PHARMACY MANAGEMENT SYSTEM

A MINI PROJECT REPORT

SUBMITTED BY

MONISH E	**2116221701038**
RAHUL SANJAY J	**2116221701045**
SOLAINARAYANAN K S	**2116221701057**
LOKESHWAR S R	**2116221701504**

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING DESIGN

RAJALAKSHMI ENGINEERING COLLEGE
THANDALAM
CHENNAI – 602105

ANNA UNIVERSITY: CHENNAI 600625

BONAFIDE CERTIFICATE

Certified that this project report **"PHARMACY MANAGEMENT SYSTEM"** is the Bonafide work of **"MONISH E (2116221701038) , RAHUL SANJAY J (2116221701045) , SOLAINARAYANAN K S (2116221701057) , LOKESHWAR S R (2116221701504)**" who carried out the project work under my supervision.

SIGNATURE

Dr.P.Kumar M.E., Ph.D.,
Professor and Head,
Computer Science and Engineering,
Rajalakshmi Engineering College,
Thandalam, Chennai – 602105.

SIGNATURE

Mr.Vijaykumar M.Tech.,
Asst. Professor (SS),
Computer Science and Engineering,
Rajalakshmi Engineering College,
Thandalam, Chennai – 602105.

EXTERNAL EXAMINER

INTERNAL EXAMINER

ACKNOWLEDGEMENT

We are highly obliged in taking the opportunity to thank our Chairman **Mr. S. Meganathan**, Chairperson **Dr.Thangam Meganathan** and our Principal **Dr.S.N.Murugesan** for providing all the facilities which are required to carry out this project work.

We are ineffably indebted to our H.O.D **Dr.P.Kumar M.E., Ph.D.,** for his conscientious guidance and encouragement to make this project a recognizable one.

We are extremely thankful to our faculty **Mr.Vijaykumar M.Tech.,** for his valuable guidance and indefatigable support and extend our heartfelt thanks to all the teaching and non-teaching staff of **Computer Science department** who helped us directly or indirectly in the completion of this project successfully.

At last but not least gratitude goes to our friends who helped us compiling the project.

Any omission in this brief acknowledgement doesn't mean lack of gratitude.

MONISH E	**2116221701038**
RAHUL SANJAY J	**2116221701045**
SOLAINARAYANAN K S	**2116221701057**
LOKESHWAR S R	**2116221701504**

ABSTRACT

The Pharmacy Management System is a web-based application designed to streamline medication inventory management and customer purchase tracking. Developed using Flask and SQLite, this system allows users to add, update, and delete medications, manage customer details, and handle purchases while maintaining accurate stock levels. The project demonstrates the application of CRUD operations, web frameworks, and database management, catering to small-scale pharmacies' needs..

TABLE OF CONTENTS

CHAPTER – 1

INTRODUCTION

Pharmacies play a critical role in healthcare, requiring efficient systems to manage inventory and customer transactions. Traditional methods often involve manual record-keeping, which is prone to errors and inefficiencies. The Pharmacy Management System addresses these challenges by providing a user-friendly platform to manage medication details, track inventory, and record customer purchases digitally. The system is designed for ease of use and ensures data accuracy, making it an invaluable tool for pharmacy operations..

1.1 SCOPE OF THE WORK

The project focuses on:-

1. Managing medication details, including name, category, price, and stock quantity.
2. Recording customer transactions and tracking purchased medications.
3. Automating stock updates after purchases to avoid discrepancies.
4. Providing a responsive and interactive user interface for seamless operations.
5. Supporting scalability for future enhancements, such as integrating analytics or e-commerce features.

1.2 PROBLEM STATEMENT

Manual inventory and customer management in pharmacies often lead to:

- Errors in stock tracking.
- Inefficient customer service.
- Time-consuming record-keeping processes.
- Challenges in handling bulk data and generating reports. This project aims to mitigate these issues by developing a robust, digital solution for pharmacy management.

CHAPTER – 2

SYSTEM SPECIFICATION

2.1 HARDWARE AND SOFTWARE SPECIFICATIONS

The following hardware and software are required:-

HARDWARE REQUIREMENTS:-

- Processor: Intel Core i5 or equivalent
- RAM: 8GB or higher
- Storage: 500GB HDD/SSD
- Display: 1280x720 resolution minimum

SOFTWARE REQUIREMENTS:-

- Operating System: Windows 10, Linux, or macOS
- Programming Language: Python 3.10+
- Framework: Flask 2.x
- Database: SQLite 3
- Tools: VS Code, Browser (for testing), SQLite Browser (optional)

CHAPTER - 3

SOFTWARE DESCRIPTION:

3.1 FEATURES

Python

Python is a high-level, interpreted, and general-purpose programming language known for its simplicity and readability. It is widely used in web development, data science, artificial intelligence, and software development due to its extensive libraries and community support. For this project, Python serves as the foundation for building the backend logic of the Pharmacy Management System.

- **Features of Python**:
  - **Ease of Use**: Simple syntax and structure make it easy to learn and implement.
  - **Cross-Platform**: Compatible with major operating systems like Windows, Linux, and macOS.
  - **Rich Libraries**: Offers powerful libraries like `os`, `sqlite3`, and `math` for diverse functionality.
  - **Dynamic Typing**: Allows variable declarations without specifying data types explicitly.
  - **Scalability**: Supports both small-scale scripts and large, complex applications.
- **Role in the Project**:
  - Acts as the programming language for writing the application logic.
  - Handles data operations such as fetching, inserting, updating, and deleting records in the SQLite database using the `sqlite3` library.

- Manages Flask’s routing and data handling functionalities.

Database:- Sqlite

SQLite is a lightweight, serverless, self-contained SQL database engine. It is a popular choice for applications requiring simple, reliable, and efficient data storage without the need for a separate database server.

- **Features of SQLite**:
  - **Serverless**: Operates directly from a file without needing a dedicated server.
  - **Lightweight**: Minimal setup and low overhead, making it ideal for small applications.
  - **Self-Contained**: All database logic is encapsulated in a single library.
  - **ACID Compliance**: Ensures reliable transactions with robust data integrity.
  - **Platform Independent**: Can be used on various operating systems seamlessly.
- **Role in the Project**:
  - Stores and manages data for medications and customer transactions in structured tables.
  - Supports SQL operations such as:
    - Creating tables (e.g., `medications`, `customers`).
    - Performing CRUD (Create, Read, Update, Delete) operations.
  - Acts as the core storage component for inventory and transaction data.

Flask

Flask is a lightweight, micro web framework written in Python. It provides developers with the tools necessary to build web applications, APIs, and more, while maintaining flexibility and scalability.

- **Features of Flask**:
  - **Micro Framework**: Minimalistic by design but highly extensible via plugins.
  - **Routing System**: Supports URL mapping to associate web pages with functions.
  - **Templating Engine**: Uses Jinja2 for dynamic HTML rendering.
  - **Integrated Development Server**: Includes a debugger for testing and debugging during development.
  - **RESTful Request Handling**: Built-in support for HTTP methods like GET, POST, PUT, and DELETE.
- **Role in the Project**:
  - **Routing**: Maps URLs to specific Python functions to handle requests (e.g., `/medications`, `/add_medication`).
  - **Dynamic Rendering**: Combines data from Python with HTML templates for an interactive user interface.
  - **Form Handling**: Processes form submissions for actions like adding medications or customer purchases.
  - **Database Integration**: Interfaces seamlessly with SQLite to fetch and display data on the web pages.
  - **Error Handling**: Provides mechanisms to catch and report errors during user interactions or data processing.

CHAPTER - 4

PROJECT DESCRIPTION

The Pharmacy Management System provides an all-in-one solution for managing medications and customers in small-scale pharmacies. It leverages Flask's lightweight framework for the backend and SQLite for data persistence, ensuring simplicity and reliability. With features like stock adjustment, purchase tracking, and customer management, the system minimizes manual errors and improves operational efficiency.

4.1 MODULE DESCRIPTION

1. **Home Module**: Displays an overview of popular medicines and their prices.
2. **Medications Module**: Manages CRUD operations and lists all medicine details.
3. **Add Medication Module**: Allows adding new medicines to the inventory.
4. **Update Quantity Module**: Enables stock increment or decrement.
5. **Customer & Purchase Module**: Handles purchases, updates stock, and logs customer transactions.

CHAPTER - 5

IMPLEMENTATION

5.1 System Workflow:-

1.Backend:

- Developed using Flask for routing and data handling.
- SQLite is used as the database for storing medication and customer details.

2.Frontend:

- HTML and Jinja templates for dynamic rendering.
- Simple forms for input and interactive buttons for operations.

3.Database:

- Two tables: `medications` for inventory and `customers` for transaction history.

4.Source Code:

- See the provided script for full implementation.

5.2 Source Code:-

app.py-

```
from flask import Flask, render_template, request, redirect, url_for
import sqlite3

app = Flask(__name__)

def get_db_connection():
    conn = sqlite3.connect('pharmacy.db')
    conn.row_factory = sqlite3.Row
    return conn

@app.route('/')
def home():
    medicines = [
```

```
        {"name": "Aspirin", "price": 50, "description": "Pain reliever"},
        {"name": "Paracetamol", "price": 30, "description": "Fever reducer"},
        {"name": "Amoxicillin", "price": 120, "description": "Antibiotic"},
    ]
    return render_template('home.html', medicines=medicines)

@app.route('/medications')
def medications():
    conn = get_db_connection()
    medications = conn.execute("SELECT * FROM medications").fetchall()
    conn.close()
    return render_template('medications.html', medications=medications)

@app.route('/add_medication', methods=['GET', 'POST'])
def add_medication():
    if request.method == 'POST':
        name = request.form['name']
        category = request.form['category']
        price = request.form['price']
        quantity = request.form['quantity']

        conn = get_db_connection()
        conn.execute("INSERT INTO medications (name, category, price, quantity) VALUES (?, ?, ?, ?)",
                     (name, category, price, quantity))
        conn.commit()
        conn.close()
        return redirect(url_for('medications'))
    return render_template('add_medication.html')

@app.route('/update_quantity/<int:med_id>', methods=['POST'])
def update_quantity(med_id):
    action = request.form['action']

    conn = get_db_connection()
    medication = conn.execute("SELECT quantity FROM medications WHERE id = ?", (med_id,)).fetchone()

    if medication:
        current_quantity = medication['quantity']
        if action == "increase":
            new_quantity = current_quantity + 1
        elif action == "decrease" and current_quantity > 0:
            new_quantity = current_quantity - 1
        else:
            new_quantity = current_quantity

        conn.execute("UPDATE medications SET quantity = ? WHERE id = ?", (new_quantity, med_id))
        conn.commit()
    conn.close()

    return redirect(url_for('medications'))

@app.route('/customers')
def customers():
    conn = get_db_connection()
    customers = conn.execute("SELECT * FROM customers").fetchall()
    conn.close()
    return render_template('customers.html', customers=customers)

@app.route('/delete_medication/<int:med_id>', methods=['POST'])
def delete_medication(med_id):
    conn = get_db_connection()
    conn.execute("DELETE FROM medications WHERE id = ?", (med_id,))
    conn.commit()
    conn.close()
    return redirect(url_for('medications'))

@app.route('/purchase_medication/<int:medication_id>', methods=['POST'])
```

```
def purchase_medication(medication_id):
    customer_name = request.form['customer_name']
    purchase_quantity = int(request.form['quantity'])

    conn = get_db_connection()
    medication = conn.execute('SELECT name, price, quantity FROM medications WHERE id = ?',
(medication_id,)).fetchone()

    if medication and medication['quantity'] >= purchase_quantity:
        new_quantity = medication['quantity'] - purchase_quantity
        total_price = medication['price'] * purchase_quantity

        conn.execute('UPDATE medications SET quantity = ? WHERE id = ?', (new_quantity, medication_id))
        conn.execute('INSERT INTO customers (name, purchased_items, quantity, total_price) VALUES (?, ?, ?,
?)',
                     (customer_name, medication['name'], purchase_quantity, total_price))

        conn.commit()
        conn.close()
        return redirect(url_for('medications'))
    else:
        conn.close()
        return "Insufficient stock", 400

'''
@app.route('/add_customer', methods=['GET', 'POST'])
def add_customer():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        phone = request.form['phone']
        purchased_items = request.form['purchased_items']

        conn = get_db_connection()
        conn.execute("INSERT INTO customers (name, email, phone, purchased_items) VALUES (?, ?, ?, ?)",
                     (name, email, phone, purchased_items))
        conn.commit()
        conn.close()
        return redirect(url_for('customers'))
    return render_template('add_customer.html')
'''

if __name__ == '__main__':
    app.run(debug=True)
```

create_db.py-

```
import sqlite3
conn = sqlite3.connect('pharmacy.db')
cursor = conn.cursor()
cursor.execute('''
    CREATE TABLE IF NOT EXISTS medications (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT NOT NULL,
        category TEXT NOT NULL,
        price REAL NOT NULL,
        quantity INTEGER NOT NULL
    )
''')

initial_data = [
    ('Aspirin', 'Pain Relief', 50.0, 100),
    ('Paracetamol', 'Fever Relief', 30.0, 200),
```

```
    ('Ibuprofen', 'Anti-inflammatory', 60.0, 150)
]

cursor.executemany('INSERT INTO medications (name, category, price, quantity) VALUES (?, ?, ?,
?)', initial_data)
conn.commit()
conn.close()
```

create table.py-

```
import sqlite3


# Connect to the database
conn = sqlite3.connect('pharmacy.db')
cursor = conn.cursor()

# Drop the existing 'customers' table if it exists
cursor.execute("DROP TABLE IF EXISTS customers;")
conn.commit()

# Create the new 'customers' table with the correct structure
cursor.execute('''
    CREATE TABLE customers (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT NOT NULL,
        purchased_items TEXT NOT NULL,
        quantity INTEGER NOT NULL,
        total_price REAL NOT NULL
    );
''')
conn.commit()

# Close the connection
conn.close()

print("The 'customers' table has been dropped and recreated successfully.")
```

home.html-

```
{% extends "base.html" %}

{% block content %}
    <header class="header">
        <h1>Welcome to the Pharmacy Management System</h1>
        <p>Your health, our priority!</p>
    </header>

    <h2>Available Medicines</h2>
    <div class="medicine-list">
        <ul>
            {% for medicine in medicines %}
                <li class="medicine-item">
                    <strong>{{ medicine.name }}</strong><br>
                    Price: ₹{{ medicine.price }}<br>
                    Description: {{ medicine.description }}
                </li>
            {% endfor %}
        </ul>
    </div>
    <a href="/customers" class="button">View Customers</a>
{% endblock %}
```

medications.html-

```
{% extends "base.html" %}

{% block title %}Medications - Pharmacy Management{% endblock %}

{% block content %}
<div class="tab-container">
```

```
    <h2>Available Medications</h2>
    <div class="medication-cards-container">
        {% for medication in medications %}
        <div class="medication-card">
            <div class="medication-card-header">
                <h3>{{ medication[1] }}</h3>
                <p><strong>Category:</strong> {{ medication[2] }}</p>
                <p><strong>Price:</strong> ₹{{ medication[3] }}</p>
            </div>

            <div class="medication-card-body">
                <p><strong>Quantity:</strong>
                    <button onclick="adjustQuantity({{ medication[0] }}, -1)">-</button>
                    <span id="quantity-{{ medication[0] }}">{{ medication[4] }}</span>
                    <button onclick="adjustQuantity({{ medication[0] }}, 1)">+</button>
                </p>
            </div>

            <div class="purchase-form">
                <form action="{{ url_for('purchase_medication', medication_id=medication[0]) }}"
method="post">
                    <label for="customer_name">Customer Name:</label>
                    <input type="text" name="customer_name" required>

                    <label for="quantity">Purchase Quantity:</label>
                    <input type="number" name="quantity" min="1" required>

                    <div class="buttons-container">
                        <button type="submit">Purchase</button>
                        <form action="{{ url_for('delete_medication', med_id=medication[0]) }}"
method="post" style="display:inline;">
                            <button type="submit" class="delete-btn">Delete</button>
                        </form>
                    </div>
                </form>
            </div>
        </div>
        {% endfor %}
    </div>
</div>

<script>
    function adjustQuantity(medicationId, delta) {
        const quantityElement = document.getElementById(`quantity-${medicationId}`);
        let currentQuantity = parseInt(quantityElement.textContent);
        currentQuantity += delta;
        if (currentQuantity < 0) currentQuantity = 0;
        quantityElement.textContent = currentQuantity;
    }
</script>
{% endblock %}
```

customer.html-

```
{% extends "base.html" %}

{% block title %}Customer Purchases{% endblock %}

{% block content %}
<h2>Customer Information</h2>


<table class="customers-table">
    <thead>
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Purchased Items</th>
            <th>Quantity</th>
            <th>Total Price</th>
        </tr>
    </thead>
    <tbody>
        {% for customer in customers %}
            <tr>
                <td>{{ customer.id }}</td>
```

```
                <td>{{ customer.name }}</td>
                <td>{{ customer.purchased_items }}</td>
                <td>{{ customer.quantity }}</td>
                <td>₹{{ customer.total price }}</td>
            </tr>
        {% endfor %}
    </tbody>
</table>
{% endblock %}
```

base.html-

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Pharmacy Management</title>
    <link rel="stylesheet" href="{{ url for('static', filename='css/styles.css') }}">
</head>
<body>
    <nav>
        <a href="/">Home</a>
        <a href="/medications">Medications</a>
        <a href="/add_medication">Add Medication</a>
        <a href="/customers">Customers</a>
    </nav>
    <div class="container">
        {% block content %}{% endblock %}
    </div>
</body>
</html>
```

add_medication.html-

```
{% extends "base.html" %}

{% block title %}Add Medication{% endblock %}

{% block content %}
<h2>Add Medication</h2>


<form action="{{ url for('add medication') }}" method="POST" class="form-container">
    <label for="name">Medication Name:</label>
    <input type="text" id="name" name="name" required>

    <label for="category">Category:</label>
    <input type="text" id="category" name="category" required>

    <label for="price">Price (₹):</label>
    <input type="number" id="price" name="price" required>

    <label for="quantity">Quantity:</label>
    <input type="number" id="quantity" name="quantity" required>

    <button type="submit" class="submit-btn">Add Medication</button>
</form>
{% endblock %}
```

add_customers.html-

```
{% extends "base.html" %}

{% block content %}
    <h1>Add Customer</h1>
    <form method="POST" action="/add_customer">
        <label for="name">Name:</label>
        <input type="text" name="name" required>
        <label for="email">Email:</label>
        <input type="email" name="email" required>
        <label for="phone">Phone:</label>
        <input type="text" name="phone" required>
```

```
        <label for="purchased_items">Purchased Items:</label>
        <input type="text" name="purchased_items" required>
        <button type="submit">Add Customer</button>
    </form>
{% endblock %}
```

styles.css-

```
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
}

nav {
  background-color: #007bff;
  padding: 10px;
  text-align: center;
}

nav a {
  color: white;
  margin: 0 15px;
  text-decoration: none;
  font-weight: bold;
}

.container {
  padding: 20px;
}

h2 {
  text-align: center;
}

.medication-list {
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
  padding: 20px;
  gap: 20px;
}

.medication-card {
  border: 1px solid #ddd;
  border-radius: 8px;
  padding: 15px;
  width: 200px;
  text-align: center;
  box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
  transition: box-shadow 0.3s ease;
}

.medication-card:hover {
  box-shadow: 0px 8px 16px rgba(0, 0, 0, 0.2);
}

.delete-button {
  margin-top: 10px;
  padding: 8px 12px;
  font-size: 14px;
  color: #fff;
  background-color: #dc3545;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.delete-button:hover {
  background-color: #c82333;
}
body {
  font-family: Arial, sans-serif;
  margin: 20px;
  background-color: #f4f4f4;
```

```
}

.header {
  background-color: #4CAF50;
  color: white;
  padding: 20px;
  text-align: center;
  border-radius: 8px;
}

h1 {
  margin: 0;
}

h2 {
  color: #2c3e50;
}

.medicine-list {
  margin-top: 20px;
}

ul {
  list-style-type: none;
  padding: 0;
}

.medicine-item {
  margin: 10px 0;
  padding: 15px;
  border: 1px solid #ddd;
  border-radius: 5px;
  background-color: white;
  transition: box-shadow 0.3s;
}

.medicine-item:hover {
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
}

.button {
  display: inline-block;
  margin-top: 20px;
  padding: 10px 20px;
  background-color: #008CBA;
  color: white;
  text-decoration: none;
  border-radius: 5px;
  transition: background-color 0.3s;
}

.button:hover {
  background-color: #005f73;
}
.tab-container {
  display: flex;
  flex-direction: column;
  align-items: center;
}

.medication-tabs {
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
}

.medication-card {
  border: 1px solid #ddd;
  border-radius: 8px;
  padding: 16px;
  margin: 10px;
  width: 250px;
  box-shadow: 2px 2px 6px rgba(0, 0, 0, 0.1);
  text-align: center;
}
```

```css
.medication-card h3 {
  margin: 0;
  font-size: 1.2em;
}

.medication-card p {
  margin: 5px 0;
}

.medication-card button {
  background-color: #4CAF50;
  color: white;
  padding: 5px 10px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

.medication-card button:hover {
  background-color: #45a049;
}

input[type="text"] {
  width: 50px;
  text-align: center;
  border: none;
  background: #f2f2f2;
}
/* General styling */
body {
  font-family: Arial, sans-serif;
  background-color: #f4f4f4;
  color: #333;
  margin: 0;
  padding: 20px;
}

header {
  background-color: #4CAF50;
  padding: 10px 0;
  text-align: center;
  color: white;
}

h1 {
  margin: 0;
}

h2 {
  text-align: center;
  margin-bottom: 20px;
}

.customers-table {
  width: 80%;
  margin: 20px auto;
  border-collapse: collapse;
  background-color: #fff;
  box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}

.customers-table th,
.customers-table td {
  padding: 12px 15px;
  text-align: left;
  border: 1px solid #ddd;
}

.customers-table th {
  background-color: #4CAF50;
  color: white;
  font-size: 16px;
}

.customers-table td {
  background-color: #f9f9f9;
```

```
}

.customers-table tr:nth-child(even) td {
  background-color: #f1f1f1;
}

.customers-table tr:hover td {
  background-color: #ddd;
}

.customers-table td {
  font-size: 14px;
}

.customers-table td:last-child {
  text-align: right;
}
/* General styling for forms */
.form-container {
  width: 50%;
  margin: 30px auto;
  background-color: #fff;
  padding: 20px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  border-radius: 8px;
}

h2 {
  text-align: center;
  margin-bottom: 20px;
}

label {
  display: block;
  font-weight: bold;
  margin-bottom: 8px;
  font-size: 14px;
}

input[type="text"],
input[type="number"] {
  width: 100%;
  padding: 10px;
  margin-bottom: 20px;
  border: 1px solid #ccc;
  border-radius: 4px;
  font-size: 14px;
}

input[type="text"]:focus,
input[type="number"]:focus {
  border-color: #4CAF50;
  outline: none;
}

button.submit-btn {
  background-color: #4CAF50;
  color: white;
  font-size: 16px;
  padding: 12px 20px;
  border: none;
  border-radius: 4px;
  width: 100%;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

button.submit-btn:hover {
  background-color: #45a049;
}

button.submit-btn:active {
  background-color: #388e3c;
}
```

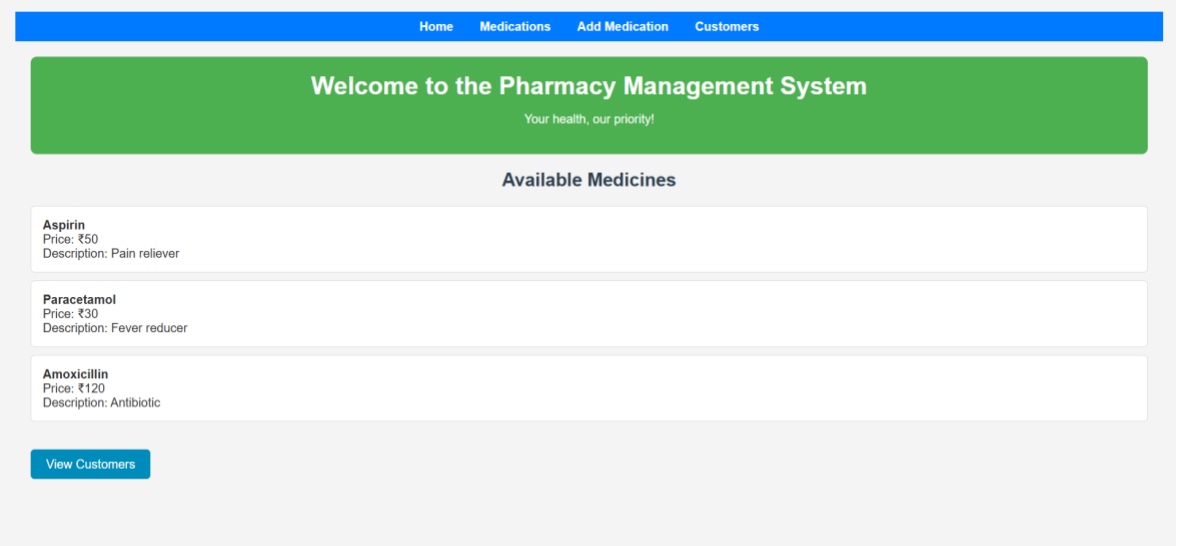
```
.form-container {
  max-width: 500px;
  margin: 0 auto;
}


@media (max-width: 768px) {
  .form-container {
      width: 90%;
  }

  label {
      font-size: 12px;
  }

  input[type="text"],
  input[type="number"],
  button.submit-btn {
      font-size: 14px;
  }
}

.medications-container {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));  /* Makes the cards responsive */
  gap: 20px;
  padding: 20px;
}


.medication-card {
  background-color: #fff;
  border-radius: 8px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  overflow: hidden;
  transition: transform 0.3s ease, box-shadow 0.3s ease;
}

.medication-card:hover {
  transform: translateY(-5px);
  box-shadow: 0 6px 12px rgba(0, 0, 0, 0.2);
}


.medication-card-header {
  background-color: #f2f2f2;
  padding: 15px;
  text-align: center;
}

.medication-card-header h3 {
  margin: 0;
  font-size: 18px;
  color: #333;
}

.medication-card-header p {
  margin: 5px 0;
  font-size: 14px;
  color: #555;
}


.medication-card-body {
  padding: 15px;
  font-size: 14px;
  color: #555;
  background-color: #fafafa;
}

.medication-card-body p {
  margin: 5px 0;
}


.medication-card-footer {
```

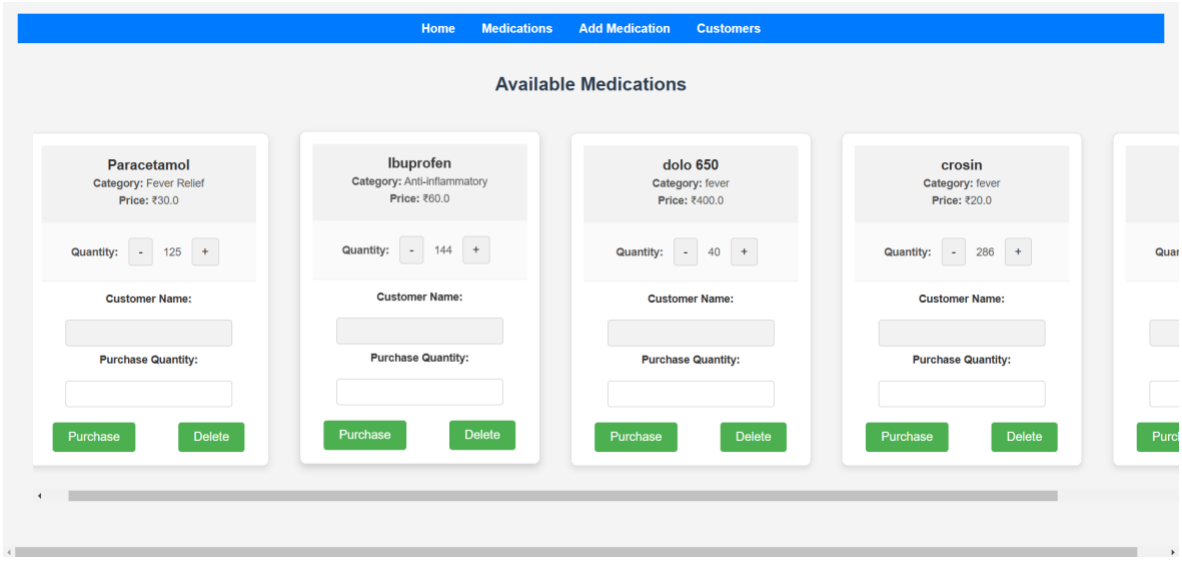
```
  display: flex;
  justify-content: space-between;
  padding: 10px;
  background-color: #fff;
  border-top: 1px solid #eee;
}

.action-btn, .delete-btn {
  background-color: #4CAF50;
  color: white;
  font-size: 14px;
  padding: 8px 12px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.action-btn:hover, .delete-btn:hover {
  background-color: #45a049;
}

.delete-btn {
  background-color: #f44336;
}

.delete-btn:hover {
  background-color: #e53935;
}


@media (max-width: 768px) {
  .medications-container {
      grid-template-columns: 1fr;
  }

  .medication-card-header h3 {
      font-size: 16px;
  }

  .medication-card-body p, .medication-card-header p {
      font-size: 12px;
  }

  .action-btn, .delete-btn {
      font-size: 12px;
      padding: 6px 10px;
  }
}
.medication-cards-container {
  display: flex;
  flex-wrap: nowrap;
  overflow-x: auto;
  gap: 20px;
  padding: 20px;
  max-width: 100vw;
  box-sizing: border-box;
  margin: 0 auto;
}

.medication-card {
  background-color: #fff;
  border-radius: 8px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  width: 280px;
  height: 400px;
  transition: transform 0.3s ease, box-shadow 0.3s ease;
  flex-shrink: 0;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
  overflow: hidden;
}

.medication-card:hover {
  transform: translateY(-5px);
```

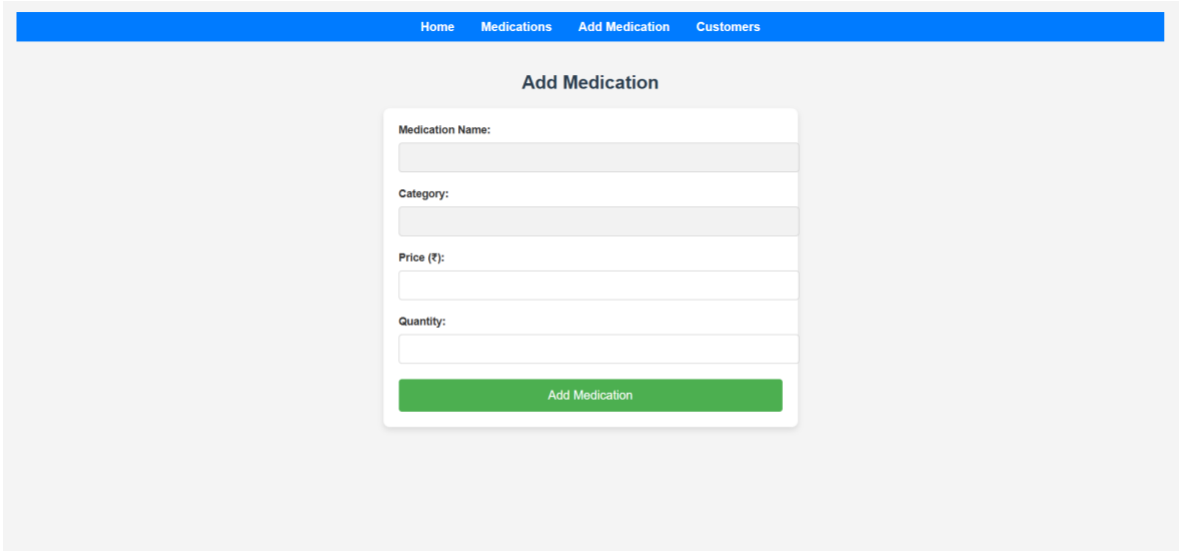
```
  box-shadow: 0 6px 12px rgba(0, 0, 0, 0.2);
}

.medication-card-header {
  background-color: #f2f2f2;
  padding: 15px;
  text-align: center;
}

.medication-card-header h3 {
  margin: 0;
  font-size: 18px;
  color: #333;
}

.medication-card-header p {
  margin: 5px 0;
  font-size: 14px;
  color: #555;
}

.medication-card-body {
  padding: 15px;
  font-size: 14px;
  color: #555;
  background-color: #fafafa;
  flex-grow: 1;
}

.medication-card-body p {
  margin: 5px 0;
  text-align: center;
}

.purchase-form {
  padding: 15px;
  background-color: #fff;
  border-top: 1px solid #eee;
  text-align: center;
}

.purchase-form input[type="text"], .purchase-form input[type="number"] {
  width: 80%;
  padding: 8px;
  margin: 10px 0;
  font-size: 14px;
  border-radius: 4px;
  border: 1px solid #ccc;
}

.buttons-container {
  display: flex;
  justify-content: space-between;
  gap: 10px;
  margin-top: 10px;
}

.purchase-form button {
  padding: 10px 20px;
  font-size: 16px;
  background-color: #4CAF50;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.purchase-form button:hover {
  background-color: #45a049;
}

.delete-btn {
  background-color: #f44336;
  color: white;
  font-size: 14px;
```

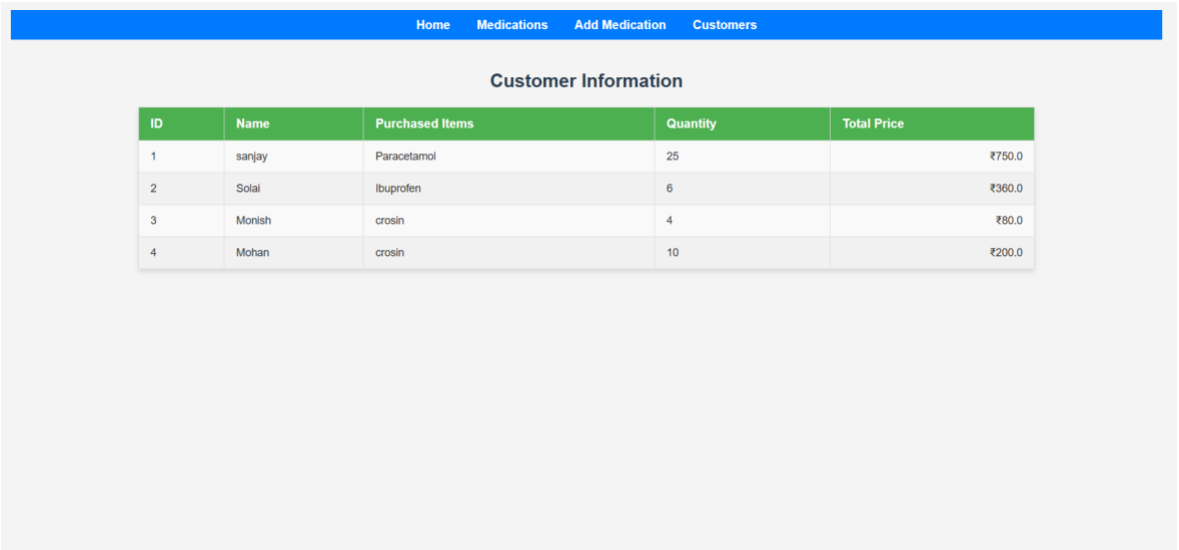
```
  padding: 8px 12px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.delete-btn:hover {
  background-color: #e53935;
}

.medication-card-body button {
  padding: 8px 12px;
  font-size: 16px;
  background-color: #f0f0f0;
  color: #333;
  border: 1px solid #ccc;
  border-radius: 4px;
  cursor: pointer;
  margin: 0 10px;
  transition: background-color 0.3s ease;
}

.medication-card-body button:hover {
  background-color: #ddd;
}

@media (max-width: 768px) {
  .medication-cards-container {
      gap: 10px;
  }

  .medication-card {
      width: 200px;
      height: 350px;
  }
}
```

5.3 Screen shots




Home
Medications
Add Medication
Customers
Available Medications
Paracetamol
Category: Fever Relief
Price: ₹30.0
Quantity: - 125 +
Customer Name:
Purchase Quantity:
Purchase
Delete
Ibuprofen
Category: Anti-inflammatory
Price: ₹60.0
Quantity: - 144 +
Customer Name:
Purchase Quantity:
Purchase
Delete
dolo 650
Category: fever
Price: ₹400.0
Quantity: - 40 +
Customer Name:
Purchase Quantity:
Purchase
Delete
crosin
Category: fever
Price: ₹20.0
Quantity: - 286 +
Customer Name:
Purchase Quantity:
Purchase
Delete


Home
Medications
Add Medication
Customers
Add Medication
Medication Name:
Category:
Price (₹):
Quantity:
Add Medication


Home
Medications
Add Medication
Customers
Customer Information
ID
Name
Purchased Items
Quantity
Total Price
1
sanjay
Paracetamol
25
₹750.0
2
Solai
Ibuprofen
6
₹360.0
3
Monish
crosin
4
₹80.0
4
Mohan
crosin
10
₹200.0

CHAPTER 6

CONCLUSION

The Pharmacy Management System provides a streamlined solution for managing medication inventory and customer transactions. It addresses the inefficiencies of manual processes and ensures data accuracy. While designed for small-scale pharmacies, the system can be expanded with additional features like analytics, reporting, or integration with external APIs.

CHAPTER 7

REFERENCES:-

1. Flask Documentation: https://flask.palletsprojects.com/
2. SQLite Documentation: https://www.sqlite.org/
3. Python Official Documentation: https://www.python.org/doc/
4. Bootstrap for UI (if used): https://getbootstrap.com/

CHAPTER – 8

BIBLIOGRAPHY:-

1. Pallets Projects. *Flask: Web Development One Flask at a Time*. 2024.
2. Hipp, D. Richard. *The Design of SQLite*. 2024.
3. Various authors. *Python Crash Course*. 2024.