GALGOTIAS UNIVERSITY

Grocery Store Management System

Team Members

Aman sharma	Leader
Yash prajapati	Member
Manish yadav	Member
Sanket Kumar Maurya	Member

Introduction

This project is a Command-Line based Grocery Store Management System developed using the Java programming language. It is designed to simulate the basic functionality of a grocery store where users can manage a list of grocery items.

The system uses a custom linked list to store and manage items, and it demonstrates the practical use of Object-Oriented Programming (OOP), File Handling, and User Input Handling in Java.

This application is completely text-based and runs on the command line, providing a menu-driven interface for ease of use.

Objective

✅ Provide a Basic Inventory System for Groceries

- Create a simple, text-based system to manage grocery items like name, quantity, and price.
- Allow operations such as add, delete, view, and calculate total.

✅ Demonstrate File Handling & Linked List in Java

- Implement a custom singly linked list to manage items dynamically.
- Use file handling to save and load data from a .txt file for data persistence.

✅ Learn Real-World Use of Java OOP Concepts

- Apply key OOP concepts like encapsulation and modularity.
- Structure code using separate classes (GroceryItem, GroceryStore, GroceryStoreApp) for better readability and maintenance.

Tools & Technologies Used

💻 Programming Language: Java

- The entire project is written in Java, a robust and object-oriented programming language.
- Used core Java features like classes, objects, constructors, and user input handling.

🧠 Concepts Implemented

Classes & Objects:

- GroceryItem class for each item (name, quantity, price, total, and next).
- GroceryStore class for managing operations (add, delete, display, total, file I/O).
- GroceryStoreApp class contains the main method and user interface logic.

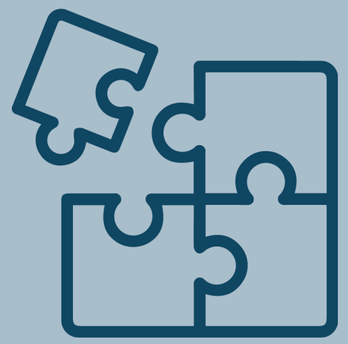
Exception Handling:

- try-catch blocks are used to manage input errors (e.g., non-numeric quantity/price) and file-related exceptions.
- Ensures that the program doesn't crash unexpectedly.

Class Design – GroceryItem (Detailed Version)

📦 Attributes (Instance Variables)

String name; // Name of the grocery item (e.g., Milk)
int quantity; // Quantity purchased (e.g., 2)
double price; // Price per unit (e.g., 30.5)
GroceryItem next; // Reference to the next item in the list



🧮 Constructor

- public GroceryItem(String name, int quantity, double price) {
- this.name = name;
- this.quantity = quantity;
- this.price = price;
- this.next = null;
- }

💰 Method: getTotalPrice()

- public double getTotalPrice() {
- return quantity * price;
- }

Data Flow / Architecture (Detailed Version)

🔍 Explanation of Flow

- ✅ 1. User Input
- The user interacts through a menu shown in the terminal.
- Options like Add, Delete, Display, Calculate, and Exit.
- ✅ 2. GroceryStoreApp – Main Class
- Takes input using Scanner
- Controls the flow using switch-case
- Calls the appropriate methods from the GroceryStore class
- ✅ 3. GroceryStore – Core Logic
- Contains methods like:
- addItem()
- deleteItem()
- displayItems()
- calculateTotal()
- saveToFile() and loadFromFile()
- Manages the linked list of items using GroceryItem references
- ✅ 4. GroceryItem – Node Class
- Each object stores:
- Item name, quantity, price
- Calculates total price
- Holds a reference to the next item in the list
- ✅ 5. File I/O
- Data is saved to grocery_items.txt when the user exits
- On start, the app reads the file and reloads the list

Sample Run (Screenshots/Code Snippets)

Sample code:-

```
import java.util.Scanner;

public class GroceryStoreApp {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
GroceryStore store = new GroceryStore();
String filename = "grocery_items.txt";

store.loadFromFile(filename); // Load saved items

while (true) {
System.out.println("\nGrocery Store Menu:");
System.out.println("1. Add Item");
System.out.println("2. Delete Item");
System.out.println("3. Display Items");
System.out.println("4. Calculate Total");
System.out.println("5. Save and Exit");
System.out.print("Choose an option: ");

int option;
try {
option = Integer.parseInt(scanner.nextLine());
} catch (NumberFormatException e) {
System.out.println("Please enter a valid number.");
continue;
}
```

```
switch (option) {
case 1:
System.out.print("Enter item name: ");
String name = scanner.nextLine();
System.out.print("Enter quantity: ");
int quantity = Integer.parseInt(scanner.nextLine());
System.out.print("Enter price per unit: ");
double price = Double.parseDouble(scanner.nextLine());
store.addItem(name, quantity, price);
break;
case 2:
System.out.print("Enter item name to delete: ");
name = scanner.nextLine();
store.deleteItem(name);
break;
case 3:
store.displayItems();
break;
case 4:
store.calculateTotal();
break;
case 5:
store.saveToFile(filename);
System.out.println("Goodbye!");
return;
default:
System.out.println("Invalid option. Try again.");
}
}
}
}
```

Sample code:-

```
Grocery Store Menu:
1. Add Item
2. Delete Item
3. Display Items
4. Calculate Total
5. Save and Exit
Choose an option: 1
Enter item name: daal
Enter quantity: 1
Enter price per unit: 80

Grocery Store Menu:
1. Add Item
2. Delete Item
3. Display Items
4. Calculate Total
5. Save and Exit
Choose an option: 4
Total bill: 110.00
```

Conclusion

🔑 Key Accomplishments

- 📦 Developed a working inventory/cart system for grocery items.
- 🔗 Implemented a custom singly linked list without using built-in data structures.
- 💾 Used File Handling (FileWriter, BufferedReader) to store and retrieve data, ensuring that the user's grocery list is preserved across sessions.
- 🧱 Applied Object-Oriented Programming (OOP) principles:
  - Encapsulation
  - Modularity
  - Reusability
- 📋 Created a user-friendly, menu-driven console interface for seamless interaction.

Thank you :-

I appreciate your attention and hope you found this project presentation informative and insightful.

This Grocery Store Management System was created with the aim of learning and demonstrating real-world use of Java programming concepts, especially:

- **Object-Oriented Programming (OOP)**
- **Linked List Implementation**
- **File Handling**
- **User Interaction via Console**

git hub link :- **https://github.com/aman2005-sudo/Grocery-store.git**