PES University, Bengaluru

Department of Computer Science & Engineering

Session: Jan - May 2023

UE19CS353 – Object Oriented Analysis and Design with Java (Mini Project)

Report on

Drug Store Inventory

By:

Pavan D N - PES1UG21CS827

Nandana C L - PES1UG21CS822

Prajwal

6th Semester, Section - I

1. Project Description

Github link: https://github.com/PAVANSHETTY23/Drug-Store-Inventory

Objectives: One of the most significant parts of the pharmaceutical industry is drug shop management. Drug retailers must examine demand and supply in order to stay competitive in the market. Buying the proper quantity is critical to the company's performance in a number of ways. Inventory is essential to meet the pharmacy store's future demands and objectives. The aim of this project is to automate inventory management for a drug store.
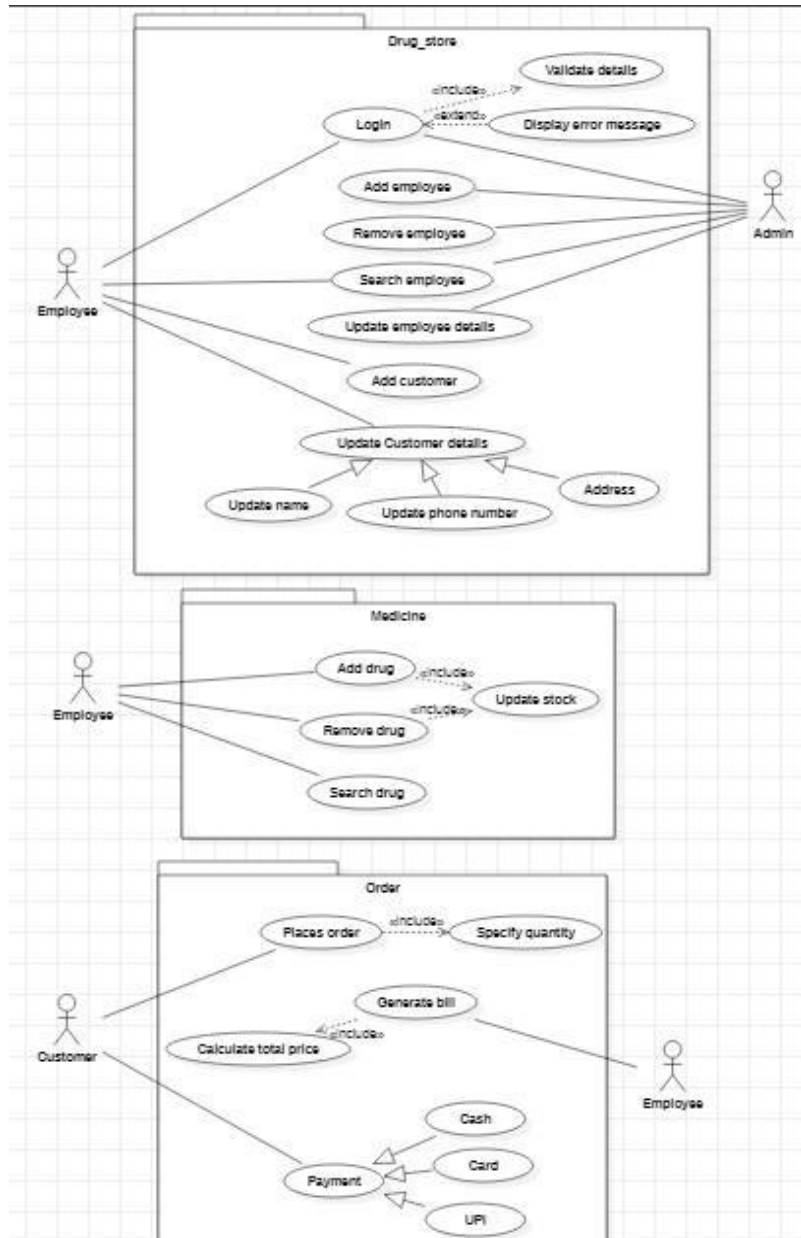
Description:
● The system is built for two types of employees, i.e., Admin and Salesman. Only valid users can access the system through a user login verification process.
● Employees have unique IDs to identify their accounts.

● Customers make requests for medicines, which are checked for availability and expiration dates. Customer then places an order and the corresponding bill is generated by the employee and the stock is updated.
● It helps to manage and record information of medicines, employees and transactions digitally.
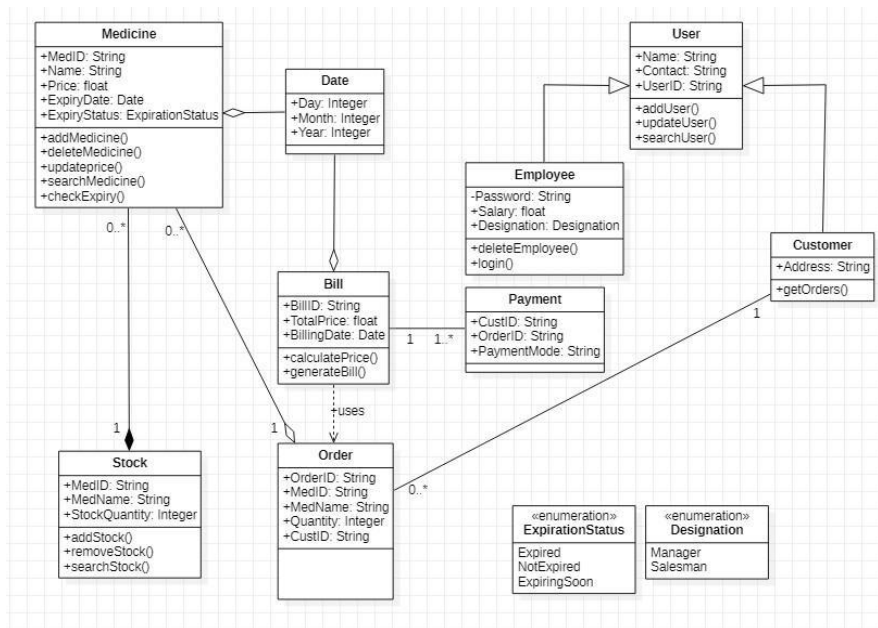
Functionality: Manages information of:
● Medicines : allows to add medicines, delete a medicine if it exceeds its expiry date, update details such as price and quantity and search for particular medicine
● Stock: management of stock whenever a purchase is made or new stock arrives

● Employees : employ people to work in the drug store, remove an employee, search for a particular employee
● Transactions: generate the bill amount using the price per unit and quantity, payment by customer
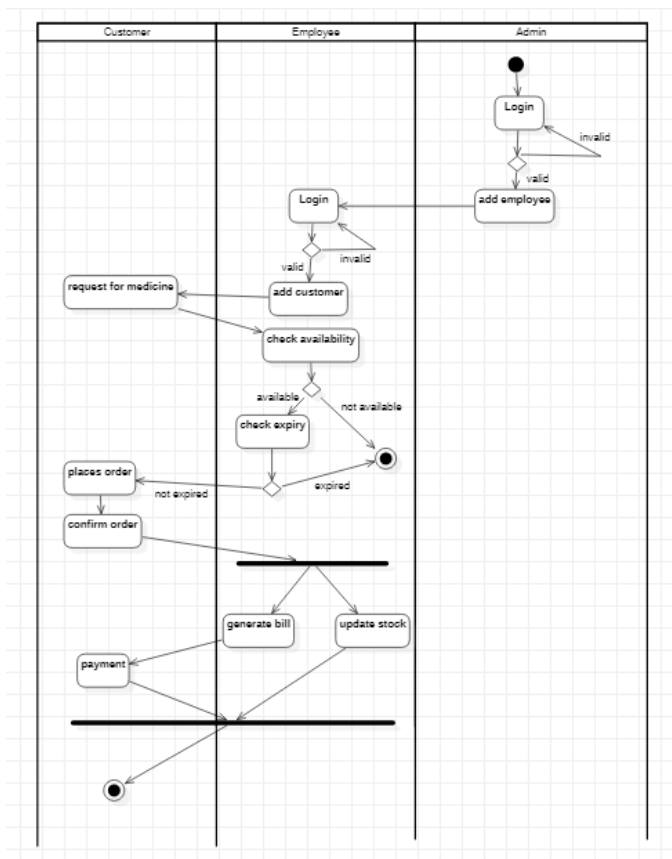
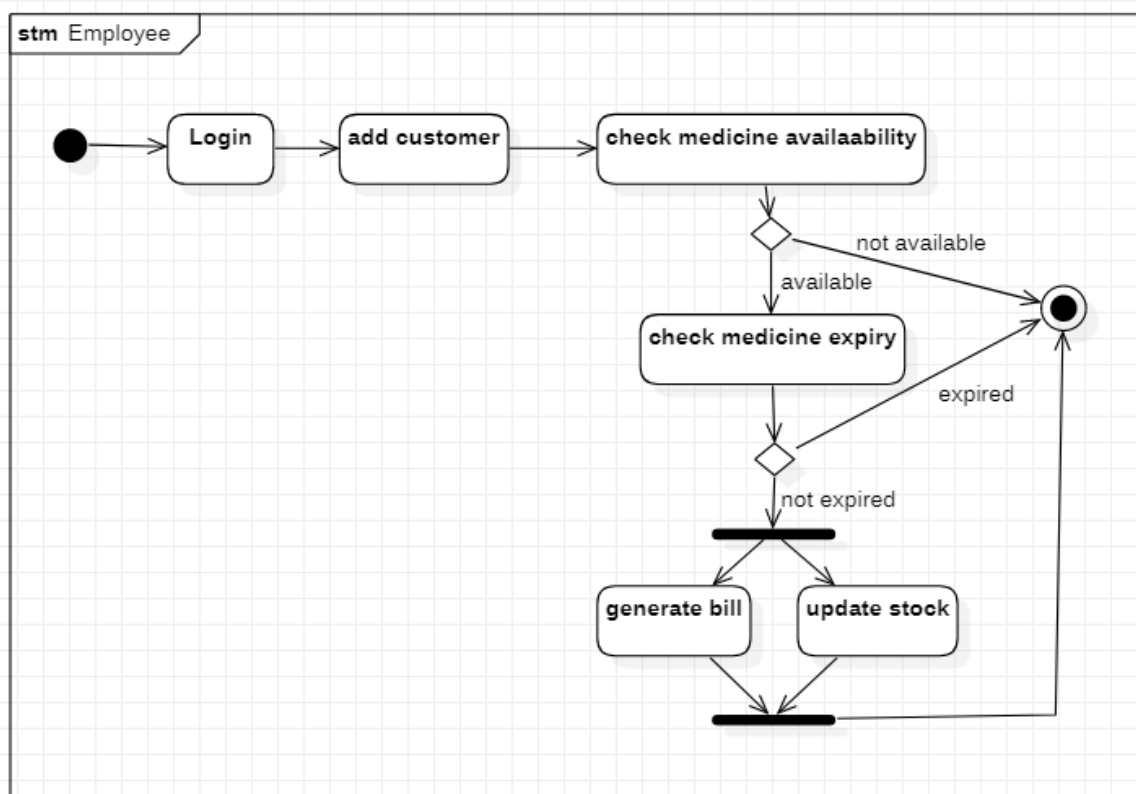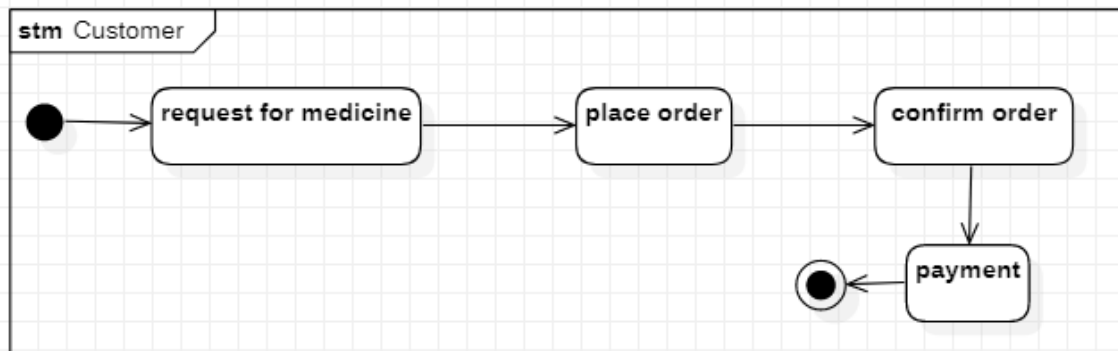# 2. Analysis and Design Models

A. Use Case

B. Class Diagram

Medicine
+MedID: String
+Name: String
+Price: float
+ExpiryDate: Date
+ExpiryStatus: ExpirationStatus
+addMedicine()
+deleteMedicine()
+updateprice()
+searchMedicine()
+checkExpiry()

Date
+Day: Integer
+Month: Integer
+Year: Integer

User
+Name: String
+Contact: String
+UserID: String
+addUser()
+updateUser()
+searchUser()

Employee
-Password: String
+Salary: float
+Designation: Designation
+deleteEmployee()
+login()

Customer
+Address: String
+getOrders()

Bill
+BillID: String
+TotalPrice: float
+BillingDate: Date
+calculatePrice()
+generateBill()

Payment
+CustID: String
+OrderID: String
+PaymentMode: String

+uses

Stock
+MedID: String
+MedName: String
+StockQuantity: Integer
+addStock()
+removeStock()
+searchStock()

Order
+OrderID: String
+MedID: String
+MedName: String
+Quantity: Integer
+CustID: String

«enumeration»
ExpirationStatus
Expired
NotExpired
ExpiringSoon

«enumeration»
Designation
Manager
Salesman

0..*   0..*   1   1   1   1..*   1   0..*

C. Activity Diagram

Customer	Employee	Admin

Login
invalid
valid
add employees
Login
valid   invalid
request for medicine
add customer
check availability
available   not available
check expiry
places order
not expired   expired
confirm order
generate bill   update stock
payment

D. State Diagram

stm Customer

● → request for medicine → place order → confirm order

confirm order → payment → ◉

stm Employee

● → Login → add customer → check medicine availaability

check medicine availaability → ◇

◇ — not available → ◉

◇ — available → check medicine expiry

check medicine expiry → ◇

◇ — expired → ◉

◇ — not expired → ▬

▬ → generate bill, update stock

generate bill, update stock → ▬ → ◉

3. Tools and Frameworks Used

- **Java :** High Level, object oriented programming language for the implementation of the project.
- **Java Swing :** GUI widget toolkit for java to implement the front end
- **MySQL :** Relational database management system for storing data relevant to the project .

4. Design Principles and Design Patterns Applied

Design Pattern : **Singleton**

The enumeration Expiration Status (Expired, notExpired, expiringSoon) is instantiated only once. The single instance of the class expirationstatus is required to control the action throughout the execution. The instance can take the values expired, not expired or expiring soon.

The instance expirationstatus has only one instance and is globally accessible to other classes.

```java
1  package Model;
2
3     public enum expirationStatus {
4         INSTANCE;
5         private String value;
6         private expirationStatus(String value) {
7             this.value = value;
8         }
9         expirationStatus() {}
10        public String getValue() {
11            return value;
12        }
13
14        public void setValue(String value) {
15            this.value = value;
16        }
17     }
18
```

+

Design Pattern : **Façade**

Facade pattern hides the complexities of the system and provides an interface to the client using which the client can access the system. Here the client can access the system through the main.

Facade hides the intricacies of an existing system by adding an interface to it. Here, Home class hides the complexities of the application from the client

Client:

```java
import View.*;

public class Main{
    public static void main(String[] args) {
        Login login = new Login();
    }
}
```

The complex code:

```java
public void actionPerformed(ActionEvent a){
if(a.getSource()==logoutBt){
new Login();
this.setVisible(false);
    }

    else if(a.getSource()==manageEmpBt){
        new ForEmp(user);
        this.setVisible(false);
}
    else if(a.getSource()==manageMedBt){
        new ManageMed(user);
        this.setVisible(false);
    }
  try {
    if(a.getSource()==sellMedBt){
        new Cart(user);
        this.setVisible(false);
    }

    else if(a.getSource()==changePasswordBt){
        new ChangePassword(user);
        this.setVisible(false);
    }}
    catch(Exception e){
        System.out.println(e.getMessage());
        }
    }

}
```
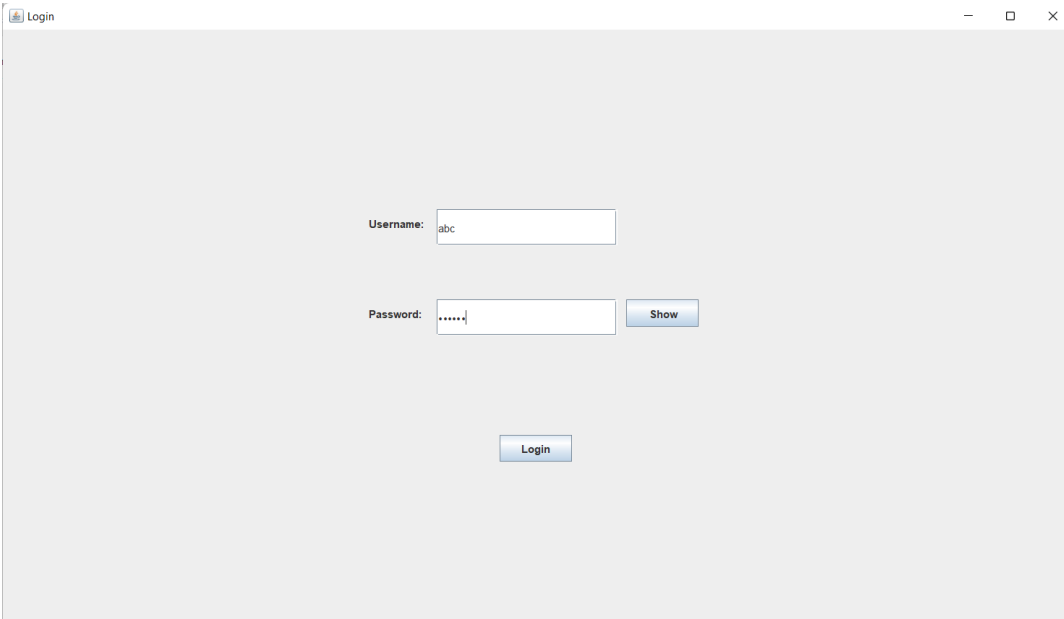
Architectural Pattern : **MVC**

The application consists of a data model, presentation information, and control information. The **Model** contains only the application data(getters and setters) for each class. The **View** presents the model's data to the user. The view knows how to access the model's data, but it has no idea what that data represents or what the user may do with it. The **Controller** exists between the view and the model. It listens to events triggered by the view and executes the appropriate reaction to these events.

5. Application Screenshots (3-4 important pages)

For employee:

Home

Logout

Manage Medicines

Search Employees

Sell medicine

Change Password

Medicine Management

Logout

Back

	Search	Show All

medId	medName	ExpiryDate	unitPrice	Quantity
m102	Xpain	2022-9-22	2.5	100
m103	Adora	2022-5-20	50.0	125
m105	Rifamex	2023-4-12	65.0	90
m106	Alacot DS	2022-6-16	150.45	122
m108	Cefotil	2022-10-30	199.34	19
m109	Phylopen Forte	2025-4-12	110.41	30

Med Id   m103   ☑ Load

Med Name   Adora

Unit price   50.0

Expiry D...   2022-5-20

Quantity   125

Add   Check almost expired

Refresh   Update

Delete Ex...

Employee Management

Logout

Back

[search box] Search   Show All

empName	contact	empId	salary	Designation
nischitha	643219876	u103	70000.0	manager
bindu	678912345	u104	60000.0	salesman
Kushi	983217654	u105	50000.0	salesman
appi	19843276	u106	40000.0	salesman
bhavvu	41987632	u107	30000.0	manager
swapna	674589123	u108	20000.0	salesman
kulal	98743216	u109	10000.0	manager
tarunika	987654322	u102	180000.0	salesman
abc	987654321	u110	20000.0	Salesman
admin	782163787	u100	2000000.0	administrator
def	9876543378	u112	30002.0	Manager

SalePoint

Logout

Back

Adora   search

medId	medName	unitPrice	ExpiryDate	Quantity
m103	Adora	50.0	2022-5-20	125

saleId	medId	medName	quantity	unitPrice	amount
6	m103	Adora	2	50.0	100.0
6	m108	Cefotil	5	199.34	996.7

Sale Id `6`

Med Id `m108` ☑ Load

Med Name `Cefotil`

Unit price `199.34`

Quantity `5` ☐ Calculate

Amount `996.7`

Add to Cart

Refresh   Update

Order ✕

⚠ Order Placed Successfully!!

OK

Calculate Total Amount

Total amount `1096.7` **Cash Paid** `1100` ☑ **Change Amount** `3`

Check Out

Logout

Back

Current Password

Show

New Password

Show

Done