# **fast-logo Project Report Inventory Management System K21-3906 K21-3907 K21-3914 Subject: Database Management S Section: BSE-5B Abstract**

The Inventory Management System is a robust and user-friendly Java-based desktop application designed to meet the inventory and transaction management needs of small to mid-sized stores. Utilizing a client-server architecture with Java and MySQL, the system offers a seamless and efficient solution for product, customer, and supplier management. Key features include real-time stock updates, transaction recording, user authentication, and a comprehensive search functionality. The graphical user interface (GUI) is developed using the Swing framework, ensuring an intuitive and accessible user experience. The MySQL database, with well-defined tables, stores essential data for products, customers, suppliers, and user activities. The system's workflow, depicted in the report, illustrates the logical flow of operations from login to reporting. Screenshots of the working project are attached, providing a visual reference for the system's interface. The project sets a solid foundation for future enhancements, such as bar-code integration and e-commerce functionality, ensuring adaptability to evolving business needs. Overall, the Inventory Management System stands as an effective tool for enhancing inventory control, streamlining transactions, and optimizing business operations for stores of varying scales.

# **1. Introduction**

The Inventory Management System is a comprehensive Java-based desktop application designed for small to mid-sized stores. This report details the architecture, features, database design, system workflow, GUI design, and future enhancements of the system.

## **2. System Architecture**

The application follows a client-server architecture. The Java application serves as the client, and MySQL acts as the back-end database. JDBC is used for seamless communication between the application and the database, ensuring a robust and scale-able system.

## **3. Features**

### **3.1 Product Management**

* ****CRUD Operations****: Users can perform Create, Read, Update, and Delete operations on products.
* ****Real-time Updates****: The system provides real-time updates on stock levels for each product.
* ****Product Code****: Each product is uniquely identified by a product code.

### **3.2 Transaction Management**

* ****Record Transactions****: Users can record both sales and purchase transactions.
* ****Automatic Inventory Adjustment****: The system automatically adjusts the inventory based on transactions.

### **3.3 User Management**

* ****User Types****: Two user types are supported: Administrator and Employee.
* ****Admin Authority****: Administrators have the authority to manage all user accounts.

### **3.4 Search Functionality**

* ****Data Retrieval****: Each section includes a search feature for easy data retrieval.
* ****Enhanced User Experience****: Users can quickly access specific information.

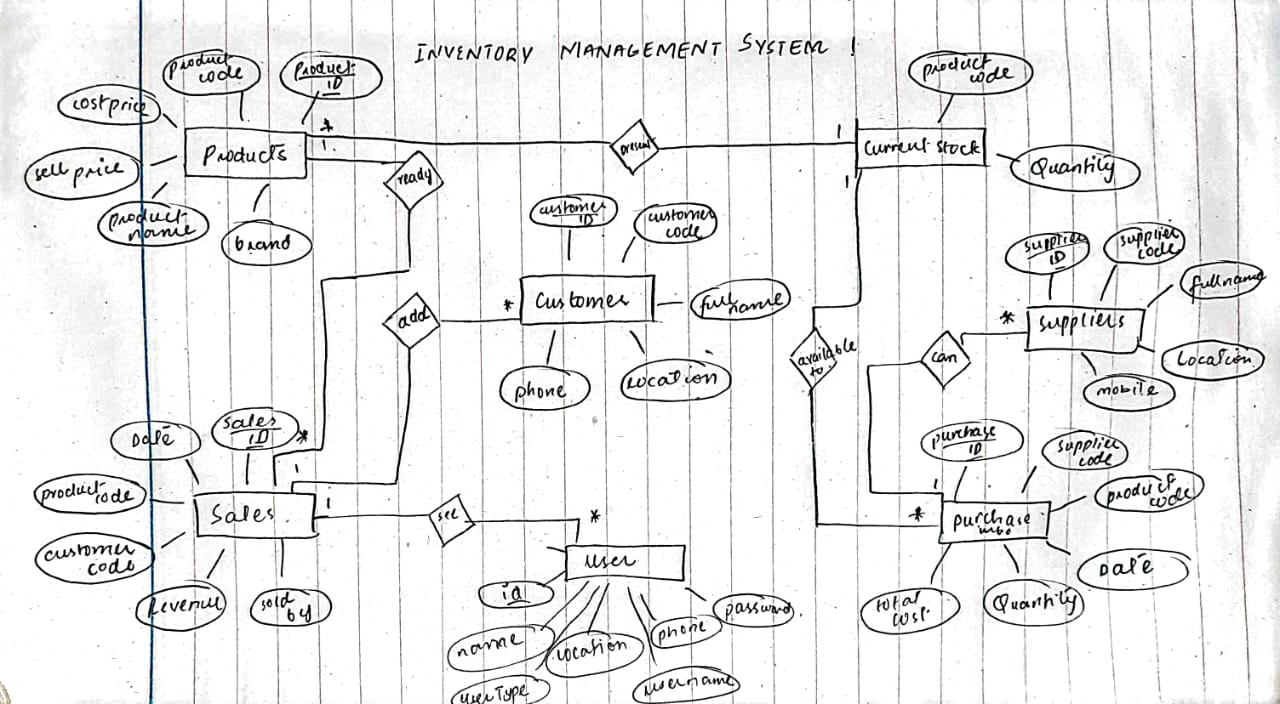
### **3.5 Seamless Sales Process**

* ****Automatic Retrieval****: During a sale, users only need to input the product code, and the system retrieves all relevant details automatically.

### **3.6 Time Log**

### ****User Activity Log****: The system maintains a time log of all users' activities within the application. **4. Database Overview Schema WhatsApp Image 2023-12-05 at 13.13.49_5ce4b38d**

#### 4.1 Database Creation

Our application is supported by a MySQL database named inventory. This database serves as the backbone for storing and organizing critical information related to our inventory management system.  
  
**ERD**

#### 4.2 Table Structure WhatsApp Image 2023-12-05 at 12.13.56_65b2d40d

Our database consists of several key tables, each designed to capture specific aspects of the inventory management process:

* ****currentstock:**** Tracks the current stock quantity of each product.
* ****customers:**** Stores information about our customers.
* ****products:**** Contains details about the products in our inventory.
* ****purchaseinfo:**** Records details of purchases, including supplier information.
* ****salesinfo:**** Logs information about sales transactions, including customer details.
* ****suppliers:**** Stores information about our suppliers.
* ****userlogs:**** Captures logs of user activities for auditing purposes.
* ****users:**** Manages user authentication and authorization.

#### 4.3 Data Population

To ensure a realistic environment for testing and development, we have populated the tables with sample data. This data provides a foundation for demonstrating the functionality of our inventory management system.

### **4.5 Database Features**

#### 4.6 Primary Keys and Indexing

Each table incorporates primary keys to enforce data uniqueness. Additionally, indexing strategies have been considered to optimize search operations for frequently accessed columns.

#### 4.7 Relationships

Our database establishes relationships between tables using foreign keys. For instance, the sales-info table references the products table through the product-code column.

#### 4.8 Normalization

We have employed normalization techniques to minimize redundancy and ensure data integrity. However, it's essential to note that the currentstock table may appear denormalized; this decision was made intentionally to address specific requirements.

#### 4.9 Transactions

The purchaseinfo and salesinfo tables manage information related to purchases and sales, respectively. These tables facilitate the tracking of inventory movements and financial transactions.

#### 4.10 User Authentication and Logging

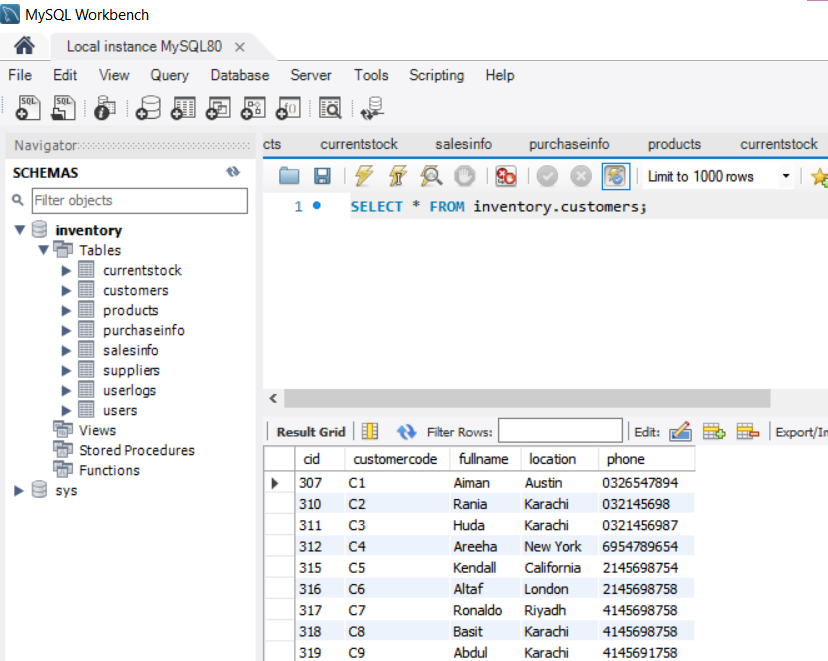
Our system incorporates user authentication through the users table, ensuring secure access. User activities are logged in the userlogs table, providing a comprehensive audit trail.

#### 4.11 Suppliers and Customers

The suppliers and customers tables store essential information about our suppliers and customers, respectively. This data is crucial for maintaining strong relationships with our business partners.

#### 4.12 Use of Double Data Type

Certain fields, such as costprice and sellprice in the products table, utilize the double data type to accommodate decimal values, ensuring precision in financial calculations.

* 

## **5. System Workflow**

### **5.1 Login**

* Users are required to log in with their credentials.
* Authentication ensures access control based on user roles.

### **5.2 Dashboard**

* Provides an overview of inventory status, recent transactions, and user activities.

### **5.3 Product Management**

* Allows users to perform CRUD operations on products.

### **5.4 Transaction Processing**

* Users can record sales and purchase transactions, and the system automatically adjusts inventory levels.

### **5.5 User Management**

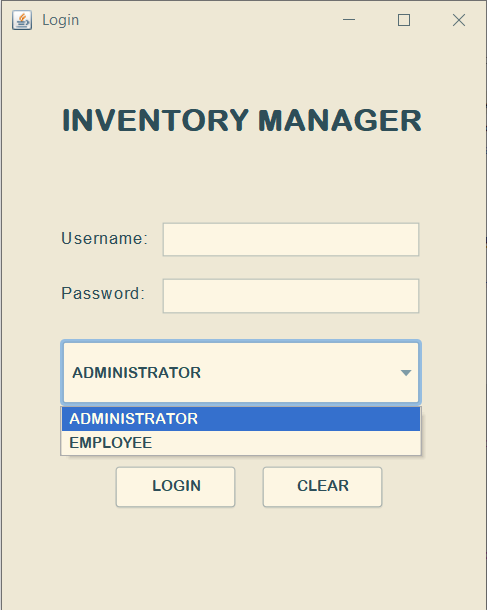
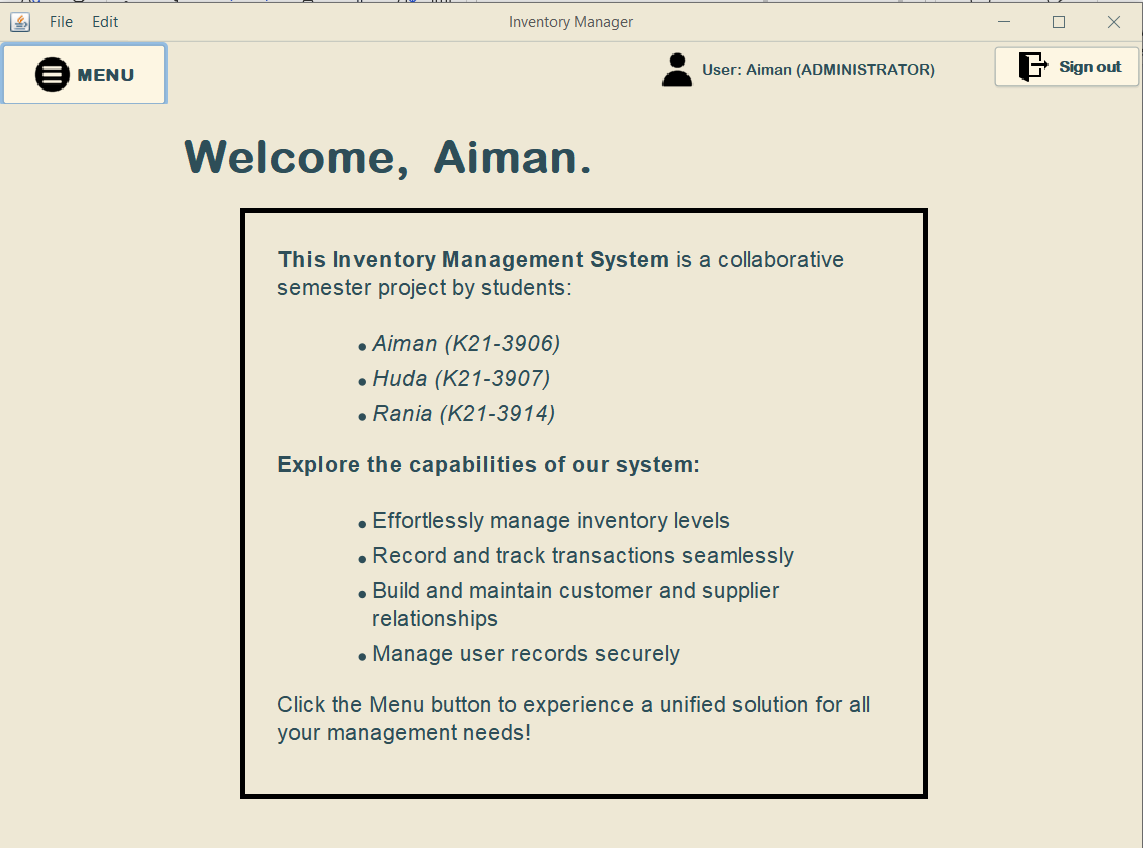
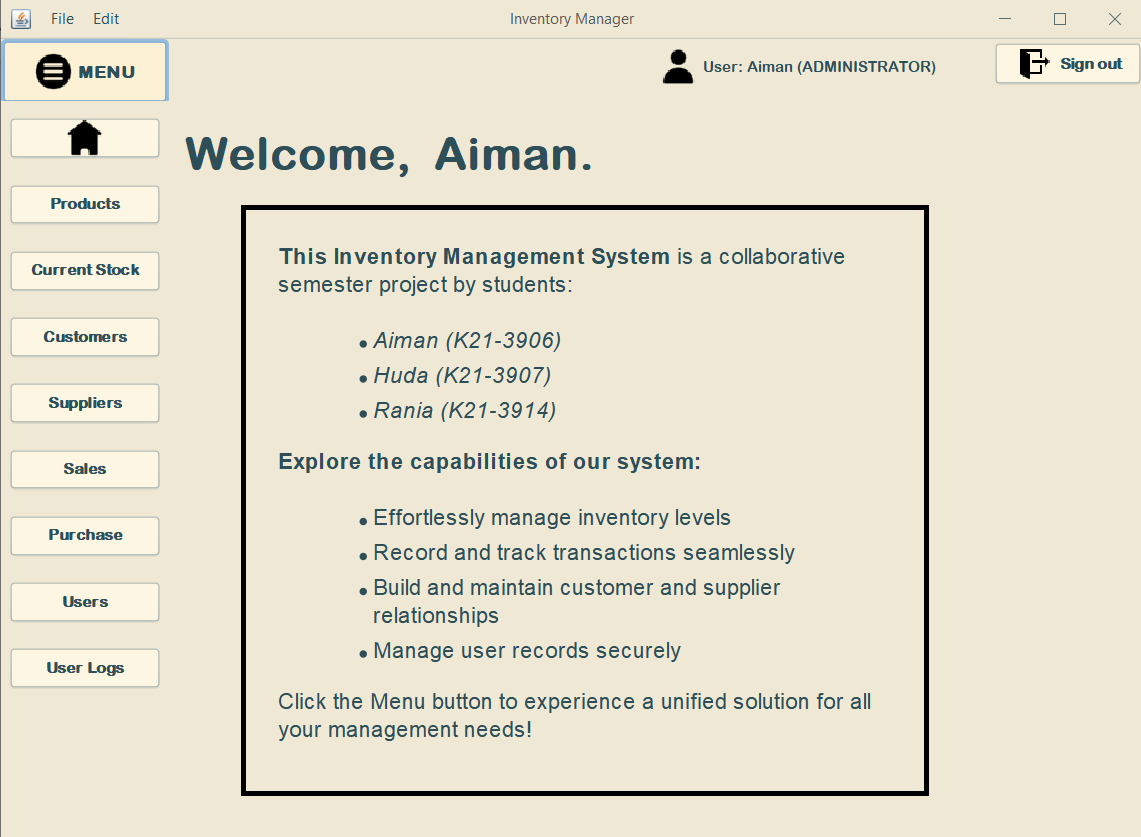
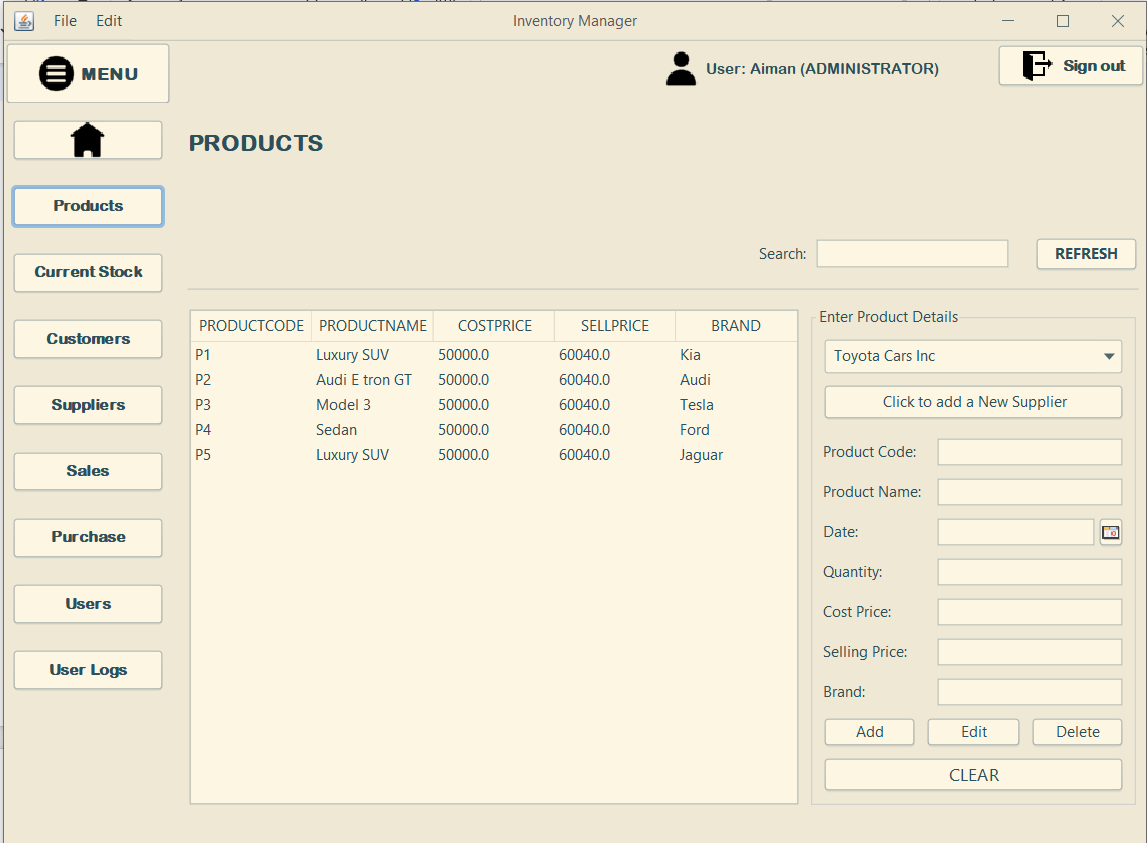
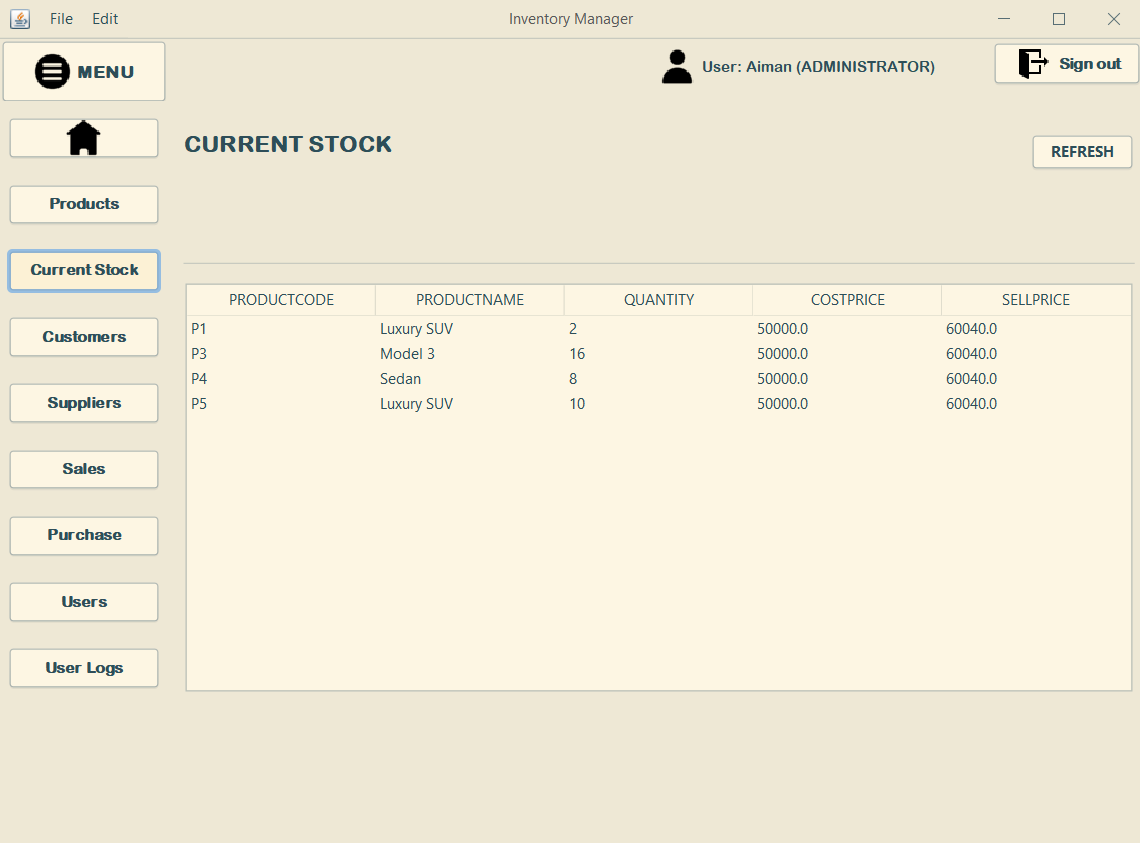
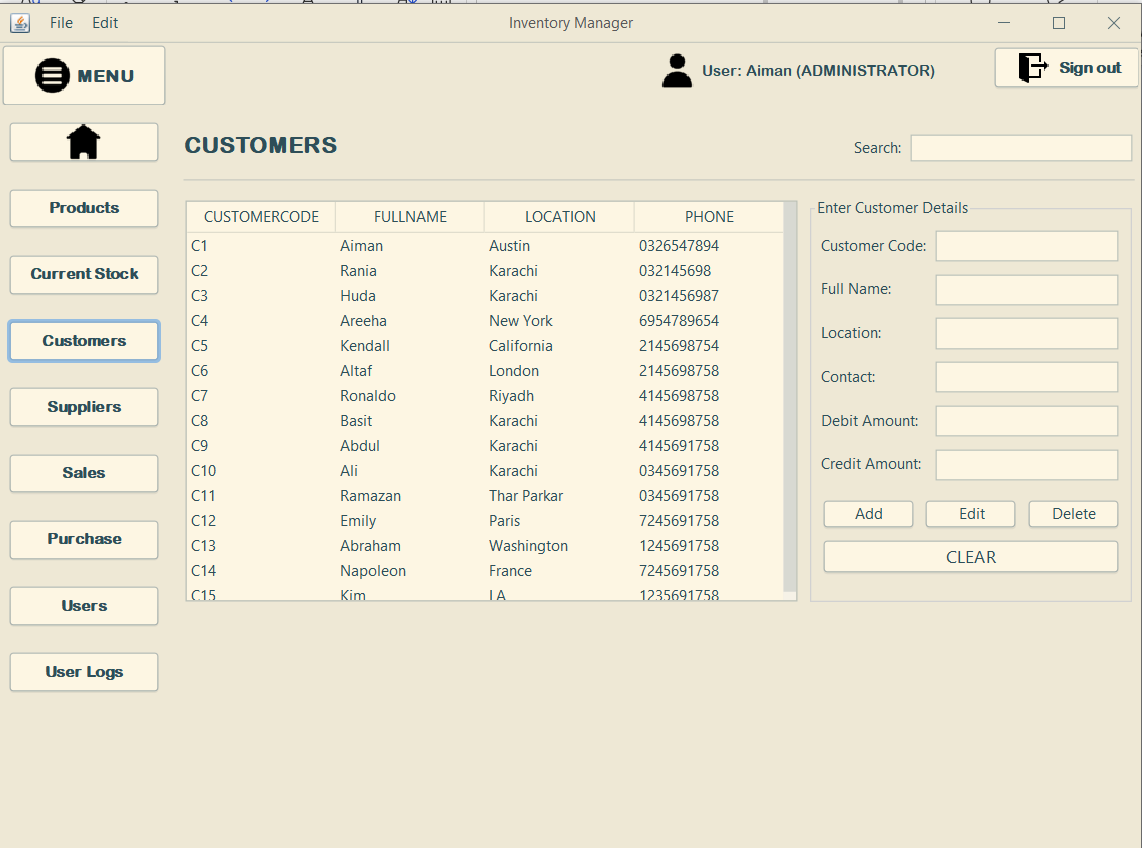
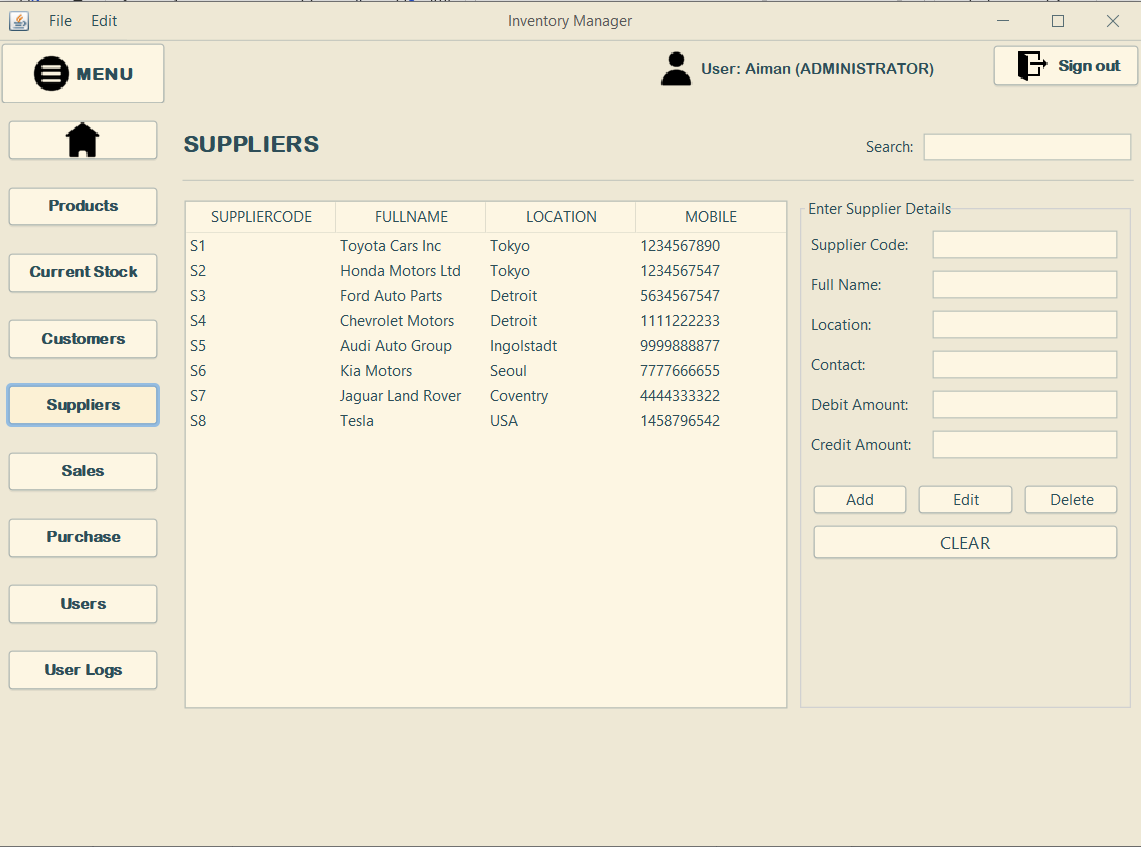
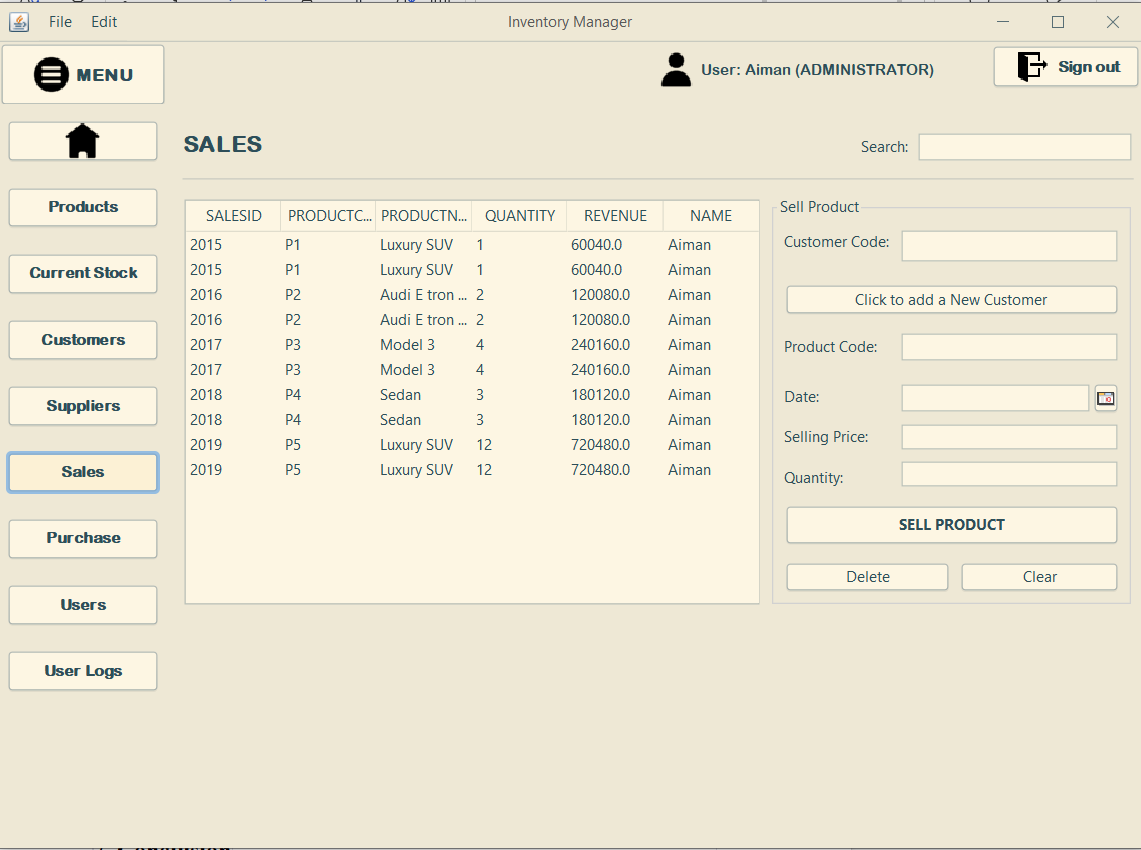
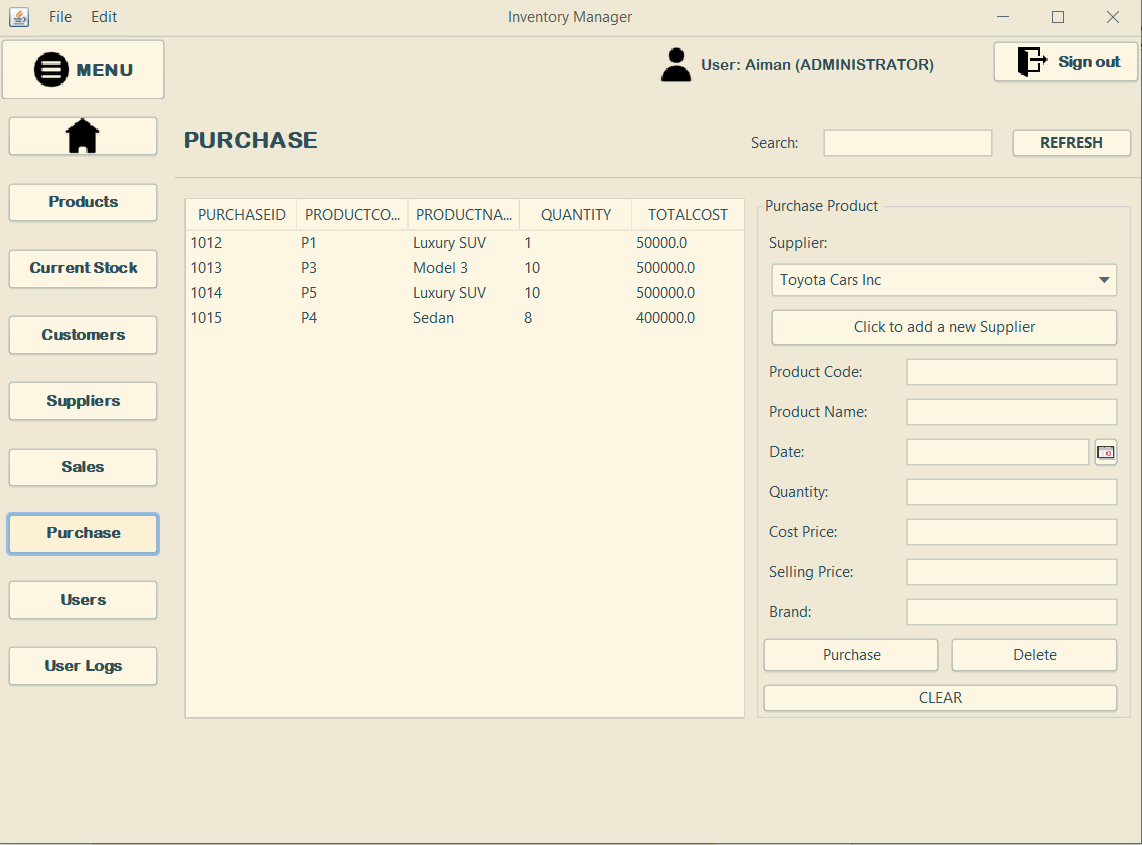
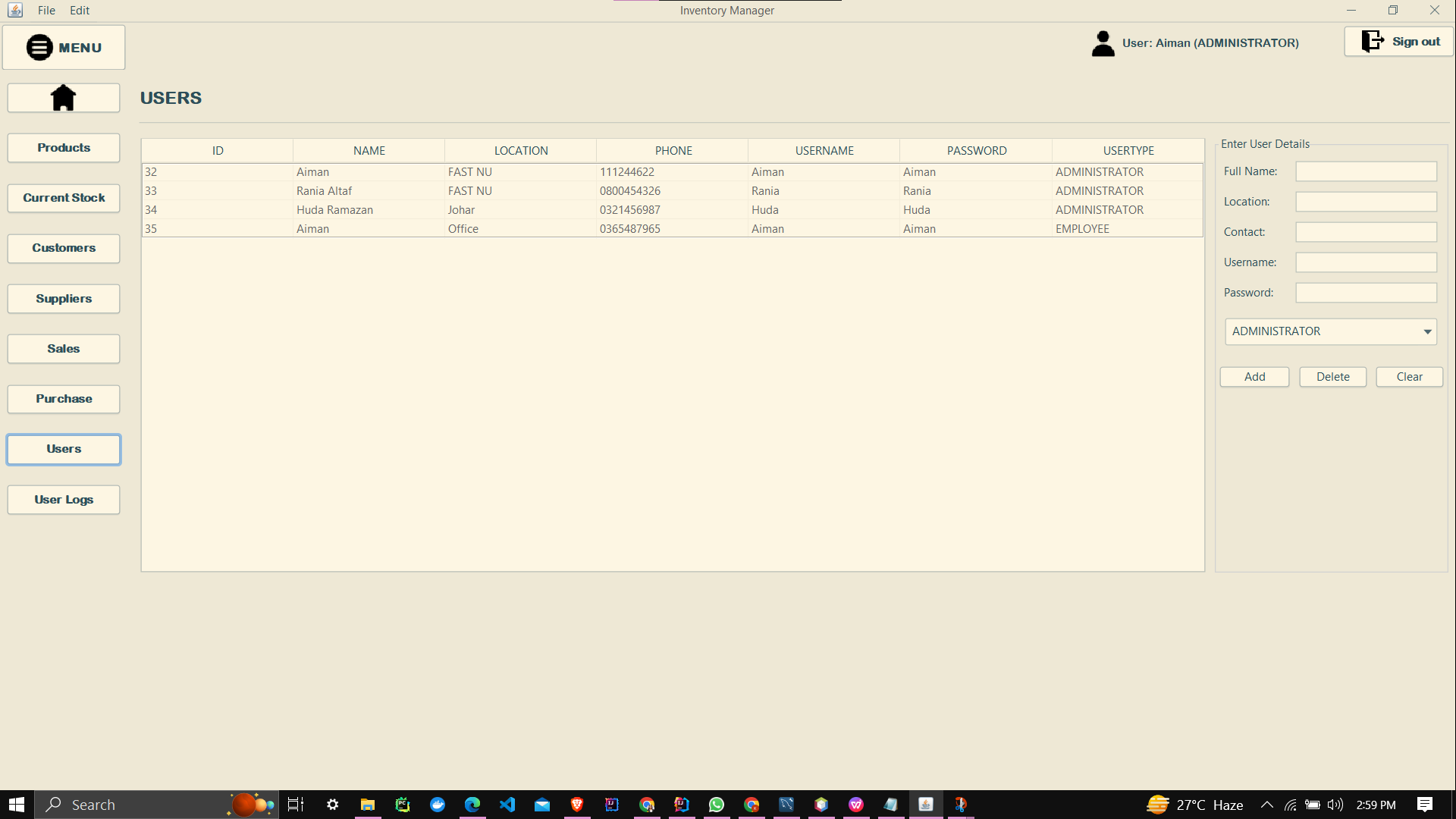
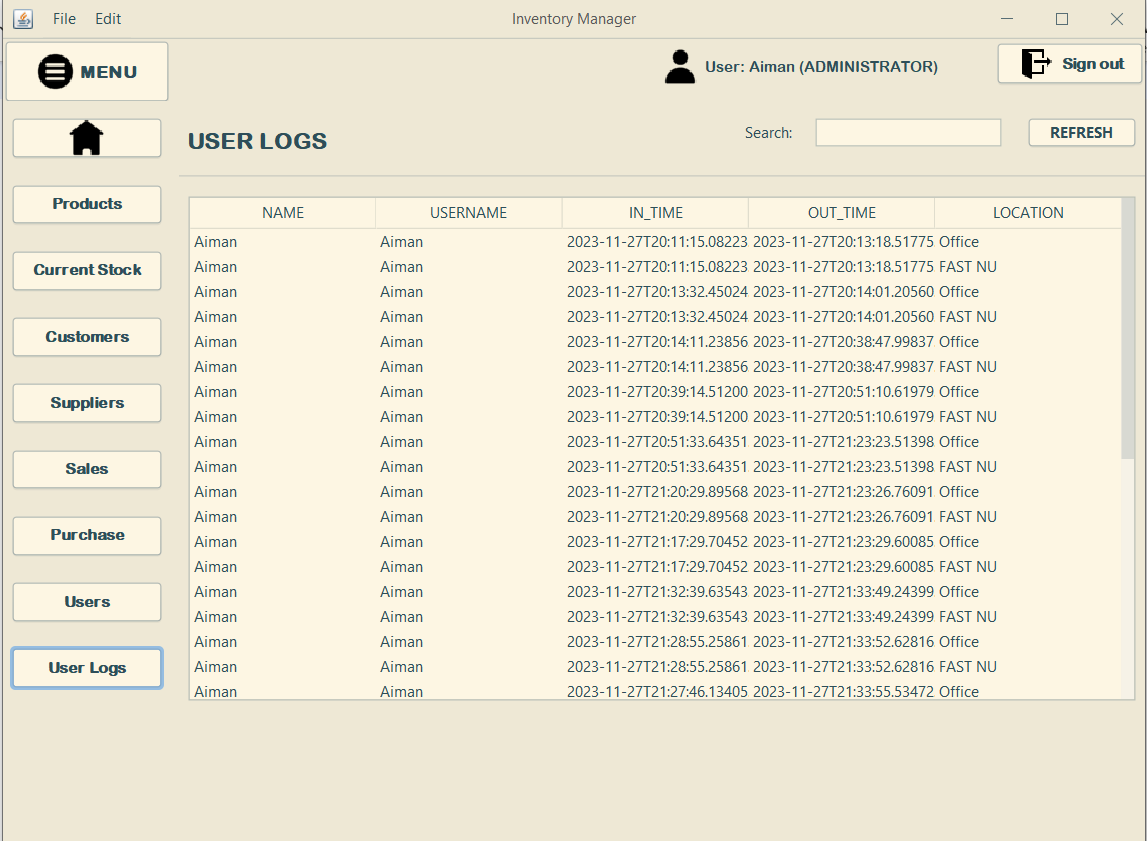
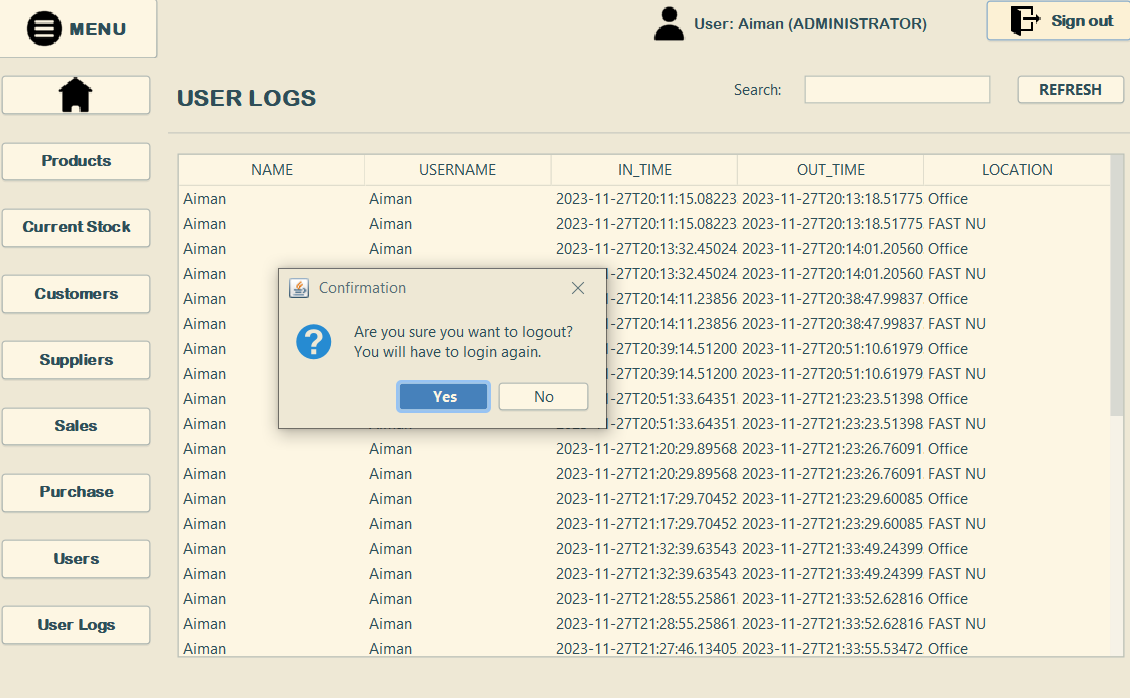
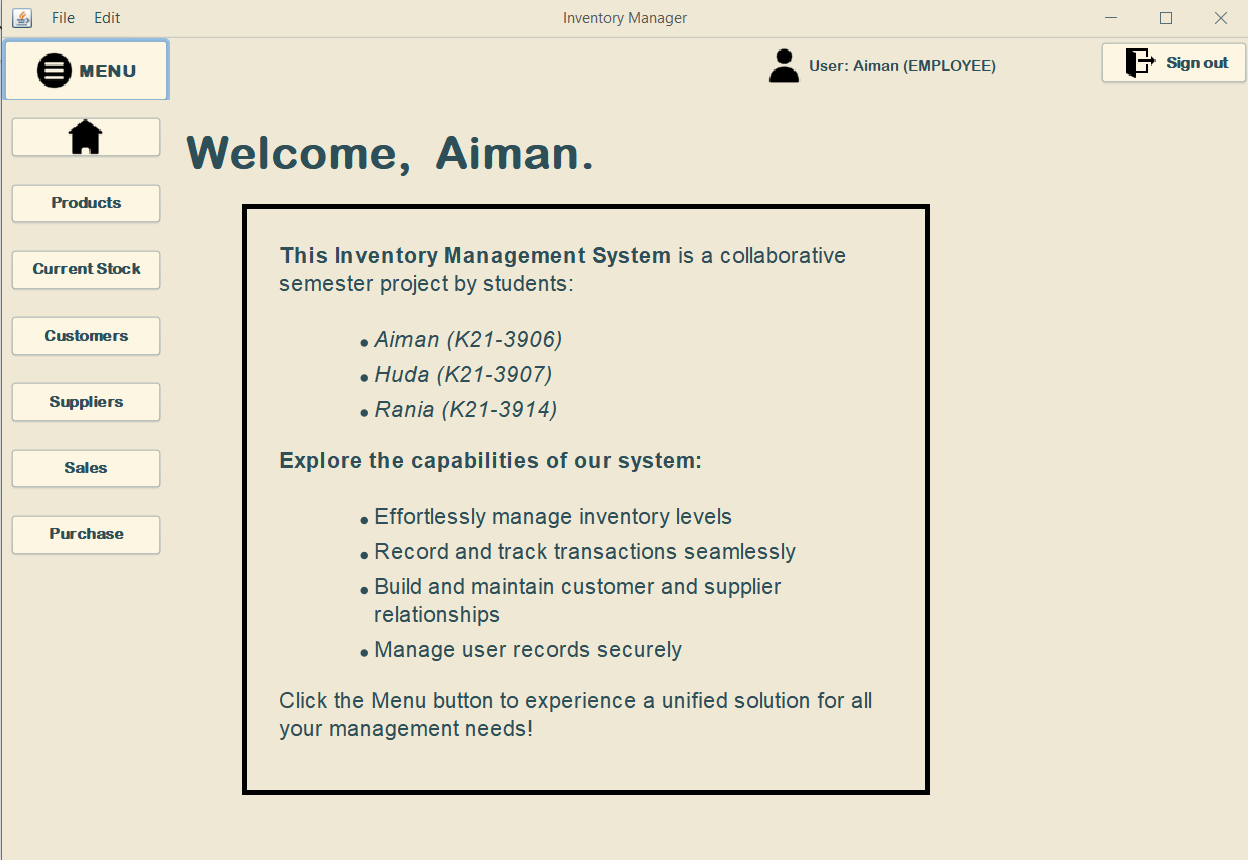
* Admins have the capability to manage user accounts.

### **5.6 Reporting**

* Generates reports on inventory status, transactions, and user activities.

## **6. GUI Design**

The GUI is developed using the Swing framework, providing an intuitive and user-friendly interface for efficient navigation and interaction. Attached are screenshots of the working project for reference.

1.   
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
   If you logged in as an employee then the options to see user and user log are no longer available.  
     
     
     
     
     
     
     
   ****Conceptual Gems****

****Java Swing:****

* + ****Description:**** Java Swing is a GUI toolkit for Java. In our project, we utilized Java Swing extensively to create a user-friendly and interactive graphical user interface (GUI) for the Inventory Management System. Swing components such as JFrame, JPanel, JButton, JTable, and more were employed to design the various screens and functionalities.
  + ****Implementation:**** The entire front-end of the application is built using Java Swing. This includes the main dashboard, product management screens, customer management screens, and other user interfaces. Buttons, text fields, tables, and other Swing components were used to facilitate user interactions.

****JDBC API:****

* + ****Description:**** JDBC (Java Database Connectivity) is a Java-based data access technology that allows Java applications to interact with a relational database. In our project, JDBC was employed to establish a connection with the MySQL database and perform database operations.
  + ****Implementation:**** We used JDBC to connect to the MySQL database where information about products, customers, suppliers, and transactions is stored. SQL queries were executed to retrieve, insert, update, and delete data as needed. This ensures the application interacts seamlessly with the underlying database.

****Error Handling:****

* + ****Description:**** Error handling is a crucial aspect of software development to ensure the application gracefully handles unexpected situations and provides meaningful feedback to the user.
  + ****Implementation:**** Throughout the project, comprehensive error handling mechanisms have been implemented. This includes validating user input to prevent invalid data entry, handling database connection errors, and displaying user-friendly error messages in case of unexpected issues. This enhances the robustness and user experience of the application.

****Vectors:****

* + ****Description:**** Vectors are dynamic arrays in Java that can grow or shrink in size. They are part of the Java Collections Framework and provide flexibility in handling data collections.
  + ****Implementation:**** Vectors were used in our project to manage dynamic lists of items, such as the list of products, customers, and suppliers. Vectors dynamically adjust their size to accommodate new data entries, ensuring efficient storage and retrieval of information.

****Runnable Interface:****

* + ****Description:**** The Runnable interface is used for multi-threading in Java. It allows a class to be instantiated and executed by a thread.
  + ****Implementation:**** Multi-threading was employed in specific functionalities of the application, such as logging user activity and updating real-time data. The Runnable interface was implemented to run these tasks concurrently without affecting the main application's performance.

By incorporating these concepts, our Inventory Management System not only leverages the power of Java Swing for a rich user interface but also ensures robust data management, effective error handling, and improved performance through multi-threading. These elements collectively contribute to a comprehensive and efficient software solution.

## **7. Conclusion**

The Inventory Management System offers a robust solution for small to mid-sized stores, enhancing efficiency in inventory control and transaction management. The system's user-friendly interface and comprehensive features contribute to seamless business operations.

## **8. Future Enhancements**

Potential future enhancements may include:

* Integration with bar-code scanners for faster product entry.
* Implementation of alerts for low-stock products.
* Integration with online platforms for e-commerce functionality.

## **9. Acknowledgments**

We acknowledge the contributions of the development team and express gratitude to all stakeholders involved in the project.

## **10. References**

* [Oracle Documentation for Java and JDBC](https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/).
* [MySQL Documentation for Database Management](https://dev.mysql.com/doc/).  
  [Documentation for java Swing](https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html)

窗体顶端

窗体底端