Software Design Specification (SDS)

Project Name: El Hamour Market
Prepared By: Mohamed Bekheet, Bassam Mohamed, Ahmed Elrashidy, Seif Aboshanab, Mohamed Mahmoud
Date: 8/11/2024

---

1. Introduction

1.1 Purpose

The purpose of this document is to describe the design and architecture of the El Hamour Market online marketplace. It outlines the functionality and design decisions to be followed during the development process, ensuring flexibility, secure, and user-friendly platform.

1.2 Scope

This Software Design Specification (SDS) covers the design and implementation details of El Hamour Market.

- This Software Design Specification (SDS) covers the design and implementation details of El Hamour Market. The software will perform the following major tasks:

- Task 1: Provide a simple shopping experience by offering a variety of products, including food, vegetables, fruits, and home tools.

- Task 2: Ensures security account management and data protection, including strong authentication and data encryption.

- Task 3: Offer advanced features like data mining, smart search, and efficient data retrieval to enhance the user experience and system performance.

---

2. System Overview

- The system consists of the following components:

- Frontend: The frontend of El Hamour Market is built using React for creating a dynamic and interactive user interface. Bootstrap is used to ensure the design is responsive and good visually.

- Backend: The backend is developed with Flask, a lightweight web framework in Python, which provides the necessary APIs for handling user requests, data management, and business logic.

- Database: Microsoft SQL is used for securely storing and managing data, including product information, user accounts, and transaction details.

---

3. System Architecture

3.1 Architectural Design

- This project follows a client-server architecture, where:

- The frontend communicates with the backend using Flask.

- The backend interacts with the Microsoft SQL database to manage the data stored and to retrieve, which provides a simple way to reach user information.

3.2 Data Flow

1. **User Interaction: The user interacts with the user interface to perform actions, such as searching for products, adding items to the cart, or managing their profile.**

2. **Request Processing: The frontend sends a request to the backend server to execute the action.**

3. **Data Handling: Enable interacting with the database, and ensure updates of the necessary data as requested.**

4. **Response: The backend sends a response back to the frontend, which updates the user interface to reflect the results of the user's action, such as displaying search results or confirming a successful purchase.**

---

4. Database Design

- The system will store data in a relational database (Microsoft SQL) with the following main entities and relationships:

- Table 1: Users

- UserID: Unique identifier for each user considered as a Primary Key(string).

- Username: to Store the username of the user who's logged to the system(varchar).

- Password: Encrypted password to provide more security for users(string).

- Email: Email address of the user to be able to log in(varchar).

- UserType: The user is a customer or admin"(varchar).

Table 2: customer(inherit from users)

- UserID: Unique identifier for each user considered as a Primary Key(string, Foreign key that provides a one to one relationship with user id in user table).

- Username: to Store the username of the user who's logged to the system(varchar).

- Password: Encrypted password to provide more security for users(string).

- Email: Email address of the user to be able to log in(varchar).

Explanation of Relationships
One-to-One Relationship: Each UserID in the Users table will correspond to exactly one record in either the Customer or Admin table.

UserType Field: This field in the Users table indicates whether a user is a customer or an admin which provide each of them access to specific things.

---

5. Technology Stack

- **Frontend: React is used for frontend development to create a dynamic and interactive user interface. Bootstrap is also used to ensure the user unterface that is simple and flexible.**

- **Backend: Flask is used as a backend for web development.**

- **Database: Microsoft SQL is used for managing and securely storing application data.**

---

6. Testing Plan

6.1 Unit Testing

Both the frontend and backend will undergo unit testing to ensure that individual components are working as expected. For the frontend, unit tests will cover user interface form validations, and individual React components. On the backend, tests will cover Flask API. data processing functions, and database interactions.

6.2 Integration Testing

Test the frontend and Backend: Testing the communication between React components and Flask APIs to ensure data flows correctly.
Backend and Database: Verifying that the backend can retrieve and update data accurately in the Microsoft SQL database.

6.3 User Acceptance Testing (UAT)

End users will test the system to confirm it meets their needs and expectations. Feedback from users will be collected to identify any usability improvements.

6.4 Performance Testing

Stress and load testing will be conducted to ensure that the system is working well without any concerns and handle the expected number of users and transactions without performance issues. Performance tests will measure response times, database query efficiency, and server load handling under high traffic.

7. Conclusion

The El Hamour Market online marketplace is designed to fulfill the specified functional and non-functional requirements as described in this Software Design Specification (SDS). The design outlined here ensures that the system will be flexible, simple for different users, and user-friendly, providing a seamless shopping experience for its users. By implementing a secure and responsive platform with efficient data management, El Hamour Market aims to deliver lasting value and satisfaction to customers.