****

**A Project documentation on**

**SUBSCRIPTION-BASED E-COMMERCE WEB APPLICATION**

**CODEFURY COMPETITION**

**Team ID – WFS3\_PUN\_TEAM3**

|  |  |
| --- | --- |
| **Name** | **PeopleSoft ID** |
| Mridul Gupta | 45393805 |
| Kritika Mittal | 45393935 |
| Sanchit Singhal | 45393470 |
| Jasleen Phutela | 45393375 |
| Vivek Singh Gusain | 45393822 |
| Sannihitha Navuluru | 45393751 |
| Kadali Yaswanth Kumar | 45393846 |
| Bokam Bhanu Prasad | 45393903 |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Chapters** | **Page No.** |
| Chapter 1 | Project Overview | 1-3 |
| Chapter 2 | Requirements | 4 |
| Chapter 3 | System Architecture | 5-7 |
| Chapter 4 | UI Design & Guidelines | 8-9 |
| Chapter 5 | UML Diagrams | 10-14 |
| Chapter 6 | Conclusion | 15 |
|  | Table of Figures | ii |

**TABLE OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Figures** | **Page No.** |
| 5.1 | Use Case Diagram | 11 |
| 5.2 | E-R Diagram | 12 |
| 5.3 | Class Diagram | 13 |
| 5.4 | Data Flow Diagram | 14 |

**CHAPTER 1**

**PROJECT OVERVIEW**

**1.1 Introduction**

E-commerce has grown to be an essential component of the global economy in the quickly changing digital landscape by giving customers access to a vast array of goods and services with unmatched ease. By fusing subscription-based services with the conventional e-commerce paradigm, the Subscription-Based E-Commerce Web Application aims to improve the online buying experience. The goal of this project is to create a feature-rich online application that serves administrators and customers alike, providing a smooth and effective platform for handling orders, subscriptions, and items.

**1.2 Overview**

The purpose of the subscription-based e-commerce web application is to cater to the needs of contemporary customers who would rather have their favorite products delivered on a regular basis. Customers can use the program to subscribe to the products of their choice, which guarantees hassle-free and frequent delivery. Additionally, the system offers a strong administrative interface for handling orders, subscriptions, and items, guaranteeing efficient operations and excellent client satisfaction.

**1.2 Key Features and Components**

**1.2.1 Customer Management**

- **Registration and Authentication**: Clients can create an account on the platform and safely log in to view their accounts. To guarantee a seamless user experience, the system verifies user input and offers error handling.

- **Subscription Management:** Consumers can choose the frequency and length of their product subscriptions, tailoring them to their own tastes. Additionally, they can change or cancel their memberships whenever they choose.

- **Order Placement and Tracking:** Orders for products that are connected to customers' subscriptions can be placed. From order placing to delivery, the technology gives users real-time status updates.

**1.2.2 Admin Management**

- **Product Management:** The product catalogue is entirely under the administrators' hands; they can add, edit, and remove products as needed. The system makes sure that the product database appropriately reflects every modification.

- **Subscription Oversight:** Administrators have the ability to monitor and control every subscription, guaranteeing that the system is current and that clients receive their orders on schedule.

- **Order Management:** Administrators have access to any order placed by a customer, can monitor its progress, and can step in to fix problems as needed. This feature guarantees a dependable and effective order fulfilment process.

**1.2.3 Database and Data Handling**

All relevant data, such as client information, product specifications, subscription plans, and order histories, are stored by the program in a relational database. The database is built with appropriate indexing and normalization to maximize efficiency, as well as data integrity and security.

**1.2.4 User Interface**

The program has an easy-to-use user interface that was created using contemporary web technologies including JavaScript, HTML, and CSS. With a header, footer, and sidebar for convenient navigation, the layout guarantees a unified appearance and feel on all pages. The user interface is made to be accessible and responsive, offering a consistent experience on various devices.

**1.2.5 Security and Error Handling**

To safeguard user data and stop unwanted access, the program has strong security features like input validation, secure authentication, and data encryption. Extensive error handling procedures guarantee that problems are quickly found and fixed, reducing user experience interruptions.

**CHAPTER 2**

**REQUIREMENTS**

**2.1 Functional Requirements**

**2.1.1 For Customers**:

- **Registration**: Ability to register with the system.

- **Login**: Secure login with valid credentials.

- **Browse Products**: View a catalogue of available food products.

- **Subscribe to Products**: Choose from various subscription plans and subscribe to products.

- **Cancel Subscription**: Option to cancel an existing subscription.

- **View Subscription History**: Access and view the history of their subscriptions.

**- Place Orders**: Ability to place orders for subscription products.

**2.1.2 For Admins:**

- **Login**: Secure login with admin credentials.

- **Manage Products**: Add, update, and delete product listings.

- **Define Subscription Types**: Set up and configure different subscription options.

- **Activate/Deactivate Subscriptions**: Control the status of subscriptions.

- **View Subscription History**: Monitor subscription activities and histories.

- **View Daily Delivery Lists**: Access lists of scheduled deliveries based on active subscriptions.

- **Manage Orders**: Oversee and manage customer orders.

**2.2 Non-Functional Requirements**

- **Usability**: The user interface should be intuitive and user-friendly.

- **Performance**: The application should handle multiple users and subscriptions efficiently.

- **Security**: Secure user authentication and authorization.

- **Scalability**: The system should be scalable to accommodate growing numbers of users and products.

- **Maintainability**: Code should be modular and easy to maintain.

**CHAPTER 3**

**SYSTEM ARCHITECTURE**

The Subscription-Based E-Commerce Web Application's system architecture is multi-layered in order to guarantee robustness, scalability, maintainability, and concern separation. The design is broken down into multiple important layers, each of which is in charge of particular functions inside the program, such as data administration, business logic, and user interface. An extensive explanation of the system architecture may be found below.

* 1. **Presentation Layer**

The application's front-end interface, or Presentation Layer, is in charge of communicating with users. With the help of web technologies like HTML, CSS, and JavaScript, it is created to be responsive and easy to use on a range of devices.

* **Designing User Interfaces (UI)**

Important elements of the user interface, such the header, footer, and sidebar, are consistent throughout all pages and make navigating simple.  
The user interface (UI) manages user inputs, shows information retrieved from the backend, and gives users feedback via alerts, notifications, and visual cues.

* **Client-Side Scripting**

DOM modification and client-side validation are managed by JavaScript.

* **CSS Frameworks**

Bootstrap is used to improve UI design. This framework facilitates a more efficient design process by offering responsive grid systems and ready-to-use components.

* 1. **Business Logic Layer**

The application's core, the Business Logic Layer (sometimes referred to as the Service Layer), is where all business rules and procedures are put into practice. To carry out the required tasks, this layer communicates with the Presentation Layer and the Data Access Layer.

* **Implementing Business Logic**

The main business logic for handling customer subscriptions, ordering, processing payments, and enforcing business rules (such product availability and subscription limits) is contained in this layer.

* **Service Classes**

The purpose of service classes is to isolate the data manipulation code from the application's rules by encapsulating the business logic. AdminService and CustomerService are the service classes.

* **Exception Handling**

This layer has strong exception handling techniques to handle mistakes that arise when business logic is being executed, guaranteeing the application's stability and user-friendliness.

* 1. **Data Access Layer (DAO Layer)**

The Data Access Layer (DAO Layer) is responsible for communicating with the database. This layer contains the Data Access Objects (DAOs) that encapsulate the logic needed to interact with the underlying database.

* **DAO Implementation**

CRUD (Create, Read, Update, Delete) activities on the database are carried out by DAOs. There is a corresponding DAO class for every entity, including Customer, Product, Order, and Subscription, which manages data operations for those entities.

* **Database Connectivity**

A connection pool is used to manage the connection to the database, guaranteeing effective use of database resources. The DAO classes manipulate data via stored procedures or SQL queries.

* 1. **Data Access Layer (DAO Layer)**

The Data Access Layer (DAO Layer) is responsible for communicating with the database. This layer contains the Data Access Objects (DAOs) that encapsulate the logic needed to interact with the underlying database.

* **DAO Implementation**

CRUD (Create, Read, Update, Delete) activities on the database are carried out by DAOs. There is a corresponding DAO class for every entity, including Customer, Product, Order, and Subscription, which manages data operations for those entities.

* **Database Connectivity**

A connection pool is used to manage the connection to the database, guaranteeing effective use of database resources. The DAO classes manipulate data via stored procedures or SQL queries.

* 1. **Database Layer**

All of the application's data is managed and kept in the database layer. It is composed of a relational database management system (RDBMS) that maintains relationships between tables and arranges data into them.

* **Relational Database**

Entity-Relationship (ER) modelling is used in the database design process to specify the relationships between entities like Admin, Customer, Product, Order, and Subscription.  
To guarantee data integrity and effective querying, tables are made for each entity with the proper primary keys, foreign keys, indexes, and constraints.

* **Database Schema**

The schema includes definitions for tables, columns, data types, relationships (one-to-one, one-to-many, many-to-many), and indexes.

**CHAPTER 4**

**UI DESIGN & GUIDELINES**

**4.1 User Interface Design**

**4.1.1 Home Page:**

- User Registration: Form for new user registration with validation.

- Login: Form for user login.

**4.1.2 Main Page (Admin):**

- Admin Dashboard: Provides access to all admin functionalities.

**4.1.3 Main Page (Customer):**

- Customer Dashboard: Provides access to all customer functionalities.

**4.1.4 Layout:**

- Header, footer, and sidebar with navigation links maintained across all pages.

**4.2 Testing**

**JUnit Test Cases:**

- Registration Tests: Verify user registration functionality.

- Login Tests: Check authentication processes.

- Subscription Tests: Ensure correct handling of subscriptions and cancellations.

- Product Management Tests: Test the addition, update, and deletion of products.

- Order Processing Tests: Validate order placement and management.

**4.3 Implementation Guidelines**

**4.3.1 Team Organization:**

- UI Team: Responsible for front-end development using HTML, CSS, JavaScript, Bootstrap.

- Back End Team: Responsible for business logic, DAO, and database management.

**4.3.2 Architecture:**

- Employ a layered architecture with loose coupling.

**4.3.3 Validation:**

- Implement thorough validation for all user inputs with proper error handling.

**4.3.4 UI Layout:**

- Use a consistent layout with header, footer, and sidebar.

**4.3.5 Optional Enhancements:**

- Explore CSS frameworks to enhance UI design.

**CHAPTER 5**

**UML DIAGRAMS**

**5.1 Use Case Diagram:**

Illustrates interactions between users (Customer and Admin) and the system, detailing the functionalities provided to each type of user.

**5.2 Class Diagram (UML):**

Shows the classes involved in the application, their attributes, methods, and relationships. Key classes include Admin, Customer, Product, Subscription, and Order.

**5.3 Data Flow Diagram (DFD):**

- Level 0: High-level view of the system and its major processes.

- Level 1: Detailed view of individual processes like user registration, product management, and order processing.

**5.4 Entity-Relationship (ER) Diagram:**

Depicts entities (Admin, Customer, Product, Subscription, Order) and their relationships, showing how data is organized and interconnected in the database.

**5.5 RDBMS Table Structures:**

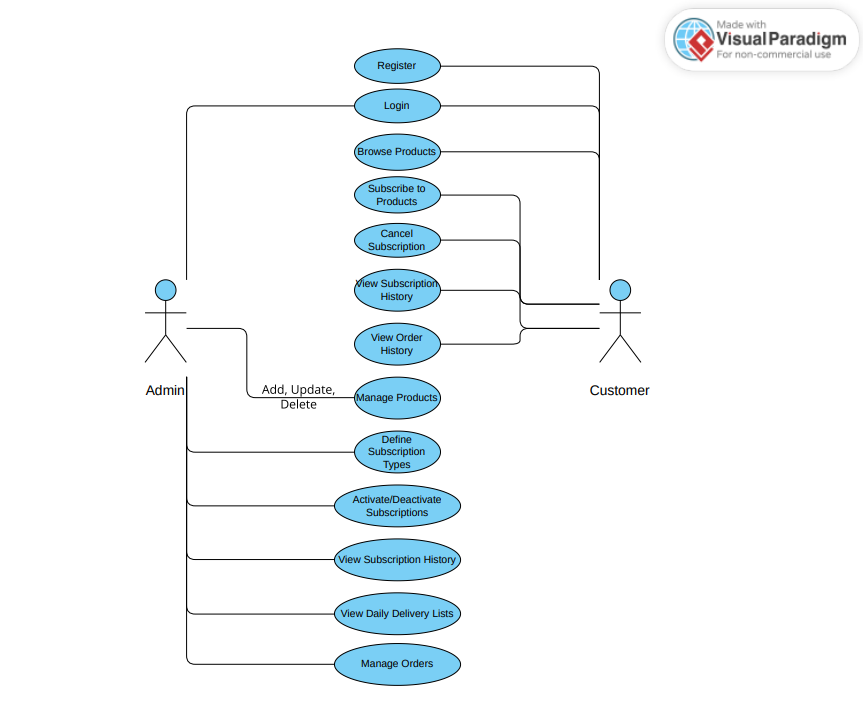
- Admin Table: Stores Admin information.

- Customer Table: Stores customers information.

- Product Table: Stores product details.

- Subscription Table: Stores subscription information linking users and products.

- Order Table: Stores order details including customer and product information.

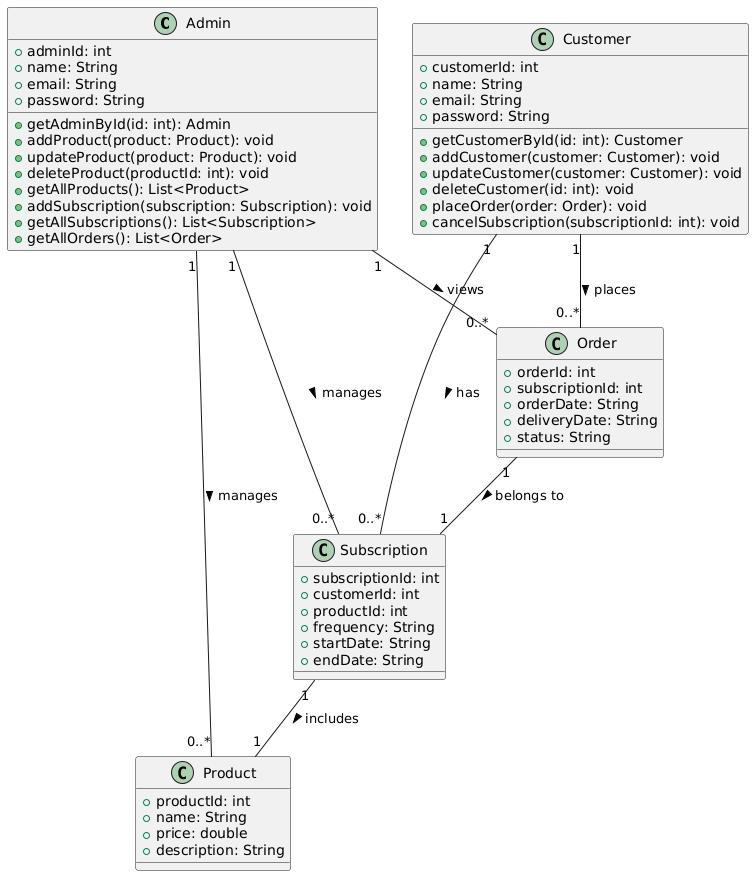
****

**Fig 5.1 Use Case Diagram**

**A diagram of a diagram

Description automatically generated with medium confidence**

**Fig 5.2 E-R Diagram**

** Fig 5.3 Class Diagram**

**A diagram of a company

Description automatically generatedFig 5.4 Data Flow Diagram**

**CHAPTER 6**

**CONCLUSION**

The feature-rich Subscription-Based E-Commerce Web Application is made to satisfy the demands of administrators and customers alike. The application provides a distinctive value proposition in the online retail market by fusing subscription services with conventional e-commerce features. In order to provide a dependable and effective e-commerce solution, the project integrates secure data management, solid backend technologies, and user-friendly interfaces. This program seeks to improve customers' entire shopping experiences while giving administrators the resources they need to efficiently oversee operations through thoughtful planning, design, and execution.

The objective of this project is to create a web application that will enable users to recurrently subscribe to different food goods. The application delivers products consistently and supports a variety of subscription plans, including weekly, bi-weekly, and monthly. Customers can manage their subscriptions using the system, while administrators can manage both items and subscriptions.