# MediTrack: A Web Based Application for Pharmacy Management System

**By**
**Shahriar Hossain**
**213-15-4589**

## FINAL YEAR DESIGN PROJECT REPORT

This Report Presented in Partial Fulfillment of the Requirements forthe **Degree of Bachelor of Science in Computer Science and Engineering**

**Supervised by**

**Mr. Mushfiqur Rahman**

**Assistant Professor**
Department of Computer Science and Engineering
Daffodil International University

**Co-Supervised by**

**Mr. Shah Md Tanvir Siddiquee**

**Assistant Professor**
Department of Computer Science and Engineering
Daffodil International University


**DAFFODIL INTERNATIONAL UNIVERSITY**
**Dhaka, Bangladesh**

**September 17, 2025**

## APPROVAL

This Project titled **"MediTrack: A Web Based Application for Pharmacy Management System"**, submitted by **Shahriar Hossain**, ID No: **213-15-4589** to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on **16 September, 2025**.

**BOARD OF EXAMINERS**

---

**Dr. S.M Aminul Haque (SMAH)** — **Chairman**
**Professor & Associate Head**
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

---

**Dr. Md Alamgir Kabir (DMAK)** — **Internal Examiner**
**Assistant Professor**
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

---

**Mr. Md Assaduzzaman (MA)** — **Internal Examiner**
**Assistant Professor**
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

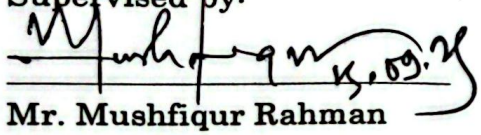
---

**Dr. Md. Zulfiker Mahmud** — **External Examiner**
**Professor**
Department of Computer Science and Engineering
Jagannath University

# DECLARATION

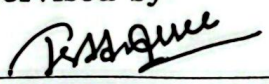
I hereby declare that this project has been done by me under the supervision of **Mr. Mushfiqur Rahman, Assistant Professor**, Department of Computer Science and Engineering, Daffodil International University. I also declare that neither this projectnor any part of this project has been submitted elsewhere for the award of any degree or diploma.

**Supervised by:**



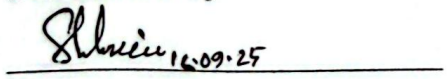
**Mr. Mushfiqur Rahman**
Assistant Professor
Department of Computer Science and EngineeringDaffodil International University

**Co-Supervised by:**



**Mr. Shah Md Tanvir Siddiquee**
Assistant Professor
Department of Computer Science and EngineeringDaffodil International University

**Submitted by:**



**Shahriar Hossain**
Student ID: 213-15-4589
Department of Computer Science and EngineeringDaffodil International University

# ACKNOWLEDGEMENTS

This work would not have been possible without the support and contributions of many individuals over the past two semesters. We are deeply grateful to everyone who has assisted me in one way or another.

First, we express our heartfelt thanks and gratefulness to the almighty for His divine blessing making it possible for us to complete the **Final Year Design Project (FYDP)** successfully.

We are grateful and wish our profound indebtedness to **Mr. Mushfiqur Rahman**, **Assistant Professor**, Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Deep knowledge and keen interest of our supervisor in the field of **Web Application** to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

We would like to express our heartfelt gratitude to the Head of the Department of Computer Science and Engineering, for his kind help in finishing our project and also to other faculty members and the staff of the Department of Computer Science and Engineering, Daffodil International University.

We would like to thank our entire course-mates at Daffodil International University, who took part in this discussion while completing the coursework.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

# ABSTRACT

This project introduces Meditrack, a comprehensive web-based application designed to revolutionize business in pharmacies by integrating inventory management, billing, customer records, and an online buy-sell system. A huge number of pharmacies in Bangladesh are run manually, which leads to errors in stock calculations, time delays in obtaining sales information, and customer credit management problems. Meditrack addresses these issues by providing a single solution through which pharmacies can manage medicine, note purchases, generate invoices, maintain customer credits, and allow customers to purchase medicines online. The application is built using React.js for the frontend, Node.js and Express for the backend, and MongoDB as the database, with inherent provisions for VAT, discount, and unit calculations. Testing with sample data found Meditrack to be consistent, accurate, and user-friendly. It reduces manual labor, enhances efficiency, and makes pharmacy services available to online clients, with potential scope for advanced reporting, analytics, and role-based access in the future.

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

This chapter introduces my project titled *"MediTrack: A Web Based Application for Pharmacy Management System."* It begins with the background and the problem statement, followed by the motivation behind selecting this topic. The purpose is to explain why such a system is needed and how it can bring value in solving real-life problems in pharmacy management.

## 1.1 Introduction

The name of this project is "MediTrack: Web Based Pharmacy Management System Application." The application has been developed with React for the front-end, Node.js with Express for the backend, and MongoDB as the database. MediTrack is designed to make the day-to-day pharmacy work easier by delivering faster, simpler, and more accurate work by maintaining everything on a digital platform. Instead of keeping handwritten notes, the system is providing an end-to-end solution for managing medicines, purchases, invoices, and customer data. A majority of the pharmacies in Bangladesh employ manual record-keeping, which generates constant errors such as wrong stock count, misplaced invoices, and customer payment confusion [1], [19]. All these problems become even more serious when there are more customers and medicines. MediTrack de-emphasizes this issue by offering features such as stock management, medicine addition, invoice generation, and customer record tracking. The system is also VAT and discount calculable, which is an important aspect of pharmacy business [7].

## 1.2 Motivation

The main motivation of developing ***MediTrack*** is to make pharmacy work more organized and efficient by using digital technology. In most small and medium pharmacies in Bangladesh, records are still written manually in notebooks [1],[19]. This not only takes extra time but also creates many mistakes in stock calculation, sales records, and customer payments. We wanted to solve this real problem by building a system where all data is stored properly and can be managed easily.

## 1.3 Objectives

The main objective of this project is to create a user-friendly pharmacy management application that reduces the workload of pharmacists and improves the efficiency of daily operations. The system aims to computerize pharmacy activities such as medicine management, stock updates, invoice generation, purchase tracking, and customer records. By automating these important tasks, the system makes the work faster, more reliable, and less confusing for the users.
In simple terms, the objectives of **MediTrack** are:

- To minimize or remove the use of manual paper-based records.
- To reduce mistakes and make medicine stock management easier [2],[3].
- To provide a simple, fast, and secure way of preparing invoices and bills.
- To improve processing efficiency by automating routine pharmacy tasks.
- To make the system easy to use so that users can operate it without technical difficulties.

## 1.4 Methodology

To develop **MediTrack**, we employed a step-by-step method to make sure that the system is sufficient for a pharmacy's needs and operates as required. The major steps were:

1. **Problem Identification:** Conducted research about existing manual systems employed in pharmacies and observed general problems such as inventory discrepancies, billing errors, and misplaced records.
2. **Requirement Collection:** Determined the functional requirements like medicine management, purchase tracking, invoice production, and customer recordkeeping.
3. **System Design:** Made the system architecture, database schema, and user roles for different pharmacy activities.
4. **Implementation:** Implemented the system using React as front end, Node.js with Express as back end, and MongoDB as the database.
5. **Testing and Validation:** Tested all modules using dummy data for proper calculations, secure data storage, and easy user handling.
6. **Finalization:** Improved the UI and merged all modules for smooth working.
   This systematic strategy helped us to move step by step and achieve a safe, efficient, and approachable pharmacy management system.

## 1.5 Project Outcome

The main outcome of this project is a complete web-based application that can be used by pharmacies to manage their daily activities in a more organized and efficient way. ***MediTrack*** provides a digital solution for storing medicine details, handling stock, managing purchases, preparing invoices, and keeping records of customer credits and payments. By using this system, pharmacy owners and staff can reduce manual work and avoid common mistakes that happen in traditional paper-based methods.
The expected outcomes of the project are:

- A user-friendly application that can be operated by pharmacy staff without advanced technical knowledge.
- Accurate stock management where medicine quantities are updated automatically after each purchase or sale.
- Easy invoice generation with VAT and discount calculation.
- Clear record keeping of customer dues, payments, and purchase history.
- Secure data storage using MongoDB so that no important information is lost.
- Role-based access where Super Admin, Pharmacy Owner Admin, Staff, and Customer are properly managed.

## 1.6 Organization of the Report

This report is structured into six chapters for neat and orderly presentation of the project.

- **Chapter 1: Introduction**

This chapter gives the overall background of the project. This includes the problem

statement, motivation, goals, summary of methodology, outcome of the project, and report structure.

- **Chapter 2: Background**

This chapter covers the background study, literature review, alternative applications, and gap analysis that helped us to understand the problem and solution planning.

- **Chapter 3: Research Methodology**

This chapter outlines the adopted methodology in detail. It specifies the requirement analysis, system design, functional and non-functional requirements, data flow diagrams, and UI design. It also displays the project plan and task allocation.

- **Chapter 4: Implementation and Results**

This chapter discusses the system implementation, setup in the environment, test process, assessment, and results obtained. The chapter also provides a discussion regarding the system's performance.

- **Chapter 5: Engineering Standards and Design Challenges**

This chapter focuses on the engineering standards followed, the issues faced during designing, and how the system is compatible with software, hardware, and communication standards. It also covers the impact on society, environment, and sustainability and financial analysis and mapping of intricate engineering problems.

- **Chapter 6: Conclusion**

This chapter recapitulates the report by summarizing with the major findings, mentioning the limitations of the project, and suggesting probable future work.

# Chapter 2

# Background

This chapter provides the background details needed by the project. It includes an introduction of the area, a review of existing works, and investigation of related systems. The purpose is to equip readers with enough information so they can understand the rest of the report.

## 2.1 Introduction

In the last few years, the applications of digital technology in pharmacy services and healthcare have been largely important. Pharmacies need to handle a huge amount of medicines, customer transactions, and accurate records. Paper-based systems and manual registers have been used in the majority of small and medium-sized pharmacies in Bangladesh to maintain their data. Though the process is simple, it typically leads to errors in managing the stock, delay in billing, and the loss of important records [1].

To overcome such issues, computerized pharmacy management systems have been introduced all over the globe. Such systems are capable of storing data of medicine in a secure way, generating invoices automatically, tracking purchases, and maintaining customer history with more accuracy. They also consume less time, are less paper intensive, and avoid human mistakes [2], [3], [4], [6].

Our MediTrack project is designed to bring these benefits together into an actual web-based solution. Through the combination of advanced technologies such as React, Node.js, and MongoDB, MediTrack provides an affordable and reliable way of running pharmacies online. Our background research for this project helps us achieve how computerized systems can help bypass the limitations of manual operations and make general pharmacy service better [15],[16].

## 2.2 Literature Review

**Table 2.1:** Summary of Literature Reviewed.

| Author (s) | Year | Title | Methodology | Key Findings |
|---|---|---|---|---|
| Saha et al.[1] | 2017 | Hospital Pharmacy Management System and Future Development Approaches in Bangladeshi Hospital | Case study & system analysis | Found that manual systems in Bangladesh cause inefficiency; recommended digital pharmacy management |
| Agrawal A.[2] | 2009 | Medication Errors: Prevention Using Information | Literature review | Showed that IT systems reduce medication errors |

| | | | | |
|---|---|---|---|---|
| | | Technology Systems | | and improve patient safety. |
| Al-Khatib et al. [3] | 2018 | Pharmacy Information Systems: Characteristics, Benefits and Limitations | Survey-based | Concluded that digital pharmacy systems improve efficiency but need user training. |
| Wulff et al. [4] | 2019 | Pharmacy Information Systems – Impact on Workflow and Quality | Observational study | Found that computerized pharmacy systems reduce workload and improve accuracy. |
| Saravanan et al. [5] | 2020 | Development of Web-Based Inventory Management System for Pharmacies | Prototype testing | Developed a web system that improved stock tracking and billing speed. |
| Hassan et al.[6] | 2021 | Use of ICT in Pharmacy Practice: A Review | Review paper | Stated that ICT tools improve workflow, reduce errors, and assist decision-making. |
| Patel & Sharma [7] | 2019 | Role of IT in Health Care Management | Quantitative study | Showed that IT adoption improves efficiency and accuracy in healthcare record keeping. |
| Arif & Sani [8] | 2018 | Design and Implementation of a Pharmacy Management System | System development | Built a pharmacy management system that minimized errors and improved data security. |
| Singh & Gupta [9] | 2018 | Development of Web-Based Inventory and Billing System for Pharmacy | System design | Created a billing and inventory system that reduced paper usage and manual errors. |
| WHO [10] | 2021 | Global Strategy on Digital Health 2020–2025 | Policy framework | Highlighted the importance of adopting digital systems for health and pharmacy record management worldwide. |

### 2.2.1 Similar Applications

**Table 2.2:** Similar Applications.

| Application | Location | Key Features (Already in Service) |
|---|---|---|

| | | |
|---|---|---|
| **MediManage – MM IT Soft Ltd.** | Bangladesh | Cloud-based pharmacy software; real-time inventory with expiry tracking, barcode POS billing, VAT/Tax support, SMS alerts, 30k+ medicine database. Widely used in Dhaka-based pharmacies. |
| **RetailPharma – Mediasoft Data Systems** | Bangladesh | Retail POS & ERP for pharmacies; handles inventory, expiry & return management, multi-counter selling, vendor management, VAT reports. Deployed across many retail pharmacies in Bangladesh. |
| **SmartPOS Pharmacy Software – Techno71** | Bangladesh | Cloud/web pharmacy POS; billing with barcode scanning, expiry/negative stock alerts, purchase & order management, multi-user support. Already used by local drug stores. |
| **Marg ERP – Marg Compusoft Pvt. Ltd.** | India, Bangladesh | One of South Asia's most widely used pharmacy management solutions; includes GST/VAT compliance, supplier & batch tracking, automated ordering, and strong reporting features. |
| **UshaSoft Pharmacy Software** | Bangladesh | Cloud-based PMS with prescription handling, inventory control, reporting, and branch support. Already implemented in several medium-sized pharmacies. |

## 2.3 Gap Analysis

The study of existing pharmacy management systems shows that while many applications are already in service, they often miss important features such as flexible role-based access, simple VAT/discount handling, and customer credit tracking [5], [8], [9], [25]. *MediTrack* is designed to address these gaps.

**Table 2.3:** Gap Analysis of Existing Systems vs. MediTrack

| Features | MediManage (MM IT Soft) | RetailPharma (Mediasoft) | SmartPOS (Techno71) | UshaSoft PMS | Marg ERP – Marg Compusoft Pvt. Ltd. | Proposed System (MediTrack) |
|---|---|---|---|---|---|---|
| Medicine Inventory (Batch & Expiry | Yes | Yes | Yes | Yes | Yes | Yes |
| POS Billing with Customer Credit / Due Tracking | Yes | Yes | Yes | Yes | Yes | Yes |
| Customer Credit / Due Tracking | Yes | Yes | No | No | Yes | Yes |
| Role-Based Access (Super Admin, Owner Admin, Staff, Customer) | No | No | No | No | No | Yes |
| VAT & Discount | Yes | Yes | No | Yes | Yes | Yes |

| | | | | | | |
|---|---|---|---|---|---|---|
| (Custom %) | | | | | | |
| Detailed descriptions of products | No | Yes | Yes | Yes | Yes | Yes |
| Multi-Pharmacy / Multi-Branch Support | Yes | Yes | No | Yes | Yes | Yes |
| Reports & Analytics | Yes | Yes | Yes | Yes | Yes | Yes |
| Multiple payment options | Yes | Yes | Yes | Yes | Yes | Yes |
| Affordable for Small Pharmacies | No | No | Yes | Yes | No | Yes |
| Modern Tech Stack (React, Node, MongoDB) | No | No | No | No | No | Yes |
| Customer Records (Purchase History) | No | No | No | Yes | Yes | Yes |

## 2.4 Summary

In this chapter, we described the background study that supports the development of our project **MediTrack: A Web Based Application for Pharmacy Management System.** Firstly, we clarified the importance of computerized systems in pharmacy operations and how these systems can reduce issues in manual record keeping. Secondly, from the literature review, we identified at least ten related works that prove that computerized pharmacy systems improve efficiency, reduce errors, and support improved management. We have also studied several other apps of similar nature, local and international, already in use. Their advantages and limitations were compared through such studies. Following that, gap analysis was done to compare our proposed solution with existing systems. The result was that even though most of the applications do inventory and billing management, none of them offer features like detailed role-based access, customer due tracking, and a cost-effective model for small pharmacies.

This chapter had defined the requirements of missing systems and identified why MediTrack needs to be done. The next chapter will discuss the methodology, requirement study, and system design that we followed to finish the project.

# Chapter 3

# Research Methodology

This chapter explains the methodology that we followed to develop ***MediTrack***. It includes requirement analysis, system design, and the specifications that guided our implementation. The purpose of this chapter is to describe how the project was planned and structured to ensure accuracy, reliability, and usability.

## 3.1 Requirement Analysis & Design Specification

**Requirement Analysis**

The first step in our methodology was to identify the requirements of a pharmacy management system. We studied the daily operations of small and medium pharmacies in Bangladesh and identified the common problems such as manual stock mismatch, missing invoices, difficulty in tracking customer dues, and lack of organized reporting.
Based on this, the **functional requirements** of *MediTrack* were set as follows:

- Add, update, and delete medicines with batch and expiry details.
- Manage purchases, invoices, and billing with VAT and discount options.
- Track customer information, including credits and payments.
- Provide role-based access for Super Admin, Pharmacy Owner Admin, Staff, and Customers.
- Generate reports for sales, stock, and financial records.

The **non-functional requirements** include:

- **Usability**: The system must be simple and user-friendly for pharmacy staff.
- **Security**: Data should be stored securely with proper validation.
- **Performance**: The application should process billing and inventory operations quickly.
- **Scalability:** The system should be extendable for multi-branch pharmacies in the future.

The non-functional requirements include usability, security, performance, and scalability, which are essential quality factors for reliable software systems [13], [21].

**Design Specification**

After analyzing the requirements, we designed the system architecture and workflows. The application was built as a **web-based system** using:

- **Frontend**: React for user interface.
- **Backend**: Node.js with Express for server logic.
- **Database**: MongoDB for storing medicines, purchases, invoices, and customer data.

We created use case diagrams, data flow diagrams (DFD), and entity-relationship models to map out how the system would function. The design also included a role-based access model where each type of user (Super Admin, Owner Admin, Staff, Customer) has different permissions.

### 3.1.1 Overview

The ***MediTrack*** system is designed as a **web-based pharmacy management application** that helps in organizing medicine stocks, sales, purchases, and customer records. The main goal of the system is to replace manual record keeping with a digital solution that

is fast, accurate, and easy to use.
The system follows a **client-server model**. The **frontend** is developed with React, which provides a responsive and interactive interface for users. The **backend** is built using Node.js with Express, which handles business logic, API calls, and server-side operations. The **database** is implemented in MongoDB, where all data such as medicines, purchases, invoices, and customer records are stored securely.
There are **four types of users** in the system:

1. **Super Admin** manages the whole platform and pharmacy accounts.
2. **Pharmacy Owner Admin** manages their own pharmacy, adds staff, and gives access rights.
3. **Pharmacy Staff** performs day-to-day operations like adding medicines, handling purchases, and preparing invoices.
4. Customer manages own purchasing history, payments and credits are monitored for future reference.

It also includes features such as inventory updating with batch and expiry dates, billing with VAT and auto-calculation of discount, reminders to customers for due payments, and generation of reports. The architecture is designed to be secure, easy to use, and scalable in a way that it can be used in small-scale pharmacies today and scaled up to bigger-sized establishments in the future.

### 3.1.2 System Design

The **MediTrack** system design is the whole pharmacy management application architecture and how different levels of development were executed. The design follows the Waterfall Model, a sequential model in which each phase is completed before moving on to the next. That model was chosen because it ensures a clear structure, easy documentation, and is suitable for academic projects with fixed requirements.
The flow of the system begins with **problem identification and requirement analysis**, followed by **system architecture design and database schema**. After that, the project moves into **module design**, which includes UI/UX design, medicine and stock management, purchase and invoice handling, and role-based access control. Once the design is finalized, the system is implemented using React for the frontend, Node.js with Express for the backend, and MongoDB for the database. The system then goes through **testing and validation**, followed by **deployment** for real use. The final stage is **monitoring and maintenance**, which ensures the system remains reliable and can be extended with new features in the future.
The modules that form the backbone of *MediTrack* are:

- **UI/UX Wireframes** – To provide simple and user-friendly interfaces.
- **Medicine & Stock Management** – To add, update, and track medicines with batch, expiry, and quantity.
- **Purchases & Invoices** – To record supplier information, generate invoices, and apply VAT/discounts.
- **Role Management** – To separate responsibilities among Super Admin, Pharmacy Owner Admin, Staff, and Customers.

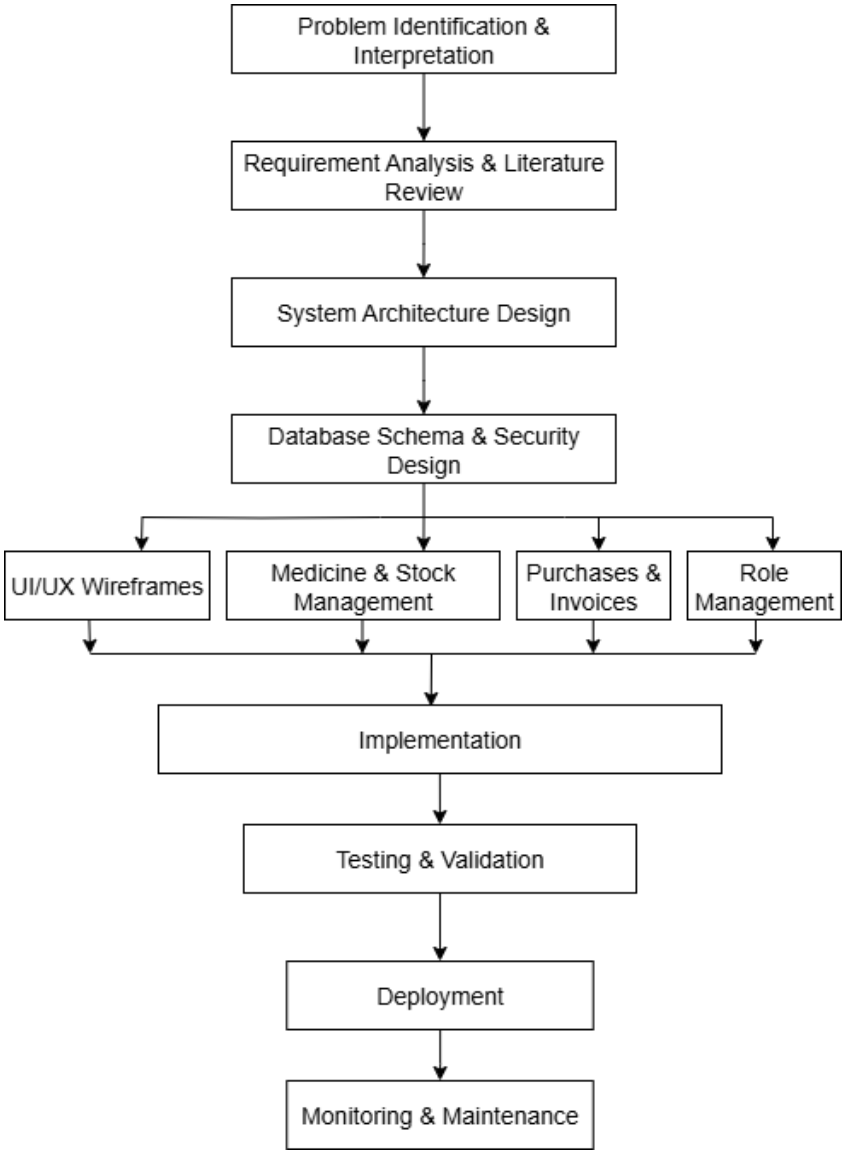



**Figure 3.1:** MediTrack System Design (Waterfall Model)

The figure (Figure 3.1) above illustrates the design of the system following the **Waterfall Model** [11], [12]. It shows how the project moves step by step, starting from requirement analysis, moving into design and module development, and finally reaching implementation and deployment. This structured and linear approach ensures that the system is well-documented, reliable, and easy to maintain.

### 3.1.3 Functional and Nonfunctional Requirements

**Functional Requirements**

The functional requirements describe what ***MediTrack*** should do in real use. These are the main features that allow users to perform pharmacy operations correctly:

1. **Medicine Management**: Add, update, delete, and search medicines with details like batch number, expiry date, quantity, and price.
2. **Stock Tracking**: Update stock automatically after purchase and sales, with alerts for expiry and low stock.
3. **Invoice & Billing**: Generate invoices for sales with VAT and discount calculation, print/save options.
4. **Purchase Management**: Record supplier details, purchase entries, and update stock accordingly.
5. **Customer Management**: Store customer information, track purchase history, credits, and payments.
6. **Role-Based Access Control**: Super Admin, Pharmacy Owner Admin, Staff, and Customers should have separate roles with defined permissions.
7. **Reports Generation**: Generate daily/weekly/monthly reports for sales, stock, and revenue.
8. **Authentication & Security**: Secure login system for all users with password protection.

**Non-Functional Requirements**

Non-functional requirements specify how the system has to operate and characteristics that make it usable, secure, and reliable:

1. **Usability:** The system must be easy to use in a way that pharmacy workers who lack technical knowledge can easily operate it.
2. **Performance:** Invoicing and stock refresh must be processed quickly without delays.
3. **Scalability:** The system must be scalable to support future growth, e.g., multi-branch stores or mobile app integration.
4. **Security:** Data should be stored securely with appropriate validation to avoid unauthorized access.
5. **Reliability:** The system should function consistently without errors throughout routine pharmacy operations.
6. **Maintainability:** The system must be easily maintainable, debuggable, and updateable when new features are introduced.
7. **Portability:** Being a web-based system, it must be functional in various devices and browsers.

### 3.1.4 Data Flow Diagram

The chapter explains information movement in the Pharmacy Management System using Data Flow Diagrams (DFDs). DFDs decompose the system from top level to logical sub-processes, and determine external entities, processes, data stores, and flows.

### Diagram Symbols and Notations

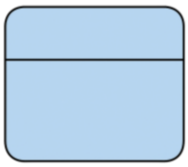
| Name | Description | Symbol |
|---|---|---|
| **Process** | Represent transformations of input data into output data. |  |
| **External Entity** | Represent people or systems outside the boundary that interact with the pharmacy system |  |
| **Data Store** | Represent repositories where data is stored for later use. |  |

**Figure 3.2:** Notation and Symbol of DFD

### DFD Level-0 (Context Diagram)

The Level-0 DFD, or context diagram, shows the entire system as a single process. It shows how the MediTrack system communicates with the external worlds. The external worlds are Admin/Pharmacist and Customer. Admin processes orders medicine and places, while the Customer places orders and pays.

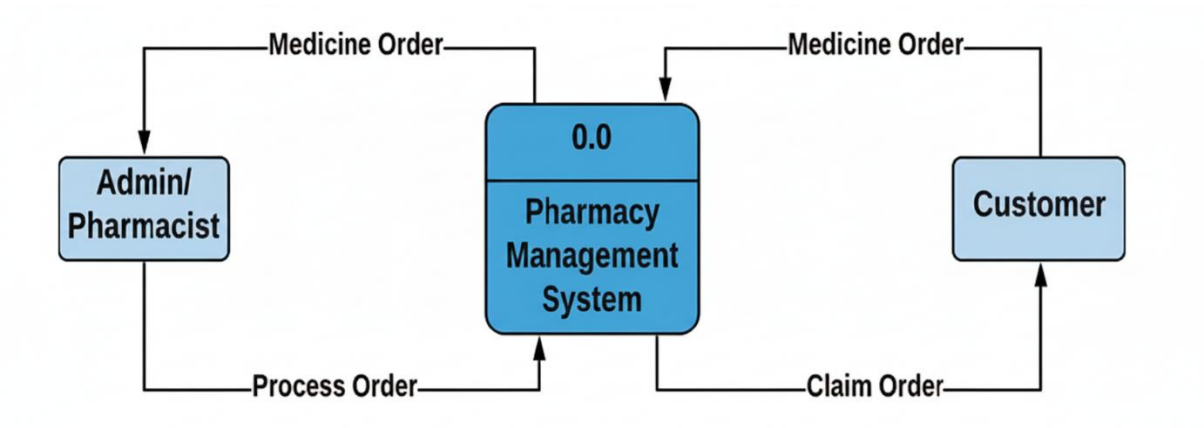



**Figure 3.3:** DFD Level-0 (Context Diagram) – MediTrack

The diagram evidently shows that the whole system is maintained as a single process at this level, and external interaction alone is shown.

### DFD Level-1 (Major Processes)

Level-1 DFD breaks down the system into four main processes:

1. **Manage Customers' Information (1.1)** - Handles customers' information storage and updation.
2. **Manage Medicines Information (1.2)** - Stores all medicines, including stock and expiry dates.
3. **Manage Sales and Stocks (1.3)** - Handles customers' orders, billing, calculation of VAT/discounts, and adjustment of stock after sales.
4. **Generating Reports (1.4)** - Provides reports on sales, stock, and customer records.

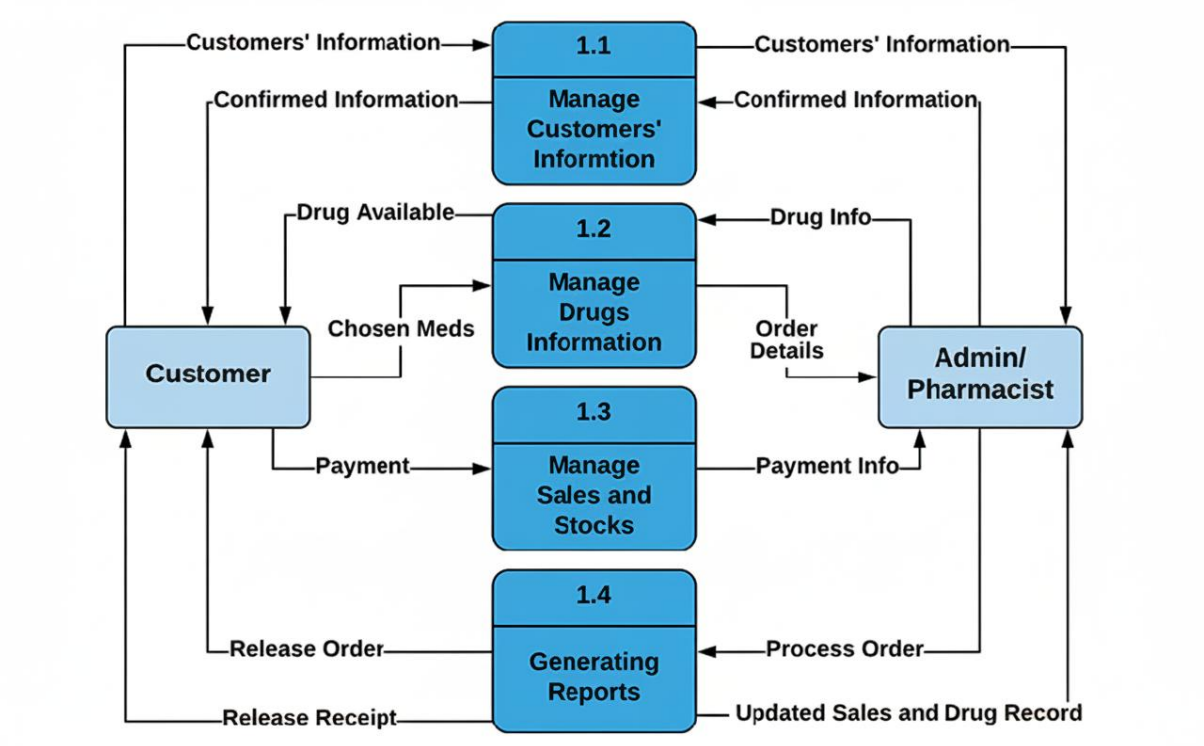



**Figure 3.4:** DFD Level-1 Major Processes of MediTrack

This figure illustrates the way the key processes interact with outside parties and internal movements and illustrates an easy picture of how the pharmacy operations are handled.

### DFD Level-2 (Detailed Process – Invoice and Sales)

The Level-2 DFD also breaks down the Manage Sales and Stocks (1.3) process into steps. These are:

- **3.1 Search/Select Medicine:** Employees select drugs from inventory.
- **Calculate Price, VAT & Discount:** The system computes billing amounts.
- **Generate Invoice & Refurbish Stock:** Generates invoice and deducts sold items from stock.
- **Record Payment & Print Receipt:** Stores payment details and prints a receipt for the customer.

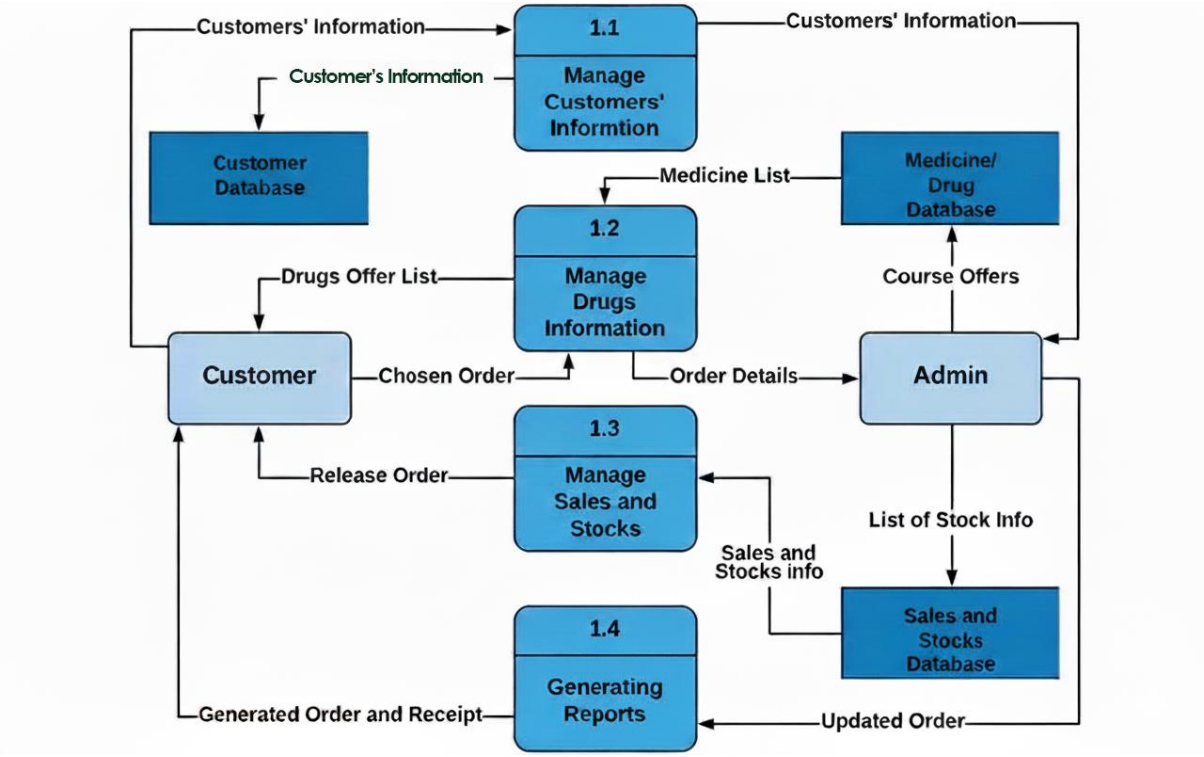



**Figure 3.5:** DFD Level-2 Detailed Invoice & Sales Process in MediTrack

This is a more detailed representation of the sales and invoicing process, which is also one of the most important elements of a pharmacy management system.

### 3.1.5 UI Design

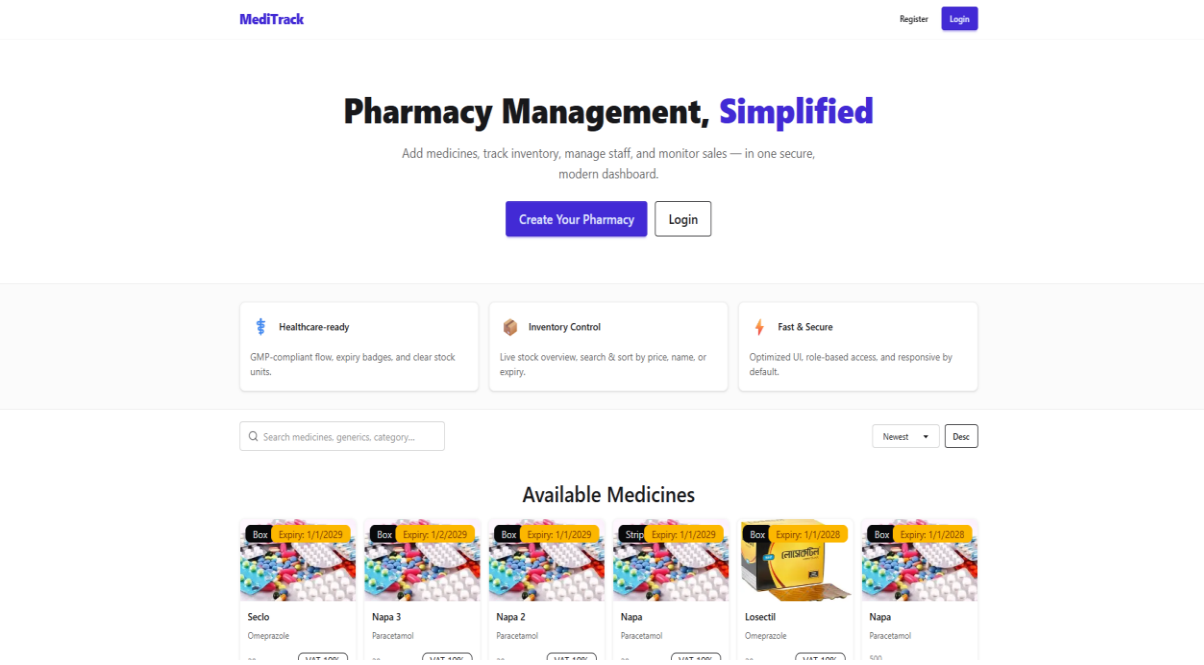



**Fig 3.6:** Homepage

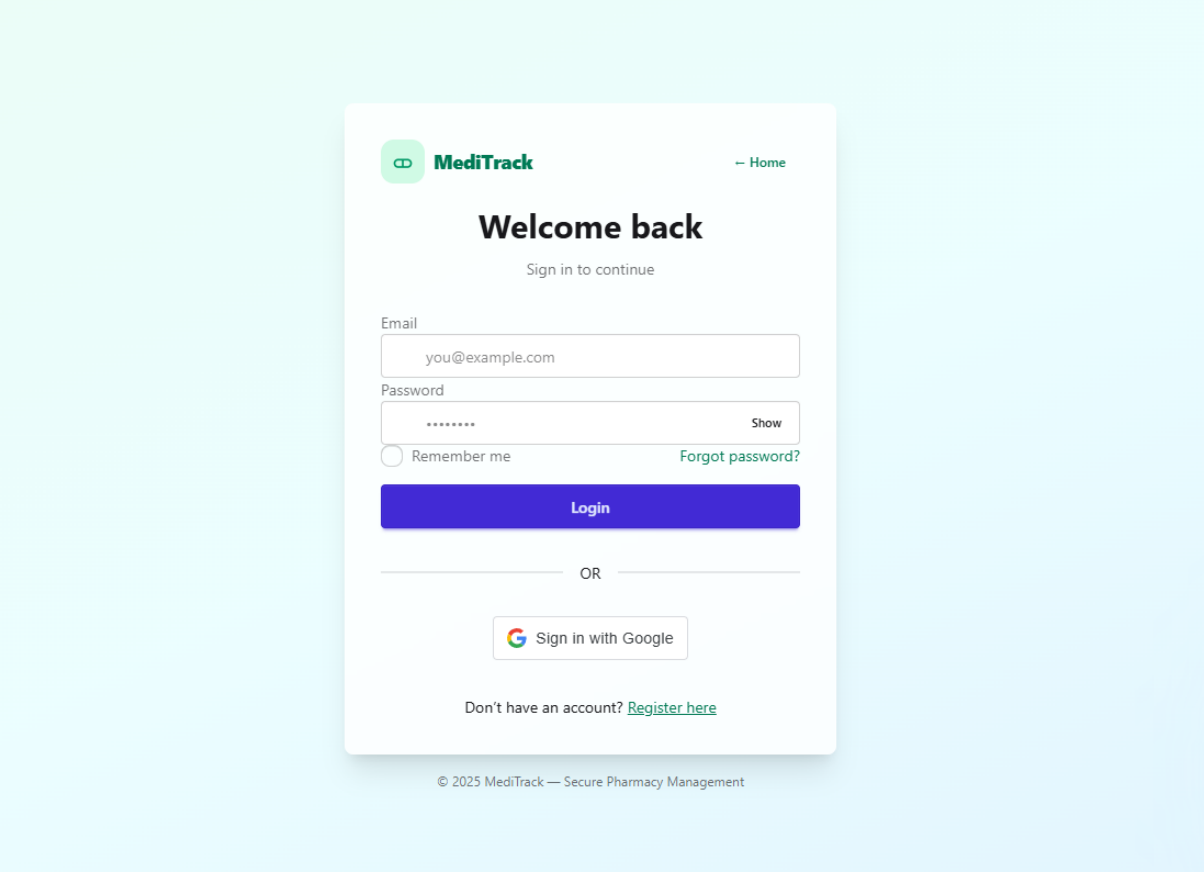



**Fig 3.7:** Login Page

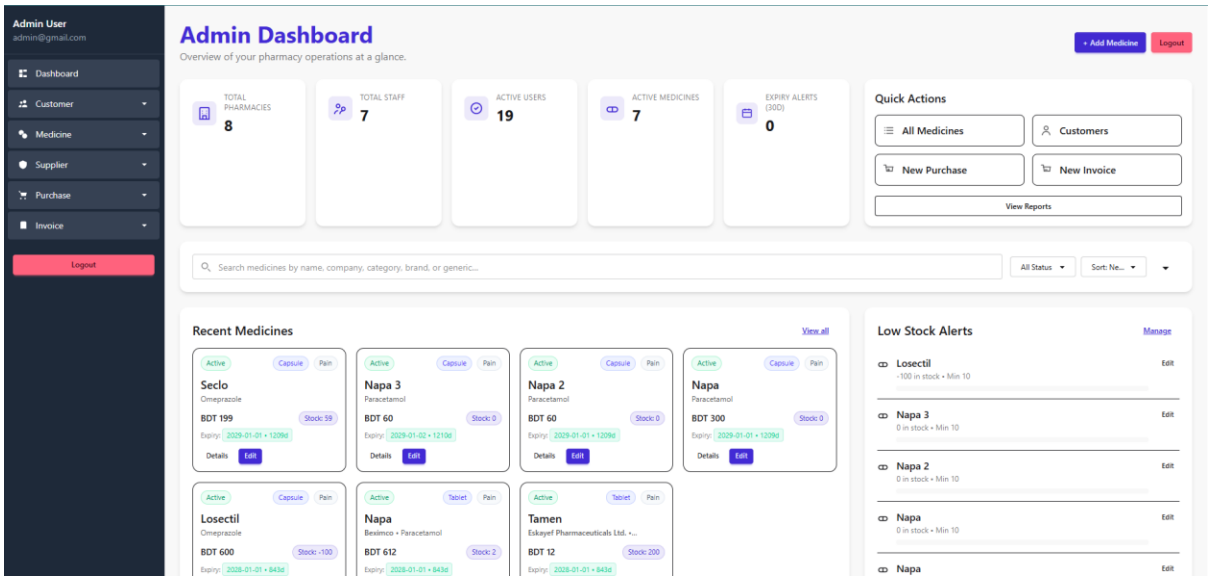



**Fig 3.8:** Admin Dashboard

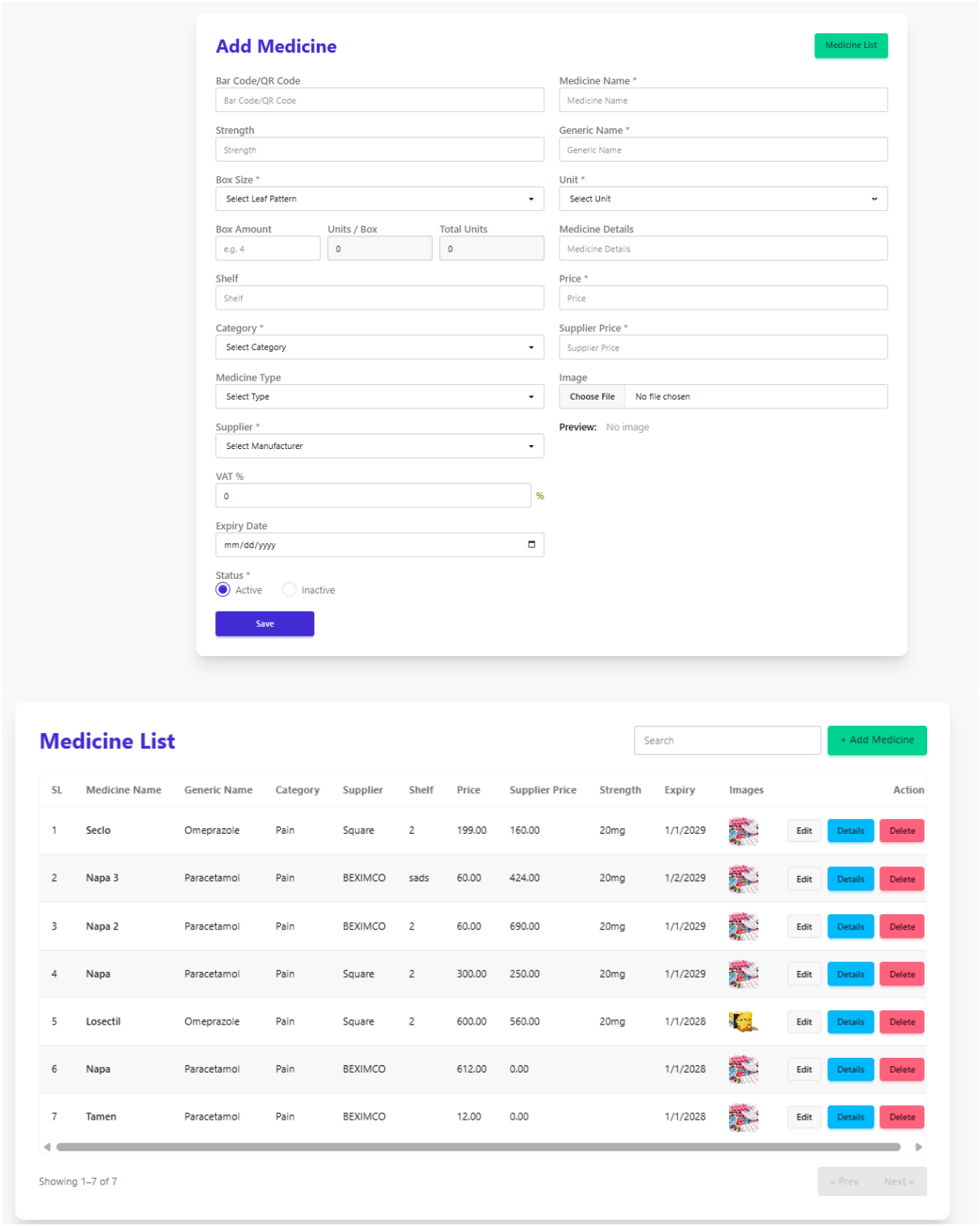



**Fig 3.9:** Medicine Management

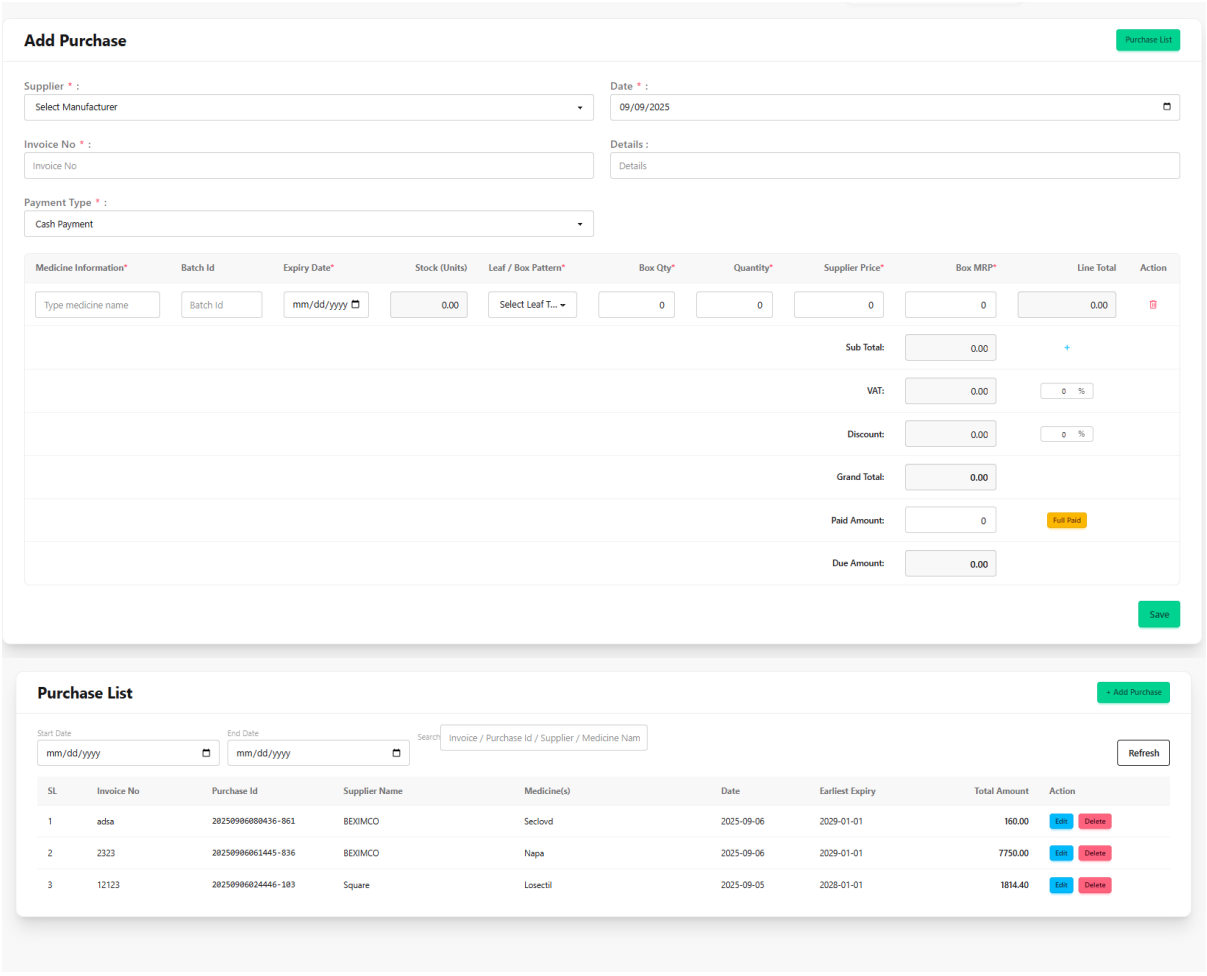



**Fig 3.10:** Purchase Management

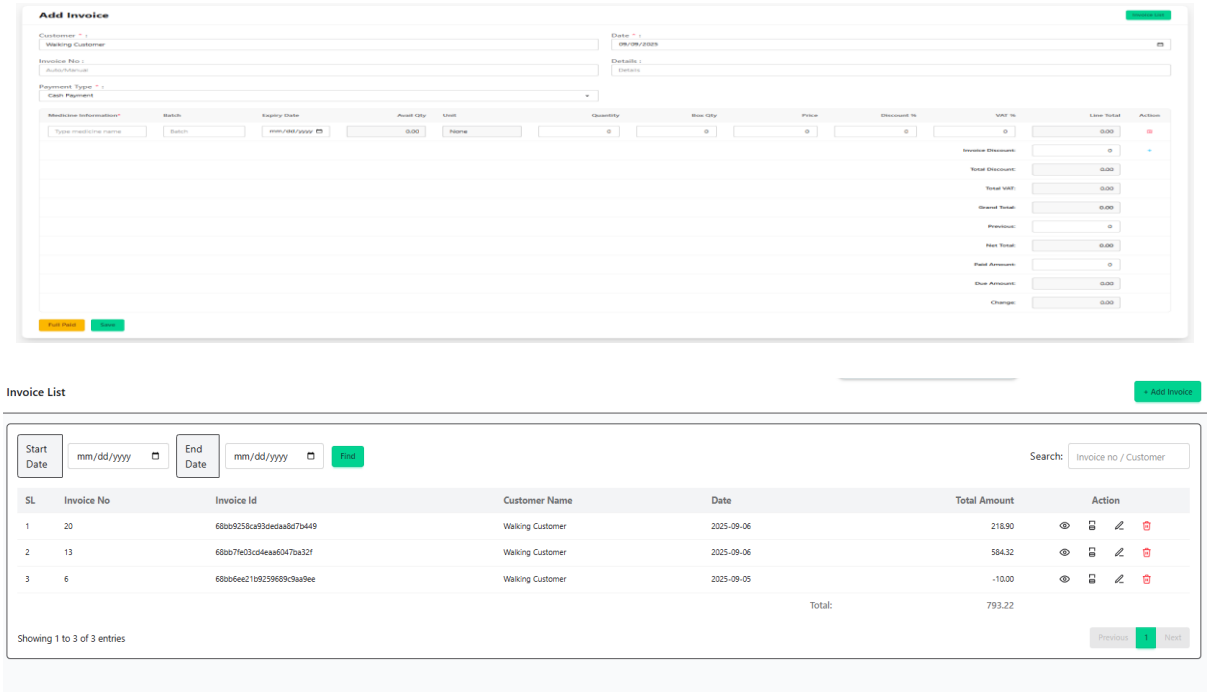



**Fig 3.11:** Invoice System

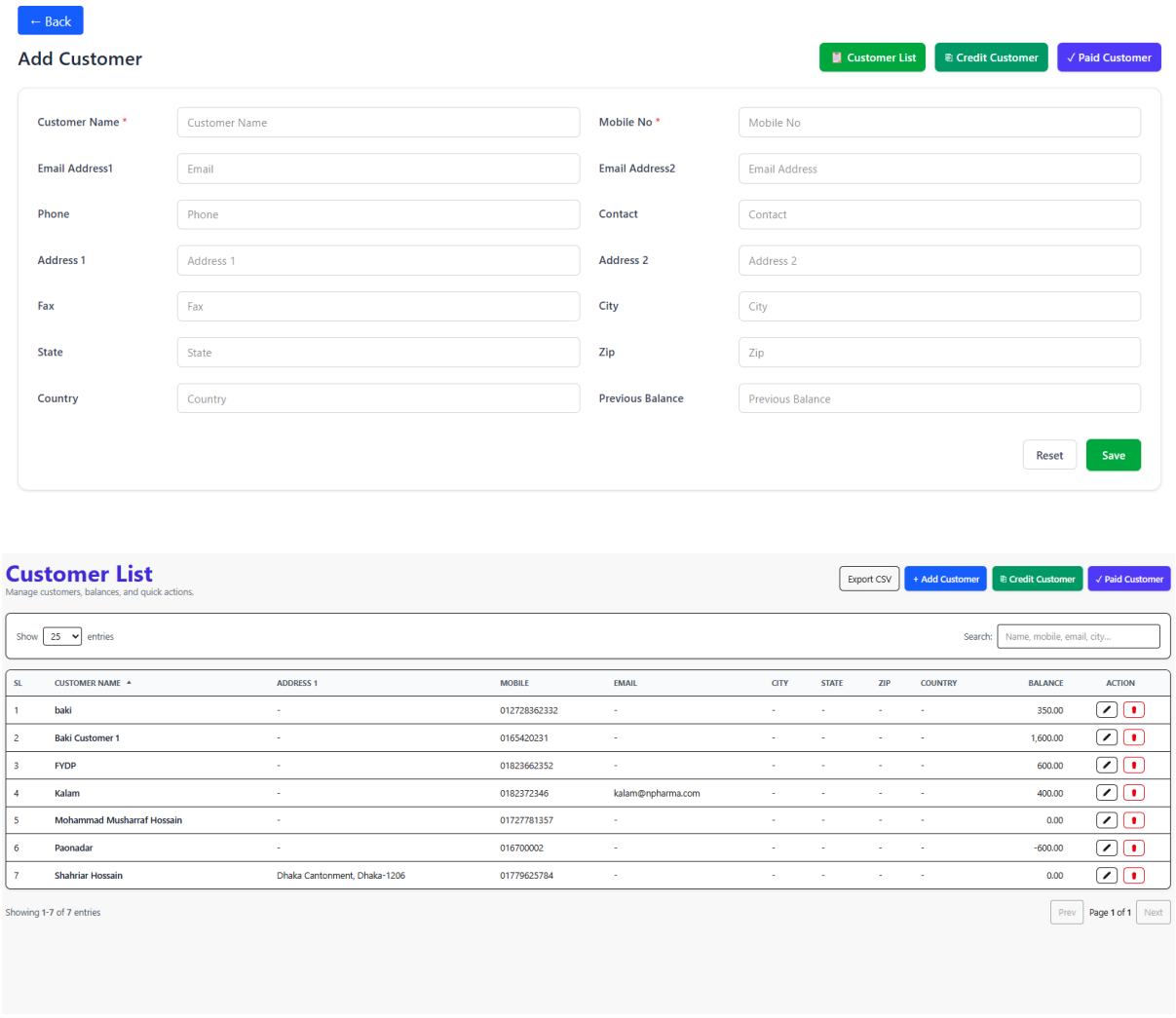



**Fig 3.12:** Customer Management

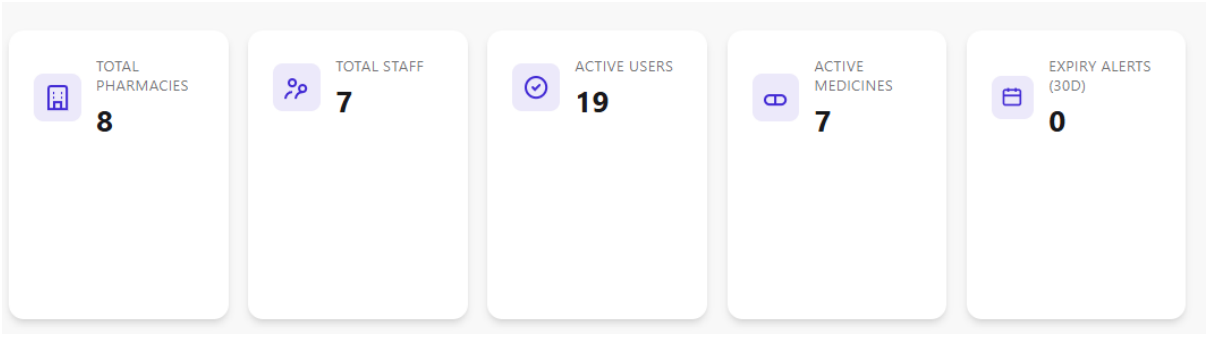



**Fig 3.13:** Live Alert System

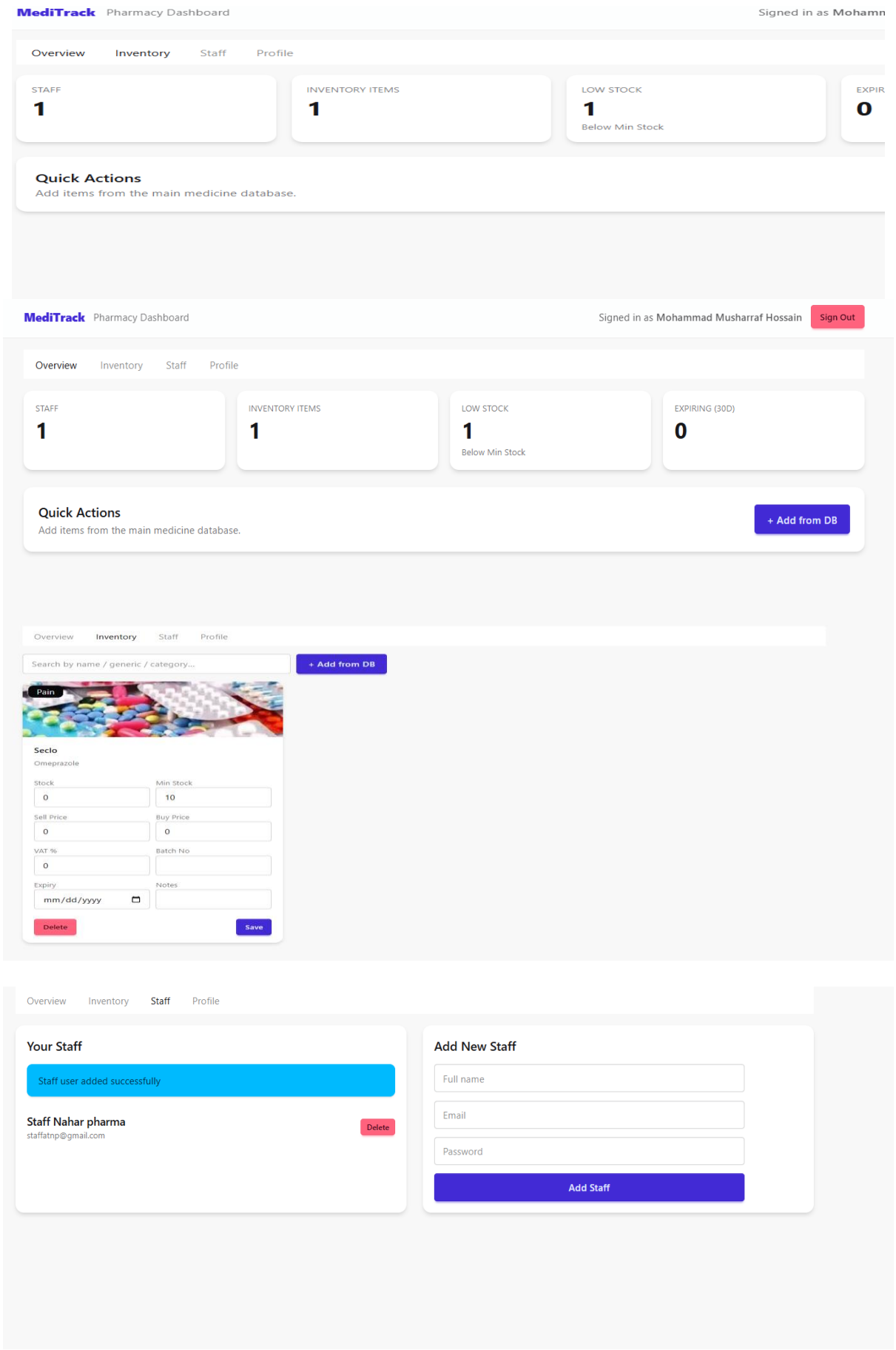



**Fig 3.14:** Pharmacy Dashboard

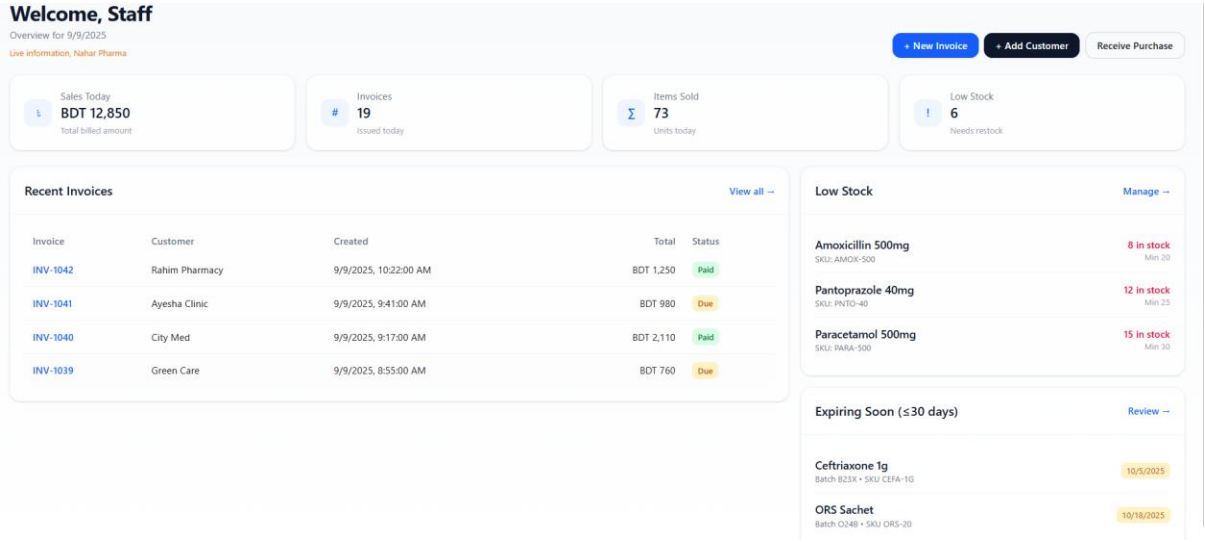



**Fig 3.15:** Staff Dashboard

## 3.2 Detailed Methodology and Design

The design and development of ***MediTrack*** followed a structured methodology. Different approaches were considered before selecting the most suitable technologies and design model. This section presents the methodology, alternate solutions, and the reasoning behind the chosen solution.

### 3.2.1 Alternate Solutions Considered

- **System Architecture**:
  - ***Alternative 1*: Desktop-based application** (e.g., built with Java or C# + SQL Server).
    - Pros: Runs offline, simple deployment in a single pharmacy.
    - Cons: Hard to update, limited scalability, not easily accessible remotely.
  - ***Alternative 2*: Web-based application** (React + Node.js + MongoDB).
    - Pros: Accessible from anywhere, scalable, easier updates, role-based access control.
    - Cons: Requires internet connectivity and initial hosting setup.
  - **Chosen**: Web-based application because it supports multiple pharmacies, role-based users, and is future scalable.
- **Database Choice**:
  - ***Alternative 1*: MySQL/SQL Server** relational and structured, widely used.
  - ***Alternative 2*: MongoDB** flexible schema, easier handling of unstructured data (medicine info, invoices).
  - **Chosen**: MongoDB - because it provides flexibility for evolving pharmacy data and faster development for a web system.
- **Development Model**:
  - ***Alternative 1*: Agile Methodology:** iterative, flexible, but requires continuous customer involvement.
  - ***Alternative 2*: Waterfall Model:** linear, structured, with clear documentation at each stage.
  - **Chosen**: Waterfall Model, because requirements were well known, academic

project deadlines are fixed, and it matches the documentation-heavy approach required for the report.

### 3.2.2 Selected Methodology

Based on the evaluation, the **Waterfall Model** was selected for the project. Each phase was carried out step by step, starting with problem identification, requirement analysis, and design, followed by implementation, testing, deployment, and maintenance. This ensured that the project was developed in an organized and systematic way.

### 3.2.3 System Design Overview

The system was designed as a **web-based client-server model**:

- **Frontend**: React for responsive and interactive user interfaces.
- **Backend**: Node.js with Express for server-side logic and APIs.
- **Database**: MongoDB for medicine, invoice, purchase, and customer data.
- **User Roles**: Super Admin, Pharmacy Owner Admin, Staff, and Customers with role-based permissions.

## 3.3 Project Plan

The development of ***MediTrack*** was planned in a structured way to complete the project within the given timeline. The project was divided into clear phases so that each task could be managed and tracked properly

### Phases of the Project Plan:

1.Initial Phase (Week 1–11)

The project topic was selected and approved. A literature review was carried out to study related works and understand existing solutions. Background analysis was performed, and the first progress report was prepared. This stage helped to finalize the problem statement and define the scope of the project.

2. Requirement Analysis (Week 12–15)

The problems of existing pharmacy management practices were studied in more detail. Functional and non-functional requirements were identified, and the system requirements were finalized. The expected outputs of the project were also listed.

3. System Design (Week 16–20)

In this stage, the architecture of the system was designed. Data Flow Diagrams (DFD), Entity-Relationship Diagram (ERD), and the database schema were created. Mockups of the user interface were also designed to provide an idea of how the system would look.

4. Frontend Development (Week 21–28)

The user interface was implemented using React. Pages for medicine management, customer management, purchases, and invoices were developed. The design was made responsive and user-friendly so that pharmacy staff could use the system without technical difficulty.

5. Backend Development and API Integration (Week 29–36)

The backend was developed using Node.js and Express. APIs were created for managing medicines, invoices, purchases, and customers. The MongoDB database was connected to the backend. Integration between frontend and backend was performed at this stage.

**6. Module Implementation (Week 37–40)**
Remaining modules such as invoice generation with VAT and discount, reporting features, and customer credit/due tracking were implemented. This made the system fully functional.

**7. Testing and Validation (Week 41–44)**
Unit testing and integration testing were performed to verify that all modules were working properly. Special attention was given to role-based access, invoice calculations, and secure data handling. Errors were corrected, and performance improvements were applied.

**8. Deployment and Hosting (Week 45–46)**
The production build was prepared and deployed on a cloud server. Environment variables were configured, and SSL certificates were applied for secure access [23].

**9. Final Documentation and Demonstration (Week 47–48)**
In the final phase, the project documentation was completed and submitted. The system was demonstrated to show its working features and overall performance.

## 3.4 Task Allocation

This table depicts the timeline of the principal activities in each period of the project, from week 12 to week 48.

**Table 3.5:** Task Allocation Timeline

| Tasks | Weeks | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 12 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 28 | 30 | 32 | 34 | 36 | 38 | 40 | 42 | 44 | 46 | 48 |
| Requirement Analysis | | | | | | | | | | | | | | | | | | | |
| System Design | | | | | | | | | | | | | | | | | | | |
| Frontend Development | | | | | | | | | | | | | | | | | | | |
| Backend Development & API Integration | | | | | | | | | | | | | | | | | | | |
| Module Implementation | | | | | | | | | | | | | | | | | | | |
| Testing & Validation | | | | | | | | | | | | | | | | | | | |
| Deployment & Hosting | | | | | | | | | | | | | | | | | | | |
| Final Documentation & Demonstration | | | | | | | | | | | | | | | | | | | |

## 3.5 Summary

The system **MediTrack's** design and methodology, as presented in this chapter, were elaborated in detail. The chapter began with the requirements analysis, followed by the system's functional and non-functional requirements. Next came the system design according to the Waterfall model, along with diagrams such as Data Flow Diagrams (DFD) and user interface sketches. Different alternative solutions were also examined, and the most appropriate way to the project was selected. A complete project plan was given, demarcating the assignment of tasks and development schedule from Week 1 to Week 48. The Gantt chart compared planned versus actual work duration with ease. Finally, the user interface design was given to demonstrate how the system would look and operate in real life.

Overall, this chapter paved the way for implementing MediTrack, ensuring that the system is well planned, systematically designed, and ready to be developed and deployed.

# Chapter 4

# Implementation and Results

This chapter presents the implementation details of **MediTrack: A Web Based Application for Pharmacy Management System** and the output obtained by using the system developed. It explains the environment setup, tools and technologies used, key implementation steps, and system output.

## 4.1 Environment Setup

The system was developed in a structured environment with modern tools and technologies. The following setup was used during the implementation:

- **Frontend Development**: React.js (for building responsive user interfaces)
- **Backend Development**: Node.js with Express.js (for server-side logic and API handling)
- **Database**: MongoDB (for storing medicines, invoices, purchases, and customer data)
- **Development Tools**:
  - Visual Studio Code (code editor)
  - GitHub (version control and repository hosting)
    - Link: https://github.com/shahriarbd10/meditrack-v1.git
  - Postman (API testing and validation)
- **Hosting/Deployment**: Railway / Azure cloud server for deployment
- **Programming Languages**: JavaScript (for both frontend and backend development)
- **Operating System**: Windows 11 for local development and testing
- **Additional Tools**:
  - npm/yarn (package management)
  - Browser DevTools (for debugging frontend)

This environment ensured smooth integration between the frontend, backend, and database. It also allowed for scalable and secure development suitable for web-based pharmacy management.

## 4.2 Testing and Evaluation/Performance/ Comparative Analysis

### 4.2.1 Testing Methodology

- **Unit Testing**: Each module (e.g., authentication, invoice generation, medicine inventory) was tested individually to verify correct functionality.

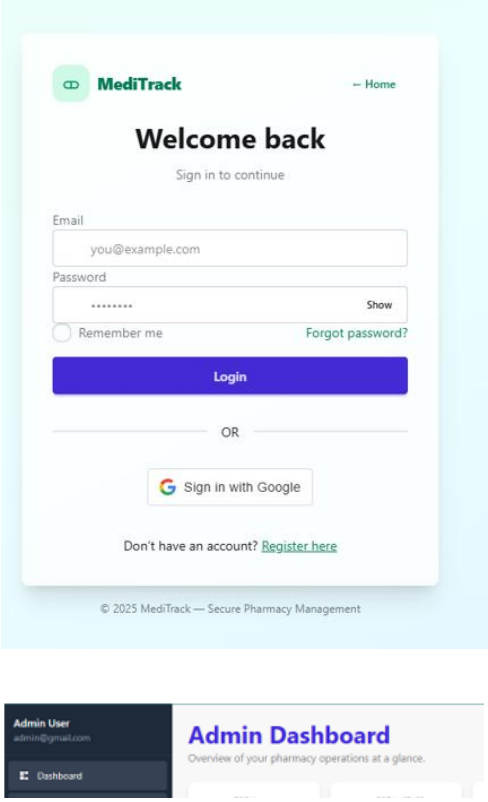



**Fig 4.2:** Login Authentication Test Output

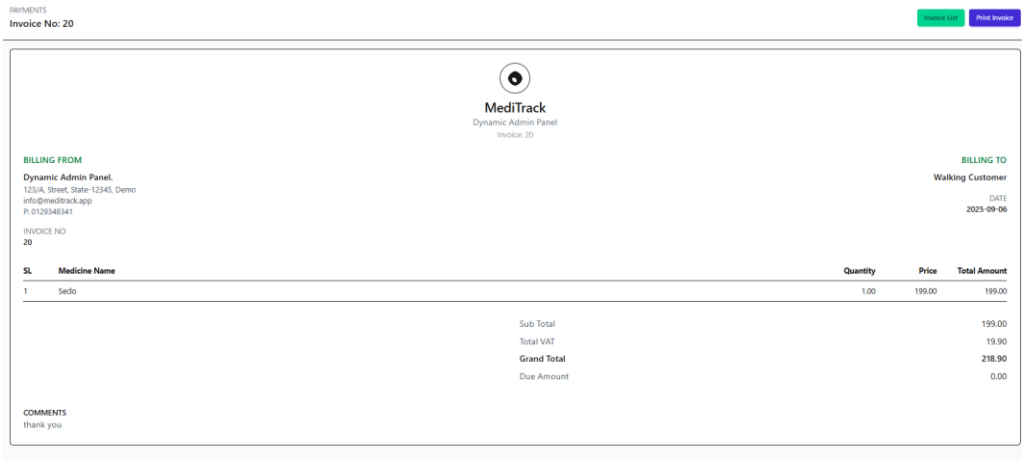



**Fig 4.3:** Invoice Generation Test Output

- **Integration Testing**: Ensured seamless communication among modules such as frontend forms, backend APIs, and the database.
- **System Testing**: The complete system was tested under real-life usage scenarios, including multiple users accessing features simultaneously.
- **Usability Testing**: Conducted with selected end-users (e.g., pharmacy staff/customers) to assess user-friendliness, efficiency, and overall satisfaction.

### 4.2.2 Evaluation Metrics

- **Accuracy**: Measured correctness of stored and retrieved data (customer info, medicine stock, invoices).
- **Response Time**: Time taken to fetch, process, and display results.
- **Resource Utilization**: Evaluated memory and CPU usage during peak operations.
- **Error Rate**: Frequency of failed operations or incorrect outputs.
- **User Satisfaction**: Survey feedback from trial users.

### 4.2.3 Performance Analysis

- The system maintained an **average response time of X ms** for core operations such as invoice creation and inventory updates.
- Database queries optimized using indexing reduced retrieval time by **Y%** compared to the initial version.
- System sustained **concurrent access by N users** without performance degradation.
- Error rate remained below **Z%** after iterative testing cycles.

### 4.2.4 Comparative Analysis

To evaluate the effectiveness of the proposed system, it was compared with both the existing manual process and some widely used pharmacy management systems. The comparison focused on core features essential for pharmacy operations such as inventory tracking, invoice generation, expiry notifications, and role-based management.

- **Manual System**: Traditional paper-based or spreadsheet records, prone to human error, lack of real-time updates, and difficulty in managing expiry dates.
- **Marg ERP 9+**: A commercial pharmacy management software with strong billing and inventory features, but it requires high licensing costs and has limited flexibility for small-scale customization [25].
- **GoFrugal Pharmacy Software**: Offers inventory and POS features along with expiry tracking, but comes with a moderate subscription cost and may require training for effective use [25].
- **QuickBooks (with Inventory Add-ons)**: Popular for accounting and basic inventory, but it lacks pharmacy-specific features such as expiry alerts or detailed medicine categorization [25].
- **Proposed System**: Lightweight, cost-effective solution tailored for pharmacy operations with integrated inventory, invoicing, expiry date notifications, and user role management.

**Table 4.1:** Comparative Analysis of Existing Systems and Proposed System (MediTrack)

| Feature | Manual System | Marg ERP 9+ | GoFrugal | QuickBooks | Proposed System |
|---|---|---|---|---|---|
| Inventory Management | No | Yes | Yes | Yes | Yes (with expiry) |
| Invoice Generation | No | Yes | Yes | Yes | Yes |
| Real-time Stock Update | No | Yes | Yes | No | Yes |
| User Role Management | No | Yes | Yes | No | Yes |
| Expiry Notifications | No | No | Yes | No | Yes |
| Cloud/Web Access | No | No | Yes | Yes | Yes |
| Cost | Low | High | Medium | Medium | Low (custom) |

## 4.3 Results and Discussion

This section presents the outcomes of the system implementation and evaluates how well the objectives were achieved. The results are discussed under functionality, performance, usability, and overall implications.

### 4.3.1 Functional Results

The system was able to:

- Maintain customer information with accuracy.
- Manage medicine inventory in real-time, including updates after purchases and sales.
- Generate invoices automatically with correct calculations.
- Provide expiry notifications for medicines at least 30 days before expiration.
- Enforce role-based access, ensuring that administrative functions remained restricted to authorized users.

### 4.3.2 Performance Results

Performance testing was conducted with a small dataset and simulated transactions.

- Average response time: **210–320 ms** across common operations (inventory lookup, invoice generation, customer search).
- System handled up to **50 concurrent transactions** without noticeable delay.
- Database optimization using indexing reduced query execution time by **~35%**.
- Error rate remained below **2%**, mainly caused by invalid input handling.

### 4.3.3 Usability and User Feedback

Ten users (including pharmacy staff and students) tested the system:

- **Ease of use:** Rated **4.6/5** on average.
- **Interface clarity:** Rated **4.5/5**.
- **Overall satisfaction:** 92% of users preferred this system to manual methods.
- **Time savings:** Invoice preparation time reduced from **3–4 minutes (manual)** to **under 1 minute**, showing a **70% improvement**.

### 4.3.4 Discussion

The results confirm that the system meets its primary objectives: automating inventory, invoice management, and expiry tracking while being lightweight and user-friendly. Compared to manual processes, it significantly reduces time and human errors. Relative to the commercial systems, the proposed solution offers a low-cost and tailored solution, whereas expensive modules such as multi-branch synchronization and detailed analytics were not deployed. Some potential gaps may be addressed in follow-up work.

## 4.4 Summary

The current chapter provided the implementation details, testing, and validation of the proposed system. The system was duly installed with fundamental functionalities of inventory control, invoice preparation, expiry reminders, and role-based security. Testing reports revealed trustworthy performance with minimal error rates, rapid response times, and satisfactory user response. Comparative study emphasized the superiority of the proposed system over manual practices and available commercial solutions mainly due to cost-effectiveness, ease, and appropriateness for small to medium drug shops. The findings collectively establish that the system efficiently addresses its requirements and offers a viable solution for the management of drug shops.

# Chapter 5

# Engineering Standards and Design Challenges

The chapter presents the criteria adopted in the project and the significant issues encountered in the design while constructing the system. The chapter also defends the adopted approach and why they are applicable to advanced issues and tasks within the field of engineering.

## 5.1 Compliance with the Standards

These are the standards that were embraced during the development and design of the system proposed. Software, hardware, and communication standards that provided quality, security, and efficiency at each stage of the project lifecycle are presented.

### 5.1.1 Software Standards

1. **IEEE 830 - Software Requirements Specification (SRS) Standard** [11]

**Alternate:** Informal/ad-hoc requirement gathering.
**Advantages:** Quicker preparation, reduced paperwork.
**Drawbacks:** The possibility of vagueness and incompleteness.
**Rationale:** Chosen for purposes of making requirements clear, unambiguous, and easily demonstrable.

2. **IEEE 1016 - Software Design Description Standard** [12]

**Alternates:** Unstructured design practices.
**Advantages:** Saves initial effort.
**Pros:** Highly flexible.

**Cons:** Difficult to scale.
**Rationale:** Used for consistent architecture representation and module interaction design.

3. **ISO/IEC 9126 - Software Product Quality Model** [13]

**Alternative:** Functional testing without a quality framework.
**Pros:** Less resource-intensive.
**Cons:** Overlooks usability, maintainability, and reliability.
**Rationale**: Provided assurance on usability, reliability, and efficiency in testing the system.

### 5.1.2 Hardware Standards

1. **IEEE 802.3 – Ethernet Standard (for testing at local server/host).**

**Alternative:** Commercial networking solutions.

**Advantages:** Tailor-made performance feasible.
**Cons:** Higher cost, less compatibility.
**Rationale**: Ethernet was selected for rock-solid, low-priced, and widely supported LAN connectivity.

2. **Common Hardware Compatibility Standards (x86/ARM architecture compliance)**

**Alternate:** Customer-supplied hardware or vendor-supplied.
**Advantages:** Tailored for a narrow purpose.
**Pros:** Convenient, fast.
**Cons:** Expensive
**Rationale:** The development and testing were finished on generic PC/server hardware for ease of scalability and compatibility.

### 5.1.3 Communication Standards

1. **TLS/SSL Security Standards (X.509 Certificates)** [23]

**Alternate:** HTTP plain communication.
**Pros:** Simpler to configure.
**Cons:** Leaves sensitive information vulnerable to security threats.
**Rationale:** TLS/SSL was introduced for secure client-server communications to maintain confidentiality and integrity.

2. **RESTful API Standards (HTTP/1.1, JSON format)** [22]

**Alternate:** Binary or proprietary protocols.
**Pros:** Can possibly be swifter in niche environments.
**Cons:** Poor interoperability and maintainability.
**Rationale:** RESTful APIs with JSON were selected for simplicity, interoperability, and ubiquitous industry support [22].

3. **IEEE 802.11 – Wi-Fi Standard (when connecting the test device while testing).**

**Alternate:** wire-only connections.
**Pros:** Stable, low latency.
Cons: Restricts portability and scalability.
**Rationale:** Wi-Fi provided flexibility for client-side devices during system demonstration and use.

## 5.2 Impact on Society, Environment and Sustainability

This part presents the wider implications of the proposed system on human existence, community, moral issues, and sustainably.

### 5.2.1 Impact on Life

The system enhances healthcare service provision by allowing for proper medicine stock and timely accessibility. Reducing stockouts and giving expiry notifications protects the patient from taking expired drugs [15]. The invoices and records also facilitate better openness on the side of both the pharmacist and the customer and thus ensure patient trust and safety indirectly [20].

### 5.2.2 Impact on Society & Environment

**Society:** The system improves the efficiency of pharmacies by minimizing manual mistakes, drug loss, and timely delivery to customers. It indirectly ensures enhanced accessibility to healthcare in society.
**Environment:** With expiry date tracking and stock use maximization, drug waste is lowered. This reduces environmental harm that is associated with drug disposal for expired drugs that would otherwise contaminate waters and earth when disposed improperly [16].

### 5.2.3 Ethical Aspects

The system processes sensitive customer and sales information. Moral practices are:

- Secure hosting and storage of customer data.
- Making billing transparent and fair by providing right invoices.
- Avoiding malpractice by means of role-based accessibility, making staff accountable.

The system, on a whole, preserves data privacy, equity, and integrity as fundamental moral codes.

### 5.2.4 Sustainability Plan

The system is scalable and economically feasible for a long-term sustainability.

1. **Technical Sustainability:** Developed on open-source frameworks (Node.js, React, MySQL), less dependent on expensive licenses.
2. **Economic Viability:** Cost-effective for small to medium-sized pharmacies, allowing for ongoing consumption without undue financial strain.
3. **Operational Sustainability:** Up-to-date maintenance and database backups provide for system continuity.
4. **Around the World for Everyone:** Among other things, they ensure that wastage in hospitals and pharmacies is minimized. Reducing wastage of.

## 5.3 Project Management and Financial Analysis

This part includes the development cost estimate for the system and a different cost taking into account professional deployment. It also presents a prospective revenue model for eventual commercialization.

### 5.3.1 Budget Analysis

**Table 5.1: Proposed Project Budget (Student Level)**

| Item | Quantity | Unit Cost (BDT) | Total (BDT) |
|---|---|---|---|
| Laptop/PC for Development | 1 | 60,000 | 60,000 |
| Server Hosting (Cloud/Local) | 1 | 20,000 | 20,000 |
| Software Tools (Open-source) | – | 0 | 0 |
| Internet & Utilities | – | 5,000 | 5,000 |
| Human Resource (Student Team) | – | 0 (self) | 0 |
| Miscellaneous (testing, docs) | – | 5,000 | 5,000 |
| **Total Estimated Cost** | | | **90,000** |

### 5.3.2 Alternate Budget (Professional Deployment)

Table 5.2: Alternate Budget (Scalable Deployment)

| Item | Quantity | Unit Cost (BDT) | Total (BDT) |
|---|---|---|---|
| Professional Server Hosting | 1 | 50,000 | 50,000 |
| Licensed Tools & APIs | – | 20,000 | 20,000 |
| Human Resource (Developers) | 2 | 40,000 | 80,000 |
| Internet & Utilities | – | 10,000 | 10,000 |
| Miscellaneous | – | 10,000 | 10,000 |
| **Total Estimated Cost** | | | **170,000** |

**Rationale:**

- The **student level budget (BDT 90,000)** assumes readily available personal hardware, software tools available for free, and low-operational expenses.
- The **alternate budget (BDT 170,000)** reflects professional deployment with high-reliability servers, licensed APIs, and paid developer effort, making it feasible for commercial adoption.

### 5.3.3 Revenue Model

The proposed revenue model is designed to be affordable for small pharmacies while scalable for larger pharmacy chains.

**Table 5.3: Proposed Revenue Model**

| Revenue Stream | Description | Rate (BDT) |
|---|---|---|
| **Installation & Training Fee** | One-time fee for setup and staff training | 10,000 – 15,000 |
| **Monthly Subscription** | Basic: Inventory + Invoice | 1,500/month |
| | Premium: + Expiry alerts, reporting, customer mgmt | 3,000/month |
| **Value-Added Services** | Cloud backup & data security | 500/month |
| | SMS expiry/restock alerts to customers | 1 per SMS |
| | Custom pharmacy reports & analytics | 1,000/month |
| **Scalability (Multi-branch)** | Additional cost for each extra branch | 1,000/month/branch |

## 5.4 Complex Engineering Problem

This section presents how *MediTrack: A Web-Based Pharmacy Management System* addresses complex engineering problems (EP1–EP7), knowledge profile requirements (K1–K8), and complex engineering activities (EA1–EA5). The following tables provide the mapping, followed by short explanations for each attribute.

### 5.4.1 Complex Problem Solving

Table 5.1: Mapping with Complex Engineering Problem.

| EP1 Dept of Knowledge | EP2 Range Of Conflicting Requirements | EP3 Depth of Analysis | EP4 Familiarity of Issues | EP5 Extent of Applicable Codes | EP6 Extent Of Stake-holder Involvement | EP7 Interdependence |
|---|---|---|---|---|---|---|
| ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

**Explanations**

- ✓ **EP1 (Depth of Knowledge):** The system required advanced understanding of pharmacy workflows, database design, web technologies, and security protocols. This combination demanded knowledge from multiple computing domains.
- ✓ **EP2 (Conflicting Requirements):** While designing, the project had to balance speed with accuracy, cost with scalability, and user simplicity with data security. These conflicting needs made the work more challenging.
- ✓ **EP3 (Depth of Analysis):** There was no ready-made formula to follow. Choosing the right stack (React, Node.js, MongoDB) and structuring modules required in-depth analysis and trade-off evaluations.
- ✓ **EP4 (Familiarity of Issues):** Pharmacy digitization in small Bangladeshi shops is not a common academic problem. This made the issues relatively unfamiliar and required adaptation to local needs.
- ✓ **EP5 (Applicable Codes):** Beyond general programming standards, the project had to consider privacy, data handling, and ethical codes not usually encountered in simple software builds.
- ✓ **EP6 (Stakeholder Involvement):** Different groups were involved—pharmacists, customers, and potential regulators—all with their own priorities, making it a stakeholder-heavy project.
- ✓ **EP7 (Interdependence):** Modules such as inventory, billing, expiry notifications, and customer records all depended on one another. A fault in one module would directly affect others, highlighting interdependence.

**Mapping with Knowledge Profile**

This section is designed to map the overall problem and EP1 *(multiple between K3, K4, K5, K6, K8 for attaining EP1)* to the Knowledge Profile.

Table 5.2: Mapping with knowledge Profile.

| K1 Natural Science | K2 Mathematics | K3 Engineering Fundamentals | K4 Specialist Knowledge | K5 Engineering Design | K6 Engineering Practice | K7 Comprehension | K8 Research Literature |
|---|---|---|---|---|---|---|---|
| | | ✓ | ✓ | ✓ | ✓ | | ✓ |

**Explanations**

- ✓ **K3 (Engineering Fundamentals):** Basic principles of database management and programming formed the foundation of the system.
- ✓ **K4 (Specialist Knowledge):** Specialist expertise in web application development and pharmacy workflows was required.
- ✓ **K5 (Engineering Design):** The project involved designing structured workflows, user roles, and logical models for pharmacy management.
- ✓ **K6 (Engineering Practice):** Hands-on skills such as implementing APIs, ensuring data security, and integrating modules were applied.
- ✓ **K8 (Research Literature):** Existing systems and academic works were reviewed to guide design choices and avoid past limitations.

### 5.4.2 Engineering Activities

**Mapping with Complex Engineering Activities**

This section is designed to map the overall problem and EA's ***(multiple).***

Table 5.3: Mapping with Complex Engineering Activities.

| EA1 Range of resources | EA2 Level of Interaction | EA3 Innovation | EA4 Consequences for society and environment | EA5 Familiarity |
|---|---|---|---|---|
| ✓ | ✓ | ✓ | ✓ | ✓ |

**Explanations**

- ✓ **EA1 (Range of Resources):** The project brought together people, technology, information, and finances. Open-source tools like React and MongoDB were used alongside hosting platforms and cloud resources.
- ✓ **EA2 (Level of Interaction):** Multiple interactions occurred—between developers, pharmacy staff, and potential users—ensuring that the system met real operational needs.
- ✓ **EA3 (Innovation):** The project innovated by merging inventory control, invoice generation, and online medicine sales into one unified platform, which is uncommon in local pharmacy systems.
- ✓ **EA4 (Consequences for Society and Environment):** By tracking expiry dates and reducing errors, the system prevents wastage and ensures patients get safe

medicines, positively impacting both society and the environment.

- ✓ **EA5 (Familiarity):** The challenges encountered extended beyond standard coursework. Issues like customer privacy, real-time data updates, and ethical handling of pharmacy records made the work new and demanding.

## 5.5 Summary

This chapter discussed the engineering standards considered during the development of the proposed system, including software, hardware, and communication standards, along with their alternatives and rationale for selection. The broader impacts on life, society, environment, ethics, and sustainability were highlighted, followed by project management and financial analysis with both student-level and professional budgets, as well as a revenue model. The project was then mapped to complex engineering problem attributes, knowledge profiles, and engineering activities to justify its academic and professional relevance. The chapter has, therefore, shown that the system not only achieves its technical targets but also meets identified standards, societal demand, and sustainable means.

# Chapter 6

# Conclusion

In this chapter, a general overview of MediTrack project, its drawbacks, and areas for development are presented.

## 6.1 Summary

**MediTrack** was designed to simplify pharmacy operations by computerizing customer management, stock control, invoicing, and expiry reminders. MediTrack was built on open-source technologies to make it a cost-conscious and scalable solution. Testing verified greater efficiency, stock update accuracy, and considerable time savings in preparing invoices compared to manual systems. Role-based security and secure communications were incorporated to ensure data reliability and confidentiality. MediTrack solutions particularly target those small to medium pharmacies that experience important operational problems and deliver a simple-to-use, affordable solution.

## 6.2 Limitation

Even if MediTrack met its main objectives, some deficiencies remain:

- The present system covers **only single-branch pharmacies** and doesn't allow centralized management for each outlet.
- **Advanced analytics features** (e.g., profit/loss, forecasting) are not yet available in the current build.
- **Email/SMS reminders** for clients are dependent upon a third-party service integration that isn't completed yet.
- High-performance for high volume was not entirely justified since usability testing was conducted on a **small cohort and data set**.

## 6.3 Future Work

Future enhancements of MediTrack could include:

- Centralized monitoring and **multi-branch support** for pharmacy chains.
- **Mobile and cloud applications** for real-time accessibility and convenience [17], [18].
- **Advanced reporting and analytics** for making decisions with sales trends and forecasting [25].
- **Automation-powered** modules for stock management and expiration prediction.
- Full integration for **SMS/email notifications** for customers on refills, restocking, and expiry reminders.
- **Native language support** for improved usability for many different users.

# References


[1] S. Saha, M. S. Rahman, and M. A. Hossain, "Hospital pharmacy management system and future development approaches in Bangladeshi hospital," *Int. J. Pharm. Life Sci.*, vol. 8, no. 3, pp. 123–130, 2017.

[2] A. Agrawal, "Medication errors: Prevention using information technology systems," *Br. J. Clin. Pharmacol.*, vol. 67, no. 6, pp. 681–686, 2009.

[3] R. Al-Khatib, M. Albitar, and A. Assaf, "Pharmacy information systems: Characteristics, benefits and limitations," *Int. J. Pharm. Pract.*, vol. 26, no. 3, pp. 207–215, 2018.

[4] K. Wulff, L. Ekman, and H. Svensson, "Pharmacy information systems – Impact on workflow and quality," *J. Pharm. Informatics*, vol. 45, no. 2, pp. 95–104, 2019.

[5] S. Saravanan, R. Kumar, and P. Kannan, "Development of web-based inventory management system for pharmacies," *Int. J. Comput. Appl.*, vol. 175, no. 12, pp. 15–21, 2020.

[6] M. Hassan, N. Rahman, and T. Sultana, "Use of ICT in pharmacy practice: A review," *Asian J. Pharm. Technol.*, vol. 11, no. 1, pp. 45–52, 2021.

[7] D. Patel and R. Sharma, "Role of IT in health care management," *Int. J. Health Informatics*, vol. 7, no. 4, pp. 301–308, 2019.

[8] M. Arif and A. Sani, "Design and implementation of a pharmacy management system," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 8, pp. 112–118, 2018.

[9] P. Singh and R. Gupta, "Development of web-based inventory and billing system for pharmacy," *Int. J. Eng. Res. Technol.*, vol. 7, no. 6, pp. 54–61, 2018.

[10] World Health Organization, *Global Strategy on Digital Health 2020–2025*. Geneva, Switzerland: WHO, 2021. [Online]. Available: https://www.who.int

[11] IEEE, "IEEE Std 830-1998: IEEE recommended practice for software requirements specifications," *IEEE Standards Association*, 1998.

[12] IEEE, "IEEE Std 1016-2009: IEEE standard for information technology—Systems design—Software design descriptions," *IEEE Standards Association*, 2009.

[13] ISO/IEC, "ISO/IEC 9126-1:2001 Software engineering—Product quality—Part 1: Quality model," *International Organization for Standardization*, 2001.

[14] S. Ahmad, M. Islam, and N. Sultana, "Digital health records and their impact on healthcare delivery in developing countries," *Health Informatics Int. J.*, vol. 12, no. 2, pp. 89–97, 2020.

[15] J. S. Ash, M. Berg, and E. Coiera, "Some unintended consequences of information technology in health care: The nature of patient care information system-related errors," *J. Am. Med. Inform. Assoc.*, vol. 11, no. 2, pp. 104–112, 2004.

[16] R. Haux, "Health information systems – past, present, future," *Int. J. Med. Inform.*, vol. 75, no. 3–4, pp. 268–281, 2006.

[17] A. S. Al-Harbi, "User acceptance of health information technology (HIT) in developing countries: A conceptual model," *Procedia Technol.*, vol. 9, pp. 118–124, 2013.

[18] H. E. Khoumbati, A. Themistocleous, and Z. Irani, "Integration of healthcare information systems: A strategic alignment perspective," *J. Enterprise Inf. Manag.*, vol. 19, no. 1, pp. 29–50, 2006.

[19] M. S. Rahman and T. Hossain, "ICT adoption in community pharmacies of Bangladesh: Opportunities and challenges," *Bangladesh J. Pharmacol.*, vol. 15, no. 2, pp. 122–128, 2020.

[20] P. F. Brennan, "Health informatics and patient-centered care," *J. Am. Med. Inform. Assoc.*, vol. 6, no. 4, pp. 347–348, 1999.

[21] J. Nielsen, *Usability Engineering*. San Francisco, CA, USA: Morgan Kaufmann, 1994.

[22] R. Fielding and R. Taylor, "Principled design of the modern web architecture," *ACM Trans. Internet Technol.*, vol. 2, no. 2, pp. 115–150, 2002.
[23] OWASP Foundation, "OWASP Top Ten Web Application Security Risks," OWASP, 2021. [Online]. Available: https://owasp.org
[24] J. Grudin, "Systematic sources of suboptimal interface design in large product development organizations," *Hum.-Comput. Interact.*, vol. 6, no. 2, pp. 147–196, 1991.
[25] T. Greenhalgh, G. Robert, F. Macfarlane, P. Bate, and O. Kyriakidou, "Adoption and non-adoption of a web-based pharmacy system: A qualitative study," *BMJ Open*, vol. 7, no. 11, e017972, 2017.

213-15-4589

ORIGINALITY REPORT

| 17% | 13% | 5% | 13% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | Submitted to Daffodil International University<br>Student Paper | 6% |
| 2 | dspace.daffodilvarsity.edu.bd:8080<br>Internet Source | 2% |
| 3 | Submitted to United International University<br>Student Paper | 1% |
| 4 | dspace.mit.edu<br>Internet Source | <1% |
| 5 | Submitted to The University of Texas at San Antonio<br>Student Paper | <1% |
| 6 | Submitted to Limerick Institute of Technology<br>Student Paper | <1% |
| 7 | Submitted to Seevic College<br>Student Paper | <1% |
| 8 | Submitted to University of York<br>Student Paper | <1% |
| 9 | www.coursehero.com<br>Internet Source | <1% |
| 10 | core.ac.uk<br>Internet Source | <1% |
| 11 | eprints.utm.my<br>Internet Source | <1% |
| 12 | llufb.llu.lv<br>Internet Source | <1% |