# Retail Medical Store Management System (Desktop - Offline Version)

## 1. Introduction

The Retail Medical Store Management System is a desktop-based offline application designed to help a single retail pharmacy manage daily operations efficiently. It allows the store owner to handle billing, medicine inventory, supplier management, purchases, expiry alerts, and reporting—all within a user-friendly interface built using Python (PyQt) and SQLite.

## 2. Objectives

• Streamline billing and inventory management for a single medical store.

• Provide expiry and low-stock alerts automatically.

• Maintain supplier and purchase records efficiently.

• Offer offline functionality for reliability without internet dependence.

## 3. System Overview

The system includes modules for Billing, Medicines, Purchase, Reports, Suppliers, Backup, and Settings. All data is stored locally using SQLite, ensuring fast access and full data control.

## 4. Features

* Dashboard displaying key metrics such as Total Medicines, Low Stock, Today's Sales, and Expiry Alerts.
* Billing system for generating bills with automatic stock updates.
* Inventory management for adding, editing, and deleting medicines.
* Supplier management for tracking sources and purchase details.
* Purchase entry system for new stock updates.
* Reports for sales, stock, and expiry data.
* Data backup and restore functionality.
* User authentication and store settings customization.

## 5. Database Schema

Below is the proposed SQLite database schema for the application:

Tables:  
1. Medicines  
 - medicine\_id (INTEGER, PK)  
 - name (TEXT)  
 - batch\_no (TEXT)  
 - expiry\_date (DATE)  
 - quantity (INTEGER)  
 - purchase\_price (REAL)  
 - sale\_price (REAL)  
 - supplier\_id (INTEGER, FK -> Suppliers.supplier\_id)  
  
2. Suppliers  
 - supplier\_id (INTEGER, PK)  
 - name (TEXT)  
 - contact\_no (TEXT)  
 - address (TEXT)  
  
3. Purchases  
 - purchase\_id (INTEGER, PK)  
 - supplier\_id (INTEGER, FK -> Suppliers.supplier\_id)  
 - date (DATE)  
 - total\_amount (REAL)  
  
4. Sales  
 - sale\_id (INTEGER, PK)  
 - date (DATE)  
 - total\_amount (REAL)  
  
5. SaleItems  
 - id (INTEGER, PK)  
 - sale\_id (INTEGER, FK -> Sales.sale\_id)  
 - medicine\_id (INTEGER, FK -> Medicines.medicine\_id)  
 - quantity (INTEGER)  
 - price (REAL)  
  
6. Users  
 - user\_id (INTEGER, PK)  
 - username (TEXT)  
 - password (TEXT)  
 - role (TEXT)

## 6. PyQt Interface Mockups

The main dashboard includes widgets and buttons for key operations such as Billing, Medicines, Purchase, Reports, and more. Each section opens a form with table views and entry fields for managing data. The dashboard design follows a clean green-white theme with cards showing stock summary, expiry alerts, and recent activities.

## 7. Implementation Details

• Technology Stack: Python, PyQt5, SQLite

• Offline-first design with local database storage.

• Modular architecture for easier maintenance.

• Backup and restore functionality via CSV or SQLite file export.

## 8. Future Enhancements

* Integration of barcode scanning for faster billing.
* Automated daily backup and expiry notifications via email.
* Advanced reporting with charts and analytics.
* Multi-user support with role-based access control.