

# Smart Inventory and Supply Chain Management System (SIMS)

## Project Report

Project Title: Smart Inventory and Supply Chain Management System

### Origin of the Idea

We came up with the idea based on the growing need for businesses to optimize inventory processes and reduce waste. The concept was inspired by real-world challenges in supply chain management, such as demand forecasting and expiration tracking. We referred to resources like Java programming tutorials, data structure textbooks, and websites like GeeksforGeeks and Oracle's Java documentation for implementation details.

### Project Overview

The Smart Inventory and Supply Chain Management System (SIMS) is a Java-based application designed to streamline inventory and supply chain operations for businesses. The system provides robust features like inventory tracking, expiration alerts, stock management, daily sales reporting, and user support functionalities.

### Five Core Tasks of the Project

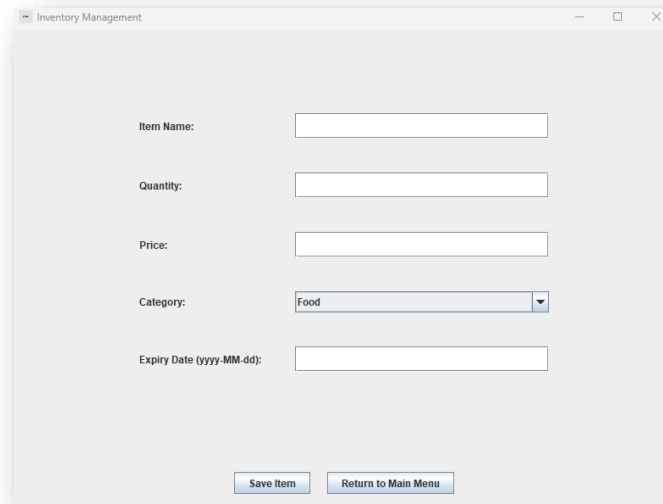
#### 1. Login System:

- Handles user authentication by validating usernames and passwords securely.



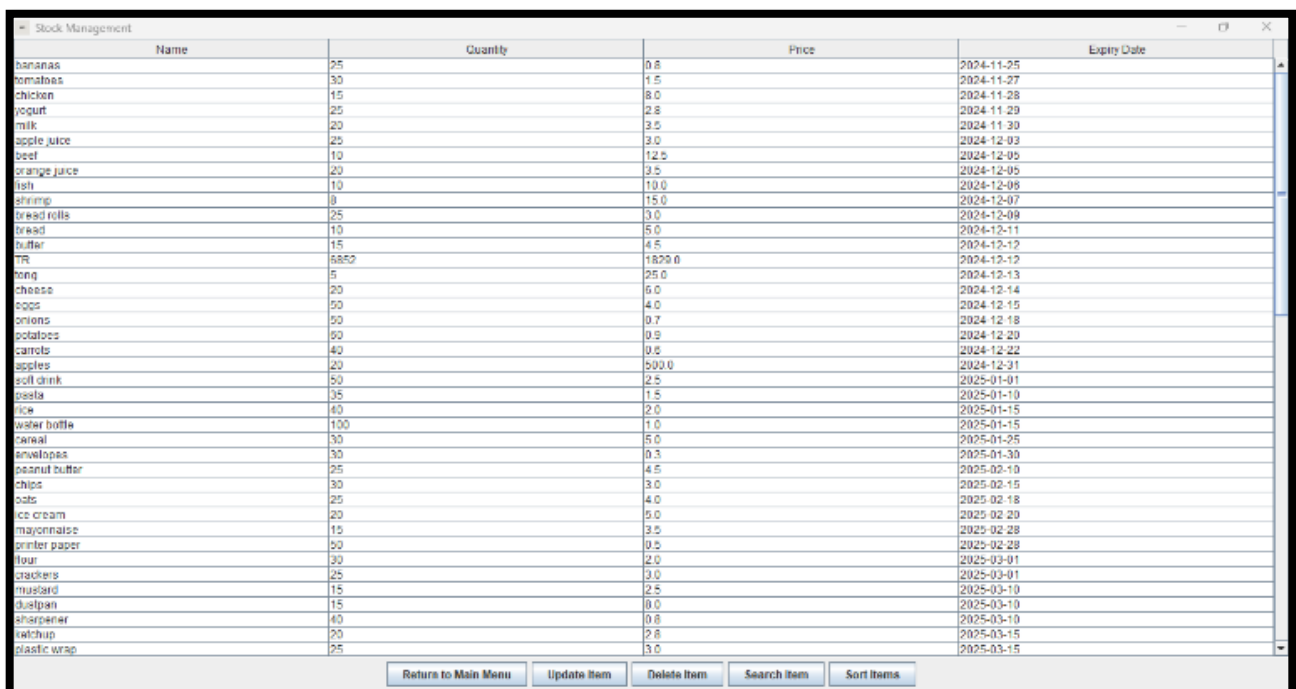
## 2. Inventory/Stock Management:

- Enables searching, sorting, and updating inventory items efficiently.
- Includes functionality to delete items and save changes to persistent storage.
- Each item has properties such as name, quantity, price, and expiration date.



A screenshot of a software window titled "Inventory Management". It contains a form with the following fields and controls:

- Item Name:** A text input field.
- Quantity:** A text input field.
- Price:** A text input field.
- Category:** A dropdown menu with "Food" selected.
- Expiry Date (yyyy-MM-dd):** A text input field.
- Buttons:** "Save Item" and "Return to Main Menu" at the bottom.



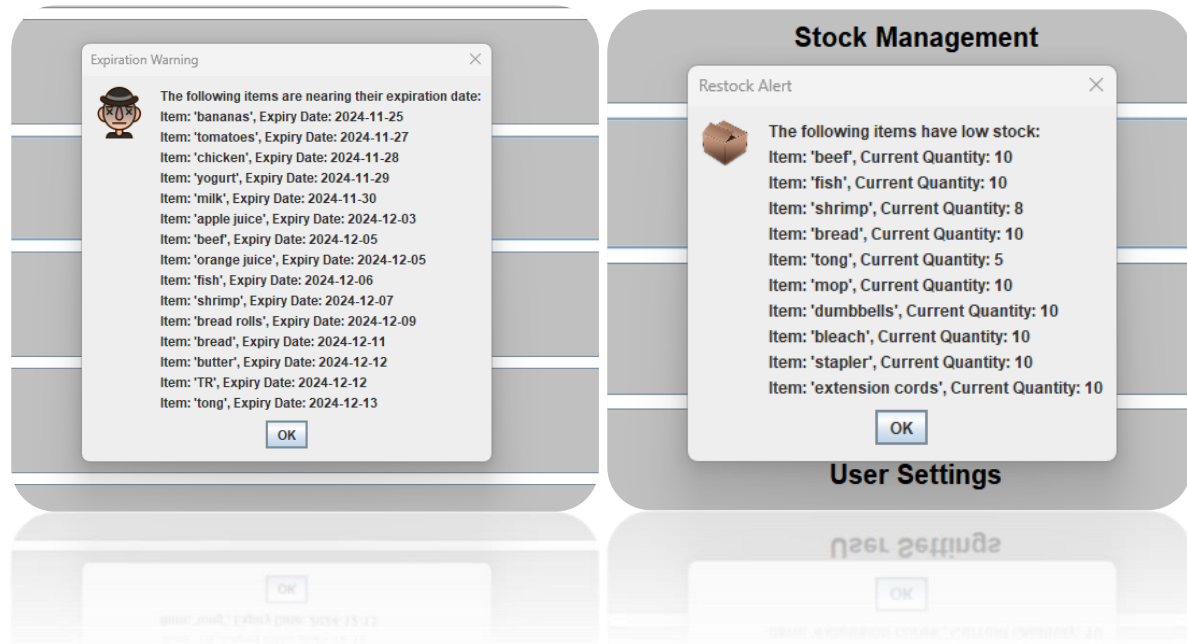
A screenshot of a software window titled "Stock Management" displaying a table of inventory items. The table has five columns: Name, Quantity, Price, and Expiry Date. The data is as follows:

Name	Quantity	Price	Expiry Date
bananas	25	0.8	2024-11-25
tomatoes	30	1.5	2024-11-27
chicken	15	8.0	2024-11-28
yogurt	25	2.8	2024-11-29
milk	20	3.5	2024-11-30
apple juice	25	3.0	2024-12-03
beef	10	12.5	2024-12-05
orange juice	20	3.5	2024-12-05
fish	10	10.0	2024-12-05
shrimp	9	15.0	2024-12-07
bread rolls	25	3.0	2024-12-09
bread	10	5.0	2024-12-11
butter	15	4.5	2024-12-12
TB	5850	1820.0	2024-12-12
long	5	25.0	2024-12-13
cheese	20	6.0	2024-12-14
eggs	50	4.0	2024-12-15
onions	50	0.7	2024-12-18
potatoes	50	0.9	2024-12-20
carrots	40	0.6	2024-12-22
apples	20	500.0	2024-12-31
selt drink	50	2.5	2025-01-01
pasta	35	1.5	2025-01-10
rice	40	2.0	2025-01-15
water bottle	100	1.0	2025-01-15
canal	30	5.0	2025-01-25
avocadoes	30	0.3	2025-01-30
peanut butter	25	4.5	2025-02-10
chips	30	3.0	2025-02-15
oats	25	4.0	2025-02-18
ice cream	20	5.0	2025-02-20
mayonnaise	15	3.5	2025-02-28
printer paper	50	0.5	2025-02-28
flour	30	2.0	2025-03-01
crackers	25	3.0	2025-03-01
mustard	15	2.5	2025-03-10
dishpan	15	0.0	2025-03-10
sharpener	40	0.8	2025-03-10
ketchup	20	2.8	2025-03-15
plastic wrap	25	3.0	2025-03-15

At the bottom of the window are four buttons: "Return to Main Menu", "Update Item", "Delete Item", and "Sort Items".

### 3. Expiration and Stock Alerts:

- Automatically checks for items nearing expiration (within 3 days) and low stock levels (below 10).
- Alerts users via popup messages.



### 4. Daily Sales Reporting:

- Tracks sales transactions and calculates total revenue.
- Generates a text-based daily sales report and saves it to a file for future reference.

Daily Sales Report

Item Name	Quantity Sold	Price per Unit	Total Price
apples	5	500.0	2500.0
pens	2	0.5	1.0
tea	12	4.0	48.0

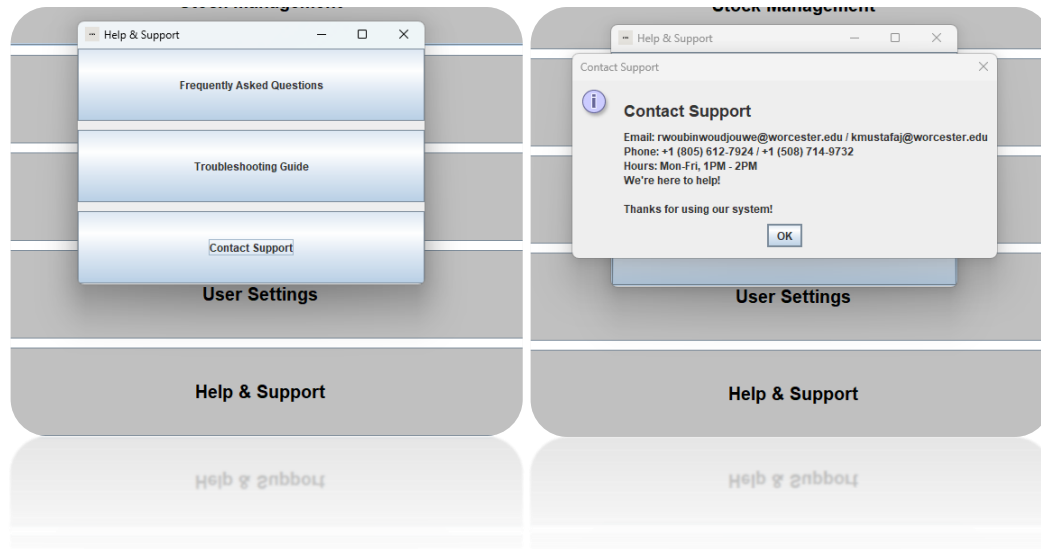
**Total Revenue: \$2549.00**

Buttons: Add Sale, Save Report, Refresh, Back to Stock Management

**Daily Revenue**

## 5. Help & Support:

- Includes a dedicated module with FAQs, troubleshooting guides, and contact support information.



## Data Structures Used

### 1. HashMap:

- Used for storing and retrieving inventory items efficiently.
- Provides constant-time complexity for lookups.

### 2. Linked List:

- Utilized for dynamically managing sales records in the daily reporting module.

### 3. ArrayLists:

- Used for dynamically managing inventory items

## Algorithms and Methods

### 1. Sorting:

- Implemented sorting functionality for inventory items based on user-selected criteria (e.g., name, quantity, and price).

### 2. Searching:

- Developed a search algorithm to locate specific items in the inventory quickly.

### 3. Recursive Parsing:

- Used for validating and parsing expiration dates.

### Object-Oriented Design

- Classes:
  - **CheckExpiration**: For tracking and alerting expiration-related issues.
  - **StockManagementGUI**: For managing inventory operations like search, update, and delete.
  - **Login**: For user authentication functionality.
  - **HelpAndSupportGUI**: For providing user guidance and troubleshooting.
  - **DailyReportGUI**: Handles sales tracking and reporting via a graphical interface.
  - **MainMenuGUI**: Provides a central navigation hub for all functionalities.
- Encapsulation:
  - Ensures data integrity by exposing item properties through getters and setters.
- Inheritance:
  - GUI components inherit properties and methods from Java's Swing library.

### Changes from Proposal

Initially, the proposal included supplier management and advanced demand forecasting features. Due to time constraints, these features were deferred to future versions. Additionally, we initially planned to include a live supply chain tracker but dropped it due to time constraints and the complexity of implementing real-time updates within our timeline. Also, we made adjustments to the data structures used during the development phase. For instance, while the initial proposal suggested using Priority Queues to manage restocking alerts, we switched to ArrayLists due to how some functionalities needed to work, such as easier iteration and dynamic updates. This change simplified the implementation while still meeting the system's requirements.

### Challenges and Solutions

1. Challenge: Implementing efficient file handling for data persistence.
  - Solution: Leveraged Java's serialization and buffered reading/writing mechanisms.
2. Challenge: Creating an intuitive and responsive GUI.
  - Solution: Utilized Java Swing's layout managers and event-driven programming.
3. Challenge: Generating accurate alerts for expiration and low stock.
  - Solution: Used LocalDate and ArrayLists for real-time calculations and prioritization.

## Conclusion and Future Work

### Conclusion:

The project successfully demonstrates a robust inventory management system leveraging data structures, algorithms, and GUI design. The implementation addresses key challenges businesses face in managing inventory and generating actionable insights.

### Future Work:

1. Integrate advanced demand forecasting algorithms using machine learning.
2. Transition to a cloud-based database for scalability and remote access.
3. Enhance the user interface with modern frameworks like JavaFX.
4. Incorporating the ability to generate reports in PDF format for improved accessibility and sharing.

## Work Distribution

**Rick:** Primarily worked on the GUI, including the InventoryManagementGUI, Login, and CheckExpiration classes.

**Klea:** Focused on implementing functionality for saving and retrieving data into/from files.

**Both Members:** Collaborated on the Inventory and Stock Management implementations, thoroughly tested and debugged the system, and jointly prepared the presentation slides and the project report.

## References/citations

1. Oracle Java Documentation: <https://docs.oracle.com/javase/>
2. Java Swing Tutorials: <https://www.javatpoint.com/java-swing>
3. Data Structures and Algorithms in Java by Robert Lafore.
4. Data Structure Zybooks
5. NAIR, P. S. (2017). Java programming fundamentals: Problem solving through object-oriented analysis and design. CRC Press.
6. W3schools.com. W3Schools Online Web Tutorials. (n.d.). <https://www.w3schools.com/java/default.asp>