Your Partner for Excellence

Texas

College of Management & IT

Java Mid Term Project

Store Management Software

Submitted By :

Aayush Lamichhane

LC ID: LC00017001850

BIT 1st Year 2nd Semester

Table of Content

Github Link : https://github.com/AaaayushXD/java_midTerm_project

Proposal

Store Management System

Abstract:
The store management system software is a software designed for the retailer to manage their stock and inventory. Manually storing stock information in pen and paper is very difficult and takes a lot of time. This software makes that job easier. It stores all the stock information in a file and can be easily updated by the employee. Store manager can easily check the stock, manage it accordingly and update it if any changes has occurred. It is easy to use console based system. It also keeps the logs of any chances made in the software.

Objective:
Its primary purpose is to make the stock management easier and bring profit to the store. It will keep track of the stocks which will make it easier for the employee to add or remove accordingly. It will also maintain a log of every sales made in the store. It also keeps the log of any changes made by the user in the software. Finally, it will be console based software, so it is user friendly and easy to use.

Tools:

- Programming Language: This software will be built entirely using Java and its concepts like OOPs and File Handling concept.
- Integrated Development Environment (IDE) : It will be built in Intellij IDEA with jdk 1.8.
- Database: The project output will be stored in text file in proper order for user to view the stocks, logs and balance.
- Version Control: Git will be used for keeping track of changes.
- Storage: It will be stored in GitHub to make it easily available and update easily.
- Documentation: A detailed documentation of the project will be created to make it easy for the user to interact with the software.

Introduction

Store management software for managing store inventory. The store contains a collection of things that are hard to keep track of. Nowadays, all stores store their digital data with ease. When there are many items, it can be difficult to remember how many items are in stock and what they cost. This program is user-friendly and will assist reps in tracking items in their inventory.

Features and Functionality:

This software stores the name of product, total quantity in stock, their price and the date it got added or updated.

1. Add new items: We can easily add a new item to the list of items with its name, price, quantity and date.

2. Update an item: We can change the quantity of product, or update price of product.

3. Buy an item: We can buy item from other vendors or manufacturer if we run out of stock. It will increase the quantity of item in the list.

4. Sell an item: We can sell item to customer. It decrease the quantity from our inventory and increases store balance.

5. View items

6. Remove an item

7. Item’s are stored in a file.

8. Changes are stored in log file.

Implementation Detail:

In this software, I have implemented basic concepts of java programming language. I used four principles of OOP ie. Inheritance, Polymorphism, Abstraction and Encapsulation. I used list and arraylist libraries to work with elements and manipulate this data. I also use other concepts like file handling and exception handling.

Main file :

Main.java × Items.java × LogManager.java × FileManager.java × Output.java × Store.java × StoreManager.java

```java
import java.io.IOException;

AaaayushXD
public class Main {
    AaaayushXD
    public static void main(String[] args) {
        try {
            Store store = new Store();
            store.features();
        } catch (IOException e) {
            System.out.println("Something went wrong. Please try again" );
        }
    }
}
```

Items class file

```java
10 usages  AaaayushXD
public class Items {
    no usages  AaaayushXD
    Items() {}
    no usages  AaaayushXD
    Items(double quantity) { this.quantity = quantity; }
    2 usages  AaaayushXD
    Items(String name, double quantity, double price, String date) {
        this.name = name;
        this.quantity = quantity;
        this.price = price;
        this.date = date;
    }

    10 usages  AaaayushXD
    public String getName() { return name; }

    no usages  AaaayushXD
    public void setName(String name) { this.name = name; }

    8 usages  AaaayushXD
    public double getQuantity() { return quantity; }

    no usages  AaaayushXD
```

```java
    public void setQuantity(double quantity) { this.quantity = quantity; }

    8 usages  AaaayushXD
    public double getPrice() { return price; }

    no usages  AaaayushXD
    public void setPrice(double price) { this.price = price; }

    3 usages  AaaayushXD
    public String getDate() { return date; }

    no usages  AaaayushXD
    public void setDate(String date) { this.date = date; }

    3 usages
    private String name;
    4 usages
    private double quantity;
    3 usages
    private double price;
    3 usages
    private String date;
}
```

Store Manager Class

```java
public abstract class StoreManager extends FileManager {


    9 usages   1 implementation   AaaayushXD
    public abstract void title();

    1 usage   1 implementation   AaaayushXD
    public abstract void menuPage();


    no usages   AaaayushXD
    public void setBankBalance(double bankBalance) { this.bankBalance = bankBalance; }

    3 usages
    private double bankBalance = getBankBalance();


    /// Features
    1 usage   AaaayushXD
    public void addItems() {
        featureInfo( title: "Add Items",  info: "It is used to add new items to our stock");
        userInput();
        outputText("Item added.");
    }
```

```java
    public void removeItem() {
        featureInfo( title: "Remove item",  info: "It is used to remove any existing item from the stock");
        Scanner scanner = new Scanner(System.in);
        System.out.println("Please enter the name of product you want to remove: ");
        String name = scanner.nextLine();
        String date = todayDate();

        //removing from list
        loadFromFile();
        boolean itemRemoved = itemsList.removeIf(item -> item.getName().equalsIgnoreCase(name));
        if (itemRemoved) {
            String output = "Item deleted successfully: " + name;
            outputText(output);
            updateLog(name,  action: "removed", date);
        } else {
            String output = "Item not found";
            outputText(output);
        }

        //removing from file
        removeFromFile(name);

    }
```

Main.java × Items.java × LogManager.java × FileManager.java × Output.java × Store.java × StoreManager.java ×

1 usage AaaayushXD

```java
public void updateItem() {
    featureInfo( title: "Update item",  info: "It is used to update price and quantity of any existing item from the stoc
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter name of item to update: ");
    String name = scanner.nextLine();
    String date = todayDate();
    loadFromFile();
    for (int i = 0; i < itemsList.size(); i++) {
        Items item = itemsList.get(i);
        if (item.getName().equalsIgnoreCase(name)) {
            double oldQuantity = item.getQuantity();
            double oldPrice = item.getPrice();
            itemsList.remove(item);
            removeFromFile(item.getName());
            System.out.println("1. Quantity \n2. Price ");
            int choice = scanner.nextInt();
            scanner.nextLine();
            switch (choice) {
                case 1:
                    System.out.println("Enter updated quantity: ");
                    double quantity = scanner.nextDouble();
                    scanner.nextLine();
```

Main.java × Items.java × LogManager.java × FileManager.java × Output.java × Store.java × StoreManager.java ×

```java
switch (choice) {
    case 1:
        System.out.println("Enter updated quantity: ");
        double quantity = scanner.nextDouble();
        scanner.nextLine();
        addToListAndFile(name, quantity, oldPrice, date);
        String output = name + "'s quantity is updated from " + oldQuantity + " to " + quantity + " on " + date;
        outputText(output);
        break;
    case 2:
        System.out.println("Enter new Price: ");
        double price = scanner.nextDouble();
        scanner.nextLine();
        addToListAndFile(name, oldQuantity, price, date);
        String outputPrice = name + "'s price is updated from Rs" + oldPrice + " to Rs" + price + " on " + date;
        outputText(outputPrice);
        break;
    default:
        String outputInvalid = "Invalid choice. Choose 1 or 2.";
        outputText(outputInvalid);
}
break;
}
```

Main.java × Items.java × LogManager.java × FileManager.java × Output.java × Store.java × StoreManager.java ×

```java
    public void deleteFile() {
        featureInfo( title: "Delete file",  info: "It is used to delete file which holds store's stock information.");
        String fileName = "store.txt";
        File file = new File(fileName);
        if (!file.exists()) {
            outputText("File doesn't exist.");
        }
        String date = todayDate();
        outputText("Successfully deleted the file :" + fileName);
        updateLog(fileName,  action: "deleted", date);
        file.delete();
    }
    1 usage   AaaayushXD
    public void sellItem() {
        featureInfo( title: "Sell item",  info: "It is used to hold information of any product being sold.");
        Scanner scanner = new Scanner(System.in);
        System.out.println("Name of selling product: ");
        String name = scanner.nextLine();
        String date = todayDate();
        loadFromFile();

        for (int i = 0; i < itemsList.size(); i++) {
            Items item = itemsList.get(i);
            if (item.getName().equalsIgnoreCase(name)) {
```

Main.java × Items.java × LogManager.java × FileManager.java × Output.java × Store.java × StoreManager.java ×

```java
        for (int i = 0; i < itemsList.size(); i++) {
            Items item = itemsList.get(i);
            if (item.getName().equalsIgnoreCase(name)) {
                System.out.println("How many would you like: ");
                double quantity = scanner.nextDouble();
                scanner.nextLine();
                if (quantity <= item.getQuantity()) {
                    double cost = quantity * item.getPrice();
                    outputText("Great. It would cost you Rs " + cost);
                    double quantityLeft = item.getQuantity() - quantity;
                    itemsList.remove(item);
                    removeFromFile(name);
                    addToListAndFile(name, quantityLeft, item.getPrice(), date);
                    updateLog( name: quantity + " " + name,  action: " sold", date);
                    updateBalance(cost,  action: "added", date);
                } else if (item.getQuantity() == 0) {
                    outputText("Sorry, we are out of stock right now.");
                } else {
                    outputText("Sorry! Unfortunately we do not have enough in stock.");
                }
                break;
            }
```

```java
public void buyItem() throws IOException {
    featureInfo( title: "Restock item" ,  info: "It is used to hold information about restocked items to store");
    bankBalance = loadBalance();
    Scanner scanner = new Scanner(System.in);
    System.out.println("What would you like to buy again? ");
    String name = scanner.nextLine();
    String date = todayDate();
    loadFromFile();
    for(int i = 0; i<itemsList.size(); i++) {
```

```java
                updateBalance(totalCost,  action: "deducted", date);
            } else {
                outputText("Insufficient balance. Please try again.");
            }
            break;
        }
    }
}
```

1 usage AaaayushXD

```java
public void viewItem() {
    featureInfo( title: "View items",  info: "It is used to view all items from stock");
    loadFromFile();
    System.out.println("-------------Result---------------");
    System.out.println("----------------------------------");
    for(Items item: itemsList) {
        System.out.println("Product name: " + item.getName()
                + ", Quantity (in stock):  " + item.getQuantity()
                + ", Price (in NRs): " + item.getPrice()
                + ", Date (yyyy-mm-dd): " + item.getDate());
    }
    System.out.println("----------------------------------");
    System.out.println("----------------------------------");
}
```

```
1 usage   AaaayushXD
    public void exit() {
        featureInfo( title: "Exit",  info: "It is used to exit from the software");
        outputText("See you soon ! Bye");
    }


}
```

Store class

```java
public class Store extends StoreManager{

    9 usages   AaaayushXD
    public void title() {
        System.out.println("--------------------------------");
        System.out.println("Logged in as Aayush Lamichhane - LC00017001850");
        System.out.println("This is store's stock management software");
        System.out.println("--------------------------------");
    }
    1 usage   AaaayushXD
    public void menuPage() {
        System.out.println("-----------------------------");
        System.out.println("1. Add new items to your stock");
        System.out.println("2. Update any item from the stock.");
        System.out.println("3. Buy any item to add to your stock.");
        System.out.println("4. Sell any item from your stock");
        System.out.println("5. View all your items.");
        System.out.println("6. Remove any item from stock.");
        System.out.println("7. Delete file");
        System.out.println("8. Exit");
        System.out.println("-----------------------------");
    }

```

```java
    public void features()  throws IOException{
        int choice = 0;
        while(choice != 8) {
            title();
            menuPage();
            System.out.println("Please select any option from given list (1-8): ");
            Scanner scanner = new Scanner(System.in);
            choice = scanner.nextInt();
            scanner.nextLine();
            switch (choice) {
                case 1:
                    title();
                    addItems();
                    break;
                case 2:
                    title();
                    updateItem();
                    break;
                case 3:
                    title();
                    buyItem();
                    break;
```

```java
            case 4:
                title();
                sellItem();
                break;
            case 5:
                title();
                viewItem();
                break;
            case 6:
                title();
                removeItem();
                break;
            case 7:
                title();
                deleteFile();
                break;
            case 8:
                title();
                exit();
                break;
            default:
                System.out.println("Invalid input. Please select from given option only. (1-8). Thank you!!");
        }
    }
```

Output class file

```
1 usage   4 inheritors   AaaayushXD
public class Output {
    17 usages   AaaayushXD
    public void outputText(String output) {
        System.out.println("-------------Result---------------");
        System.out.println("----------------------------------");
        System.out.println(output);
        System.out.println("----------------------------------");
        System.out.println("----------------------------------");
    }

    8 usages   AaaayushXD
    public void featureInfo(String title,String info) {
        System.out.println("----------------------------------");
        System.out.println("This feature is called " + title);
        System.out.println(info);
        System.out.println("----------------------------------");
    }
}
```

File Manager class

```java
import java.io.*;
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

1 usage  2 inheritors  AaaayushXD
public class FileManager extends LogManager{

    21 usages
    List<Items> itemsList = new ArrayList<>();

    //to validate the date entered by the user
    6 usages  AaaayushXD
    public String todayDate() { return LocalDate.now().toString(); }

    /// to add new item to the array list and file
    5 usages  AaaayushXD
    public void addToListAndFile(String name, double quantity, double price, String date) {
        Items item = new Items(name, quantity, price, date);
        itemsList.add(item);
        saveToFile();
    }

```

```java
    1 usage  AaaayushXD *
    public void saveToFile() {
        try {
            File file = new File( s: "store.txt");
            if (!file.exists()) {
                file.createNewFile();
            }
            FileWriter fileWriter = new FileWriter(file,  b: true);
            BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);

            int index = itemsList.size() - 1;
            String fileLine = "Product name: " + itemsList.get(index).getName()
                    + ", Quantity (in stock):  " + itemsList.get(index).getQuantity()
                    + ", Price (in NRs): " + itemsList.get(index).getPrice()
                    + ", Date (yyyy-mm-dd): " + itemsList.get(index).getDate();
            bufferedWriter.write(fileLine);
            bufferedWriter.newLine();

            bufferedWriter.close();
            fileWriter.close();
        } catch (IOException e) {
            System.out.println("Something went wrong while saving data to file");
            e.printStackTrace();
        }
```

```java
public void removeFromFile(String name) {
    try{
        File file = new File( s: "store.txt");
        FileWriter fileWriter = new FileWriter(file);
        BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);
        for(Items item: itemsList) {
            if(item.getName().equalsIgnoreCase(name)) {
                String output = "Item successfully deleted: " +item.getName();
                outputText(output);
            }
            else {
                String outputLine = "Product name: " + item.getName()
                        + ", Quantity (in stock):  " + item.getQuantity()
                        + ", Price (in NRs): " + item.getPrice()
                        + ", Date (dd-mm-yyyy): " + item.getDate();
                bufferedWriter.write(outputLine);
                bufferedWriter.newLine();
            }
        }
        bufferedWriter.close();
        fileWriter.close();
    } catch (IOException e) {
        System.out.println("Couldn't remove item from the file.");
        e.printStackTrace();
```

```java
}
1 usage   AaaayushXD
public void userInput() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter name of product: ");
    String name = scanner.nextLine();
    System.out.println("Enter quantity of product: ");
    double quantity = scanner.nextDouble();
    System.out.println("Enter price (in NRs): ");
    double price = scanner.nextDouble();
    scanner.nextLine();
    String date = todayDate();

    addToListAndFile(name, quantity, price, date);
    updateLog(name, action: "added", date);
}
```

```java
public void loadFromFile() {
    try {
        File file = new File( s: "store.txt");
        if (!file.exists()) {
            file.createNewFile();
        }
        FileReader fileReader = new FileReader(file);
        BufferedReader bufferedReader = new BufferedReader(fileReader);
        String line;
        itemsList.removeAll(itemsList);
        while((line = bufferedReader.readLine()) != null) {
            String[] parts = line.split( s: ",");
            if(parts.length == 4) {
                String name = parts[0].split( s: ": ")[1];
                double quantity = Double.parseDouble(parts[1].split( s: ": ")[1]);
                double price = Double.parseDouble(parts[2].split( s: ": ")[1]);
                String date = parts[3].split( s: ": ")[1];
                Items item = new Items(name, quantity, price, date);
                itemsList.add(item);
            }
        }
        bufferedReader.close();
        fileReader.close();
    } catch (Exception e)  {
        System.out.println("Couldn't read from file. Something went wrong.");
        e.printStackTrace();
    }
```

Log Manager Class

```java
public class LogManager extends Output{
    1 usage   AaaayushXD
    public double getBankBalance() { return bankBalance; }

    no usages   1 override   AaaayushXD
    public void setBankBalance(double bankBalance) { this.bankBalance = bankBalance; }

    8 usages
    private double bankBalance ;
    6 usages   AaaayushXD *
    public void updateLog(String name, String action, String date) {
        try{
            File file = new File( s: "log.txt");
            if(!file.exists()) {
                file.createNewFile();
            }
            FileWriter fileWriter = new FileWriter(file,  b: true);
            BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);
            bufferedWriter.write( s: name + " is " + action + " on " +date + "\n");
            bufferedWriter.close();
            fileWriter.close();
        } catch (IOException e) {
            System.out.println("Something went wrong while updating the log file");
            e.printStackTrace();
        }
    }
```

```java
Main.java × Items.java × LogManager.java × FileManager.java × Output.java × Store.java × StoreManager.java ×
    public void updateBalance(double updatedAmount, String action, String date) {
        try {
            File file = new File( s: "transaction.txt");
            File tmpFile = new File( s: "tmp.txt");
            FileWriter fileWriter = new FileWriter(tmpFile);
            BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);
            bankBalance = loadBalance();

            if(action.equalsIgnoreCase( s: "added")) {
                bankBalance += updatedAmount;
                outputText("New amount = "+bankBalance);
            } else if(action.equalsIgnoreCase( s: "deducted")) {
                bankBalance -= updatedAmount;
                outputText("New Amount = " +bankBalance);
            }
            bufferedWriter.write( s: "Total Balance = Rs " + bankBalance);
            bufferedWriter.newLine();
            bufferedWriter.write( s: "Rs " +updatedAmount+ " is " + action + " on " + date);
            bufferedWriter.newLine();
            bufferedWriter.close();
            fileWriter.close();
            file.delete();
            tmpFile.renameTo(file);

        } catch (IOException e) {
            System.out.println("Something went wrong while updating balance.");
            e.printStackTrace();
```

```
2 usages   AaaayushXD
public double loadBalance() throws IOException {
    File orginalFile = new File( s: "transaction.txt");
    if(!orginalFile.exists()) {
        createTransactionFile();
    }
    FileReader fileReader = new FileReader(orginalFile);
    BufferedReader bufferedReader = new BufferedReader(fileReader);
    String line;
    double bankBalance = 1000.0;
    while((line = bufferedReader.readLine()) !=null) {
        if(line.startsWith("Total")) {
            String[] parts = line.split( s: "Rs ");
            bankBalance = Double.parseDouble(parts[1]);
            break;
        }
    }
    bufferedReader.close();
    fileReader.close();
    return bankBalance;
}
```

```
1 usage   AaaayushXD
public void createTransactionFile() {
    try {
        File file = new File( s: "transaction.txt");
        FileWriter fileWriter = new FileWriter(file);
        BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);
        bufferedWriter.write( s: "Total Balance = Rs "+ 1000.0);
        bufferedWriter.close();
        fileWriter.close();
    } catch (IOException e) {
        System.out.println("Something went wrong");
        e.printStackTrace();
    }
}
}
```

Summary

Store management software is designed to allow store staff to keep track of their inventory. The program is console-based and simple to utilize. It contains all the essential needs for administration. This computer program offers highlights like including things to records, upgrading things, buying/selling things, seeing things, and evacuating things from the list. It employments numerous fundamental programming concepts such as OOP, record dealing with, and exemption taking care of. It employments legacy to share factors and strategies between subclasses and parent classes. It uses getters and setters to urge or set the esteem of a private variable within the bundle. In this software, data is stored in a file using file management. All changes made to the software are accurately stored in the log files. This software will provide all the basic feature needed for a management software. This software helps any store to increase the productivity of the company and be more efficient. Employee can manage and keep track of the inventory. In large store, there are thousands of items to be kept in track. So, software like these can come in handy.

Challenges and Solution :

Challenges:

- Handling error and provide clear error message.
- It was troublesome to include or get information from the record.
- Guaranteeing date approval to preserve a legitimate log record.
- ConcurrentComodification exemption whereas expelling thing from a list whereas emphasizing over it.
- As the program is console-based, it was truly difficult to create a great user-friendly framework.

Solutions:

- I made a menu with clear alternatives for highlights to create it more user-friendly.
- I utilized standard expressions to approve the user's input.
- I included clear and point by point blunder message to assist client direct through issue and offer assistance me amid investigating.
- I illuminated the ConcurrentComodification special case by changing the cycle circle from for-each to conventional for circle.

Conclusion

In brief, ready to make scaled down store management software. In this program, we have included numerous highlights and made it exceptionally user-friendly. I also store all updates in a log for easy viewing of future changes. This program has made a difference and I have understood many Java concepts very well. OOP, file administration, and exemption dealing with are a few of the foremost critical concepts to have in programming. This extend covers this subject and makes me get it it superior.