PROJECT DOCUMENTATION

FINAL REPORT

TOPIC:

PHARMACY MANAGEMENT SYSTEM

TEAM:

BITF24M004	**ALI HAIDER**
BITF24M005	**Abdul HADI**
BITF24M006	**HANZALA AKRAM**
BITF24M031	**ABDULLAH JAMSHAID**

SUBMITTED TO :

SIR. ANZAR AHMED

TABLE OF CONTENTS

PROJECT REPORT

- **Problem Statement:** A pharmacy requires a digital solution to replace manual bookkeeping. Key challenges include inefficient manual stock tracking, slow sales calculation, and lack of role-based security (separating Admins from Pharmacists).
- **Solution Description:** We have developed a "Pharmacy Pro Management" suite using Java Swing for the interface and SQLite for the database.
- **Architecture: The system follows a 2-Tier Architecture:**
    - **Frontend:** PharmacyGUI.java handles all user interactions (Login, Sales, Admin Dashboard).
    - **Backend:** DatabaseHandler.java manages all SQL connections, queries, and transaction logic.

INTRODUCTION

In the modern healthcare and retail sector, efficiency and accuracy are crucial. Manual bookkeeping in pharmacies often leads to human error, stock mismanagement, and slower customer service. This project, titled "Pharmacy Pro Management - Full Suite", is designed to address these challenges by automating the core operational and administrative tasks of a pharmacy.

This software is a desktop-based application developed using Java Swing for a robust Graphical User Interface (GUI) and SQLite for lightweight, serverless data management. The system follows modular architecture, distinctively separating the roles of the Administrator and the Pharmacist to ensure data security and operational integrity.

Key functionalities of the system include:

- Role-Based Access Control: The system secures sensitive administrative functions (like staff and supplier management) behind a secure login, while granting Pharmacists access to the Point of Sale (POS) features.

- Real-Time Inventory Management: The application provides immediate feedback on stock levels. When a sale is finalized, the system automatically decreases the stock quantity from the database, preventing the sale of out-of-stock items.
- Point of Sale (POS) Efficiency: The sales module features a dynamic search filter that allows Pharmacists to instantly locate medicines by name. It compiles selected items into a transaction cart, calculates the grand total, and generates a permanent invoice record in the database.
- Data Persistence: Utilizing SQLite, the system ensures that all records ranging from supplier details to daily transaction log are persistently stored and easily retrievable for reporting and management purposes.

By transitioning from manual ledgers to this digital solution, the Pharmacy Pro Management System aims to minimize operational time, eliminate calculation errors, and provide the management with a clear, real-time view of their business performance.

USE-CASE DOCUMENTATION

UC ID: PH-01

Title: Process Sale & Generate Invoice

Primary Actor: Pharmacist

Pre-Conditions: Pharmacist is logged in; Medicines exist in the database with StockQty > 0.

Main Flow:

1. Pharmacist initiates the "Sales" module.
2. System displays the "Search Medicine" bar and current "Cart".
3. Pharmacist types a medicine name (e.g., "Panadol").
4. System dynamically filters and displays matching medicines from the database.

   4.1. Medicine Not Found:

   - 4.1.1. Pharmacist types a name that does not exist in the database.
   - 4.1.2. System displays an empty search table (no results found).
   - 4.1.3. Pharmacist clears the search bar and tries a different keyword or generic name.

5. Pharmacist clicks a medicine.
6. System asks for Quantity.
7. Pharmacist enters quantity (e.g., "2").

   7.1. Invalid Input Format:

- 7.1.1. Pharmacist enters non-numeric text (e.g., "two") instead of a number.
- 7.1.2. System throws a NumberFormatException (implicitly handled in UI logic).
- 7.1.3. The action is cancelled, and the item is not added to the cart.

8. System checks if entered_qty <= current_stock.

   8.1. Insufficient Stock (Quantity > Stock):

   - 8.1.1. Pharmacist enters a quantity greater than the available stock displayed in the dialog.
   - 8.1.2. The system's validation logic fails (Integer.parseInt(qS) <= stock check returns false).
   - 8.1.3. System ignores the input and does *not* add the item to the cart.
   - 8.1.4. Pharmacist must re-select the item and enter a valid quantity.

9. System adds item to the Cart table and updates the "Total Bill".
10. Pharmacist clicks "FINALIZE & PRINT BILL".
11. System saves the transaction to INVOICES and INVOICE_DETAILS tables.

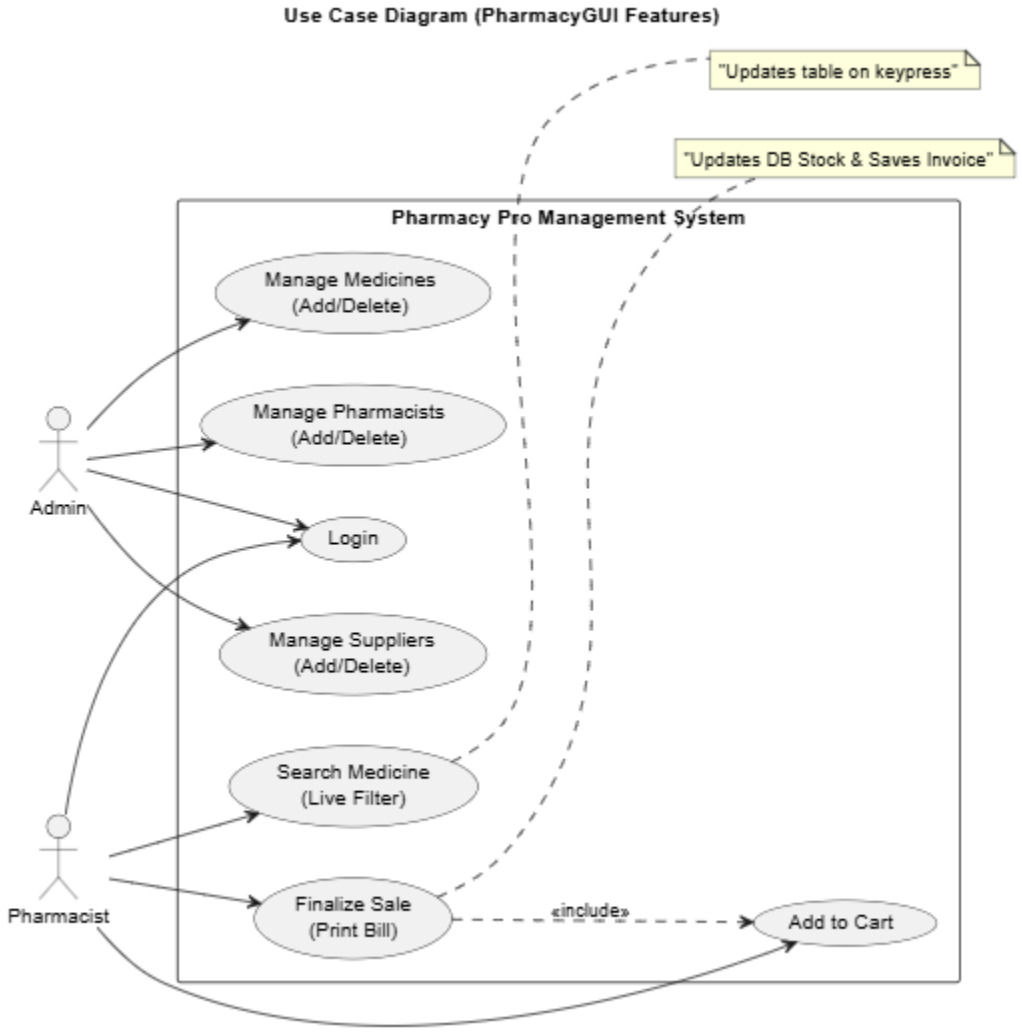
    11.1. Cart is Empty:

    - 11.1.1. Pharmacist clicks **"FINALIZE & PRINT BILL"** without adding any items.
    - 11.1.2. System checks the row count (cartModel.getRowCount() == 0).
    - 11.1.3. System returns without creating an invoice.

12. System automatically decrements the stock count in MEDICINES table.

13.System displays "Sale Completed!" message and resets the cart.

USE-CASE DIAGRAM

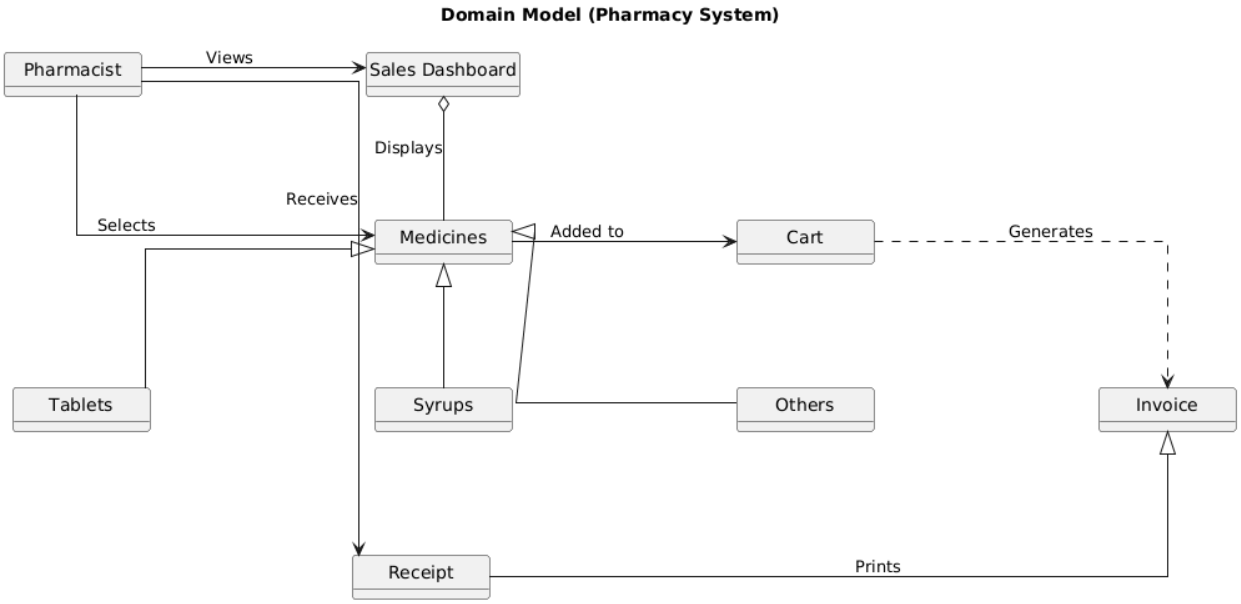


DOMAIN MODEL

The Pharmacist logs into the system using their secure credentials. The interface displays the Sales Dashboard. The Pharmacist searches for a Medicine (e.g., 'Panadol'). The system queries the Category and Stock details. The Pharmacist selects the medicine, creating a line item in the Cart. Upon checkout, the system generates a unique Invoice, records the Invoice Details, and updates the Supplier stock records internally.

Entities:

Admin	**Managing entity.**
Pharmacist	**Operational user.**
Medicine	**The core product, includes Price and StockQty.**
Invoice	**Represents a finalized sale (CREATE TABLE INVOICES).**
Supplier	**Entity that provides medicines.**


Domain Model (Pharmacy System)
Pharmacist
Views
Sales Dashboard
Displays
Receives
Selects
Medicines
Added to
Cart
Generates
Tablets
Syrups
Others
Invoice
Receipt
Prints

ACTIVITY DIAGRAM