Final project proposal: Store Management System

Project Overview:

In this project, we used **4 advanced elements. They are database, GUI, network, and multithreading. We also used encryption to make the password safer.**

Our project introduces a Store Management System designed to facilitate online store ordering.

On the store side, managers can update inventory by modifying stock levels, removing items, or adding new products. Additionally, they can update the status of orders, informing customers about the availability for pickup.

For clients, the system offers a user-friendly **interface** to browse store inventory, add items to their shopping cart, and place orders. Customers can also review their past orders through their order history.

To ensure security and personalized service, each user must create a unique account with a username and password upon registration.

We use a robust SQLite **database** to store user data, inventory information, and order history.

We implement secure and efficient **network** communication using Java Sockets. The system supports simultaneous operations from multiple clients by using **multithreading**.

We use SHA-256 to encrypt the password and there is no decrypt function. So, even the developer cannot see the password the user set for their account.

To run this system on your local machine, follow these steps:

Step 1: Install Java and Eclipse. We used Java 8, so it is recommended that you also install Java 8 or newer. Also, make sure you have **SQLite** on your computer. Install one if you do not have.

Step 2: Download the zip file for the code and unzip it. **PLEASE save the project folder on Internal disk, not** external disk like portable hard drive.

Step 3: Import the project folder in Eclipse.

Step 4: Start by running the **"Server.java".**

Step 5: After you see **“Server is running and ready to accept connections...”** in the Server Logs window**,** run **“Start.java”.** Make sure that **the port** in the start.java is the same as the server.

Step 6: You will be required to set up a new admin account if there are none. You can follow the demo video to learn how to do all the operations. Remember, **always use “Start.java”** to open a new login window.

You can update items, change order status, and add new admin account when you login an **“admin”** account. You can place order, view the inventory, check your order history when login a **“user”** account. You can create new user account by clicking **register,** and login an account by clicking login button in login part.

Step 7: Type **“close”** or close the window of the server to close the server. You should see **"Server is shutting down..."** and the window is closed.

Features:

Order Management (Client Side):

- User-friendly GUI for browsing store inventory and placing orders.
- Real-time communication with the server to check item availability and place orders.
- Order history and tracking status functionality for users.

Stock Management (Server Side):

- GUI for store administrators to update and manage inventory, including adding new items, updating existing ones, and setting stock levels.
- Process orders and change order status.
- Set up a new admin account.

Account login/registration:

- GUI for login/registration. If the account belongs to the store, it will open the Stock Management GUI. Or it will open Order Management GUI. A new account can be created and stored in the database if the user does not have an account.
- If there is no account, the system requires you to set up a new admin account.

Networking:

- Secure and efficient communication between client and server applications using Java Sockets.
- Communication is accomplished by sending and receiving JSON.
- Implementation of client-server architecture to support multiple clients simultaneously.

Database Integration:

- Robust database schema for storing user data, inventory information, order history, and users’ information by using SQLite.

- Efficient data retrieval and update mechanisms to ensure system performance.

Multithreading:

- Each client will start a new thread in the server, and support simultaneous operations from multiple clients.

Security:

- Use SHA-256 to encrypt the password.

Architecture:

The system architecture is divided into two main components:

- Client Application: A frontend application for customers to browse inventory, place orders, and track their order history.
- Server Application: A backend application for store administrators to manage inventory, process orders, and interact with clients.

Project Implementation:

- Phase 1: Develop the GUI for both the client and server using the Swing library.
- Phase 2: Implement inventory management and order processing functionalities, including database integration for persistent data storage.
- Phase 3: Set up the basic server and client architecture with simple communication and a preliminary database schema and transfer current functions to be compatible with the network.
- Phase 4: Conduct thorough testing, bug fixing, and performance optimization.

Conclusion:

The "Store Management System" project aims to create a seamless and efficient bridge between customers and stores, enhancing the shopping experience through technology. By leveraging Java's powerful libraries and features, the system will offer a robust, scalable, and user-friendly platform for online ordering and inventory management.