Software Design Description

for

Teipon Gadget System

Version *[1.0]*

Prepared by

MOHAMAD HAZIK HAIKAL BIN RAZAK (2024779495)
EIMAN DAMIEN BIN ROHMAN (2024779447)
WAN MUHAMMAD NAQIB BIN WAN MOHD NAZRI (2024926951)
MUHAMMAD TAUFIQ BIN SHOEID (2024542603)
MUHAMMAD LUQMAN MU`IZZ BIN ANUAR (2024974353)
MOHAMAD SYAZMIR B MOHD SAIDI (2024556469)
MUHAMMAD MAHDI BIN MOHD NORIZAL (2024947555)
SYARIFUL HUSAINI BIN SHARI FUDDIN (2024568373)

19 December 2024

Table of Contents

1 Introduction 1

1.1 Purpose 1

1.2 Definitions, Acronyms and Abbreviations 1

1.3 Intended Audience and Reading Suggestions 2

1.3.1 Suggested Sequence for Reading the Document 3

1.4 Product Scope 4

1.4.1 Included functionalities 4

1.4.2 Excluded functionalities 5

1.5 References 5

2 High Level Design 6

2.1 Design Considerations 6

2.1.1 Design Options 7

2.1.2 Assumptions 9

2.1.3 Constraints 10

2.2 System Level Desired Behavior 13

2.2.1 Customer Activity Diagram 13

2.2.2 Staff Activity Diagram 14

2.2.3 Admin Activity Diagram 15

2.3 Logical Representation of the Architecture 16

2.4 Architectural Component Overview 18

2.4.1 Software Dependencies 20

2.4.2 Third Party Component Description 21

2.5 Process Architecture 22

2.6 Deployment Architecture 22

3 Detailed Design 24

3.1 Class Diagram 24

3.1.1 Scenarios 24

3.2 Class Summary 25

3.2.1 Admin 25

3.2.2 Customer 28

3.2.3 Staff 31

3.2.4 Payment 33

4 GUI Mockups 33

5 Open Issues 44

Appendix A: Approval 45

Appendix B: Glossary 46

Appendix C: Sequence Diagram 47

Revision History

Name	Date	Reason For Changes	Version
MOHAMAD HAZIK HAIKAL BIN RAZAK	19/12/2024	SDD	1.0

1 Introduction

1.1 Purpose

The product specified in this document is the Teipon Gadget Transaction Management System (TMS), Version 1.0, designed to transition the shop's manual transaction and inventory management processes to a digital platform.

This SDD covers the core functionality of the system, including staff capabilities to create, read, update, and delete phone listings, manage inventory details, and access transaction history and customer interaction records. It also includes customer functionalities, such as searching and sorting phones based on specific criteria like brand, price range, and features, through a user-friendly interface.

The scope focuses on the foundational infrastructure to support immediate operational needs and scalability for future enhancements, such as online transactions, customer accounts, and analytics. This SDD does not include advanced subsystems like payment gateways or third-party integrations, which may be added in future iterations.

1.2 Definitions, Acronyms and Abbreviations

Acronym	Definition
TMS	Transaction Management System
SDD	Software Design Description
DBMS	Database Management System
PHP	Hypertext Preprocessor
CSS	Cascading Style Sheets
HTML	Hypertext Markup Language
JavaScript	A programming language for web development
IDE	Integrated Development Environment
Laragon	Cross-platform web server solution stack
MySQL	An open-source relational database management system
Dialogflow	A tool by Google for building conversational interfaces like chatbots.
Ngrok	A tunneling service to expose a local server to the internet.

Webhook	A way for one application to send real-time data to another application events.based on specific
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
SQL	Structured Query Language
MVC	Model-View-Controller

1.3 Intended Audience and Reading Suggestions

Stakeholders	Description
Project Managers	Project managers oversee project timelines, resource allocation, and deliverables. They analyze the product scope, constraints, and dependencies to plan effectively, manage risks, and coordinate the team for successful implementation.
Developers	Developers are responsible for designing, coding, and implementing the system’s features. They focus on understanding technical architecture, core functionalities (CRUD operations, inventory management, etc.), and ensuring scalability for future enhancements.
Marketing Staff	Marketing staff promote the system by highlighting its key features, such as search functionalities, inventory management, and user-friendly interfaces. They translate technical benefits into messages that resonate with potential users.
Testers	Testers validate the system to ensure it meets functional requirements and performs reliably. They develop test cases, identify bugs, and verify core functionalities like phone listings, inventory management, and search features.
End-Users	End-users include staff and customers who interact directly with the system. Staff manage inventory and transactions, while customers search for phones. A focus on usability and seamless navigation ensures a smooth user experience.
Documentation Writers	Documentation writers create user manuals and technical guides. They simplify complex system details into clear instructions, helping

	developers, testers, and end-users understand and use the system effectively.

1.3.1 Suggested Sequence for Reading the Document

1. **Introduction**
   i. **Purpose**: Provides an overview of the system, its goals, and the problem it addresses.
   ii. **Readers**: All reader types should start here to understand the document's scope.

2. **System Overview**
   i. **Purpose**: Offers a high-level view of the system architecture, key components, and features.
   ii. **Readers**: Project Managers, Developers, Testers, and Documentation Writers.

3. **Functional Requirements**
   i. **Purpose**: Details of the system's functionalities (CRUD operations, search features, inventory management).
   ii. **Readers**: Developers, Testers, Project Managers, and End-Users for system understanding.

4. **User Functionalities**
   i. **Purpose**: Focuses on staff and customer interactions, covering use cases and user workflows.
   ii. **Readers**: End-Users, Marketing Staff, Documentation Writers.

5. **Technical Specifications**
   i. **Purpose**: Contains platform, database, and system design details needed for development.
   ii. **Readers**: Developers, Testers, and Documentation Writers.

6. **User Interface Descriptions**
   i. **Purpose**: Explains the system's interface with mockups, screen layouts, and navigation flow.

ii. **Readers**: End-Users, Marketing Staff, Developers, and Documentation Writers.

7. **Test Scenarios**
    i. **Purpose**: Lists test cases and validation scenarios to ensure system quality.
    ii. **Readers**: Testers and Developers.

8. **Project Milestones**
    i. **Purpose**: Outlines the project timeline, implementation phases, and future plans for scalability.
    ii. **Readers**: Project Managers and Developers.

1.4 Product Scope

1.4.1 Included functionalities

1. **Inventory Management**
    - CRUD operations for phone listings (Create, Read, Update, Delete).

2. **Customer Search and Filtering**
    - Search and filter phones based on:
        i. Brand
        ii. Price
        iii. Features

3. **Transaction Management**
    - Store and access customer interaction and transaction history.

4. **User Management**
    - Admin and staff functionalities for managing user accounts and profiles

5. **User Authentication**
    - Basic login and registration processes for Admins, Staff, and Customers.

6. **Sales Reporting**
    - Generating sales reports (Admin functionality).

7. **Web-based Interface**
   - User-friendly, web-based system for accessibility and ease of use.

8. **Scalability for Future Enhancements**
   - Foundational infrastructure to support future features like analytics and online transactions.

1.4.2 Excluded functionalities

1. **Online Payment Integration**
   - Adding online banking for online phone purchases.

2. **Advanced Customer Relationship Management (CRM)**
   - Using CRM to manage customer purchases and warranty follow-ups.

3. **Third-party System Integration**
   - Syncing inventory with your supplier's system.

4. **Mobile Application Development**
   - Creating an app for customers to browse phones, accessories, and check offers.

5. **Advanced Security Protocols**
   - Multi-factor authentication.

1.5 References

- Affordable Second-Hand Smartphones, Tablets & Laptops | CompAsia MY. (2022). CompAsia E-Store. https://compasia.my/

- DirectD. (n.d.). DirectD: Retail & wholesale of mobile devices and accessories. https://www.directd.com.my/

- Halomobile. (n.d.). Halomobile: Buy, trade-in & repair used Apple devices. https://www.halomobile.com.my/

- Mobile shop Billing software | POS Software for Mobile shop. (n.d.). https://www.gofrugal.com/retail/electronics-pos/mobile-shop-billing-software.html

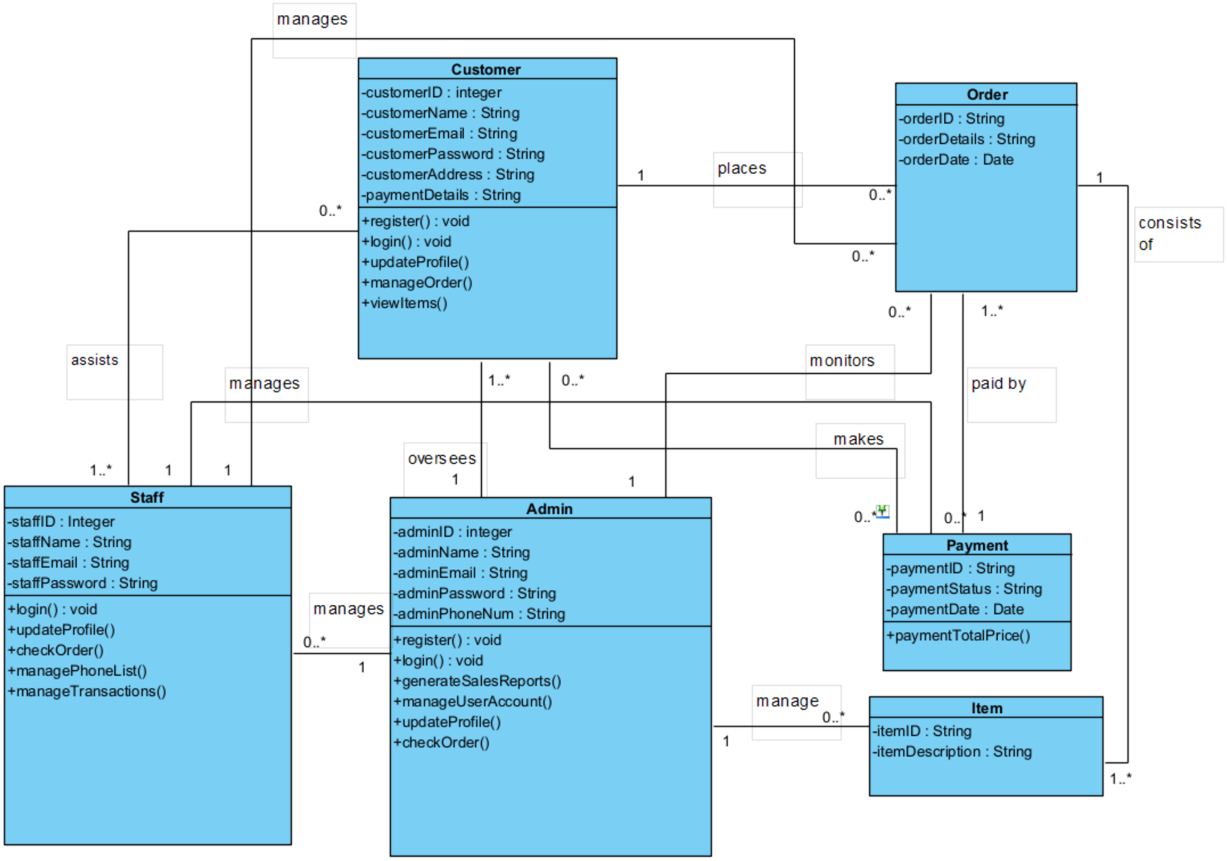
- SRO, E. (n.d.). Example Software Requirements Specification (SRS) | REQView Documentation. https://www.reqview.com/doc/iso-iec-ieee-29148-srs-example/

- [SRS] SRS.docx

- 3cat.my. (n.d.). Buy, trade-in & repair used Apple devices. https://3cat.my/

2 High Level Design

2.1 Design Considerations



2.1.1 Design Options

- High-level Design Direction

  For our project, we adopted a Model-View-Controller (MVC) design pattern without using any prebuilt frameworks. The architecture was implemented using PHP, HTML, JavaScript, CSS, and MySQL as the development environment. The MVC pattern provided a structured approach to separating application logic, data handling, and user interface components.

MVC Components	Description	Language/System
Model	Responsible for managing data and business logic	PHP and MySQL
View	Represents the user interface.	HTML, CSS, JavaScript
Controller	Handles user requests, interacts with the model, and determines the view to be displayed.	PHP

- Rationale for Chosen Design

Reasons	Description
Alignment with Project Goals	The project aimed to build a functional web application using basic web technologies. Implementing MVC design pattern provided a hands-on understanding of the architecture.
Flexibility	Full control over structure and logic, allowing us to customize the system based on project needs.
Scalability and Maintainability	The MVC design pattern made it easier to isolate bugs, modify components independently, and extend features like adding new pages or database interactions.

- Alternative Approaches Considered

Alternative Approach	Description	Rationale for Rejection
Monolithic Design Pattern	All codes reside in a single file, mixing business logic, presentation, and user input handling.	• Make the codebase unmanageable. • It is difficult to debug and maintain.

Framework-Based MVC (Laravel)	Frameworks provide a structured implementation of MVC.	• limited experience and time for teams to learn frameworks. • Added complexity and setup time.

2.1.2 Assumptions

No.	Assumptions	Description
1	Consistent Access to Technology Stack	The development team is expected to have reliable access to Laragon, which includes Apache or Nginx, PHP, JavaScript, and MySQL. Disruptions or constraints in the use of these technologies may have an impact on development schedules.
2	Availability of Skilled Development Team	The development team is assumed to be skilled in PHP, JavaScript, and MySQL, as well as expertise dealing with web-based systems based on the MVC paradigm. If further training is required or there is a dearth of experienced developers, the project timeframe may be compromised.
3	Stable Operating Environment	The system is expected to run in a reliable environment, either on Windows Server for development (with Laragon) or on a Linux-based production server. Changes in the operating system or hardware specifications may need changes to server setup or program compatibility
4	Browser Compatibility	Users are expected to access the system through modern web browsers (Chrome, Firefox, Edge, or Safari) that support JavaScript. If users use outdated or unsupported browsers, the system's functioning may be limited, and additional compatibility changes are required
5	Reliable Internet Connection	The project expects that users will have access to a reliable internet connection, particularly when performing online functions such as product searches and transactions. If network stability is disrupted, it may have an impact on the user experience
6	Database Scalability	It is assumed that MYSQL will handle the expected volume of data, including product listings, user data, and transaction records, without performance issues. If the database volume exceeds expectations, optimization or database scaling might be necessary
7	Third-Party Integration Reliability	Any third-party APIs (such as payment gateways, if they are used) are presumed to provide dependable and consistent service.

		Third-party service outages could affect how well the system works
8	Customer Data Privacy Compliance	Users are expected to handle data in accordance with privacy rules. The system might need to be adjusted to guarantee compliance if rules change or more stringent requirements are added.

2.1.3 Constraints

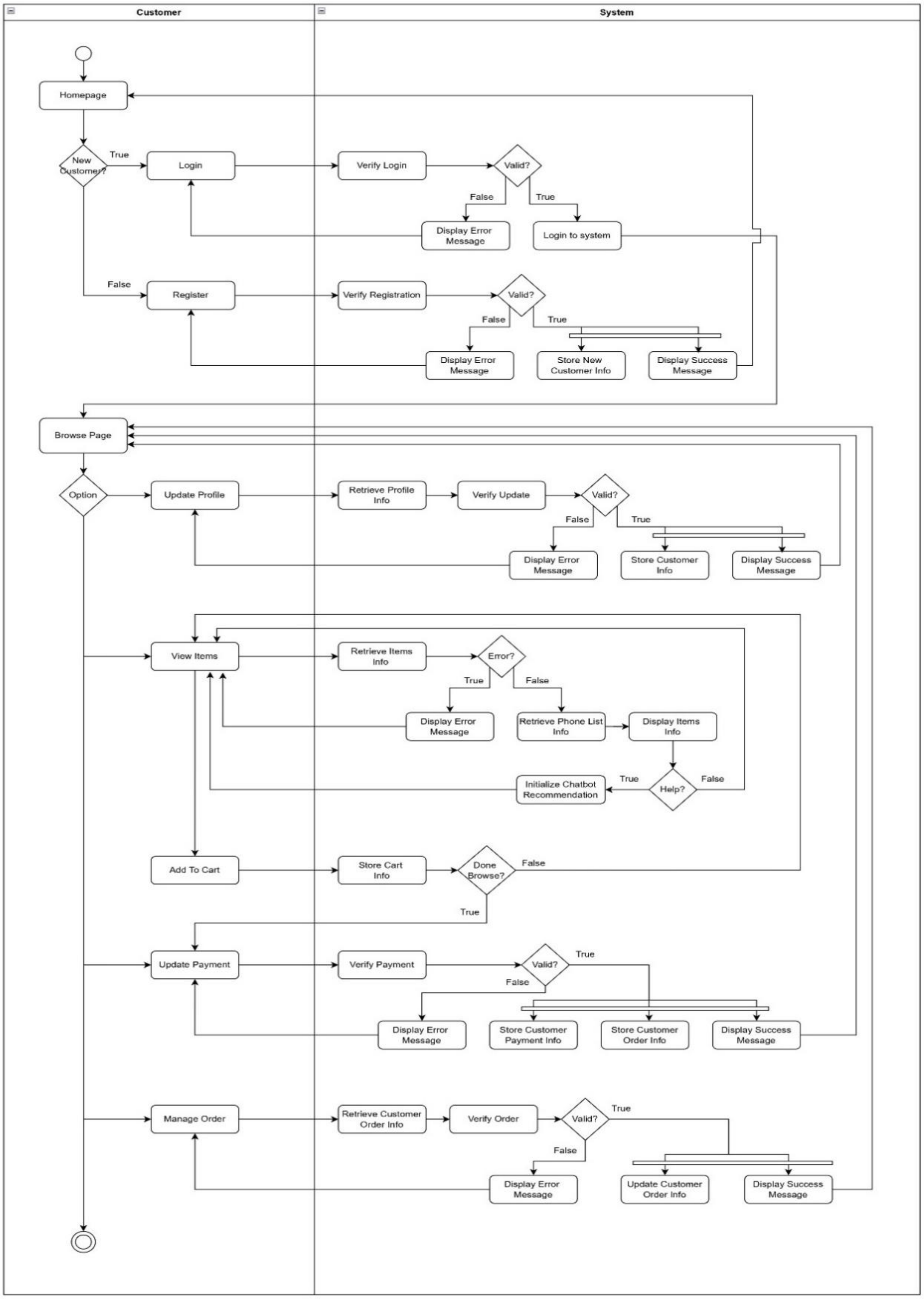
Constraint Category	Aspect	Description
Technology Stack Constraints	Server Platform	• The system must use Laragon as the local development server, requiring compatibility with PHP. • Deployment will use Apache, limiting server-side technologies.
	Programming Languages	• Backend must be developed using PHP. • Frontend scripting must use JavaScript, with preference for lightweight, widely supported libraries.
	Database	• MySQL is the designated database system, requiring SQL-compatible queries. • Limits the use of NoSQL systems like MongoDB.
	Operating System	• Development targets Windows Server (Laragon) but must also support Linux-based systems, limiting platform-specific features.
Security Constraints	User Authentication	• Enforce secure authentication using HTTPS and password hashing algorithms (e.g., bcrypt, Argon2). • Passwords must not be stored in plain text.

	Data Protection	• Follow GDPR or similar regulations for data protection. • Constraints on secure data storage, encryption, and proper PII handling
Hardware Constraints	Server Requirements	• Server hardware: minimum quad-core processor and 8 GB RAM for development. • Storage limits require database optimization for efficiency.
	Client Devices	• Must support desktops, tablets, and mobile devices. • Requires responsive design for compatibility with lower-performance hardware.
Integration and Interface Constraints	Third-Party API Integration	• APIs must be compatible with PHP and MySQL. • Require secure and widely supported APIs like REST over HTTP.
	Data Format	• Data exchanges must use JSON format, limiting the use of XML or other formats.
Development and Maintenance Constraints	Design Conventions	• Follow coding standards like PSR-12 for PHP. • Use the MVC pattern to enforce separation of concerns.
	Documentation and Code Commenting	• Code must be well-documented with comments for maintainability. • Limit obscure code practices or complex one-liners.
	Version Control	• Use a version control system like Git with strict branching and merging guidelines to prevent conflicts.

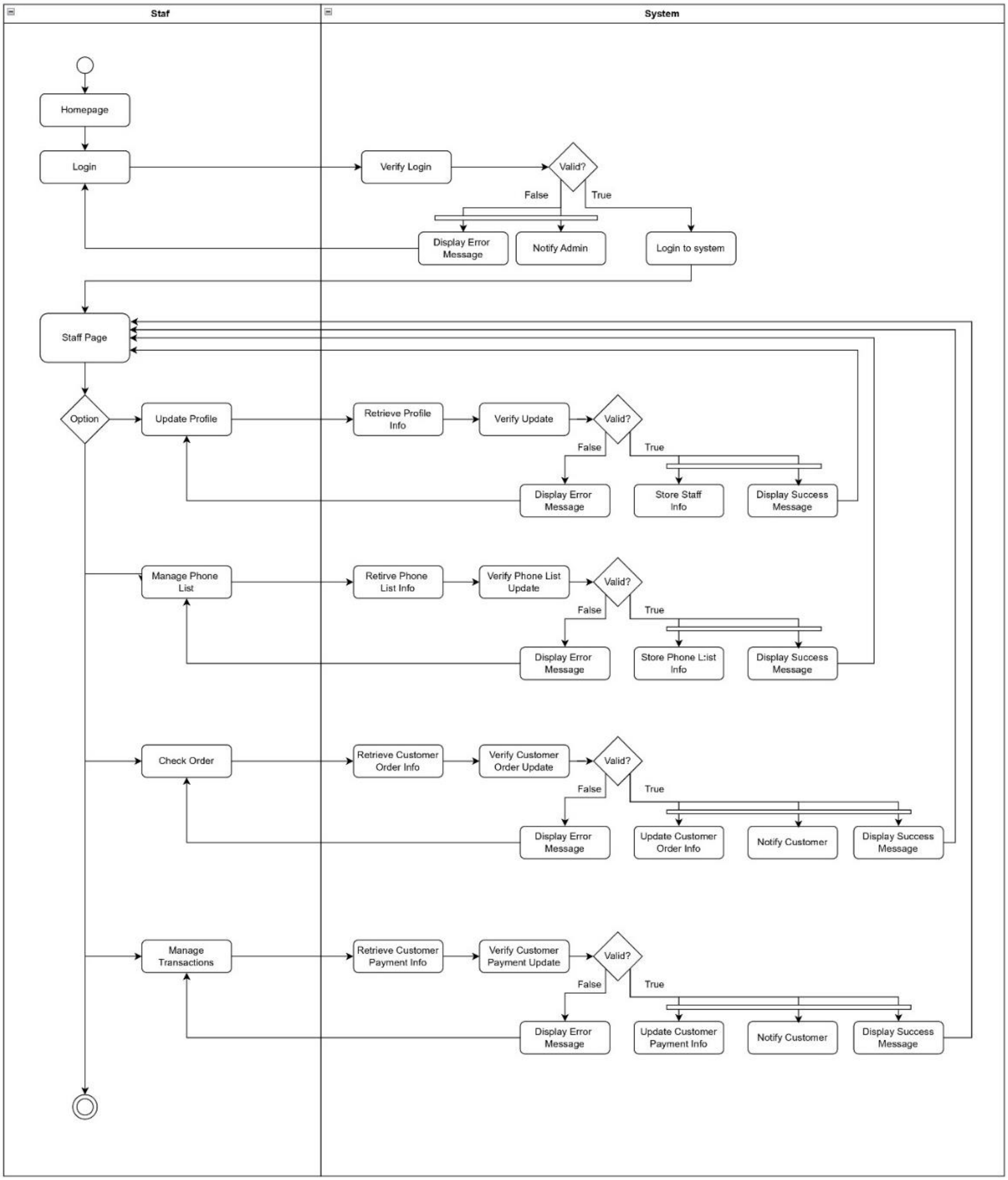
Regulatory and Compliance Constraints	Legal Compliance	• Comply with e-commerce laws and data privacy regulations. • Customer consent is required for data collection and usage.
	Audit and Logging	• Implement secure logging for inventory, sales, and user activity. • Requires non-intrusive and secure audit trails.
Communication Protocols	Network Security	• Enforce HTTPS for secure communication. • WebSocket usage is limited unless supported by Laragon.
User Interface Constraints	Browser Compatibility	• Interface must support modern browsers (Chrome, Firefox, Edge, Safari). • Experimental or browser-specific features should be avoided.
	Responsive Design	• Fully responsive design required. • Support mobile-first approach, limiting complex desktop-only features for mobile compatibility.

2.2 `System Level Desired Behaveior

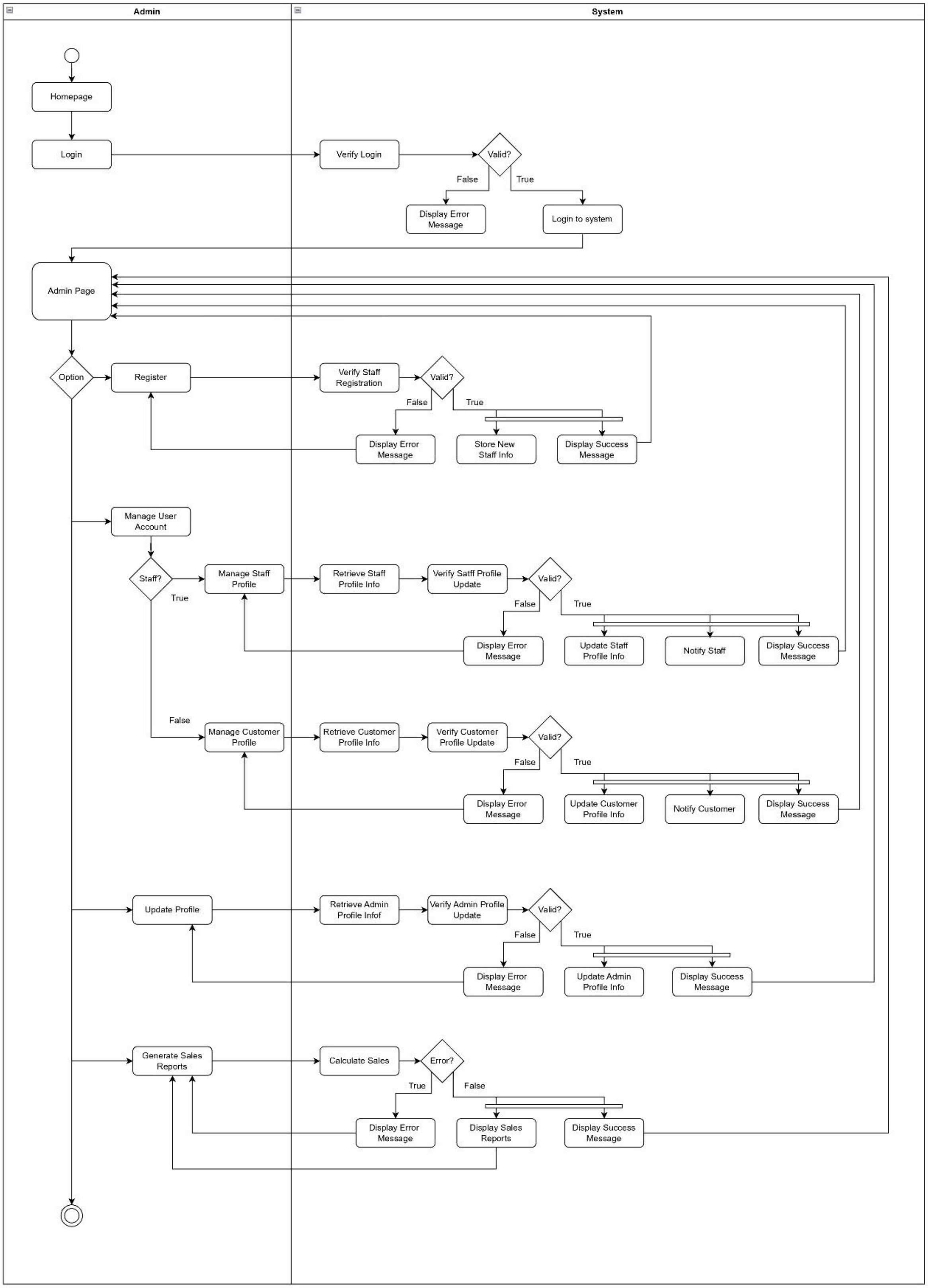
2.2.1 Customer Activity Diagram



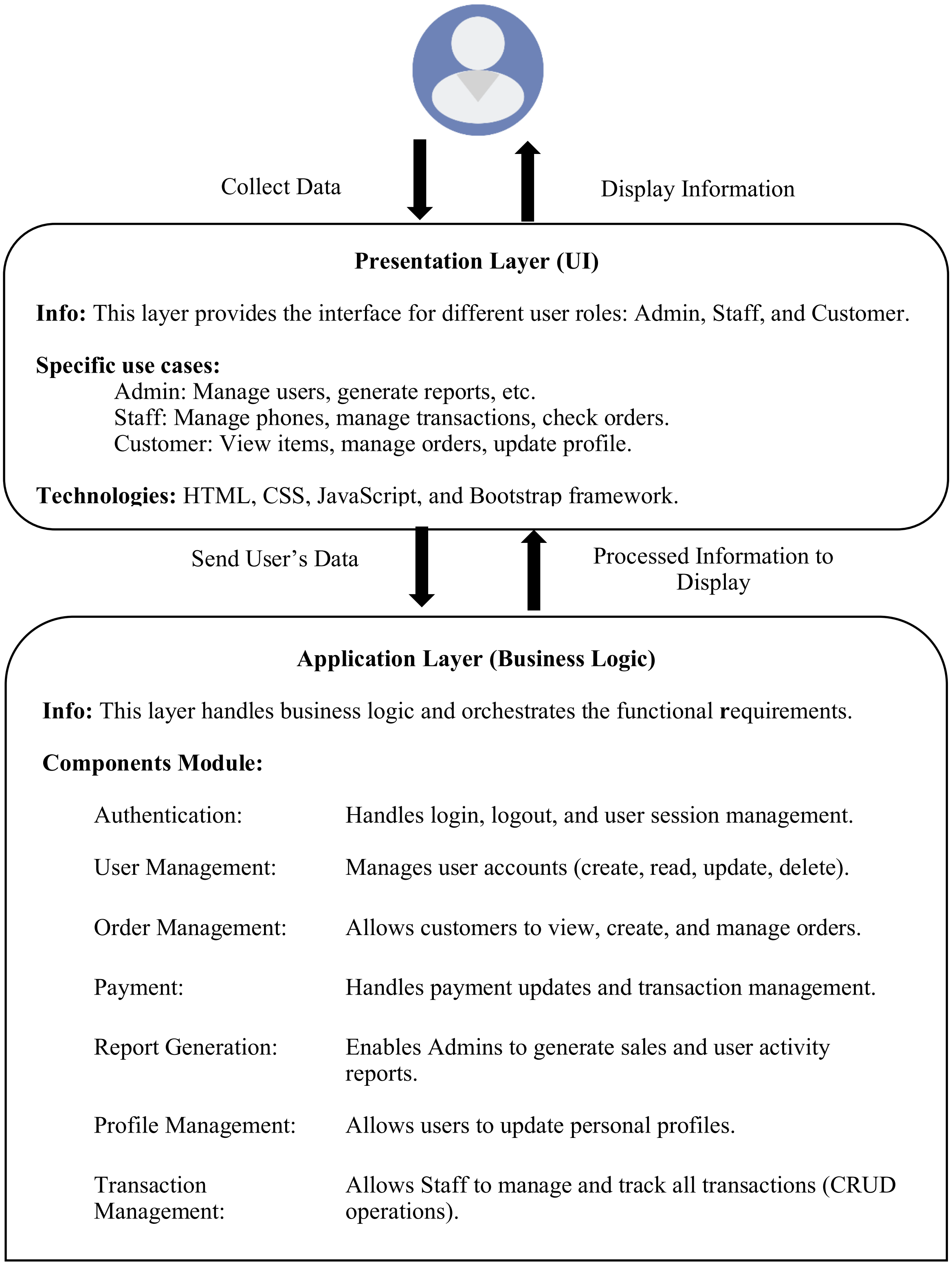
2.2.2 Staff Activity Diagram



2.2.3 Admin Activity Diagram



2.3 Logical Representation of the Architecture



Presentation Layer (UI)

Info: This layer provides the interface for different user roles: Admin, Staff, and Customer.

Specific use cases:

Admin: Manage users, generate reports, etc.
Staff: Manage phones, manage transactions, check orders.
Customer: View items, manage orders, update profile.

Technologies: HTML, CSS, JavaScript, and Bootstrap framework.

Application Layer (Business Logic)

Info: This layer handles business logic and orchestrates the functional **r**equirements.

Components Module:

Authentication: Handles login, logout, and user session management.

User Management: Manages user accounts (create, read, update, delete).

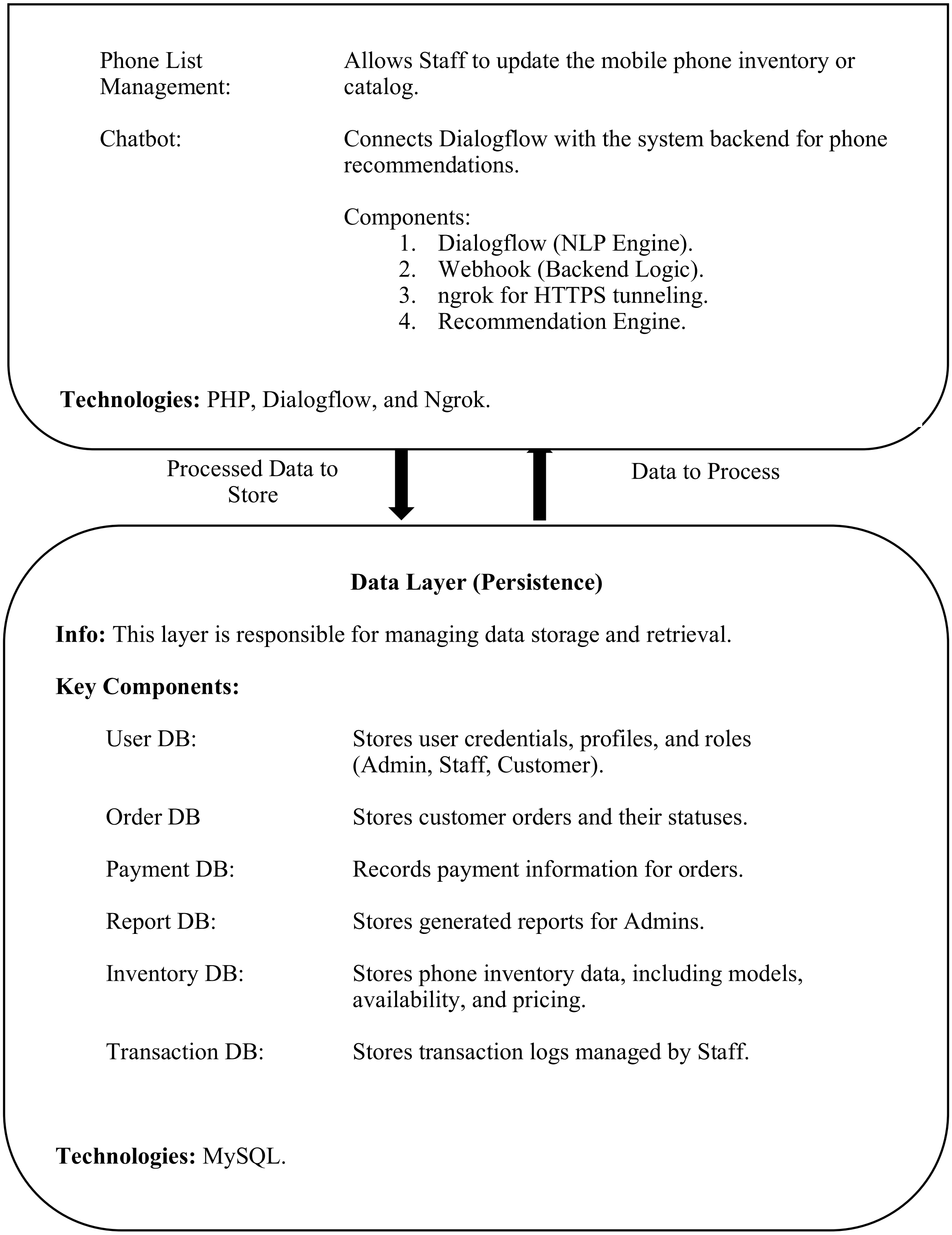
Order Management: Allows customers to view, create, and manage orders.

Payment: Handles payment updates and transaction management.

Report Generation: Enables Admins to generate sales and user activity reports.

Profile Management: Allows users to update personal profiles.

Transaction Management: Allows Staff to manage and track all transactions (CRUD operations).



Phone List Management: Allows Staff to update the mobile phone inventory or catalog.

Chatbot: Connects Dialogflow with the system backend for phone recommendations.

Components:

1. Dialogflow (NLP Engine).
2. Webhook (Backend Logic).
3. ngrok for HTTPS tunneling.
4. Recommendation Engine.

Technologies: PHP, Dialogflow, and Ngrok.

Data Layer (Persistence)

Info: This layer is responsible for managing data storage and retrieval.

Key Components:

User DB: Stores user credentials, profiles, and roles (Admin, Staff, Customer).

Order DB Stores customer orders and their statuses.

Payment DB: Records payment information for orders.

Report DB: Stores generated reports for Admins.

Inventory DB: Stores phone inventory data, including models, availability, and pricing.

Transaction DB: Stores transaction logs managed by Staff.

Technologies: MySQL.

2.4 Architectural Component Overview

1. **User Interface Component**

   Purpose: This component provides a visual and interactive interface for all user roles, including Admins, Staff, and Customers.

   Function:

   - **Admin Interface:** Enables the creation and management of user accounts, inventory updates, and generation of transaction and sales reports.
   - **Staff Interface:** Allows staff members to update inventory details, process customer orders, and manage transaction records.
   - **Customer Interface:** Provides an interactive platform where customers can:
     i. Browse and search for phones using filters like price, specifications, and availability.
     ii. Place orders and track delivery status.
     iii. Access account features such as order history and profile management.
   - The interface ensures responsiveness across devices (desktop, tablet, and mobile) using HTML, CSS, JavaScript, and Bootstrap. It enhances user experience through interactive dashboards, real-time notifications, and validation tools.

2. **Inventory Management Component**

   Purpose: To manage and monitor the availability of phones and accessories in real time, ensuring seamless transactions.

   Function:

   - Maintains detailed records of available, reserved, and sold inventory.
   - Automate inventory updates when purchases are made, or stock is replenished.
   - Provides low stock alerts and generates restocking recommendations based on historical sales trends.
   - Supports categorization of products by model, specifications, and price range, improving search and sorting capabilities for customers.
   - Integrates with the database layer to ensure data consistency and reliability

3. **Order Management Component**

   Purpose: This component handles all aspects of customer orders, from creation to delivery tracking.

   Function:

   - Processes customer purchases and updates order statuses (e.g., Pending, Confirmed, Shipped, Delivered).
   - Provides customers with real-time order tracking and delivery estimates.
   - Implements validation checks to ensure correct order details, such as customer addresses and payment status.
   - Allow staff to manage transactions, cancel orders, and handle returns or refunds efficiently.
   - Supports future integrations with third-party payment gateways (e.g., Stripe, PayPal) for secure online payments.

4. **User Authentication and Authorization Component**

   Purpose: Ensures secure access to system functionalities based on user roles.

   Function:

   - Implements role-based access control (RBAC) to restrict functionalities for Admins, Staff, and Customers.
   - Enforces secure login and registration processes using hashed passwords (bcrypt) and email verification.
   - Provides session management to track logged-in users and prevent unauthorized access.
   - Logs key user activities for audit and security monitoring purposes.

5. **Sales Reporting Component**

   Purpose: To provide analytical insights into business performance and assist with decision-making.

   Function:

   - Generates automated sales reports, displaying metrics like total sales, best-selling products, and revenue trends.
   - Allows Admins to export reports in formats like PDF and Excel for offline analysis.
   - Provides visual analytics (charts and graphs) for a clearer understanding of data, such as monthly sales performance and customer purchasing behaviors.

6. **Notification Management Component**

Purpose: Enhances user engagement and system responsiveness through real-time updates.

Function:

- Sends automated notifications for key events, such as order confirmation, low-stock alerts, and payment updates.
- Supports email-based notifications for order status changes and promotional offers.
- Integrates with future tools like SMS APIs for enhanced communication channels.

7. **Security and Data Management Component**

Purpose: Ensures the system adheres to strict security protocols and data management standards.

Function:

- Implement SSL/TLS encryption to secure data exchanges between the client and server.
- Stores sensitive customer and transaction data securely in the database using encryption.
- Comply with GDPR standards to protect customer privacy.
- Tracks critical user actions through audit trails for accountability.

2.4.1 Software Dependencies

- **Laragon**
  Used as a local development environment. Includes tools like PHP, MySQL, Apache/Nginx, and Composer for dependency management.

- **PHP**
  The main scripting language is used for server-side processing.

- **Composer**
  Dependency manager for PHP to handle library installations and updates.

- **jQuery**
  Provides advanced interactivity and simplifies JavaScript usage.

- **Bootstrap**
  Ensures responsive designs across different screen sizes.

- **JavaScript**
  Used for dynamic UI elements and real-time validation.

- **MySQL**
  Stores all persistent data, including inventory, user profiles, and transactions. Supports SQL-based querying for CRUD operations.

- **AJAX**
  Allows dynamic updates to UI without requiring full-page reloads.

- **JSON**
  Ensures efficient data exchange between the frontend and backend.

2.4.2 Third Party Component Description

1. Third-Party APIs (Optional for Future Expansion)

- **Payment Gateways (e.g., Stripe, PayPal)**:
  - **Purpose**: To handle secure online transactions.
  - **Incoming Data**: Payment confirmations, user authentication tokens.
  - **Outgoing Data**: Payment requests, transaction details.
  - **Standards Used**: REST API protocols.

2. UI Libraries and Tools

- **Bootstrap**:
  - **Purpose**: To simplify the design process and ensure consistency across pages.
  - **Role**: Helps build responsive, mobile-friendly layouts.

3. Dependency Management

- **Composer**:
  - **Purpose**: To streamline the installation and updating of PHP libraries.
  - **Role**: Reduces manual management of dependencies and ensures compatibility.

4. Data Security Enhancements

- **SSL/TLS Protocols**:
  - **Purpose**: Encrypts data exchanges to prevent interception or tampering.

- **Role**: Ensures compliance with GDPR and protects sensitive customer and transaction data.

2.5 Process Architecture

[Please refer to Figure 3.1: Class Diagram]

2.6 Deployment Architecture

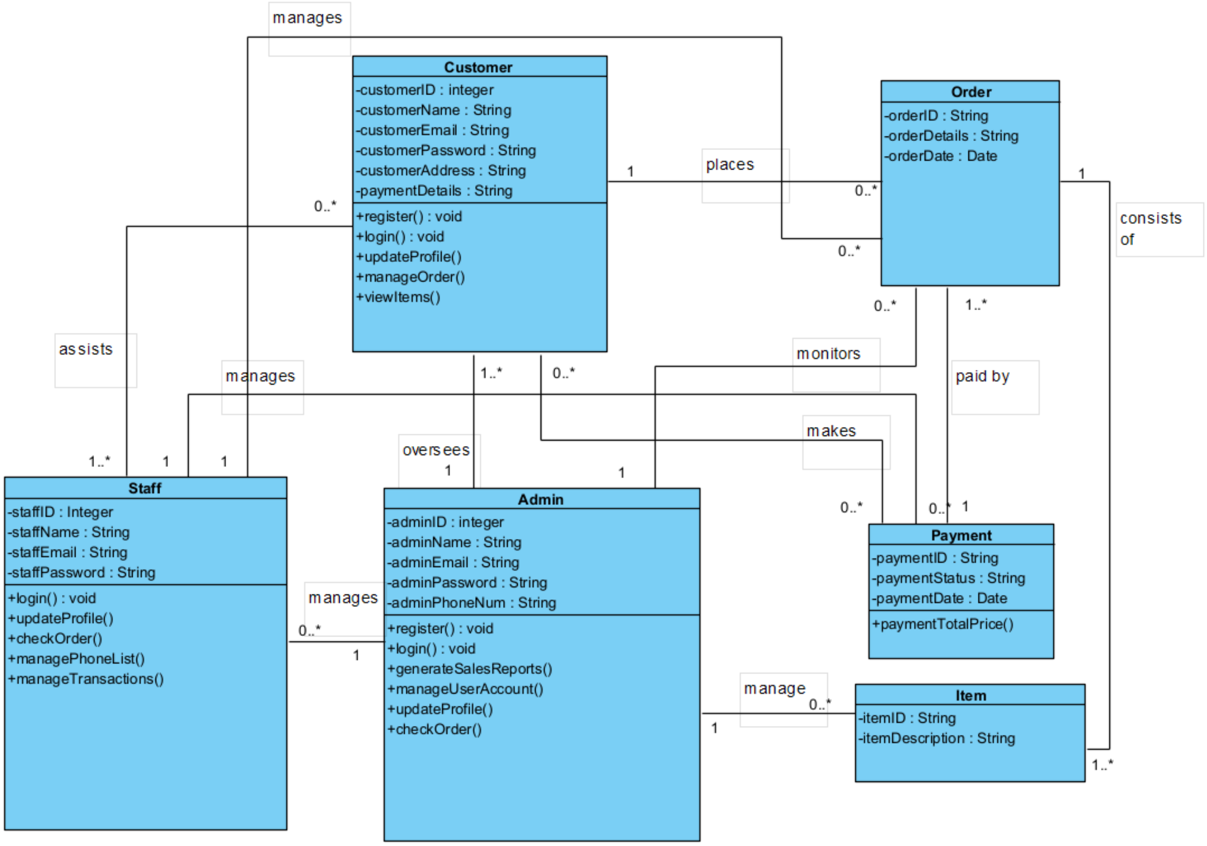
All system components are guaranteed to operate efficiently and integrate seamlessly due to the deployment design. It supports a variety of user roles and devices and is built for scalability, security, and responsiveness.

Layer	Technologies	Deployment
Presentation Layer (User Interface Component)	• HTML • CSS • JavaScript • Bootstrap	Hosted on a Content Delivery Network (CDN) to ensure low latency and high availability. Load-balanced web servers to handle high traffic.
Application Layer	• PHP	Deployed on cloud-based application servers. Microservices architecture for scalability and isolated development. RESTful API endpoints secured with OAuth2 and token-based authentication.
Database Layer (Security and Data Management Component	• Relational Database: MySQL for structured data • NoSQL for caching and	Deployed on a managed database service (e.g., AWS RDS, Azure SQL Database). Encrypted at rest using AES-256 and in transit using SSL/TLS

	real-time notifications	
Analytics and Reporting Layer (Sales Reporting Component)	• VS Code and database querying libraries for report generation	Hosted alongside the application servers with caching for performance optimization
Notification Management Layer	• Email Services: SendGrid for email notifications. • Future SMS Integration: Twilio APIs	Queue-based notification system for handling high volumes.
Security Layer (User Authentication and Authorization Component)	• Authentication: crypt for password hashing, JWT for session management. • Authorization: Role-Based Access Control (RBAC) implemented in middleware.	Security configurations enforced at both application and server levels. WAF (Web Application Firewall) and IAM policies for enhanced security

3 Detailed Design

3.1 Class Diagram



3.1.1 Scenarios

[Please refer to Appendix C: Sequence Diagram]

3.2 Class Summary

3.2.1 Admin

Attributes	*adminID, adminName, adminEmail, adminPassword, adminPhoneNum*
register()	**1. Pre-condition:** • The admin must log in first to the Teipon Gadget System. **2. Post-condition:** • The system will register the new data that is being made about staff, the Staff table will be updated and staff login to Teipon Gadget System **3. Algorithm:** • Retrieve the existing record from the Staff table. • Prepare a query to update the records from Staff table. • Execute Statement **4. Error handling/Exception processing:** • The system will request the admin to input the correct data about staff according to the accepted format.
login()	**1. Pre-condition:** • The admin must not log in first to the Teipon Gadget System. **2. Post-condition:** • The system will login the admin and display admin profile **3. Algorithm:** • Retrieve the existing record from the Admin table. • Prepare a query to check the records from Admin table and differentiate with data that is retrieved from the login page. • Execute Statement if it is equal/same.

	4. Error handling/Exception processing: • The system will request admin to input the correct data according to the accepted format and make sure the data already exists in the Admin table.
generateSales Report()	**1. Pre-condition:** • The admin must log in first to the Teipon Gadget System. **2. Post-condition:** • The system will generate sales reports in PDF format once admin requests to generate on the Admin Dashboard page. **3. Algorithm:** • Admin will click the generate button then the data will be interpreted. • Retrieved the existing record from the SalesReport table. • Display the output in PDF Format. **4. Error handling/Exception processing:** • The system generates Sales Report about the business only if admin want to do so.
manageUser Account()	**1. Pre-condition:** • The admin must log in first to the Teipon Gadget System. **2. Post-condition:** • The admin will be able to view, update and delete their customer profile information as needed. **3. Algorithm:** • Retrieved customer profile information from Customer table. • Update the data on the Customer table based on the changes made by the admin.

	- Execute statement. 4. **Error handling/Exception processing:** - The system will request the admin to input the correct data according to the accepted format.
updateProfile()	1. **Pre-condition:** - The admin must log in first to the Teipon Gadget System. 2. **Post-condition:** - The admin will be able to view and update their profile information as needed. 3. **Algorithm:** - Retrieve user profile information from the Admin table. - Prepare a query to update the records from the Admin table. - Execute Statement. 4. **Error handling/Exception processing:** - The system will request the admin to input the correct data according to the accepted format.
checkOrder()	1. **Pre-condition:** - The admin must log in first to the Teipon Gadget System. 2. **Post-condition:** - The admin will be able to check orders that have been placed by customers. 3. **Algorithm:** - Retrieve the existing record from the Order table. - Prepare a query to display the order made from the Order table. - Execute Statement. 4. **Error handling/Exception processing:**

	• If there is no order, the system will display no order statement.

3.2.2 Customer

Attributes	*customerID, customerName, customerEmail, customePassword, customerAddress, paymentDetails*
register()	1. **Pre-condition:** • The customer must not log in first in the Teipon Gadget System 2. **Post-condition:** • The system will register the new data that is being made, the Customer table will be updated and login user to Teipon Gadget System. 3. **Algorithm:** • Retrieve the existing record from the Customer table. • Prepare a query to update the records from the Customer table. • Execute Statement. 4. **Error handling/Exception processing:** • The system will request the customer to input the correct data according to the accepted format.
login()	1. **Pre-condition:** • The customer must not log in first in the Teipon Gadget System 2. **Post-condition:** • The system will login the customer and display the customer profile. 3. **Algorithm:**

	• Retrieve the existing record from the Customer table. • Prepare a query to check the records from the Customer table and differentiate them with data that is retrieved from the login page. • Execute Statement if it is equal or same. **4. Error handling/Exception processing:** • The system will request the customer to input the correct data according to the accepted format and make sure the data already exists on the Customer table.
updateProfile()	**1. Pre-condition:** • The customer must be logged in first in the Teipon Gadget System **2. Post-condition:** • Customers will be able to view and update profile information as needed. **3. Algorithm:** • Retrieve the existing records from the Customer table. • Prepare a query to update the records from the Customer table. • Execute Statement. **4. Error handling/Exception processing:** • The system will request the customer to input the correct data according to the accepted format.
manageOrder()	**1. Pre-condition:** • The customer must be logged into the Teipon Gadget System. **2. Post-condition:**

	- The customer will be able to view and update their order details as needed. **3. Algorithm:** - Retrieve order information at manage order section. - Prepare a query to update the order that has been managed by customer. - Execute Statement. **4. Error handling/Exception processing:** - The system will request the customer to double confirm if there are any changes to the order.
checkOrder()	**1. Pre-condition:** - The customer must be logged into the Teipon Gadget System. **2. Post-condition:** - Customers will be able to check or track their order made. **3. Algorithm:** - Retrieve records from Order table. - Display the information of the order according to the order made by the customer. **4. Error handling/Exception processing:** - Not applicable.

3.2.3 Staff

Attributes	*staffID, staffName, stafEmail, staffPassword*
login()	**1. Pre-condition:** • The staff must not log in first in the Teipon Gadget System **2. Post-condition:** • The system will login the staff and display staff profile. **3. Algorithm:** • Retrieve the existing record from the Staff table. • Prepare a query to check the records from the Staff table and differentiate with data that is retrieved from the login page. • Execute Statement if it is equal or same. **4. Error handling/Exception processing:** • The system will request the staff to input the correct data according to the accepted format and make sure the data already exists in the Staff table.
updateProfile()	**1. Pre-condition:** • The staff must be logged in first in the Teipon Gadget System **2. Post-condition:** • Staff will be able to view and update profile information as needed. **3. Algorithm:** • Retrieve the existing records from the Staff table. • Prepare a query to update the records from the Staff table. • Execute Statement. **4. Error handling/Exception processing:** • The system will request the staff to input the correct data according to the accepted format.

checkOrder()	1. **Pre-condition:** • The staff must be logged into the Teipon Gadget System. 2. **Post-condition:** • Staff will be able to check or track orders made by customers. 3. **Algorithm:** • Retrieve records from Order table. • Display the information of the order according to the order made by the customer. 4. **Error handling/Exception processing:** • Not applicable.
managePhone List()	1. **Pre-condition:** • The staff must be logged into the Teipon Gadget System. 2. **Post-condition:** • Staff will be able to view and update the phone list according to the number of stocks available. 3. **Algorithm:** • Retrieve the existing record from that Phone table. • Prepare a query to update the record from the Phone table. • Execute Statement. 4. **Error handling/Exception processing:** • The system will request staff to input the correct data according to the accepted format.
manage Transaction()	1. **Pre-condition:** • The staff must be logged into the Teipon Gadget System. 2. **Post-condition:** • Staff will be able to manage customer transactions either reject or approve customer transactions. 3. **Algorithm:**

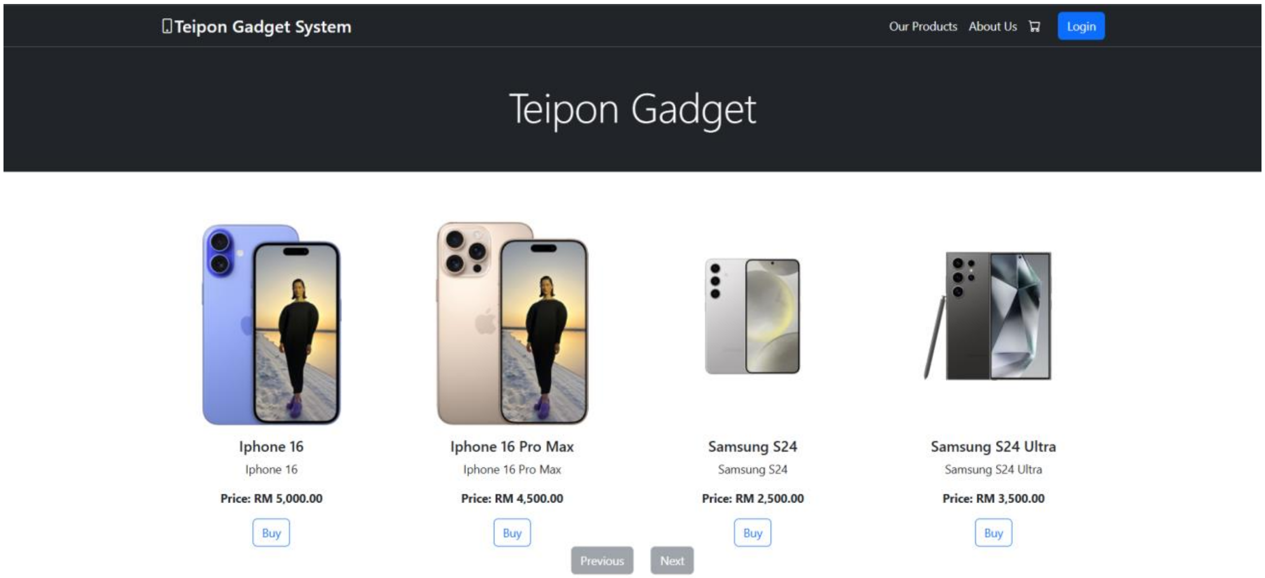
	• Retrieve records from Transaction table. • Prepare a query to update the records from the Transaction table. • Execute Statement **4. Error handling/Exception processing:** • The system will request the staff to input the correct data according to the accepted format.

3.2.4 Payment

Attributes	*paymentID, paymentStatus, paymentDate*
paymentTotal Price()	**1. Pre-condition:** • The staff and customers must not log in first to the Teipon Gadget System. **2. Post-condition:** • The system will show the total price payment made by customers, both staff and customer can see the total price. **3. Algorithm:** • Retrieve the existing record from the Payment table. • Prepare a query to display the record from the Payment table. • Execute Statement. **4. Error handling/Exception processing:** • If the customer did not make any payment, it will display no payment has been made.

4 GUI Mockups

4.1 Home Page



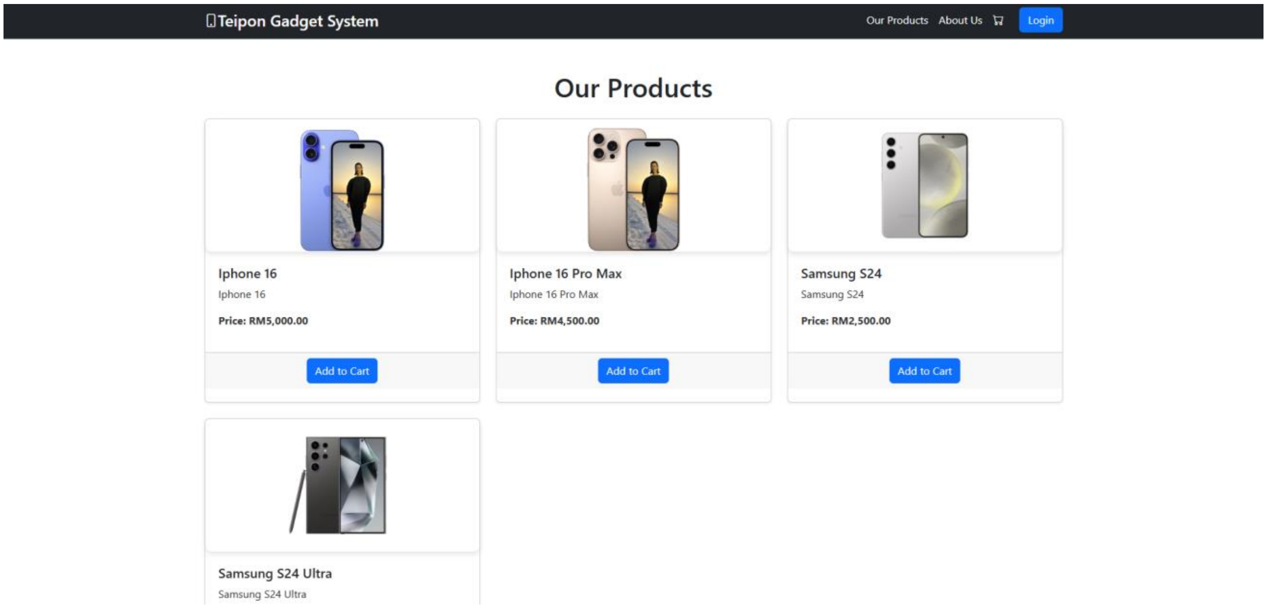
Iphone 16
Iphone 16
Price: RM 5,000.00

Iphone 16 Pro Max
Iphone 16 Pro Max
Price: RM 4,500.00

Samsung S24
Samsung S24
Price: RM 2,500.00

Samsung S24 Ultra
Samsung S24 Ultra
Price: RM 3,500.00

4.1.1 Product Listing Page



Our Products

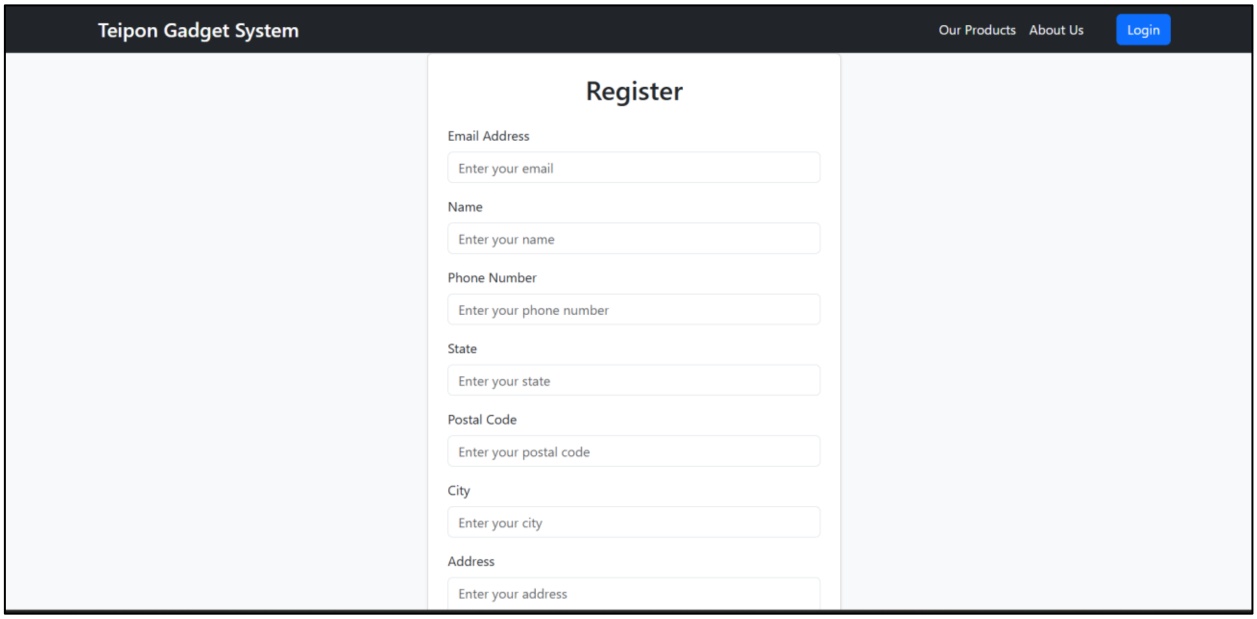
Iphone 16
Iphone 16
Price: RM5,000.00

Iphone 16 Pro Max
Iphone 16 Pro Max
Price: RM4,500.00

Samsung S24
Samsung S24
Price: RM2,500.00

Samsung S24 Ultra
Samsung S24 Ultra

4.2 Register Page



Register

Email Address

Enter your email

Name

Enter your name

Phone Number

Enter your phone number

State

Enter your state

Postal Code

Enter your postal code

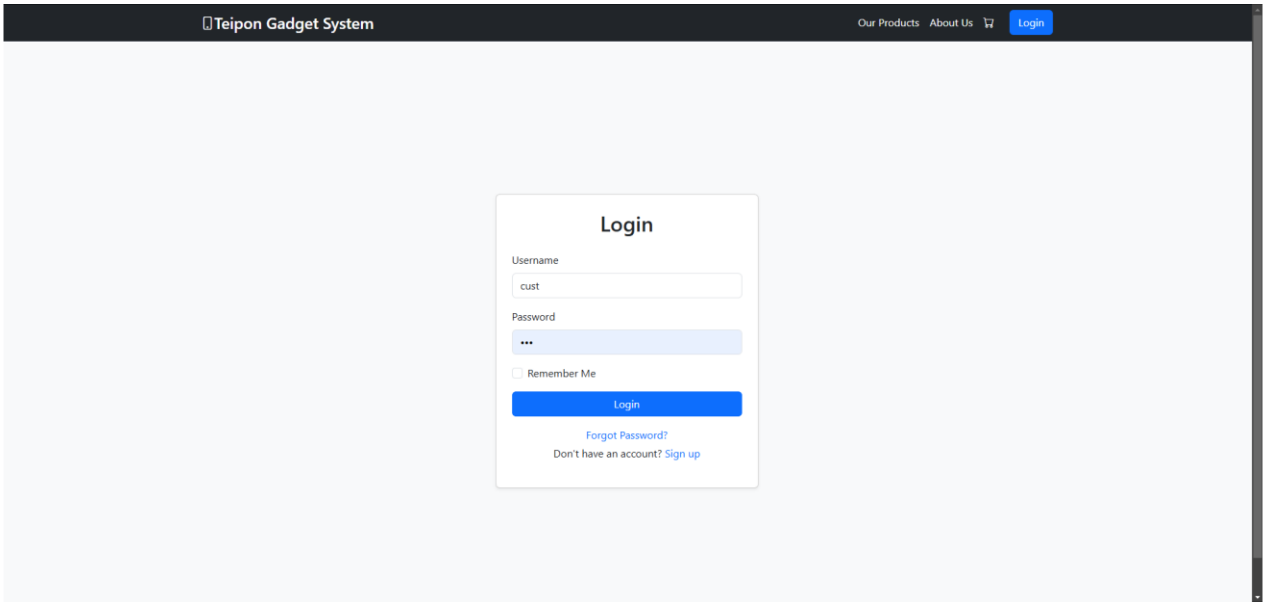
City

Enter your city

Address

Enter your address

4.2.1 Customer Login Page



Login

Username

cust

Password

•••

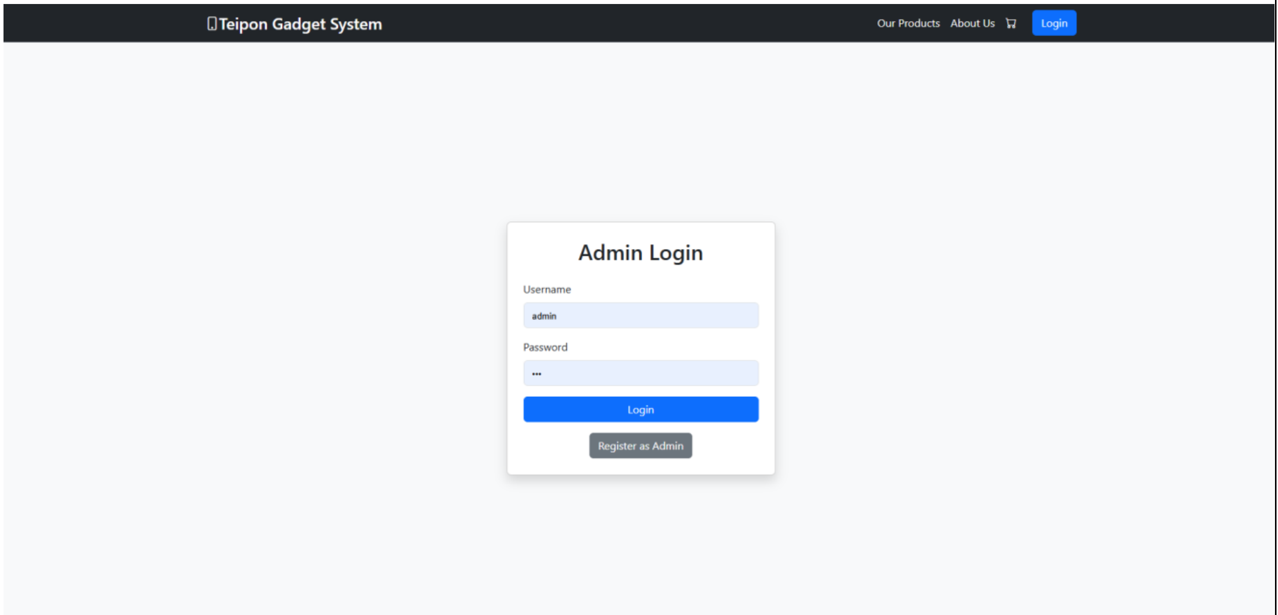
□ Remember Me

Login

Forgot Password?

Don't have an account? Sign up

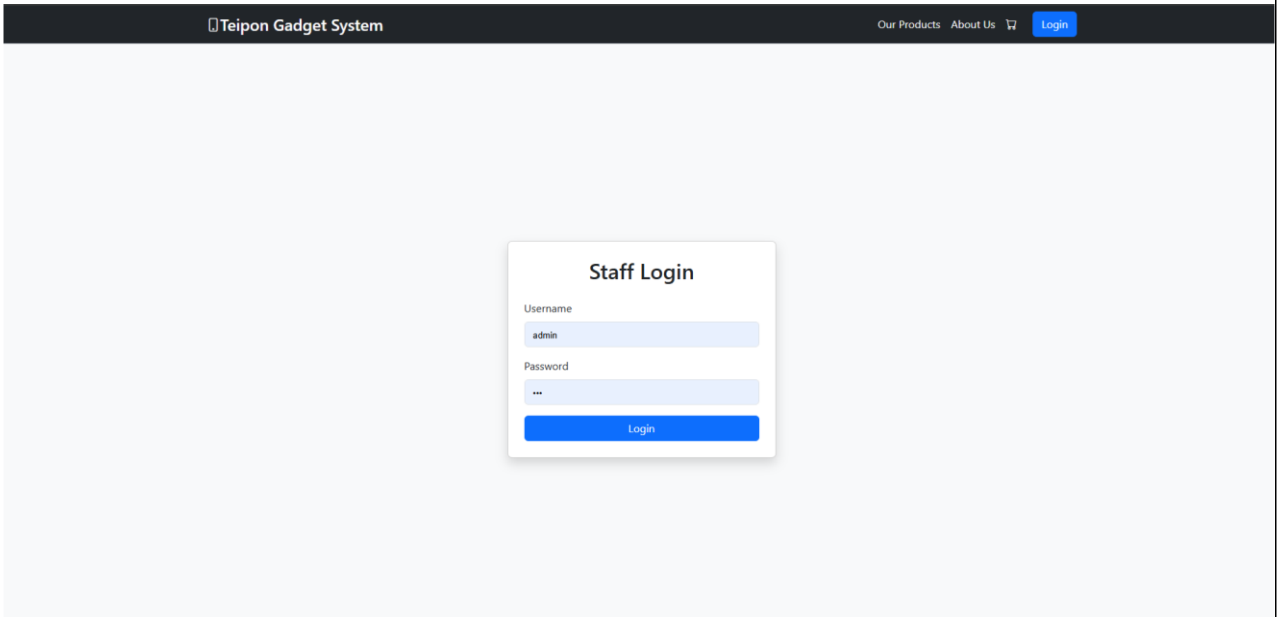
4.2.2 Admin Login Page



Username
admin

Password
•••

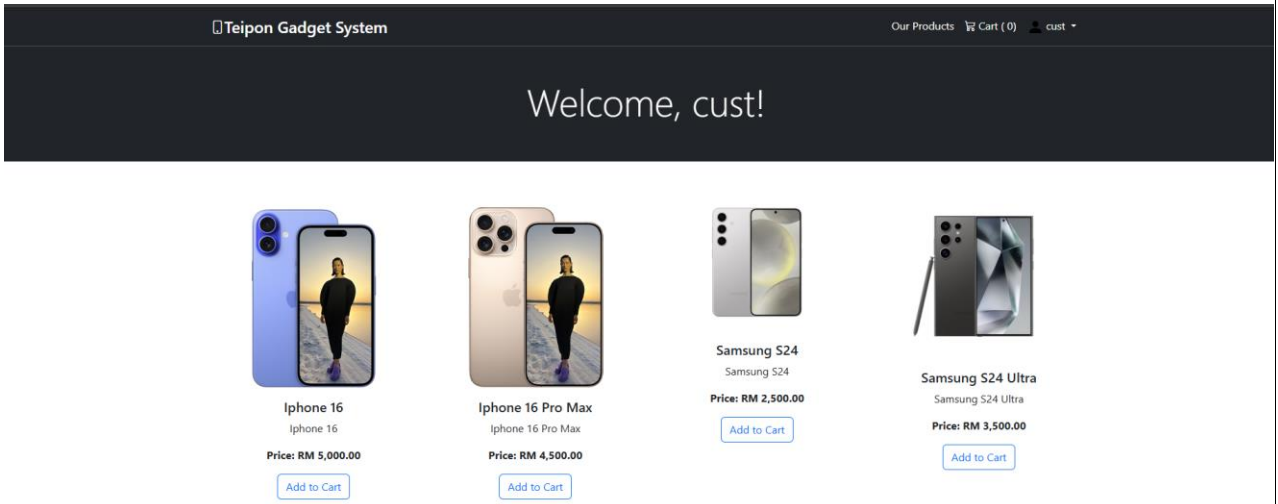
4.2.3 Staff Login Page



Username
admin

Password
•••

4.3 Customer Page



Welcome, cust!

Iphone 16

Iphone 16

Price: RM 5,000.00

Iphone 16 Pro Max

Iphone 16 Pro Max

Price: RM 4,500.00

Samsung S24

Samsung S24

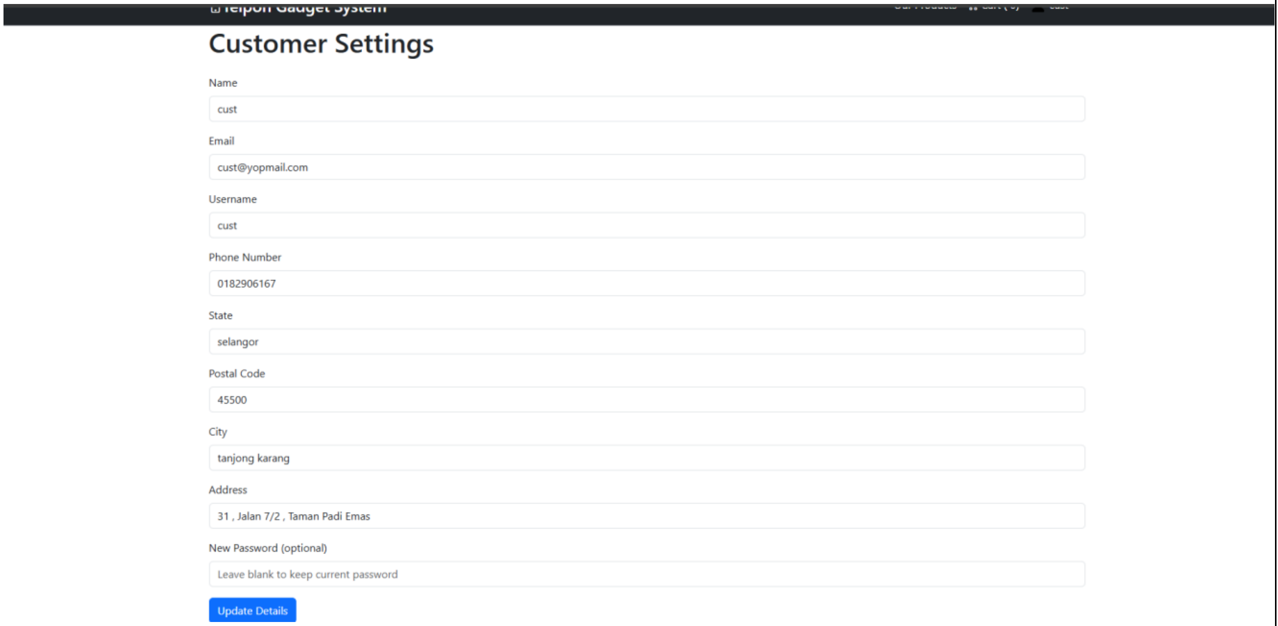
Price: RM 2,500.00

Samsung S24 Ultra

Samsung S24 Ultra

Price: RM 3,500.00

4.3.1 Customer Settings Page



Customer Settings

Name

cust

Email

cust@yopmail.com

Username

cust

Phone Number

0182906167

State

selangor

Postal Code

45500

City

tanjong karang

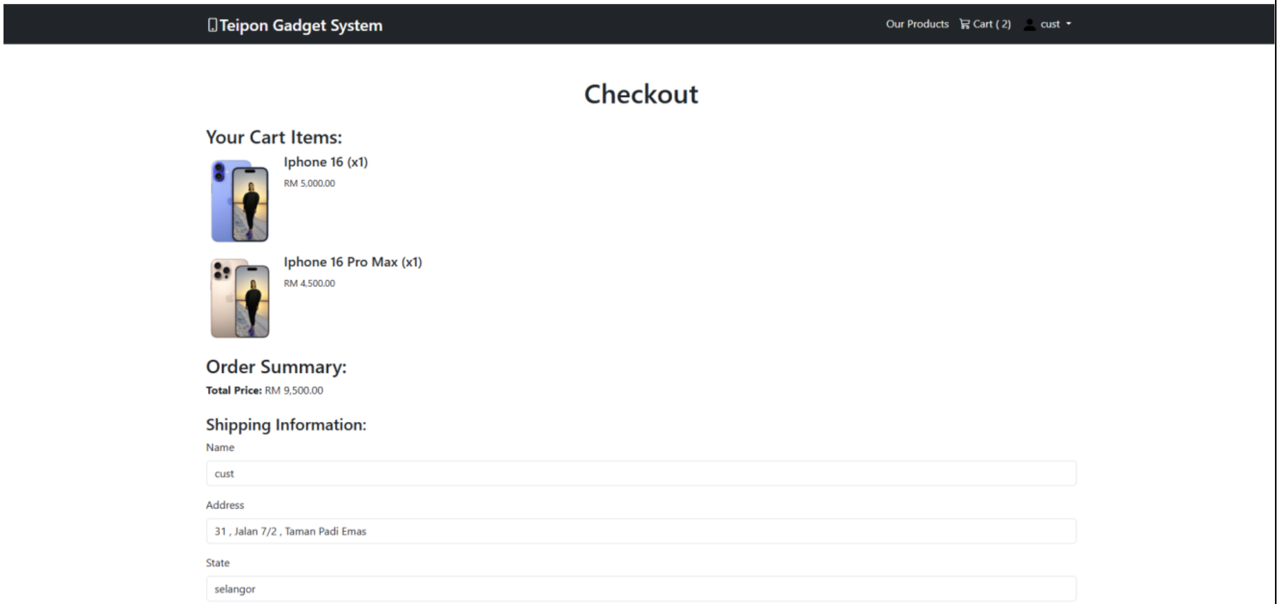
Address

31 , Jalan 7/2 , Taman Padi Emas

New Password (optional)

Leave blank to keep current password

4.3.2 Customer Checkout Page



Checkout

Your Cart Items:

Iphone 16 (x1)
RM 5,000.00

Iphone 16 Pro Max (x1)
RM 4,500.00

Order Summary:

Total Price: RM 9,500.00

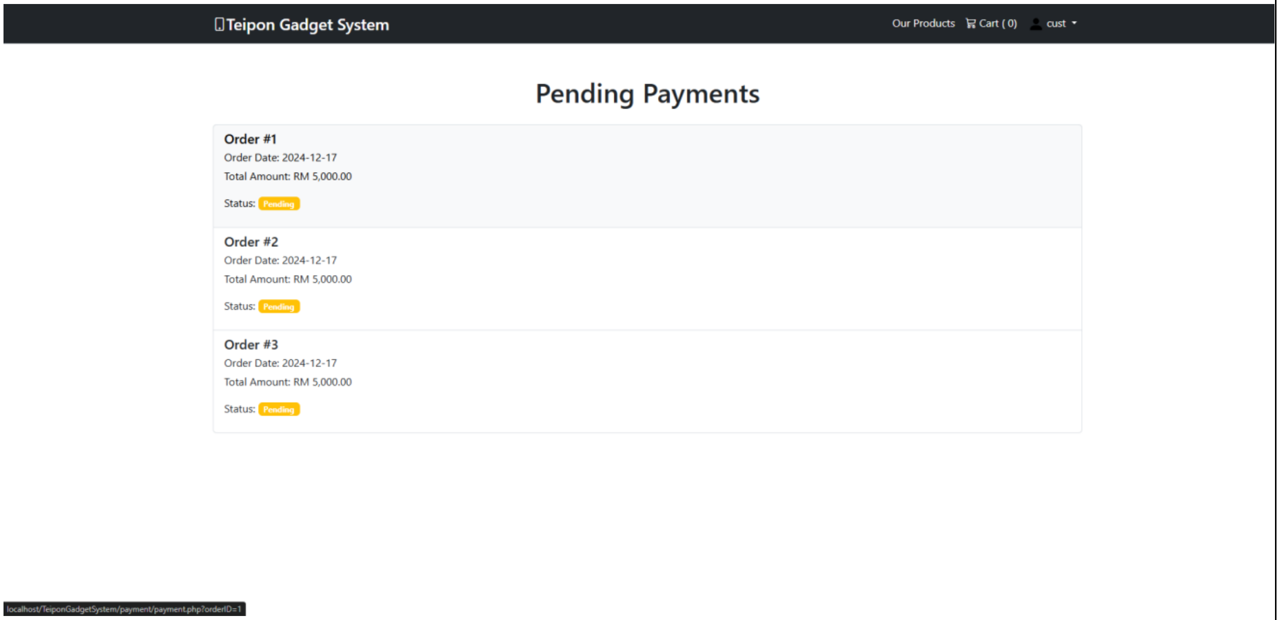
Shipping Information:

Name
cust

Address
31 , Jalan 7/2 , Taman Padi Emas

State
selangor

4.3.3 Customer Payment Page



Pending Payments

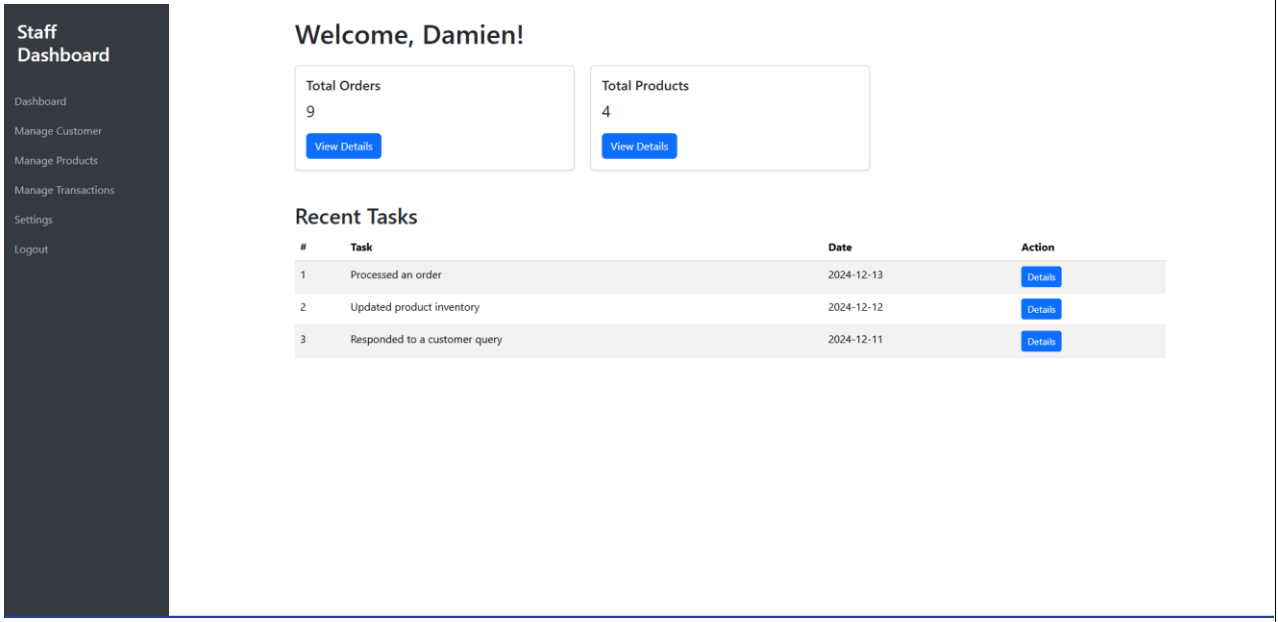
Order #1
Order Date: 2024-12-17
Total Amount: RM 5,000.00
Status: Pending

Order #2
Order Date: 2024-12-17
Total Amount: RM 5,000.00
Status: Pending

Order #3
Order Date: 2024-12-17
Total Amount: RM 5,000.00
Status: Pending

localhost/TeiponGadgetSystem/payment/payment.php?orderID=1

4.4 Staff Dashboard



Welcome, Damien!

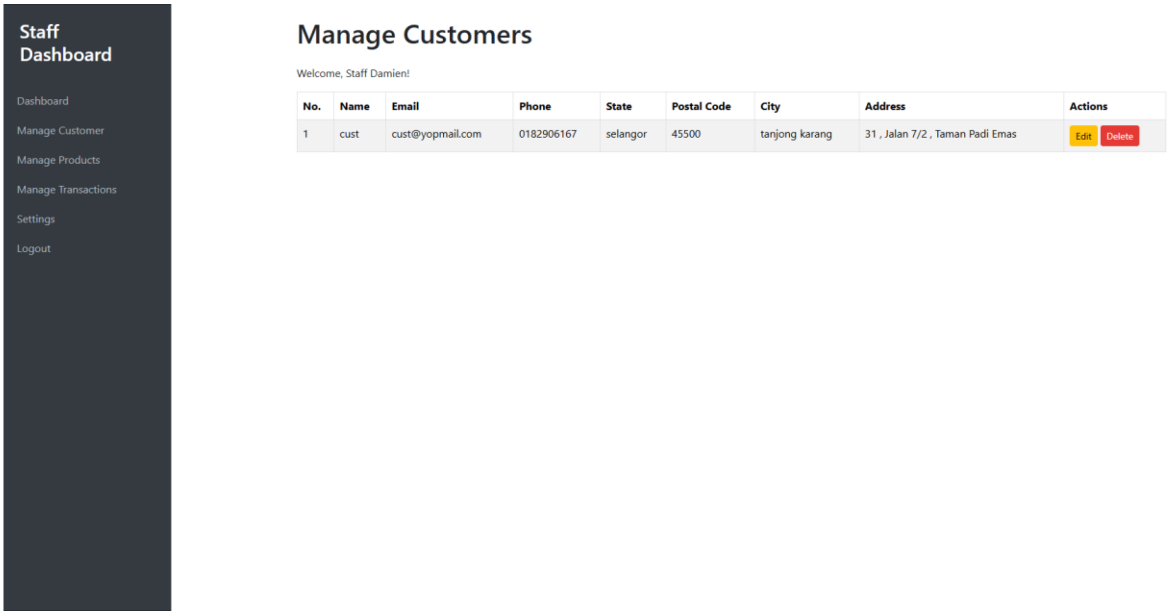
Total Orders
9
View Details

Total Products
4
View Details

Recent Tasks

#	Task	Date	Action
1	Processed an order	2024-12-13	Details
2	Updated product inventory	2024-12-12	Details
3	Responded to a customer query	2024-12-11	Details

4.4.1 Manage Customer (Staff/Admin)

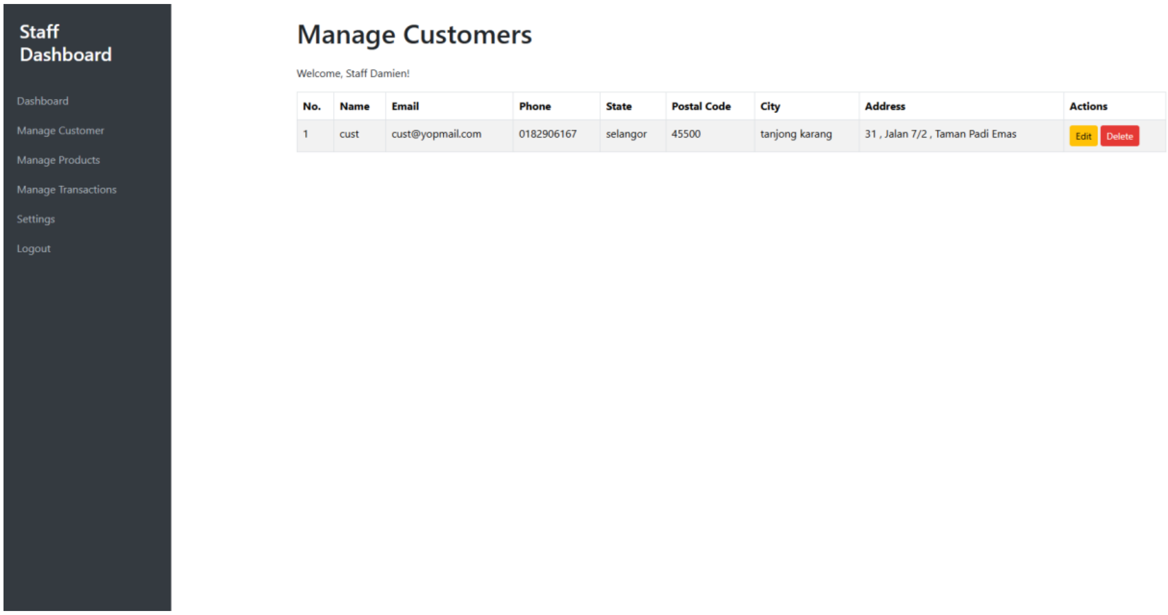


Manage Customers

Welcome, Staff Damien!

No.	Name	Email	Phone	State	Postal Code	City	Address	Actions
1	cust	cust@yopmail.com	0182906167	selangor	45500	tanjong karang	31 , Jalan 7/2 , Taman Padi Emas	Edit Delete

4.4.2 Manage Products (Staff/Admin)

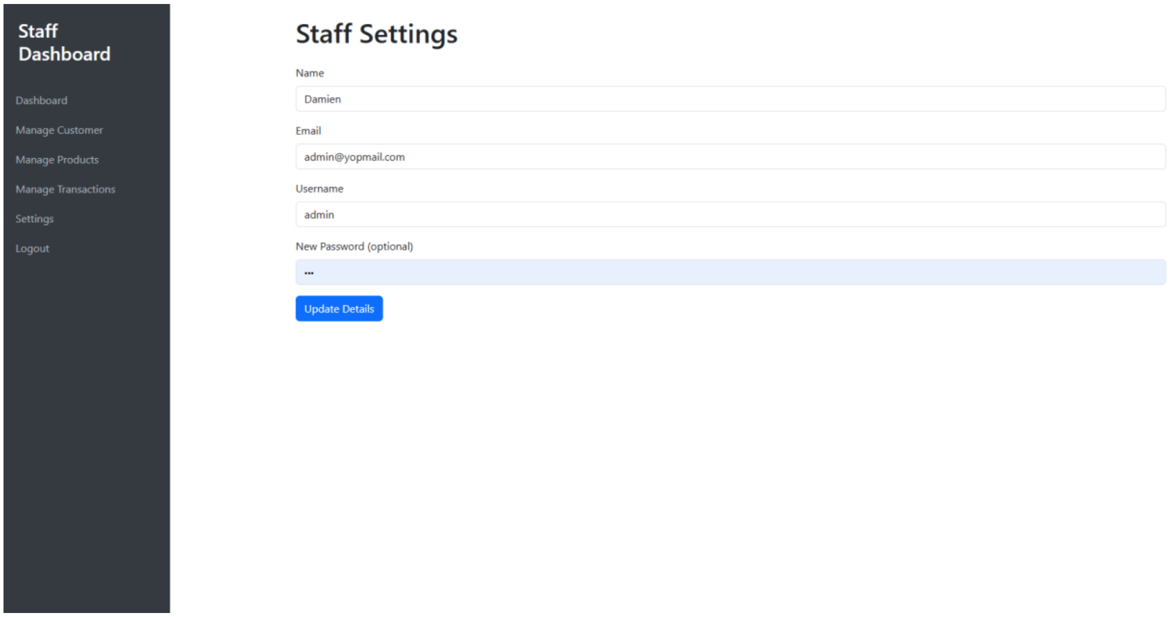


Manage Customers

Welcome, Staff Damien!

No.	Name	Email	Phone	State	Postal Code	City	Address	Actions
1	cust	cust@yopmail.com	0182906167	selangor	45500	tanjong karang	31 , Jalan 7/2 , Taman Padi Emas	Edit Delete

4.4.4 Settings (Staff/Admin)



Staff Settings

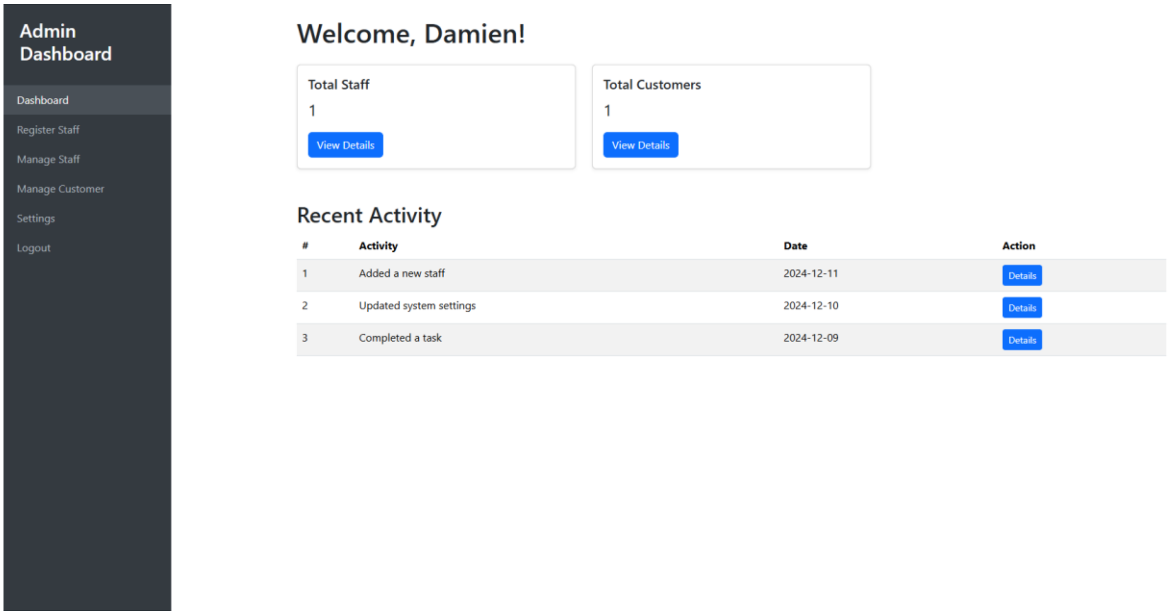
Name: Damien

Email: admin@yopmail.com

Username: admin

New Password (optional): •••

4.5 Admin Dashboard



Welcome, Damien!

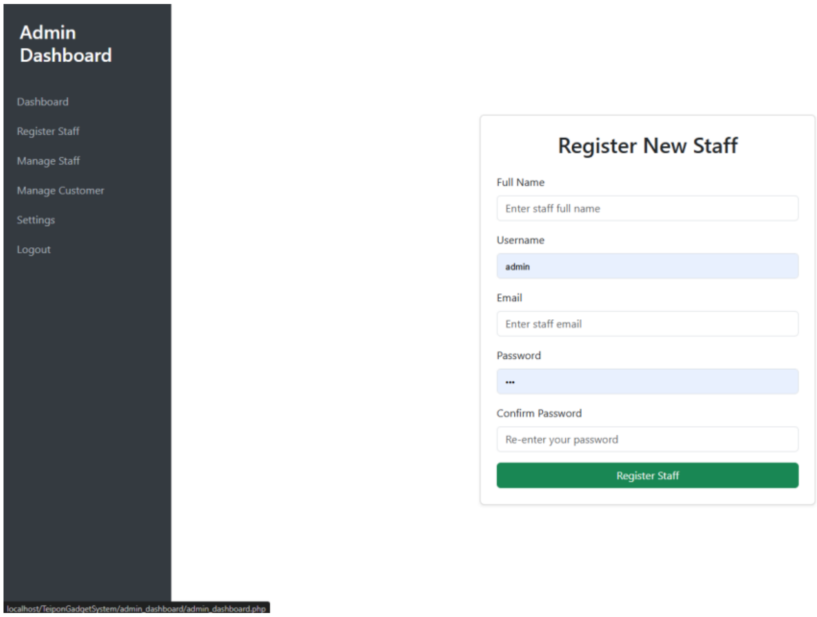
Total Staff
1

Total Customers
1

Recent Activity

#	Activity	Date	Action
1	Added a new staff	2024-12-11	Details
2	Updated system settings	2024-12-10	Details
3	Completed a task	2024-12-09	Details

4.5.1 Staff Registration (Admin)



Register New Staff

Full Name
Enter staff full name

Username
admin

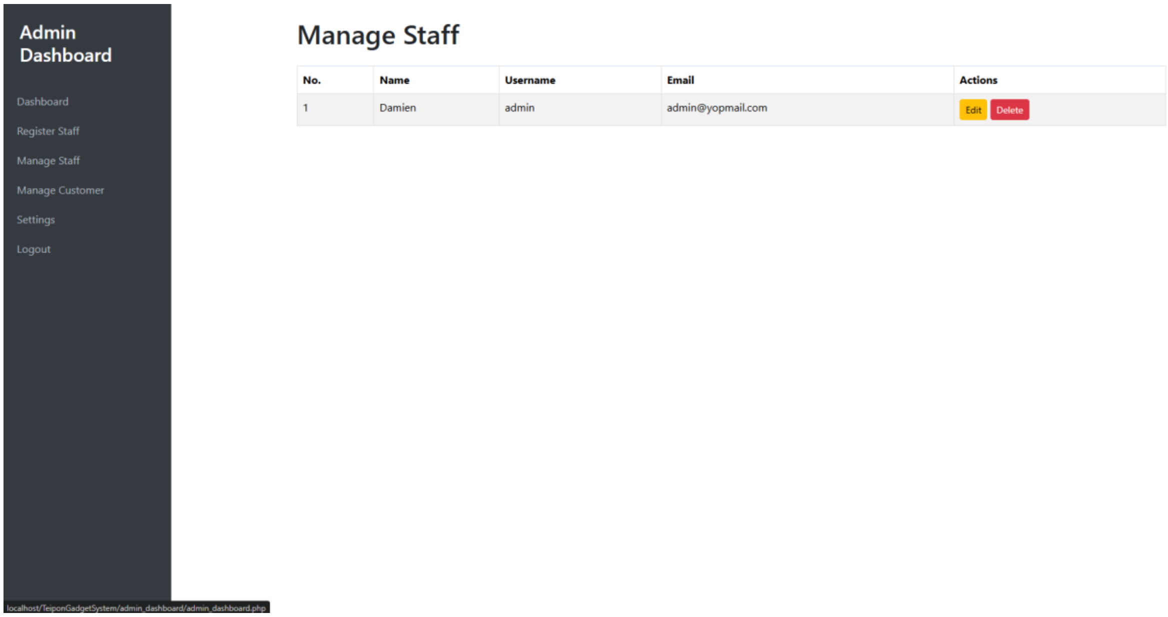
Email
Enter staff email

Password
•••

Confirm Password
Re-enter your password

localhost/TeiponGadgetSystem/admin_dashboard/admin_dashboard.php

4.5.2 Manage Staff (Admin)

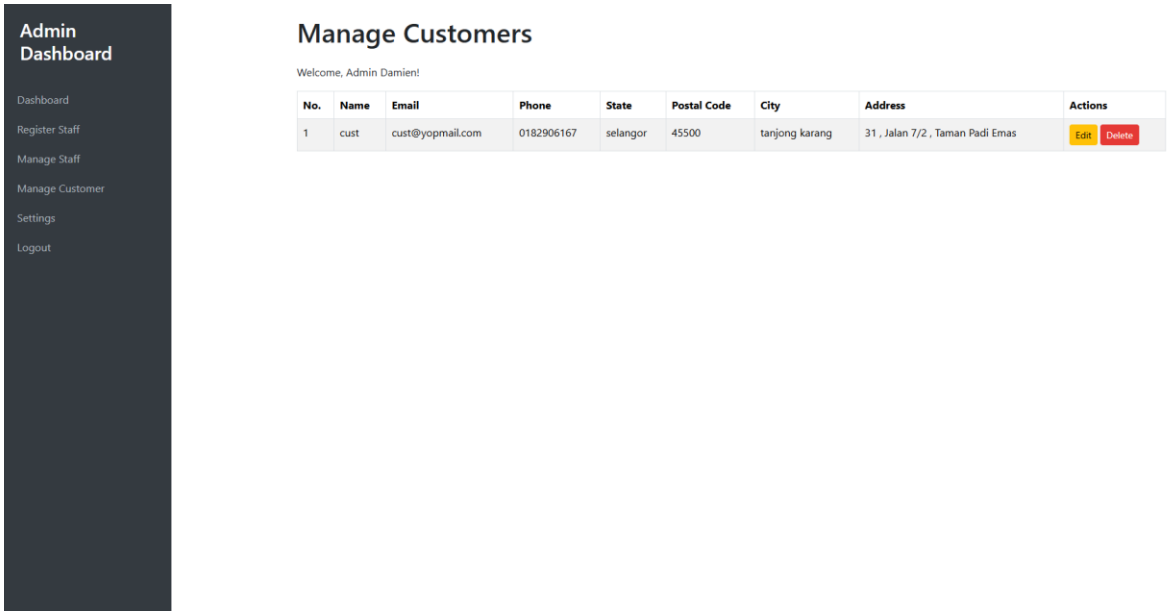


Manage Staff

No.	Name	Username	Email	Actions
1	Damien	admin	admin@yopmail.com	Edit Delete

localhost/TeiponGadgetSystem/admin_dashboard/admin_dashboard.php

4.5.3 Manage Customers (Admin)

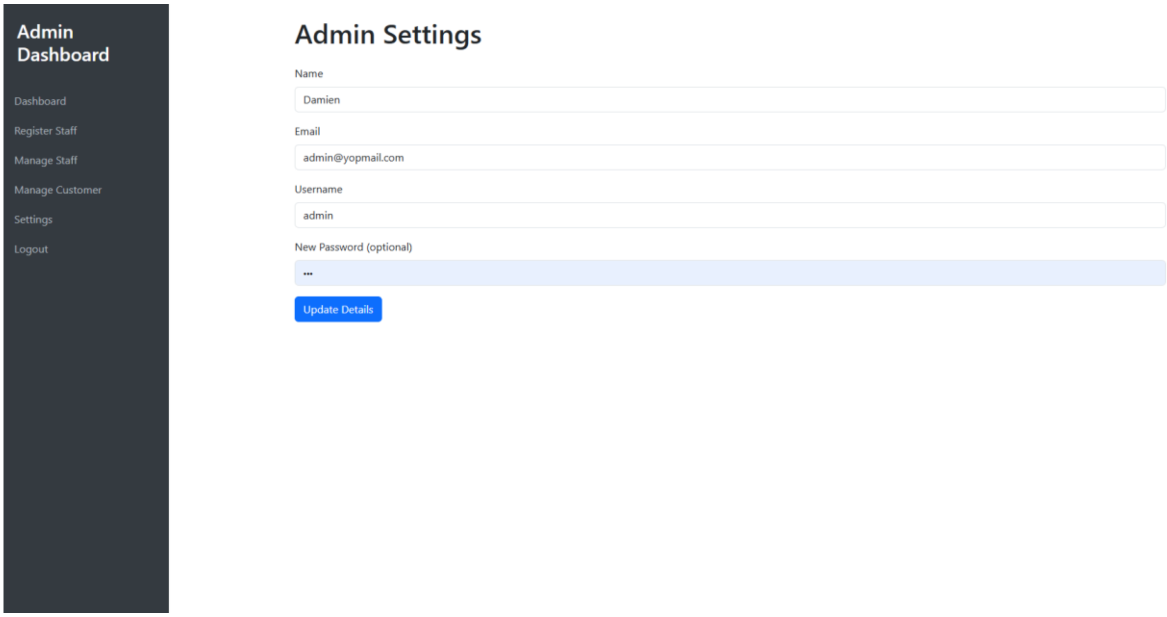


Manage Customers

Welcome, Admin Damien!

No.	Name	Email	Phone	State	Postal Code	City	Address	Actions
1	cust	cust@yopmail.com	0182906167	selangor	45500	tanjong karang	31 , Jalan 7/2 , Taman Padi Emas	Edit Delete

4.5.4 Admin Settings



Admin Settings

Name: Damien

Email: admin@yopmail.com

Username: admin

New Password (optional): •••

5 Open Issues

1. **Issue with Data Processing in Real-Time.**

   The Teipon Gadget System is facing problems with processing data in real-time, which is essential to ensure that users' data is handled smoothly and quickly, especially when dealing with many active users simultaneously. This issue can cause delays in processing immediate data that needs to be reflected in real-time, such as customer transactions, updating the status of item deliveries, and providing search results for customers looking for specific products. Such delays can lead to frustration and doubts from customers about the system's reliability in handling data processes. This problem becomes even more significant during peak hours when a high number of users are accessing the system, negatively affecting their experience. Without any improvements, this issue could result in lost sales and a decline in customer satisfaction.

2. **Issue with Security of the System from leaking and Trespassing.**

   Sensitive data like client information, transaction details, or system credentials may not be sufficiently protected by the system's current security procedures. Inadequate authentication procedures, weak encryption techniques, or holes in access control might expose the system to hostile hacking attempts, data breaches, or even unwanted access. For instance, the company's reputation might be harmed and there could be financial or legal repercussions if consumer information is disclosed. Because it jeopardizes data and erodes user confidence in the platform, this problem is crucial. For the system to function securely and safeguard users and the company, its defenses must be strengthened.

3. **Issue with User Navigation and Interaction on the Website.**

   The website's irregular page layouts and ambiguous menu structures make it difficult for many people to navigate. The system is devoid of visual aids or simple features that would help users with important activities like maintaining accounts, filtering product searches, or completing orders. This causes irritation for novice users by making it difficult for them to comprehend how to utilize the website efficiently. The problem is made worse by the lack of user-friendly design features, such adaptable layouts or easily available assistance tools, particularly for visitors who visit the website on various devices. A perplexing user interface might deter users from finishing their duties and diminish the system's allure.

Appendix A: Approval

The undersigned acknowledge they have reviewed the Teipon Gadget System Software Design Description and agree with the approach it presents. Changes to this Software Design Description will be coordinated with and approved by the undersigned or their designated representatives.

1. Signature : Date: 18/12/2024

   Name : Mohamad Hazik Haikal bin Razak

   Student Id: 2024779495

   Role : Project Manager

2. Signature : Muizz Date: 18/12/2024

   Name : Muhamaad Luqman Mu’izz bin Anuar

   Student Id: 2024974353

   Role : Software Tester

3. Signature : Date: 18/12/2024

   Name : Wan Muhammad Naqib bin Wan Mohd Nazri

   Student Id: 2024926951

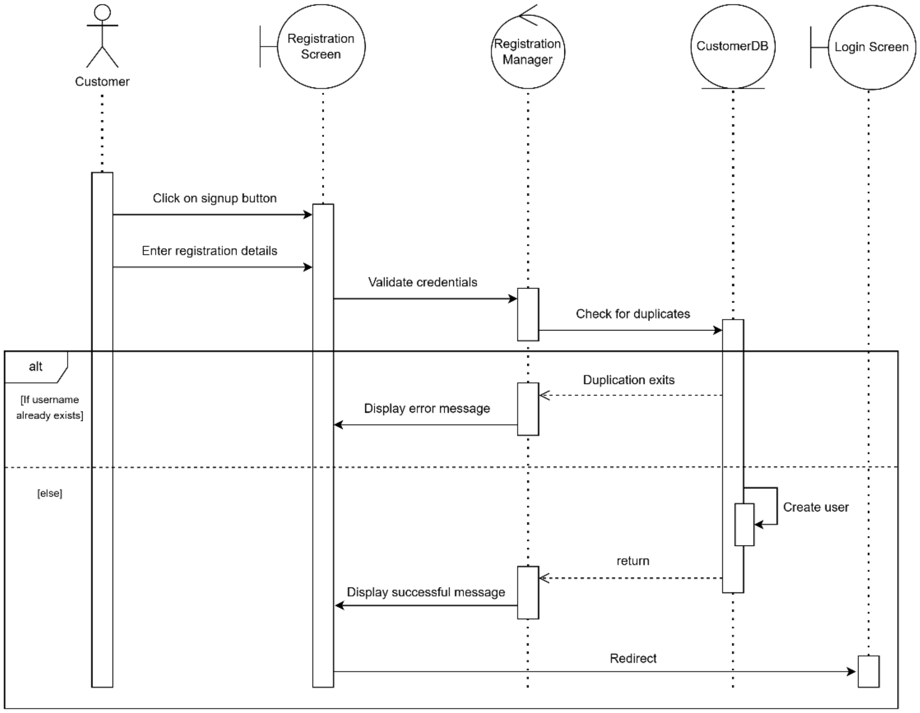
   Role : Software Requirement Analyst

Appendix B: Glossary

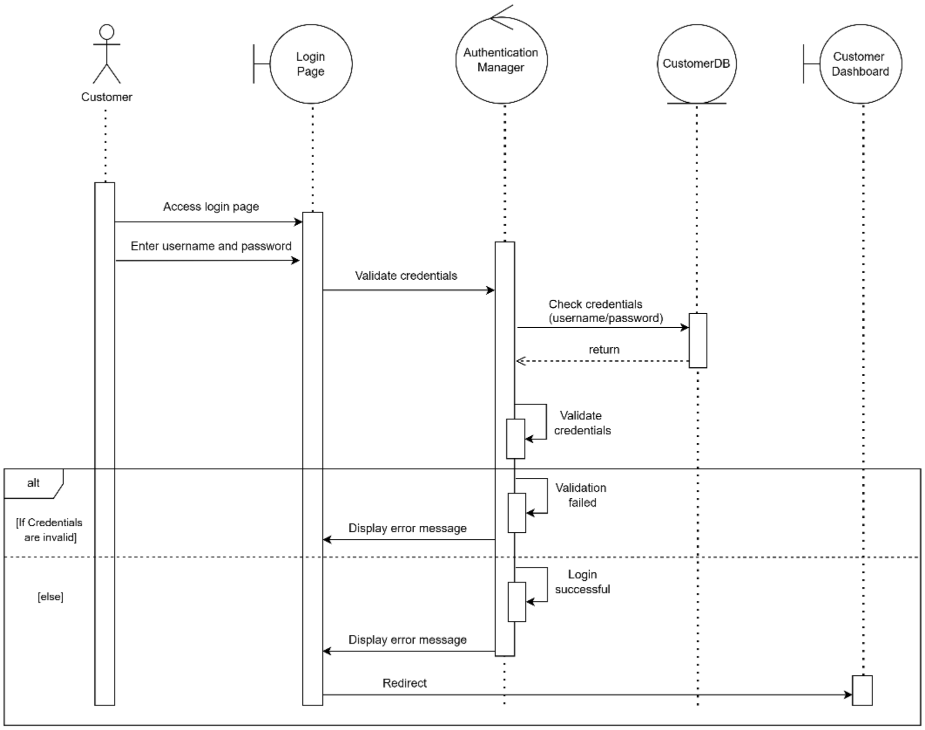
SRS	A document that describes the system's requirements and specifications.
DBMS	Software that helps manage and organize databases.
PHP	A scripting language used for creating dynamic web pages.
CSS	A style sheet language for designing how web pages look.
HTML	The standard markup language for creating web pages.
JavaScript	A programming language to make web pages interactive.
IDE	Software used by developers to write, test, and debug code.
Laragon	A tool for setting up a local web server environment.
MySQL	A system for managing relational databases.
Dialogflow	A tool by Google for building conversational interfaces like chatbots.
Ngrok	A tunneling service to expose a local server to the internet.
Webhook	A way for one application to send real-time data to another application events.based on specific
HTTP	A protocol is used to transfer data over the web.
HTTPS	A secure version of HTTP with data encryption.
SQL	A language used to manage and query databases.
MVC	Model-View-Controller

Appendix C: Sequence Diagram

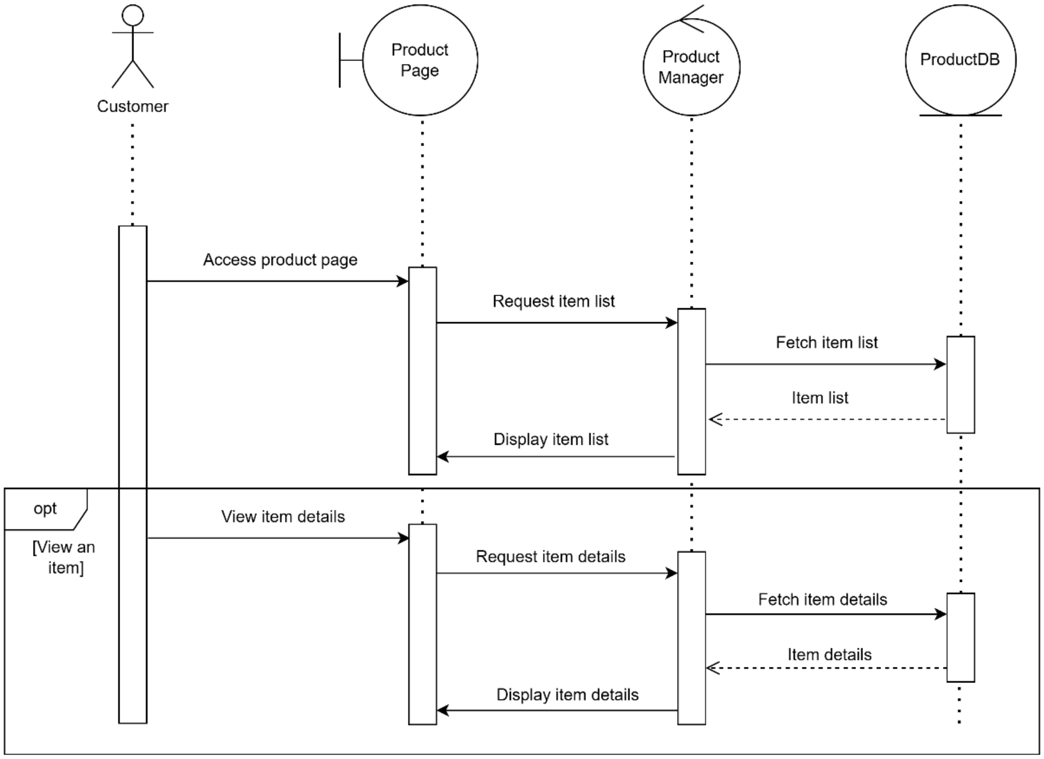
Use case 1: Customer Registration



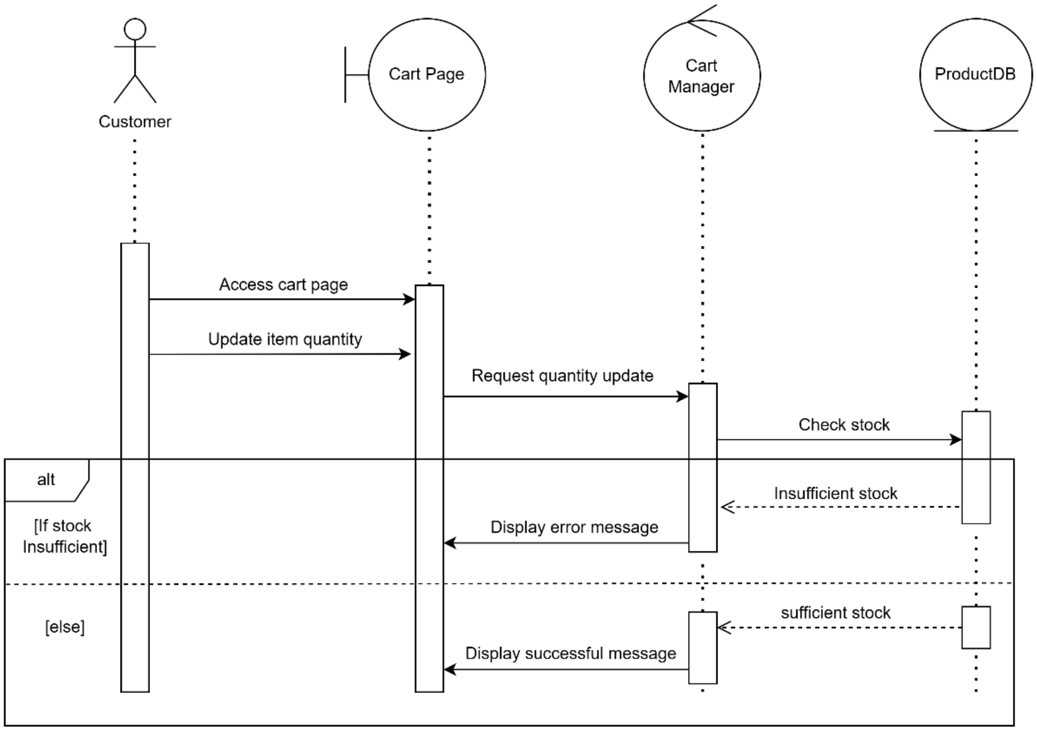
Use case 2: Customer Login



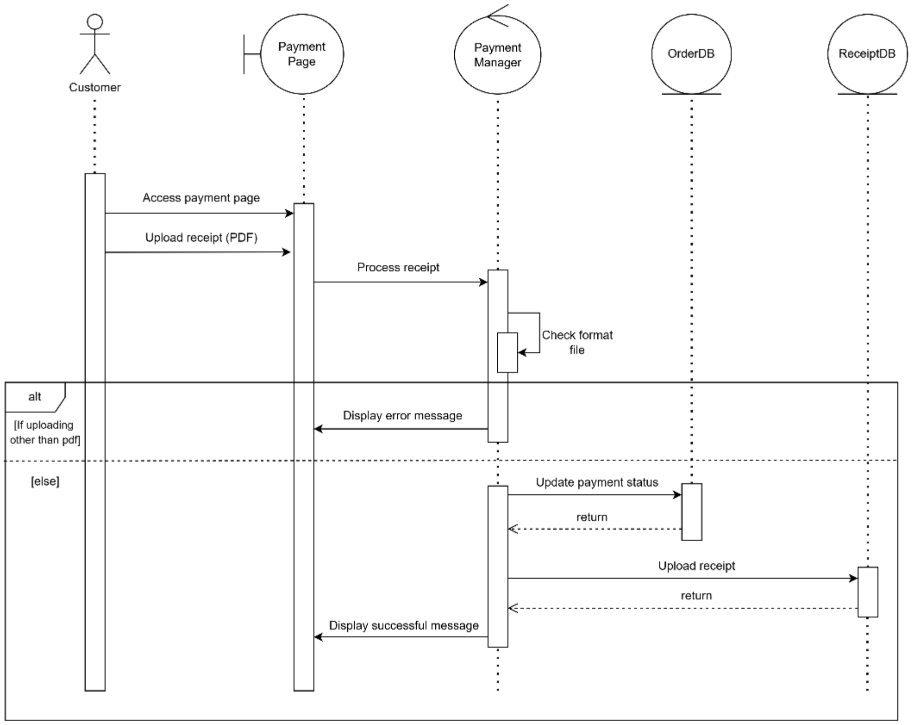
Use case 3: Customer View Item



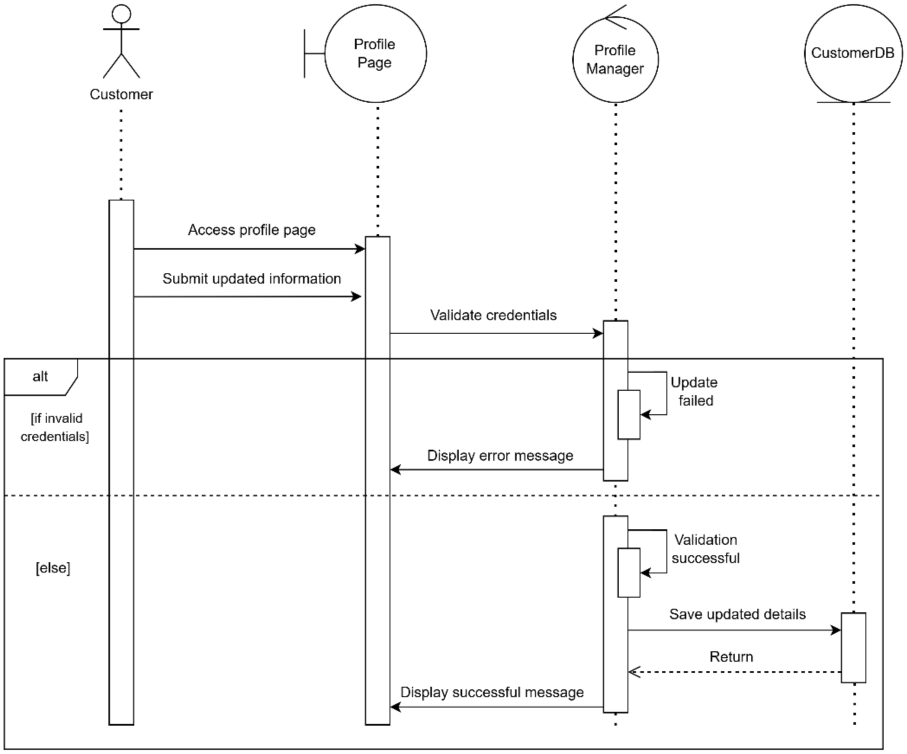
Use case 4: Customer Manage Order



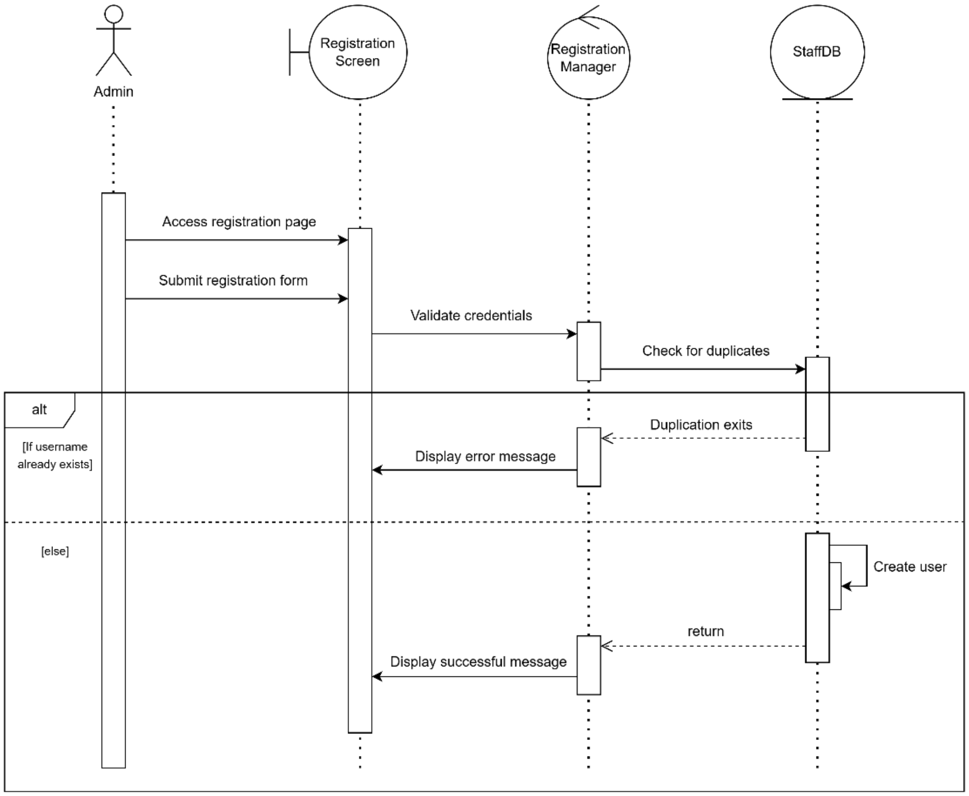
Use case 5: Customer Update Payment



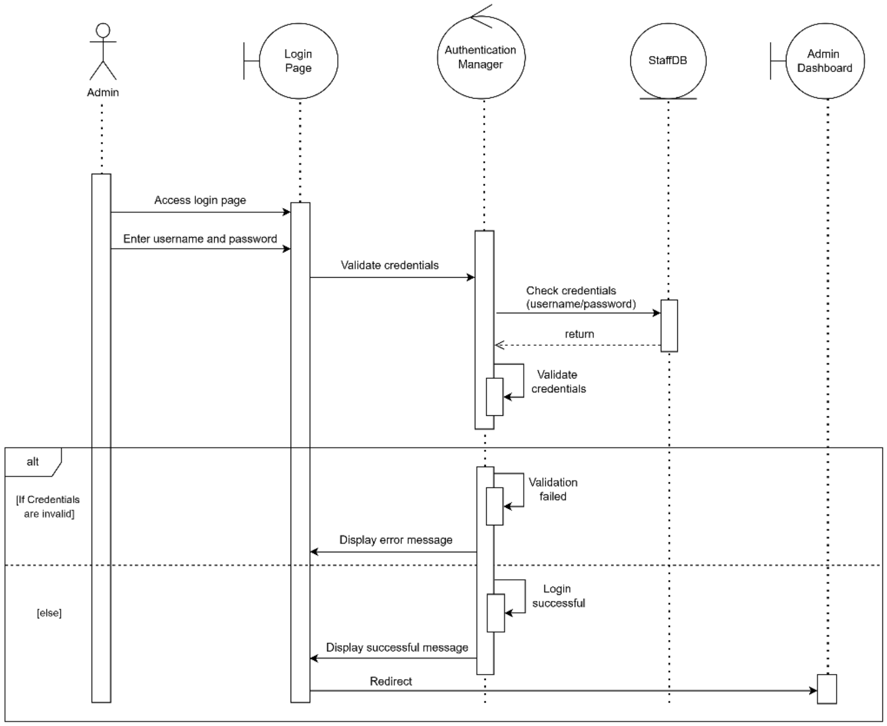
Use case 6: Customer Update Profile



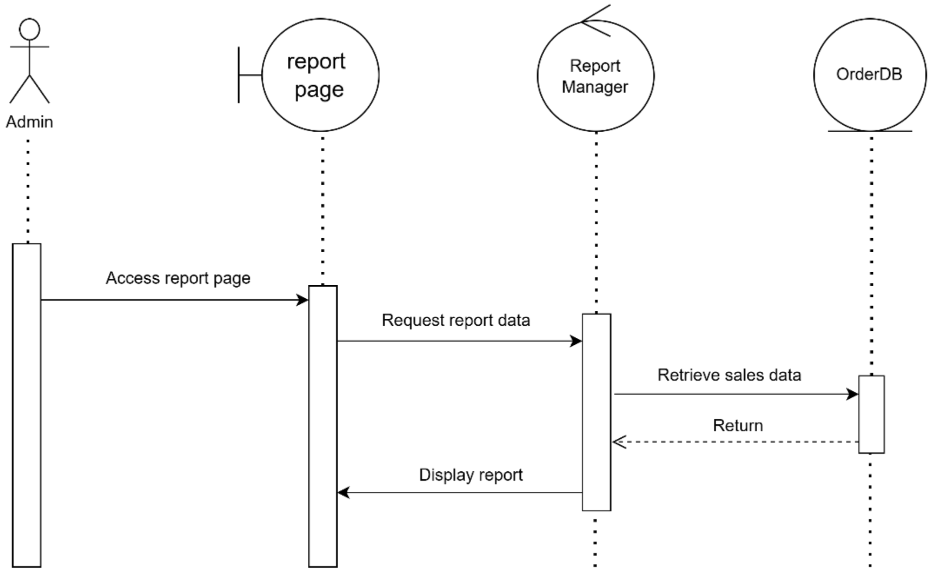
Use case 7: Admin Registration



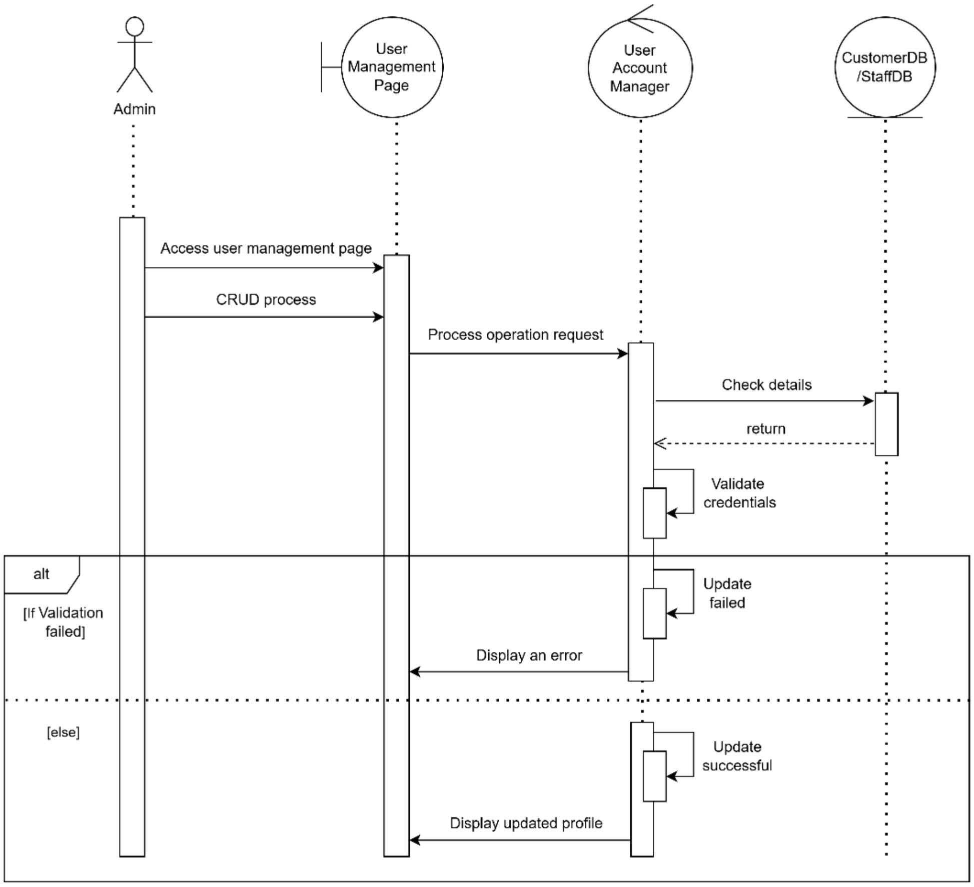
Use case 8: Admin Login



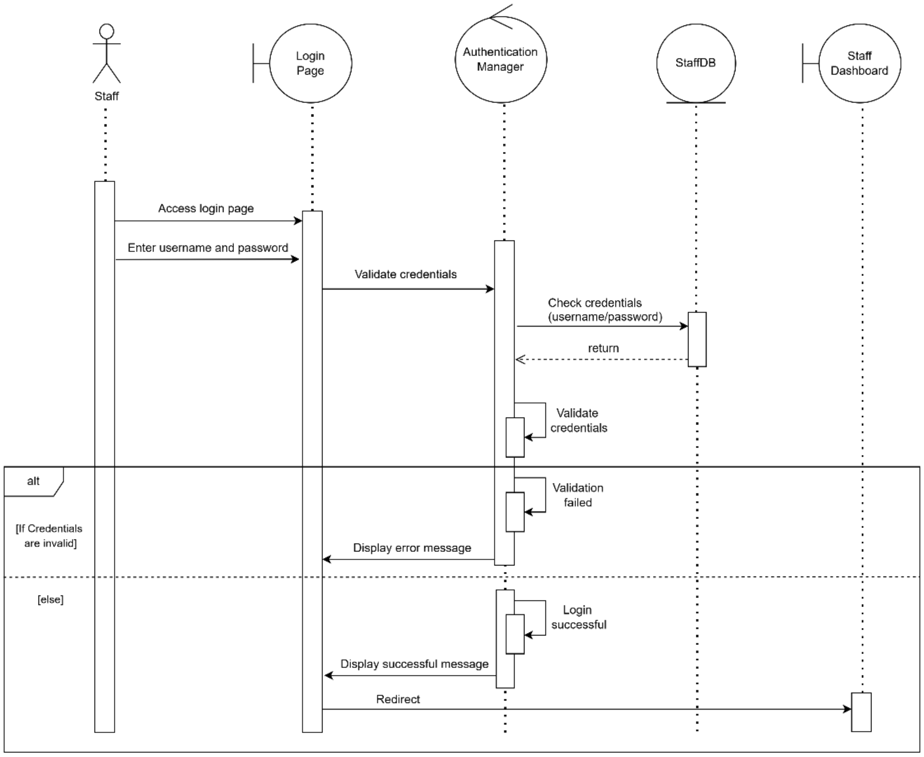
Use case 9: Admin Generating Sales Reports



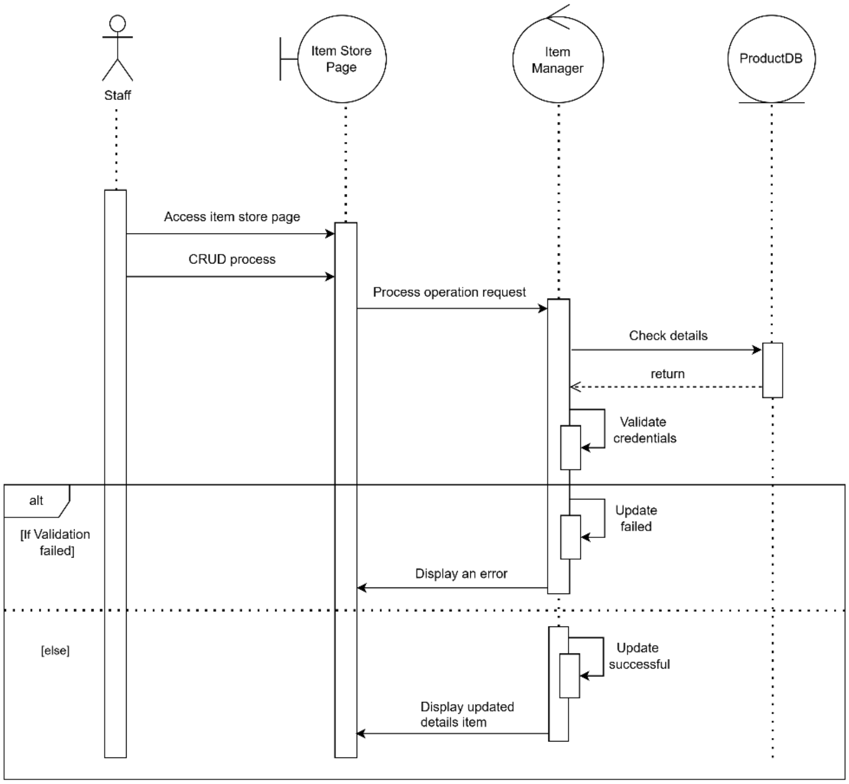
Use case 10: Admin Manage User Account



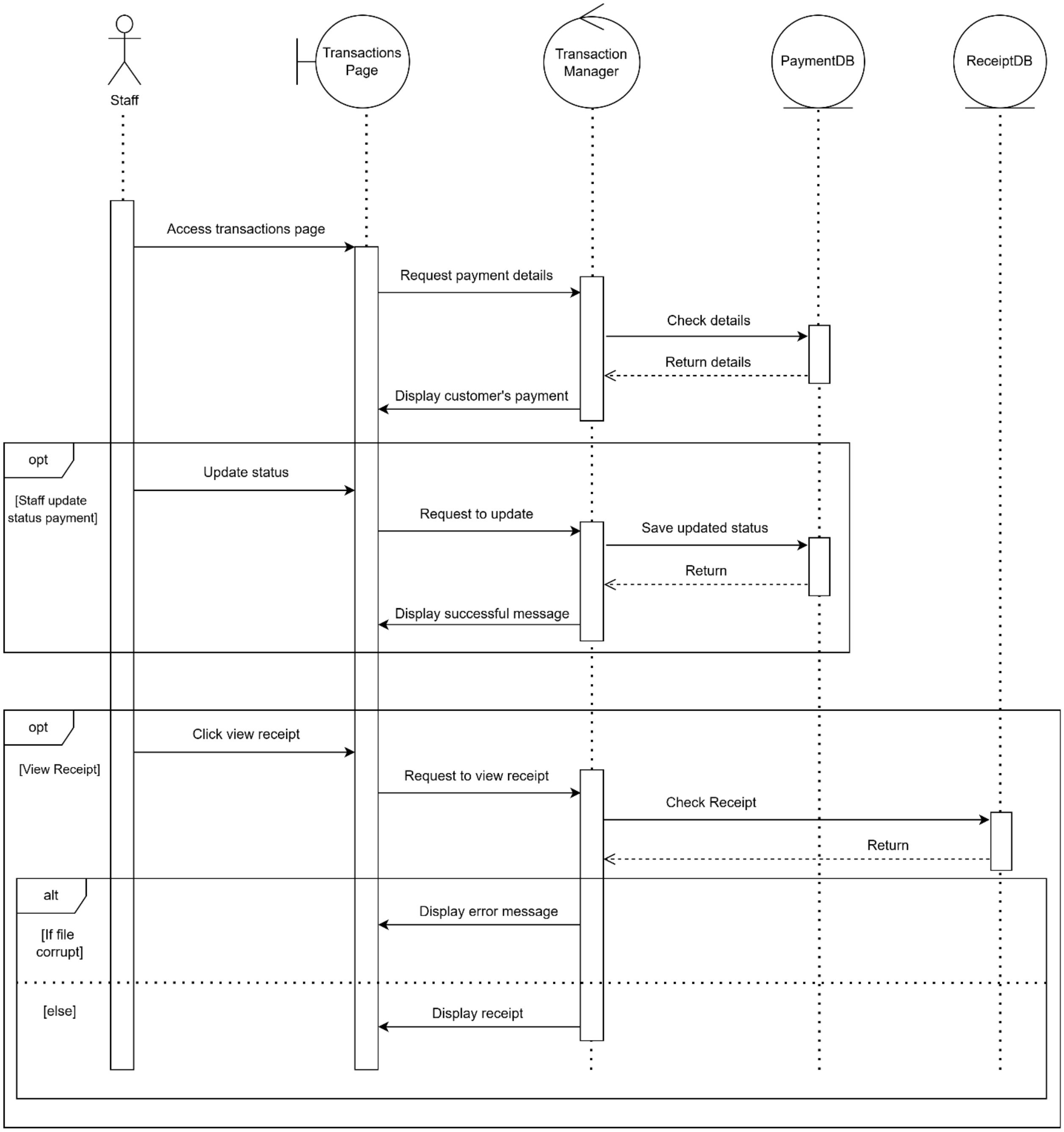
Use case 11: Staff Login



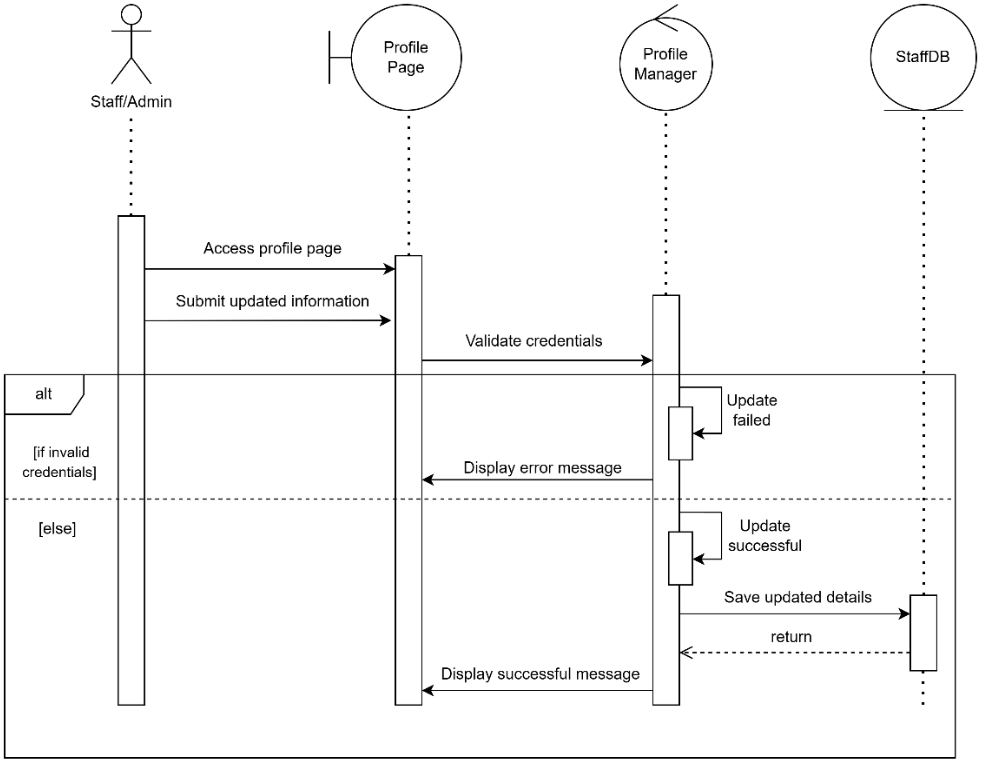
Use case 12: Staff Manage Phone List



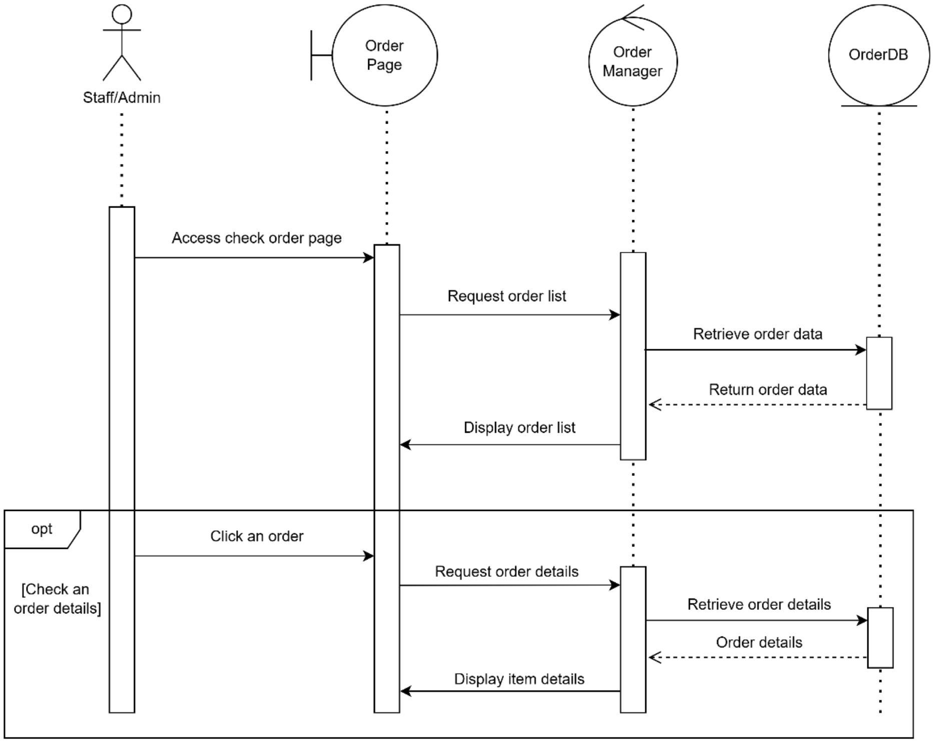
Use case 13: Staff Manage Transactions



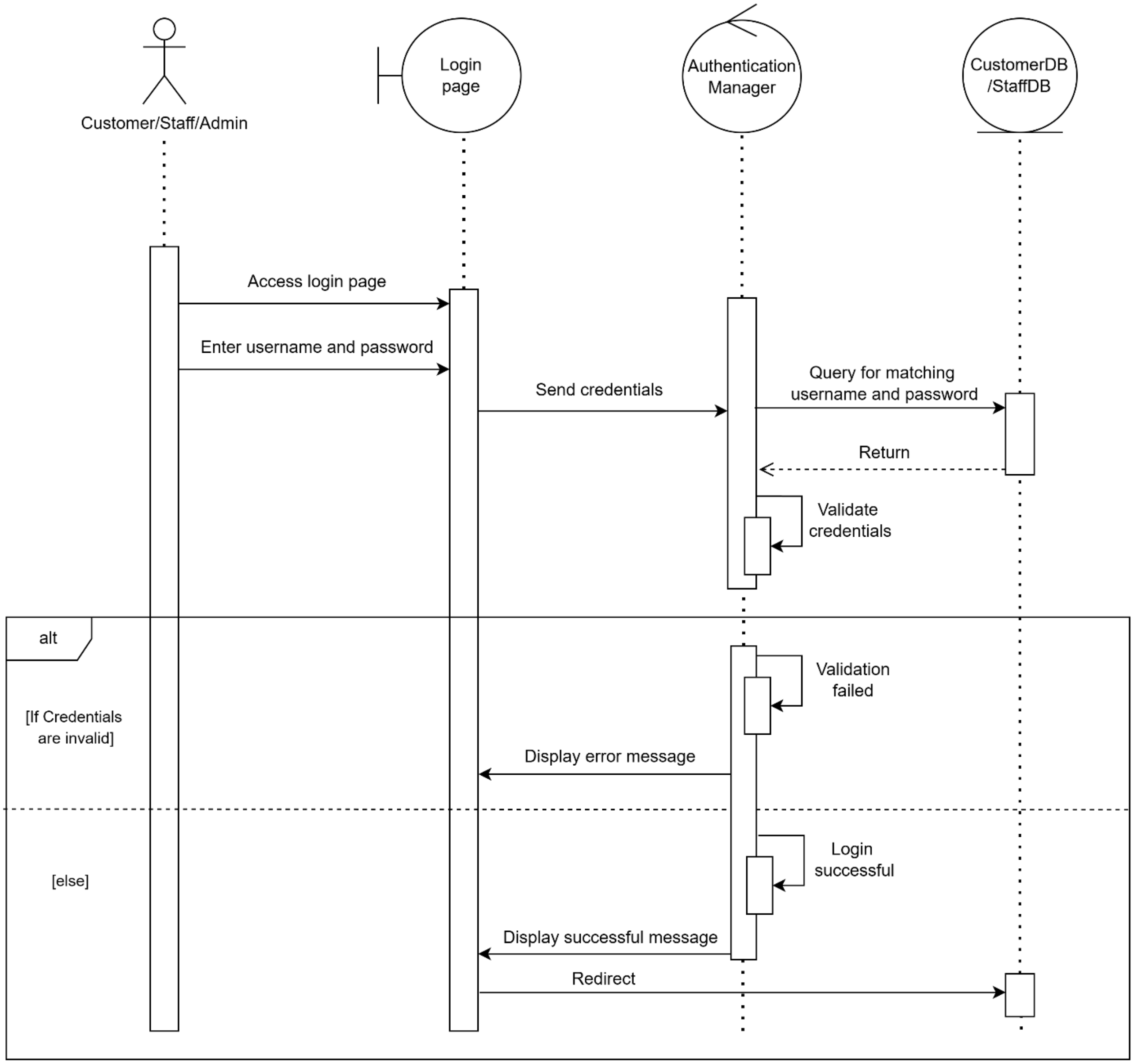
Use case 14: Admin/Staff Update Profile



Use case 15: Admin/Staff Check Order



Use case 16: Authentication



Appendix D: To Be Determined List

N/A