vu Hoang Nam** takes the lead in the Feasibility Study phase (preparing the document) and has significant contributions in Testing (35%), suggesting expertise in both planning and quality assurance.
  + **Thai Dieu Phuong** and **Nguyen Tien Ngoc Linh** contribute consistently across all phases, with higher involvement in Development (25% each) and Testing (35% for Thai Dieu Phuong, 15% for Nguyen Tien Ngoc Linh), indicating they are likely key developers and testers.
* **Timeline Efficiency:**
  + The project spans 33 days (February 25 to March 29), with the Development phase being the longest (14 days), which is expected given the complexity of coding an e-commerce system.
  + There is overlap between the Requirement Analysis and Design phases (March 1–4), suggesting parallel work to accelerate the project timeline.
  + The Testing phase (6 days) is relatively short compared to Development, which might indicate a need for more testing time to ensure quality.
* **Evaluation Distribution:**
  + The evaluation percentages generally sum to 100% per phase, except for the Requirement Analysis phase (90%), which might indicate a missing task or team member contribution.
  + The even distribution in the Deployment phase (25% each) reflects a collaborative effort to ensure a successful launch.

The Task Sheet provides a clear overview of the e-commerce project’s timeline, tasks, and team member responsibilities. The project is well-structured, with distinct phases and collaborative efforts from all team members. The Gantt Chart visually represents the timeline, making it easy to see overlaps, task durations, and individual contributions. By downloading the Gantt Chart image, you can use it for project documentation or presentations.

# Conclusion

The BronxLuggage project marks a significant achievement in our academic journey at Aptech, encapsulating a transformative experience that has enriched our technical expertise, teamwork capabilities, and understanding of real-world software development. Developed over the course of one month, from February 25, 2025, to March 29, 2025, this e-commerce platform was designed to modernize the operations of Bronx Luggage, a specialized luggage store located in Festival Marketplace, by providing a seamless online shopping experience for customers. Under the guidance of our mentor, Mr. To Hoang Anh, and through the collaborative efforts of our team—Vu Hoang Nam, Nguyen Tien Ngoc Linh, Thai Dieu Phuong, and Nguyen Hai Anh—we successfully delivered a fully functional, user-friendly, and visually appealing system that aligns with the project’s objectives while meeting the academic standards set by Aptech.

The primary goal of BronxLuggage was to create a Single-Page Application (SPA) and responsive website that allows users to browse, filter, and purchase luggage products online, addressing the limitations of the store’s current in-person business model (Section: Problem Definition). By implementing key features such as product categorization, search and filtering, shopping cart management, user authentication, and customer feedback (Section: The Proposed System), the system enhances customer accessibility, engagement, and convenience. The project adhered to a well-defined scope, focusing on core e-commerce functionalities while excluding features like real-time inventory synchronization, AI recommendations, and multi-language support (Section: Boundaries of the System). The system was successfully deployed on a local server using XAMPP, with the potential for live hosting on a domain like www.bronxluggage.com, demonstrating its readiness for practical use.

Our development process followed a structured approach, as outlined in the Task Sheet, with six distinct phases: Feasibility Study, Requirement Analysis, Design, Development, Testing, and Deployment. The Feasibility Study phase (February 25–March 4, 2025) laid the groundwork by assessing the project’s viability, with Vu Hoang Nam leading the documentation effort and the team contributing ideas collaboratively. The Requirement Analysis and Design phases (March 1–5, 2025) defined the system’s functional and non-functional requirements, resulting in a detailed Software Requirement Specification (SRS) that included user roles, use cases, screen flows, and an Entity-Relationship Diagram (ERD) (Section: Systems Designs). The Development phase (March 5–18, 2025), the longest at 14 days, saw the team coding the system using technologies like PHP, MySQL, HTML5, CSS3, and JavaScript, with Nguyen Tien Ngoc Linh and Thai Dieu Phuong taking significant roles in front-end and back-end development, respectively. The Testing phase (March 19–24, 2025) ensured quality through rigorous validation, with Thai Dieu Phuong and Vu Hoang Nam leading the effort to achieve a 95% test case pass rate. Finally, the Deployment phase (March 25–29, 2025) culminated in a successful presentation of the system, with all team members contributing equally to the deployment process.

The BronxLuggage system meets its functional requirements by providing a seamless user experience for both Guest Users and Registered Users. Guest Users can browse products, filter by gender and category, view product details, and contact the retailer, while Registered Users can additionally manage their accounts, add products to their cart, place orders, and submit reviews (Section: Customer Requirements Specification). The system’s design, detailed in the Sitemap and System Functions Design sections, ensures intuitive navigation through pages like index.php, shop.php, cart.php, and checkout.php, with dynamic features like category filtering and a cart badge enhancing usability. Non-functional requirements, such as reliability (accurate store information), security (password hashing, SQL injection prevention), and portability (responsive design across devices), were also prioritized, ensuring a robust and accessible platform (Section: Software Requirement Specification).

Despite our achievements, the project was not without challenges. One notable difficulty was the initial unfamiliarity with certain technologies, such as Git for version control and SCSS for styling, which required additional learning. We addressed this through peer-to-peer training sessions, with Vu Hoang Nam guiding the team on PHP/MySQL and Thai Dieu Phuong assisting with SCSS (Section: Acknowledgement). Another challenge was the tight timeline, particularly during the Development phase, where coding multiple features simultaneously (e.g., add\_to\_cart.php, process\_checkout.php) demanded careful coordination. Daily check-ins via WhatsApp and Nguyen Hai Anh’s oversight ensured we stayed on track, reallocating tasks as needed to meet deadlines. Additionally, ensuring security compliance, such as implementing prepared statements to prevent SQL injection, required extra effort, but our focus on best practices resulted in a secure system with no major vulnerabilities identified during testing.

The BronxLuggage project has imparted several valuable lessons that will guide our future endeavors. First, the importance of a structured development process became evident, as breaking the project into phases allowed us to manage complexity and deliver incrementally. Second, effective collaboration and communication were key to our success; our daily check-ins and weekly reviews with Mr. To Hoang Anh ensured alignment and transparency, while tools like Google Drive and Git facilitated seamless teamwork. Third, we gained a deeper appreciation for non-functional requirements, such as security and portability, which are critical for user trust and accessibility. For instance, implementing responsive design ensured the system works on devices of all sizes, while password hashing protected user data. Finally, the project reinforced the value of perseverance and adaptability, as we navigated challenges by leveraging each other’s strengths and learning from our mistakes.

Looking to the future, BronxLuggage has significant potential for expansion. The system’s modular design and normalized database structure (Section: Systems Designs) make it extensible, allowing for the addition of features like an admin panel for product management, payment integration for real transactions, and multi-language support to cater to a broader audience. Features such as AI-based product recommendations, a wishlist, and live chat support could further enhance the user experience, while integration with the physical store’s inventory system would enable real-time stock updates. Deploying the system to a live server with an SSL certificate would ensure secure access for a global audience, increasing Bronx Luggage’s market reach and business growth. These enhancements, while beyond the current scope, are feasible given the solid foundation we’ve established, and they represent exciting opportunities for future iterations of the project.

In conclusion, the BronxLuggage project has been a profoundly rewarding experience, both academically and professionally. We successfully delivered an e-commerce platform that modernizes Bronx Luggage’s operations, providing a seamless and engaging shopping experience for customers while meeting the academic objectives set by Aptech. The challenges we faced strengthened our problem-solving skills, and the lessons we learned—about teamwork, technology, and project management—will shape our approach to future projects. We are deeply grateful for the guidance of Mr. To Hoang Anh, whose mentorship was instrumental in our success, and for the support of Aptech, which provided us with the knowledge and resources to undertake this endeavor. We also take pride in the collaborative spirit of our team, whose dedication and hard work brought this vision to life. As we move forward, we carry with us the skills, experiences, and confidence gained from this project, eager to apply them to new challenges and opportunities in our careers.