PHARMACY MANAGEMENT SYSTEM

DBMS MINI PROJECT REPORT

Submitted by

ABOORVAN SHANMUGAPRIYA BABU - 230701011

AWINTHIKA SANTHANAM - 230701048

In partial fulfillment for the award of the degree of

BACHELOR OF COMPUTER SCIENCE ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE,

ANNA UNIVERSITY,

CHENNAI: 602 105

2024-2025

RAJALAKSHMI
ENGINEERING COLLEGE

BONAFIDE CERTIFICATE

NAME……

ACADEMIC YEAR……SEMESTER…….BRANCH……

UNIVERSITY REGISTER No……..

Certified that this is the bonafide record of work done by the above students in the Mini Project titled "**PHARMACY MANAGEMENT SYSTEM**" in the subject **CS19443 Database Management Systems** during the year 2024 - 2025.

Submitted for the Practical Examination held on _____

Signature of Faculty – in – Charge

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

The Pharmacy Management System is an intuitive application designed to streamline and optimize pharmacy operations. By automating key tasks such as customer management, medicine inventory tracking, and sales recording, it significantly reduces manual effort while ensuring high levels of accuracy and organization. This system employs SQLite for its robust and lightweight backend database management, and Streamlit for creating an interactive and responsive frontend interface. The system is structured to provide a seamless user experience for pharmacy staff. Users can effortlessly navigate through a variety of functions, including adding and updating customer information, managing medicine inventory, recording sales transactions, and viewing essential data. The interface is designed with simplicity in mind, featuring a straightforward sidebar menu that allows users to access different functionalities quickly. Additionally, the incorporation of icons and a background image enhances the visual appeal and accessibility of the interface, making it user-friendly even for those with limited technical knowledge. One of the core strengths of the Pharmacy Management System is its ability to handle large volumes of data efficiently. The SQLite database ensures that all records are stored securely and can be retrieved swiftly, which is crucial for maintaining an up-to-date and accurate inventory. This, in turn, helps in reducing medication errors, improving stock management, and ensuring that essential medicines are always available.

CHAPTER 1 INTRODUCTION

The Pharmacy Management System project aims to modernize and streamline the operations of pharmacies by leveraging technology. This project utilizes software tools like SQLite for database management and Streamlit for creating an interactive user interface. SQLite serves as the backend database, storing and organizing crucial data such as customer details, medicine inventory, and sales records. It plays a vital role in ensuring data integrity and providing efficient data retrieval and manipulation capabilities for the system.

Streamlit, on the other hand, is utilized for building the frontend interface of the application. It allows for the creation of a user-friendly web application that pharmacy staff can easily navigate. With Streamlit, users can interact with the system through a visually appealing and intuitive interface, accessing features like adding customers, managing medicines, recording sales, and viewing essential data. The inclusion of icons and background images further enhances the visual appeal and usability of the interface, making it accessible even to users with limited technical knowledge.

By combining the power of SQLite and Streamlit, the Pharmacy Management System offers pharmacies a modern solution for managing their daily operations efficiently. Additionally, features like detailed sales tracking, customer management, and inventory control aid in regulatory compliance and business planning. Overall, the Pharmacy Management System project highlights the importance of intuitive design and automation in modernizing healthcare management systems, ultimately enhancing the quality of service provided to customers. Future iterations may include mobile app support and predictive analytics, further elevating the system’s capabilities.

This project demonstrates the effective use of technology to address real-world needs in the healthcare industry. It provides pharmacies with tools to automate tasks, minimize manual effort, ensure data accuracy, and improve overall efficiency.

1. OBJECTIVE

The objective of the Pharmacy Management System project is to develop a user-friendly application that modernizes and streamlines pharmacy operations. By leveraging technology, the project aims to automate tasks such as customer management, medicine inventory tracking, and sales recording, thereby reducing manual effort and ensuring accuracy. The system utilizes SQLite for backend database management and Streamlit for frontend interface development, providing pharmacy staff with an intuitive platform to efficiently manage daily operations. Overall, the objective is to enhance efficiency, accuracy, and customer service within the pharmacy industry through the implementation of a user-friendly and modernized management system.

2. MODULE

- **sqlite3:** Used for backend database management, including storing and retrieving data.31
- **datetime**: Utilized for obtaining the current date in the sales recording functionality.
- **streamlit**: Used for building interactive web applications with a clean and modern interface.
- **streamlit_option_menu**: Used to add icons to the sidebar navigation menu for enhanced user experience.

CHAPTER 2 SURVEY OF TECHNOLOGY

2.1 SOFTWARE

DESCRIPTION Visual studio Code

Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging.

First and foremost, it is an editor that gets out of your way. The delightfully frictionless edit-build-debug cycle means less time fiddling with your environment, and more time executing on your ideas.

2.2 LANGUAGES

2.2.1 JAVA

Java is widely used in database management systems (DBMS) for its robustness, platform independence, and extensive ecosystem of libraries. With APIs like JDBC (Java Database Connectivity), Java facilitates database interactions, allowing developers to execute SQL queries, manage database connections, and handle data effectively. Its strong typing and comprehensive error handling make it ideal for integrating with various DBMS platforms. Java's versatility and performance make it a preferred choice for enterprise-level database applications.

2.2.2 MySQL

Many of the world's largest and fastest-growing organisations including Facebook, Google, Adobe, Alcatel Lucent and Zappos rely on MySQL to save time and money powering their high-volume Web sites, business-critical systems and packaged software. Since then, the performance & scalability, reliability, and ease of use of the world's most popular open source database, characteristics that made MySQL the #1 choice for web applications, have relentlessly been improved.

2.3 LIBRARIES

2.3.1 **Streamlit**:

Streamlit is an open-source Python library used for building interactive web applications for data science and machine learning projects. It simplifies the process of creating web apps by allowing developers to write Python scripts and immediately visualize the results as a web app.

2.3.2 sqlite3:

This library provides a straightforward way to interact with SQLite databases using Python. It allows for the creation, manipulation, and querying of databases. SQLite3 provides a powerful set of features for managing relational databases, including support for transactions, indexes, and triggers. Its ease of use and portability make it a popular choice for embedded systems, mobile applications, and small-scale web development projects.

CHAPTER 3 REQUIREMENT AND ANALYSIS

3.1 REQUIREMENTS SPECIFICATION

User Requirements

User requirements include Data management, Accuracy, Accessibility, Integration and Security for efficient pharmacy management.

System Requirements

There should be a database backup of the pharmacy management system. Operating system should be Windows XP or a higher version of windows. The system should have sufficient hardware resources to run the application smoothly, including CPU, memory (RAM), and disk space.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements:

1. **Computer**: A desktop or laptop computer capable of running the required software.
2. **Processor**: A modern processor with sufficient processing power to handle database operations and web application rendering smoothly.
3. **Memory (RAM)**: At least 4GB of RAM is recommended to ensure smooth performance, especially when dealing with large datasets.

Software Requirements:

1. **Python**: Python 3.x should be installed on the system. The specific version required by the code can vary, but compatibility with Python 3.6 or higher is recommended.
2. **Python Libraries**: The required Python libraries, including sqlite3, streamlit, and Pillow, should be installed. Users can use package managers like pip to install these libraries.
3. **Web Browser:** A modern web browser like Chrome, Firefox, or Safari should be installed to access the Streamlit web application.

CHAPTER 4 ARCHITECTURE DIAGRAM

4.1 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a crucial tool utilized in system analysis to illustrate the flow of data within a process or system. It models the system's data flow by depicting external entities that interact with the system, data transformations performed within processes, and the resulting output data that may flow to other processes or external entities, such as files. The primary advantage of using DFDs lies in their ability to offer a comprehensive overview of the data that a system processes, thereby aiding in the understanding and visualization of data flow within the system. Additionally, DFDs provide insights into the inputs and outputs of each entity and process, enhancing understanding of the system's functionalities and interactions.

CONTEXT LEVEL DFD :

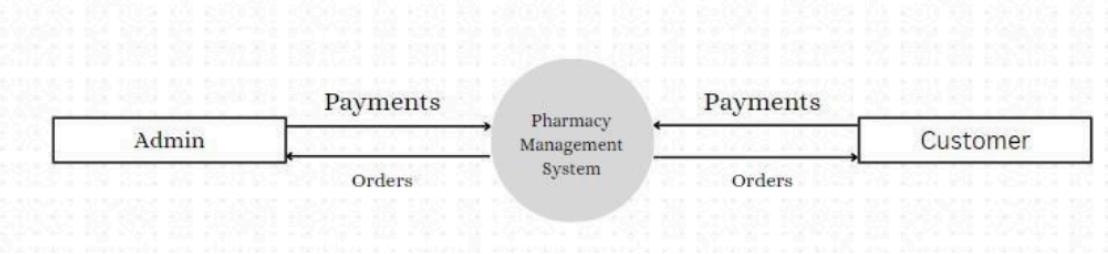


Fig 4.1.1: Context level DFD

LEVEL 1 DFD:

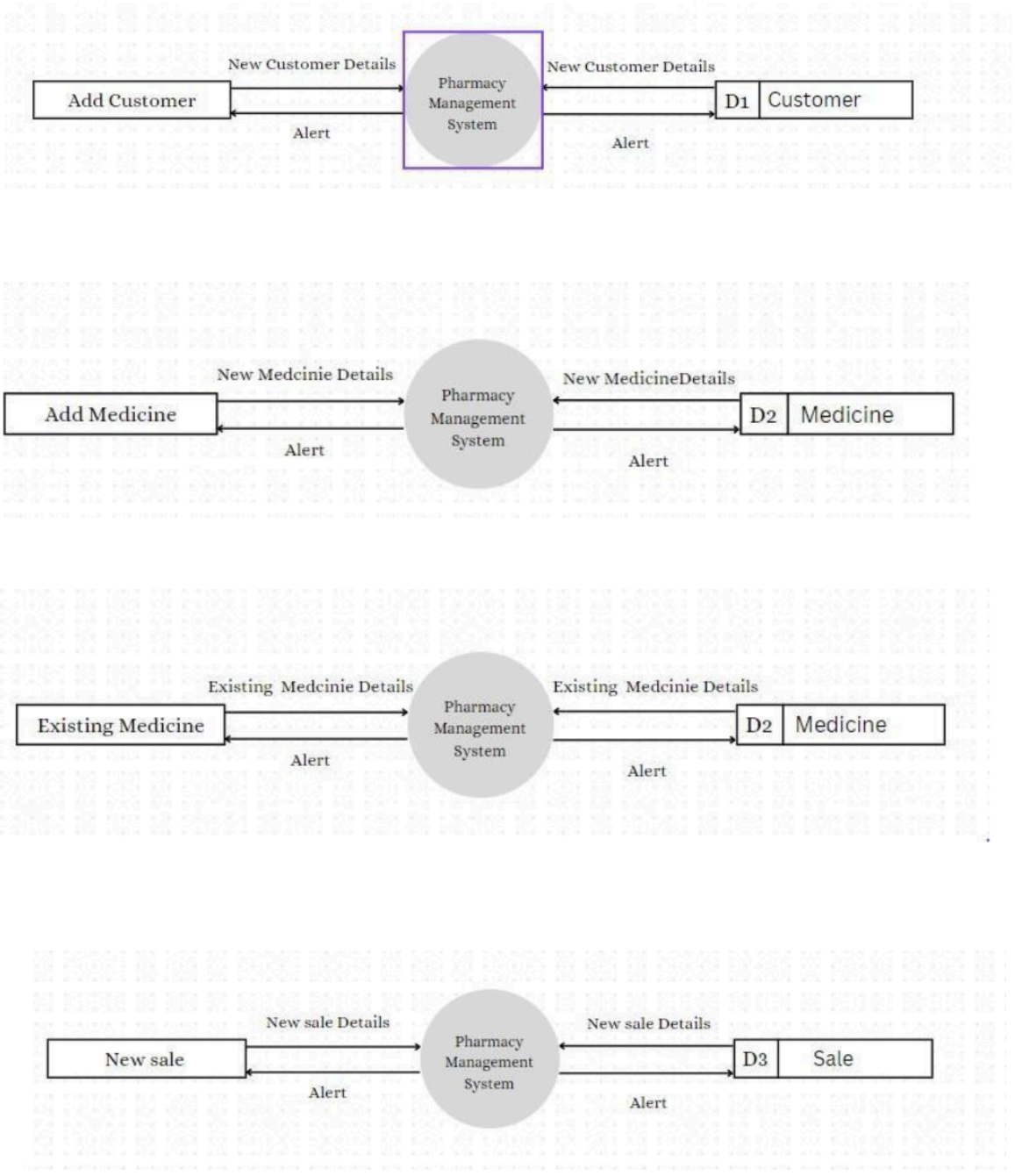
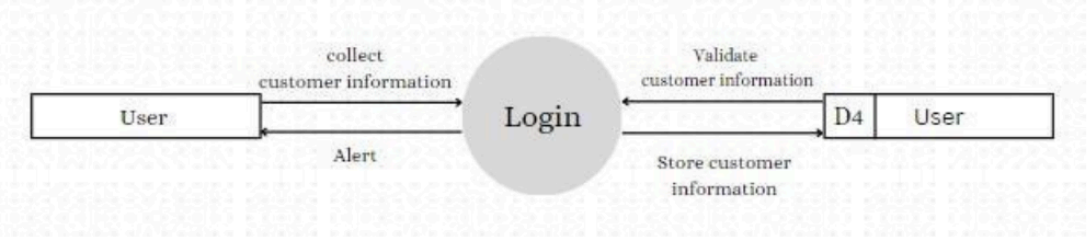
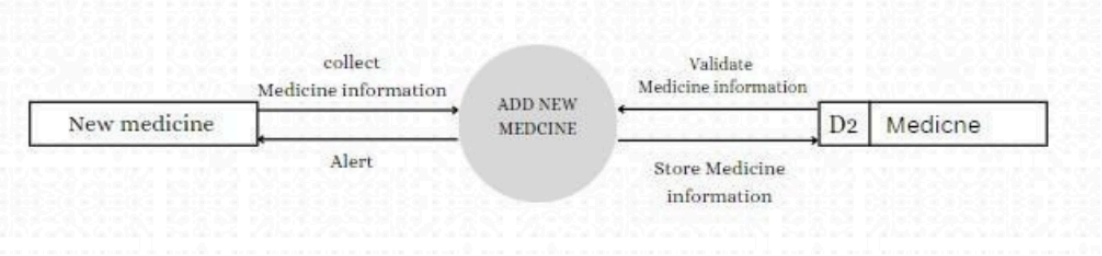
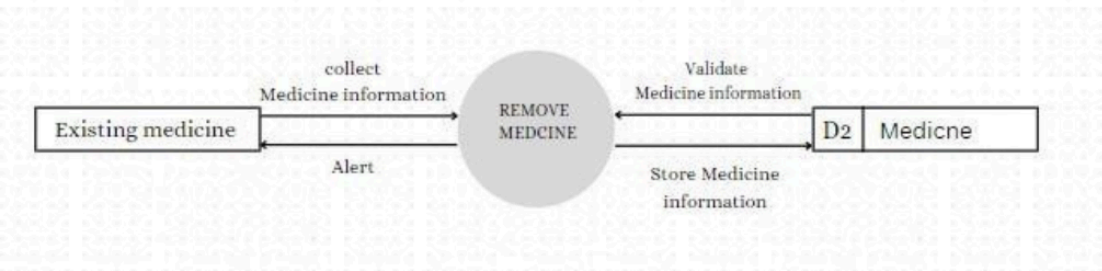


Fig 4.2: Level 1 DFD

LEVEL 2 DFD:







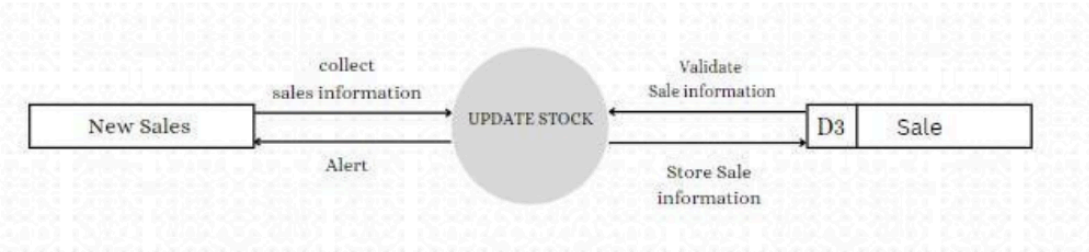


Fig 4.1.3: Level 2 DFD

3.2.DATA DICTIONARY

Customers Table

Table	Attribute	Data Type	Description
Customers			
	customer_id	INTEGER	Primary Key, Unique identifier for each customer
	name	TEXT	Name of the customer
	phone	TEXT	Contact number of the customer
	address	TEXT	Address of the customer

Table 4.2.1 Customer Table

Medicines Table

Table	Attribute	Data Type	Description
Medicines			
	medicine_id	INTEGER	Primary Key, Unique identifier for each medicine
	name	TEXT	Name of the medicine
	manufacturer	TEXT	Manufacturer of the medicine
	price	REAL	Price of the medicine
	stock	INTEGER	Number of units available in stock

Table 4.2.2 Medicines Table

Sales Table

Table	Attribute	Data Type	Description
Sales			
	sale_id	INTEGER	Primary Key, Unique identifier for each sale
	customer_id	INTEGER	Foreign Key, References `Customers(customer_id)`
	medicine_id	INTEGER	Foreign Key, References `Medicines(medicine_id)`
	quantity	INTEGER	Number of units sold
	sale_date	DATE	Date of the sale transaction

Table 4.2.3 Sales Table

4.3 E-R DIAGRAM

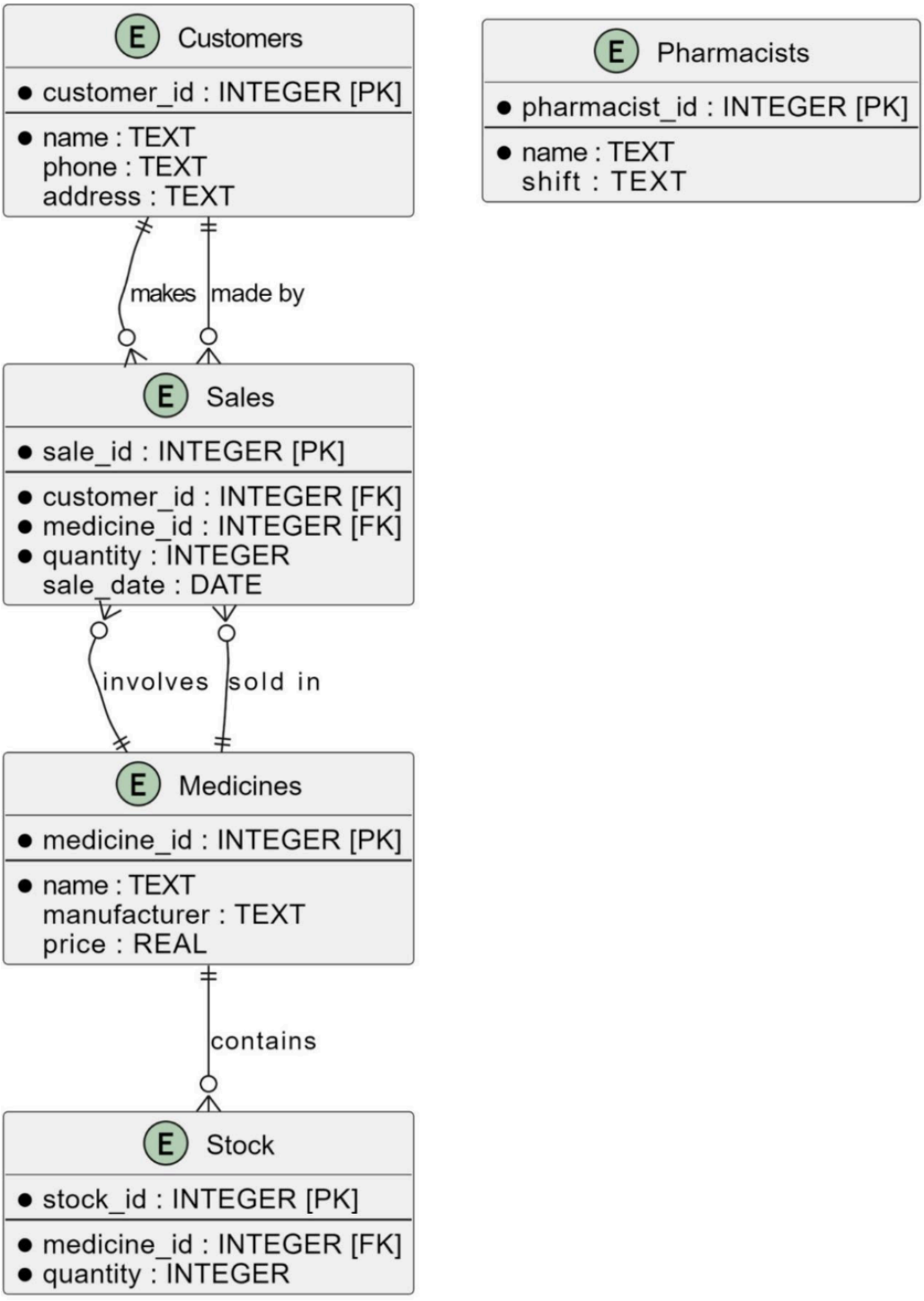


Figure 4.3 Entity Relationship Diagram

CHAPTER 5 PROGRAM CODE

5.1 CODE DETAILS

5.1.1 FRONT- END:

```
import sqlite3
from datetime import date import streamlit as st
# Database operations def initialize_db():
  try:
    conn = sqlite3.connect('pharmacy.db') cursor = conn.cursor()
    # Create tables
            cursor.execute('''CREATE TABLE IF NOT EXISTS Customers (
                    customer_id INTEGER PRIMARY KEY,
            name TEXT NOT NULL,
            phone TEXT, address TEXT
          )''')
            cursor.execute('''CREATE TABLE IF NOT EXISTS Medicines (
                    medicine_id INTEGER PRIMARY KEY,
            name TEXT NOT NULL,
            manufacturer TEXT, price REAL
          )''')
            cursor.execute('''CREATE TABLE IF NOT EXISTS Stock ( stock_id
                    INTEGER PRIMARY KEY, medicine_id INTEGER,
            quantity INTEGER,
            FOREIGN KEY (medicine_id) REFERENCES Medicines(medicine_id)
          )''')
            cursor.execute('''CREATE TABLE IF NOT EXISTS Sales ( sale_id
                    INTEGER PRIMARY KEY,
customer_id INTEGER, medicine_id INTEGER, quantity INTEGER, sale_date DATE,
            FOREIGN KEY (customer_id) REFERENCES Customers(customer_id),
            FOREIGN KEY (medicine_id) REFERENCES Medicines(medicine_id)
          )''')
            cursor.execute('''CREATE TABLE IF NOT EXISTS Pharmacists (
                    pharmacist_id INTEGER PRIMARY KEY,
```

```
                name TEXT NOT NULL,
                shift TEXT
            )'")
        conn.commit()
        except sqlite3.Error as e: st.error(f"Database error: {e}")
    finally:
        conn.close()
    def add_customer(name, phone, address): try:
        conn = sqlite3.connect('pharmacy.db') cursor = conn.cursor()
        cursor.execute('INSERT INTO Customers (name, phone, address) VALUES (?, ?,
  ?)', (name, phone, address))
        conn.commit()
        except sqlite3.Error as e: st.error(f"Database error: {e}")
    finally:
        conn.close()
    def add_medicine(name, manufacturer, price): try:
        conn = sqlite3.connect('pharmacy.db') cursor = conn.cursor()
        cursor.execute('INSERT INTO Medicines (name, manufacturer, price) VALUES (?,
  ?, ?)', (name, manufacturer, price))
conn.commit()
        except sqlite3.Error as e: st.error(f"Database error: {e}")
    finally:
        conn.close()
    def add_stock(medicine_id, quantity): try:
        conn = sqlite3.connect('pharmacy.db') cursor = conn.cursor()
        cursor.execute('INSERT INTO Stock (medicine_id, quantity) VALUES (?, ?)',
        (medicine_id, quantity)) conn.commit()
        except sqlite3.Error as e: st.error(f"Database error: {e}")
    finally:
        conn.close()
    def make_sale(customer_id, medicine_id, quantity): try:
        conn = sqlite3.connect('pharmacy.db') cursor = conn.cursor()
        sale_date = date.today().strftime('%Y-%m-%d')
```

```
        cursor.execute('INSERT INTO Sales (customer_id, medicine_id, quantity,
        sale_date) VALUES (?, ?, ?,
    ?)', (customer_id, medicine_id, quantity, sale_date))
        cursor.execute('UPDATE Stock SET quantity = quantity - ? WHERE medicine_id =
    ?', (quantity, medicine_id))
        conn.commit()
        except sqlite3.Error as e: st.error(f"Database error: {e}")
      finally:
        conn.close()
      def view_customers(): try:
        conn = sqlite3.connect('pharmacy.db') cursor = conn.cursor()
        cursor.execute('SELECT * FROM Customers')
rows = cursor.fetchall() return rows
        except sqlite3.Error as e: st.error(f"Database error: {e}")
      finally:
        conn.close()
      def view_medicines(): try:
        conn = sqlite3.connect('pharmacy.db') cursor = conn.cursor()
        cursor.execute('SELECT * FROM Medicines') rows = cursor.fetchall()
        return rows
        except sqlite3.Error as e: st.error(f"Database error: {e}")
      finally:
        conn.close()
      def view_stock(): try:
        conn = sqlite3.connect('pharmacy.db') cursor = conn.cursor()
        cursor.execute('SELECT * FROM Stock') rows = cursor.fetchall()
        return rows
        except sqlite3.Error as e: st.error(f"Database error: {e}")
      finally:
        conn.close()
      def view_sales(): try:
        conn = sqlite3.connect('pharmacy.db') cursor = conn.cursor()
        cursor.execute('SELECT * FROM Sales') rows = cursor.fetchall()
return rows
```

```
    except sqlite3.Error as e: st.error(f"Database error: {e}")
  finally:
    conn.close()

# Initialize the database initialize_db()

# Streamlit UI

st.title("Pharmacy Management System")

# Sidebar for navigation st.sidebar.title("Navigation")
page = st.sidebar.selectbox("Go to", ["Add Customer", "Add Medicine", "Add Stock",
"Make Sale", "View Data"])

  if page == "Add Customer": st.header("Add Customer") name =
    st.text_input("Name") phone = st.text_input("Phone")
  address = st.text_input("Address") if st.button("Add Customer"):
      if name and phone and address: add_customer(name, phone, address)
        st.success("Customer added successfully")
    else:
      st.error("All fields are required")

  elif page == "Add Medicine": st.header("Add Medicine")
  name = st.text_input("Medicine Name") manufacturer =
  st.text_input("Manufacturer")
  price = st.number_input("Price", min_value=0.0, format="%.2f") if st.button("Add
  Medicine"):
      if name and manufacturer and price: add_medicine(name, manufacturer, price)
st.success("Medicine added successfully") else:
      st.error("All fields are required")
  elif page == "Add Stock": st.header("Add Stock")
  medicine_id = st.number_input("Medicine ID", min_value=1, step=1) quantity =
  st.number_input("Quantity", min_value=0, step=1)
  if st.button("Add Stock"):
      if medicine_id and quantity >= 0: add_stock(medicine_id, quantity)
        st.success("Stock added successfully")
    else:
      st.error("All fields are required")
  elif page == "Make Sale": st.header("Make Sale")
```

```
customer_id = st.number_input("Customer ID", min_value=1, step=1) medicine_id =
st.number_input("Medicine ID", min_value=1, step=1) quantity =
st.number_input("Quantity", min_value=1, step=1)
if st.button("Make Sale"):
    if customer_id and medicine_id and quantity: make_sale(customer_id,
      medicine_id, quantity) st.success("Sale recorded successfully")
  else:
    st.error("All fields are required")

elif page == "View Data": st.header("View Data")
data_type = st.selectbox("View", ["Customers", "Medicines", "Stock", "Sales"]) if
data_type == "Customers":
  st.subheader("Customers") customers = view_customers() for customer in
  customers:
    st.write(customer)

  elif data_type == "Medicines": st.subheader("Medicines")
medicines = view_medicines() for medicine in medicines:
    st.write(medicine) elif data_type == "Stock":
  st.subheader("Stock") stock = view_stock() for item in stock:
    st.write(item)

  elif data_type == "Sales": st.subheader("Sales") sales = view_sales() for sale in
    sales:
    st.write(sale)
```

5.1.2 BACK END:

```
-- phpMyAdmin SQL Dump
-- version 4.5.1
-- http://www.phpmyadmin.net
--
-- Host: 127.0.0.1
-- Generation Time: Apr 03, 2018 at 09:09 PM
-- Server version: 10.1.16-MariaDB
-- PHP Version: 5.6.24

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101 SET
  @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET
  @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET
  @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `pharmacy`
--

-- --------------------------------------------------------

--
-- Table structure for table `company`
--

CREATE TABLE `company` (
  `NAME` varchar(50) NOT NULL,
  `ADDRESS` varchar(50) NOT NULL,
  `PHONE` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
--
-- Dumping data for table `company`
--

INSERT INTO `company` (`NAME`, `ADDRESS`, `PHONE`) VALUES
('Elshark', 'Egypt/\nElmansoura', '12903'),
('El_Horia', 'Damanhour/\nShobra/\nBus_Station', '01289078443'),
('Med_City', 'Damanhour /\nShobra  /\nBus Station', '010114367832');

-- --------------------------------------------------------

--
-- Table structure for table `drugs`
--

CREATE TABLE `drugs` (
  `NAME` varchar(50) NOT NULL,
  `TYPE` varchar(20) NOT NULL,
  `BARCODE` varchar(20) NOT NULL,
  `DOSE` varchar(10) NOT NULL,
  `CODE` varchar(10) NOT NULL,
  `COST_PRICE` double NOT NULL,
  `SELLING_PRICE` double NOT NULL,
  `EXPIRY` varchar(20) NOT NULL,
  `COMPANY_NAME` varchar(50) NOT NULL,
  `PRODUCTION_DATE` date NOT NULL,
  `EXPIRATION_DATE` date NOT NULL,
  `PLACE` varchar(20) NOT NULL,
  `QUANTITY` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `drugs`
--

INSERT INTO `drugs` (`NAME`, `TYPE`, `BARCODE`, `DOSE`, `CODE`,
  `COST_PRICE`, `SELLING_PRICE`, `EXPIRY`, `COMPANY_NAME`,
```

```
  `PRODUCTION_DATE`, `EXPIRATION_DATE`, `PLACE`, `QUANTITY`)
  VALUES
('Novalo', 'Bills', 'fsdgjfihjorodsf', 'normal', '3d00', 2, 3, 'Available for use',
  'Med_City', '2017-03-03', '2019-03-03', 'N-Right', 40),
('novafol', 'Bills', 'ftrkl432432md', 'normal', '2xaa', 33, 40, 'Available for use',
  'Med_City', '2016-01-01', '2017-01-01', 'N-Left', 27);

-- --------------------------------------------------------

--
-- Table structure for table `expiry`
--

CREATE TABLE `expiry` (
  `PRODUCT_NAME` varchar(50) NOT NULL,
  `PRODUCT_CODE` varchar(20) NOT NULL,
  `DATE_OF_EXPIRY` varchar(10) NOT NULL,
  `QUANTITY_REMAIN` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Table structure for table `history_sales`
--

CREATE TABLE `history_sales` (
  `USER_NAME` varchar(20) NOT NULL,
  `BARCODE` varchar(20) NOT NULL,
  `NAME` varchar(50) NOT NULL,
  `TYPE` varchar(10) NOT NULL,
  `DOSE` varchar(10) NOT NULL,
  `QUANTITY` int(11) NOT NULL,
  `PRICE` double NOT NULL,
  `AMOUNT` double NOT NULL,
  `DATE` varchar(15) NOT NULL,
  `TIME` varchar(20) NOT NULL
```

```
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `history_sales`
--

INSERT INTO `history_sales` (`USER_NAME`, `BARCODE`, `NAME`, `TYPE`,
  `DOSE`, `QUANTITY`, `PRICE`, `AMOUNT`, `DATE`, `TIME`) VALUES
('Ebrahem Samer', 'sgnfsjkfnsdjfkb', 'Breofin', 'Bills', ' Free used', 2, 6, 12,
  '12-02-2017', '05:02:06'),
('Ebrahem Samer', 'sgnfsjkfnsdjfkb', 'Breofin', 'Bills', ' Free used', 2, 6, 12,
  '12-02-2017', '05:02:26'),
('Ebrahem Samer', 'sgnfsjkfnsdjfkb', 'Breofin', 'Bills', ' Free used', 4, 6, 24,
  '12-02-2017', '05:02:40'),
('Ebrahem Samer', 'nbhdl4978549', 'Morfin', 'Injection', '1 (Day)', 2, 14, 28,
  '13-02-2017', '01:38:00'),
('Ebrahem Samer', 'nbhdl4978549', 'Morfin', 'Injection', '1 (Day)', 2, 14, 28,
  '13-02-2017', '01:38:10'),
('Ebrahem Samer', 'nbhdl4978549', 'Morfin', 'Injection', '1 (Day)', 7, 14, 98,
  '13-02-2017', '01:38:28'),
('Ebrahem Samer', 'nbhdl4978549', 'Morfin', 'Injection', '1 (Day)', 1, 14, 14,
  '13-02-2017', '01:38:46'),
('Shimaa Ahmed', 'sgnfsjkfnsdjfkb', 'Breofin', 'Bills', ' Free used', 2, 6, 12,
  '13-02-2017', '01:59:34'),
('Shimaa Ahmed', 'sgnfsjkfnsdjfkb', 'Breofin', 'Bills', ' Free used', 5, 6, 30,
  '13-02-2017', '01:59:43'),
('Ebrahem Samer', 'sgnfsjkfnsdjfkb', 'Breofin', 'Bills', ' Free used', 1, 6, 6,
  '13-02-2017', '02:12:33'),
('Ebrahem Samer', 'fsdjkbdfjkffds', 'Declofien', 'Injection', 'Free', 2, 14, 28,
  '17-02-2017', '09:55:43'),
('Ebrahem Samer', 'fsdjkbdfjkffds', 'Declofien', 'Injection', 'Free', 2, 14, 28,
  '17-02-2017', '09:55:58'),
('Ebrahem Samer', 'fsdjkbdfjkffds', 'Declofien', 'Injection', 'Free', 5, 14, 70,
  '17-02-2017', '09:56:11'),
('Ebrahem Samer', 'fsdjkbdfjkffds', 'Declofien', 'Injection', 'Free', 2, 17, 34,
  '17-02-2017', '10:04:58'),
```

('Ebrahem Samer', 'fsdjkbdfjkffds', 'Declofien', 'Injection', 'Free', 2, 17, 34, '17-02-2017', '10:05:15'),
('Ebrahem Samer', 'fsdjkbdfjkffds', 'Declofien', 'Injection', 'Free', 5, 17, 85, '17-02-2017', '10:05:26'),
('Ebrahem Samer', 'fsdjkbdfjkffds', 'Declofien', 'Injection', 'Free', 4, 20, 80, '18-02-2017', '11:16:08'),
('Ebrahem Samer', 'fsdjkbdfjkffds', 'Declofien', 'Injection', 'Free', 4, 20, 80, '18-02-2017', '11:16:28'),
('Ebrahem Samer', 'AnyBarcodedaf', 'AnyName', 'Drink', '2 Days', 4, 14, 56, '18-02-2017', '11:17:06'),
('Ebrahem Samer', 'AnyBarcodedaf', 'AnyName', 'Drink', '2 Days', 4, 14, 56, '18-02-2017', '11:17:15'),
('Ebrahem Samer', 'AnyBarcodedaf', 'AnyName', 'Drink', '2 Days', 7, 14, 98, '18-02-2017', '11:17:24'),
('Shimaa Ahmed', 'AnyBarcodedaf', 'AnyName', 'Drink', '2 Days', 6, 14, 84, '18-02-2017', '11:18:29'),
('Shimaa Ahmed', 'AnyBarcodedaf', 'AnyName', 'Drink', '2 Days', 2, 14, 28, '18-02-2017', '11:18:41'),
('Shimaa Ahmed', 'AnyBarcodedaf', 'AnyName', 'Drink', '2 Days', 2, 14, 28, '18-02-2017', '11:18:45'),
('Ebrahem Samer', 'ftrkl432432md', 'novafol', 'Bills', 'normal', 2, 40, 80, '14-04-2017', '04:50:32'),
('Ebrahem Samer', 'ftrkl432432md', 'novafol', 'Bills', 'normal', 2, 40, 80, '14-04-2017', '04:50:53'),
('Ebrahem Samer', 'ftrkl432432md', 'novafol', 'Bills', 'normal', 6, 40, 240, '14-04-2017', '04:51:01'),
('Ebrahem Samer', 'ftrkl432432md', 'novafol', 'Bills', 'normal', 1, 40, 40, '03-05-2017', '03:33:30'),
('Ebrahem Samer', 'ftrkl432432md', 'novafol', 'Bills', 'normal', 1, 40, 40, '03-05-2017', '03:33:36'),
('Ebrahem Samer', 'ftrkl432432md', 'novafol', 'Bills', 'normal', 1, 40, 40, '03-05-2017', '03:33:41');

-- --

--
-- Table structure for table `inbox`

```
--

CREATE TABLE `inbox` (
  `MESSAGE_FROM` varchar(20) NOT NULL,
  `MESSAGE_TO` varchar(20) NOT NULL,
  `MESSAGE_TEXT` varchar(200) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `inbox`
--

INSERT INTO `inbox` (`MESSAGE_FROM`, `MESSAGE_TO`,
  `MESSAGE_TEXT`) VALUES
('Ebrahem Samer', 'Shimaa Ahmed', 'Welcome shimaa'),
('Shimaa Ahmed', 'Ebrahem Samer', 'Welcome sir'),
('Ebrahem Samer', 'Shimaa Ahmed', 'Hay , Shimaa '),
('Shimaa Ahmed', 'Ebrahem Samer', 'Hay Doctor Ebrahem'),
('Shimaa Ahmed', 'Ebrahem Samer', 'Welcome Doctor Ebrahem'),
('Ebrahem Samer', 'Shimaa Ahmed', 'Hello , Shimaa'),
('Ebrahem Samer', 'Shimaa Ahmed', 'Shimaa , Please go and update\nthe drug roof '),
('Ebrahem Samer', 'Ali Mostafa', 'Welcome'),
('Ebrahem Samer', 'Shimaa Ahmed', 'This is your salary on the disk , \n3000 , close in
  12  ; good luck'),
('Ebrahem Samer', 'Shimaa Ahmed', 'good job meet me in five'),
('Shimaa Ahmed', 'Ebrahem Samer', 'Ok i will '),
('Ebrahem Samer', 'Shimaa Ahmed', 'gdfgfdgfdgfdg'),
('Ebrahem Samer', 'Shimaa Ahmed', 'haaaaaaaaaaaaaaai'),
('Ebrahem Samer', 'Shimaa Ahmed', 'What is wrong ? '),
('Shimaa Ahmed', 'Ebrahem Samer', 'I am okay thanks ');

-- --------------------------------------------------------

--
-- Table structure for table `login`
--
```

```
CREATE TABLE `login` (
  `NAME` varchar(50) NOT NULL,
  `TYPE` varchar(20) NOT NULL,
  `DATE` varchar(20) NOT NULL,
  `TIME` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `login`
--

INSERT INTO `login` (`NAME`, `TYPE`, `DATE`, `TIME`) VALUES
('Prajein', 'Employee', '14-11-2024', '10:30:24'),
('Ebrahem Samer', 'Admin', '17-02-2017', '10:30:24'),
('Ebrahem Samer', 'Admin', '17-02-2017', '10:32:48'),
('Shimaa Ahmed', 'Employee', '17-02-2017', '10:32:56'),
('Ebrahem Samer', 'Admin', '17-02-2017', '10:33:10'),
('Shimaa Ahmed', 'Employee', '17-02-2017', '10:33:37'),
('Ebrahem Samer', 'Admin', '17-02-2017', '10:36:21'),
('Ebrahem Samer', 'Admin', '17-02-2017', '10:36:53'),
('Ebrahem Samer', 'Admin', '17-02-2017', '10:49:27'),
('Ebrahem Samer', 'Admin', '17-02-2017', '11:02:23'),
('Ebrahem Samer', 'Admin', '17-02-2017', '01:40:08'),
('Ebrahem Samer', 'Admin', '18-02-2017', '10:50:29'),
('Ebrahem Samer', 'Admin', '18-02-2017', '10:51:50'),
('Ebrahem Samer', 'Admin', '18-02-2017', '10:53:33'),
('Ebrahem Samer', 'Admin', '18-02-2017', '10:58:41'),
('Ebrahem Samer', 'Admin', '18-02-2017', '11:15:39'),
('Shimaa Ahmed', 'Employee', '18-02-2017', '11:18:19'),
('Ebrahem Samer', 'Admin', '18-02-2017', '11:23:25'),
('Shimaa Ahmed', 'Employee', '18-02-2017', '11:24:19'),
('Ebrahem Samer', 'Admin', '04-04-2017', '06:32:57'),
('Shimaa Ahmed', 'Employee', '04-04-2017', '06:39:00'),
('Ebrahem Samer', 'Admin', '13-04-2017', '02:57:26'),
('Ebrahem Samer', 'Admin', '13-04-2017', '03:06:11'),
('Ebrahem Samer', 'Admin', '13-04-2017', '03:08:31'),
('Ebrahem Samer', 'Admin', '13-04-2017', '03:09:40'),
```

('Ebrahem Samer', 'Admin', '13-04-2017', '03:13:24'),
('Ebrahem Samer', 'Admin', '13-04-2017', '05:04:26'),
('Ebrahem Samer', 'Admin', '13-04-2017', '05:07:20'),
('Ebrahem Samer', 'Admin', '13-04-2017', '05:10:11'),
('Ebrahem Samer', 'Admin', '13-04-2017', '05:21:53'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:11:57'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:13:44'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:17:42'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:19:38'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:22:00'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:28:37'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:30:48'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:35:00'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:39:54'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:41:53'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:44:29'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:47:08'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:48:24'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:49:36'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:51:28'),
('Ebrahem Samer', 'Admin', '14-04-2017', '05:53:15'),
('Ebrahem Samer', 'Admin', '14-04-2017', '06:22:53'),
('Ebrahem Samer', 'Admin', '14-04-2017', '06:30:59'),
('Ebrahem Samer', 'Admin', '14-04-2017', '02:32:24'),
('Ebrahem Samer', 'Admin', '14-04-2017', '02:40:18'),
('Ebrahem Samer', 'Admin', '14-04-2017', '02:43:43'),
('Ebrahem Samer', 'Admin', '14-04-2017', '02:46:41'),
('Ebrahem Samer', 'Admin', '14-04-2017', '02:48:26'),
('Ebrahem Samer', 'Admin', '14-04-2017', '02:49:19'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '02:52:01'),
('Ebrahem Samer', 'Admin', '14-04-2017', '02:58:36'),
('Ebrahem Samer', 'Admin', '14-04-2017', '03:14:22'),
('Ebrahem Samer', 'Admin', '14-04-2017', '03:17:23'),
('Ebrahem Samer', 'Admin', '14-04-2017', '03:19:28'),
('Ebrahem Samer', 'Admin', '14-04-2017', '03:27:34'),
('Ebrahem Samer', 'Admin', '14-04-2017', '04:49:24'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '04:55:06'),

('Shimaa Ahmed', 'Employee', '14-04-2017', '05:01:50'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:03:59'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:14:50'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:17:01'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:17:50'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:21:19'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:23:30'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:26:03'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:28:53'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:32:36'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:35:04'),
('Shimaa Ahmed', 'Employee', '14-04-2017', '05:37:17'),
('Ebrahem Samer', 'Admin', '14-04-2017', '07:19:33'),
('Shimaa Ahmed', 'Employee', '15-04-2017', '02:03:12'),
('Ebrahem Samer', 'Admin', '15-04-2017', '02:47:28'),
('Ebrahem Samer', 'Admin', '15-04-2017', '02:56:16'),
('Ebrahem Samer', 'Admin', '15-04-2017', '03:06:20'),
('Ebrahem Samer', 'Admin', '15-04-2017', '03:36:58'),
('Ebrahem Samer', 'Admin', '15-04-2017', '03:42:44'),
('Ebrahem Samer', 'Admin', '03-05-2017', '01:23:14'),
('Ebrahem Samer', 'Admin', '03-05-2017', '01:51:20'),
('Ebrahem Samer', 'Admin', '03-05-2017', '01:52:35'),
('Ebrahem Samer', 'Admin', '03-05-2017', '03:31:40'),
('Ebrahem Samer', 'Admin', '03-05-2017', '03:47:32'),
('Ebrahem Samer', 'Admin', '05-05-2017', '01:24:00'),
('Ebrahem Samer', 'Admin', '05-05-2017', '03:06:19'),
('Ebrahem Samer', 'Admin', '05-05-2017', '07:54:02'),
('Shimaa Ahmed', 'Employee', '05-05-2017', '07:55:52'),
('Ebrahem Samer', 'Admin', '05-05-2017', '08:01:50'),
('Ebrahem Samer', 'Admin', '05-05-2017', '08:02:44'),
('Ebrahem Samer', 'Admin', '05-05-2017', '08:05:37'),
('Ebrahem Samer', 'Admin', '05-05-2017', '08:07:37'),
('Shimaa Ahmed', 'Employee', '05-05-2017', '08:09:23'),
('Ebrahem Samer', 'Admin', '05-05-2017', '08:14:18'),
('Shimaa Ahmed', 'Employee', '05-05-2017', '08:15:44'),
('Ebrahem Samer', 'Admin', '05-05-2017', '08:16:15'),
('Ebrahem Samer', 'Admin', '06-05-2017', '09:51:33'),

('Shimaa Ahmed', 'Employee', '06-05-2017', '09:52:46'),
('Shimaa Ahmed', 'Employee', '06-05-2017', '09:54:33'),
('Ebrahem Samer', 'Admin', '07-05-2017', '04:44:39'),
('Ebrahem Samer', 'Admin', '07-05-2017', '10:02:15'),
('Ebrahem Samer', 'Admin', '07-05-2017', '10:12:11'),
('Ebrahem Samer', 'Admin', '27-05-2017', '03:53:36'),
('Ebrahem Samer', 'Admin', '27-05-2017', '03:54:05'),
('Ebrahem Samer', 'Admin', '27-05-2017', '04:05:04'),
('Shimaa Ahmed', 'Employee', '27-05-2017', '04:06:02'),
('Ebrahem Samer', 'Admin', '30-05-2017', '03:13:41'),
('Ebrahem Samer', 'Admin', '31-05-2017', '10:57:35'),
('Ebrahem Samer', 'Admin', '31-05-2017', '11:00:02'),
('Ebrahem Samer', 'Admin', '31-05-2017', '11:06:32'),
('Ebrahem Samer', 'Admin', '07-06-2017', '08:38:00'),
('Ebrahem Samer', 'Admin', '07-06-2017', '08:40:43'),
('Ebrahem Samer', 'Admin', '07-06-2017', '08:41:28'),
('Shimaa Ahmed', 'Employee', '19-09-2017', '06:10:07'),
('Ebrahem Samer', 'Admin', '07-10-2017', '04:39:50'),
('Ebrahem Samer', 'Admin', '07-10-2017', '04:40:39'),
('Ebrahem Samer', 'Admin', '21-11-2017', '09:06:10'),
('Ebrahem Samer', 'Admin', '21-11-2017', '09:15:39'),
('Shimaa Ahmed', 'Employee', '14-12-2017', '02:56:45'),
('Ebrahem Samer', 'Admin', '24-03-2018', '07:20:36'),
('Ebrahem Samer', 'Admin', '24-03-2018', '08:47:14'),
('Ebrahem Samer', 'Admin', '24-03-2018', '08:51:01'),
('Shimaa Ahmed', 'Employee', '24-03-2018', '08:52:17'),
('Ebrahem Samer', 'Admin', '24-03-2018', '08:52:50');

-- --

--
-- Table structure for table `message_history`
--

CREATE TABLE `message_history` (
  `MESSAGE_FROM` varchar(20) NOT NULL,
  `MESSAGE_TO` varchar(20) NOT NULL,

```
  `MESSAGE_TEXT` varchar(200) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Table structure for table `purchase`
--

CREATE TABLE `purchase` (
  `BARCODE` varchar(20) NOT NULL,
  `NAME` varchar(50) NOT NULL,
  `TYPE` varchar(20) NOT NULL,
  `COMPANY_NAME` varchar(20) NOT NULL,
  `QUANTITY` int(11) NOT NULL,
  `PRICE` double NOT NULL,
  `AMOUNT` double NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `purchase`
--

INSERT INTO `purchase` (`BARCODE`, `NAME`, `TYPE`, `COMPANY_NAME`,
  `QUANTITY`, `PRICE`, `AMOUNT`) VALUES
('fsdgjfihjorodsf', 'Novalo', 'Bills', 'Med_City', 40, 2, 80);

-- --------------------------------------------------------

--
-- Table structure for table `sales`
--

CREATE TABLE `sales` (
  `BARCODE` varchar(20) NOT NULL,
  `NAME` varchar(50) NOT NULL,
  `TYPE` varchar(10) NOT NULL,
```

```
  `DOSE` varchar(10) NOT NULL,
  `QUANTITY` int(11) NOT NULL,
  `PRICE` double NOT NULL,
  `AMOUNT` double NOT NULL,
  `DATE` varchar(15) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--
-- Table structure for table `users`
--

CREATE TABLE `users` (
  `ID` int(11) NOT NULL,
  `NAME` varchar(50) NOT NULL,
  `DOB` varchar(20) NOT NULL,
  `ADDRESS` varchar(100) NOT NULL,
  `PHONE` varchar(20) NOT NULL,
  `SALARY` double NOT NULL,
  `PASSWORD` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `users`
--

INSERT INTO `users` (`ID`, `NAME`, `DOB`, `ADDRESS`, `PHONE`, `SALARY`,
  `PASSWORD`) VALUES
(1, 'Awinthika S', '23-12-2005', 'Chennai/\r\nst/India', '01128284736', 5000,
  'awin123'),
(2, 'Aboorvan', '3-2-2005', 'Damanhour/\nShobra', '01290789432', 2000, 'aboo123'),
(3, 'Prajein', '3-2-1971', 'Egypt/\nElmanaoura', '01147893423', 4000, 'praj123'),
(4, 'Ali Mostafa', '7-8-1977', 'Egypt/\nEl_mansoura/\nshobra', '011804368743', 3000,
  'alimohammed');

--
```

```
-- Indexes for dumped tables
--


--
-- Indexes for table `company`
--
ALTER TABLE `company`
  ADD PRIMARY KEY (`NAME`);

--
-- Indexes for table `drugs`
--
ALTER TABLE `drugs`
  ADD PRIMARY KEY (`BARCODE`);

--
-- Indexes for table `purchase`
--
ALTER TABLE `purchase`
  ADD PRIMARY KEY (`BARCODE`);

--
-- Indexes for table `users`
--
ALTER TABLE `users`
  ADD PRIMARY KEY (`ID`);

/*!40101 SET
   CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET
   CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET
   COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

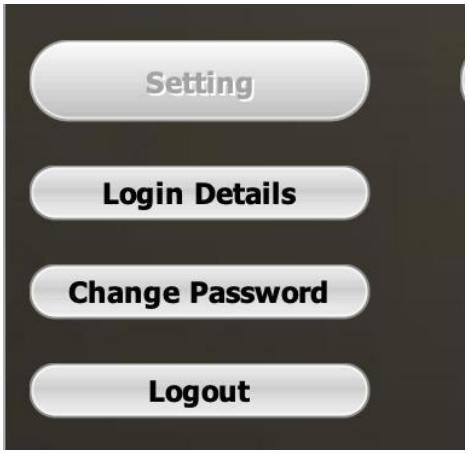
CHAPTER 6

RESULT AND DISCUSSION

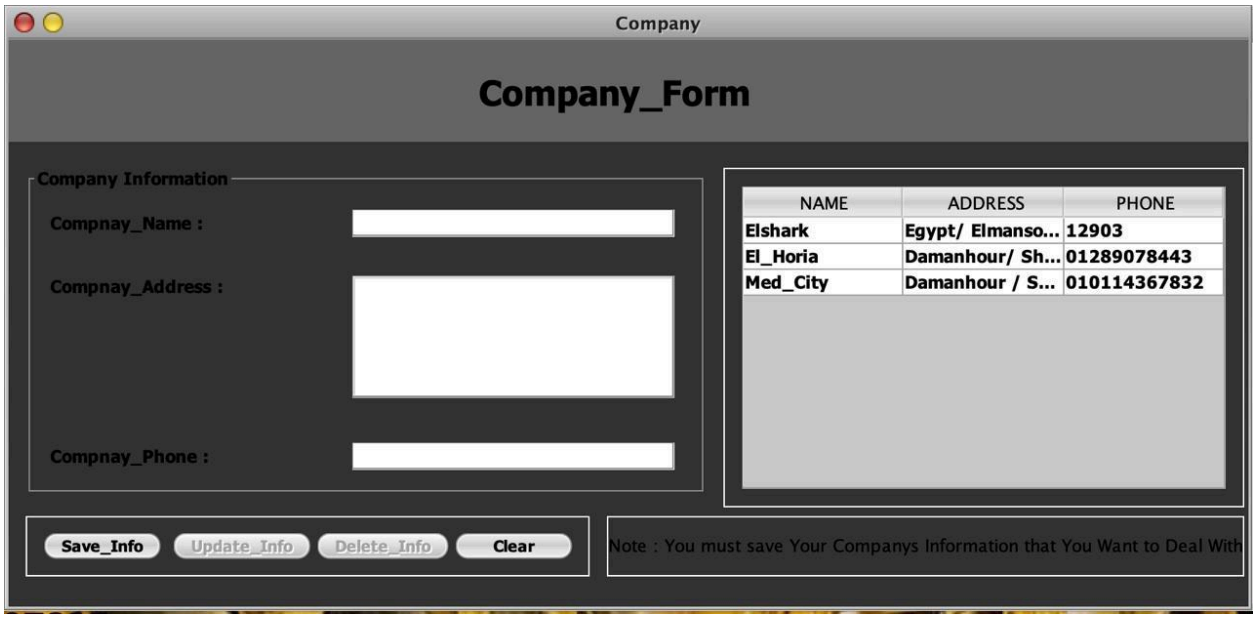
5.1 DATABASE DESIGN

The database design for the pharmacy management system includes five tables: Customers, Medicines, Stock, Sales, and Pharmacists. The Customers table stores customer details like ID, name, phone, and address. The Medicines table holds medicine information, including ID, name, manufacturer, and price. The Stock table tracks the quantity of each medicine, referencing the Medicines table via the medicine ID. The Sales table records sales transactions, including customer ID, medicine ID, quantity sold, and sale date, referencing both the Customers and Medicines tables. The Pharmacists table contains pharmacist details like ID, name, and shift. The system supports basic CRUD operations for each entity, allowing the addition of new customers, medicines, and stock, recording sales, and viewing stored data. Referential integrity is maintained through foreign key constraints between related tables, ensuring consistency in the data.

6.2 OUTPUT:

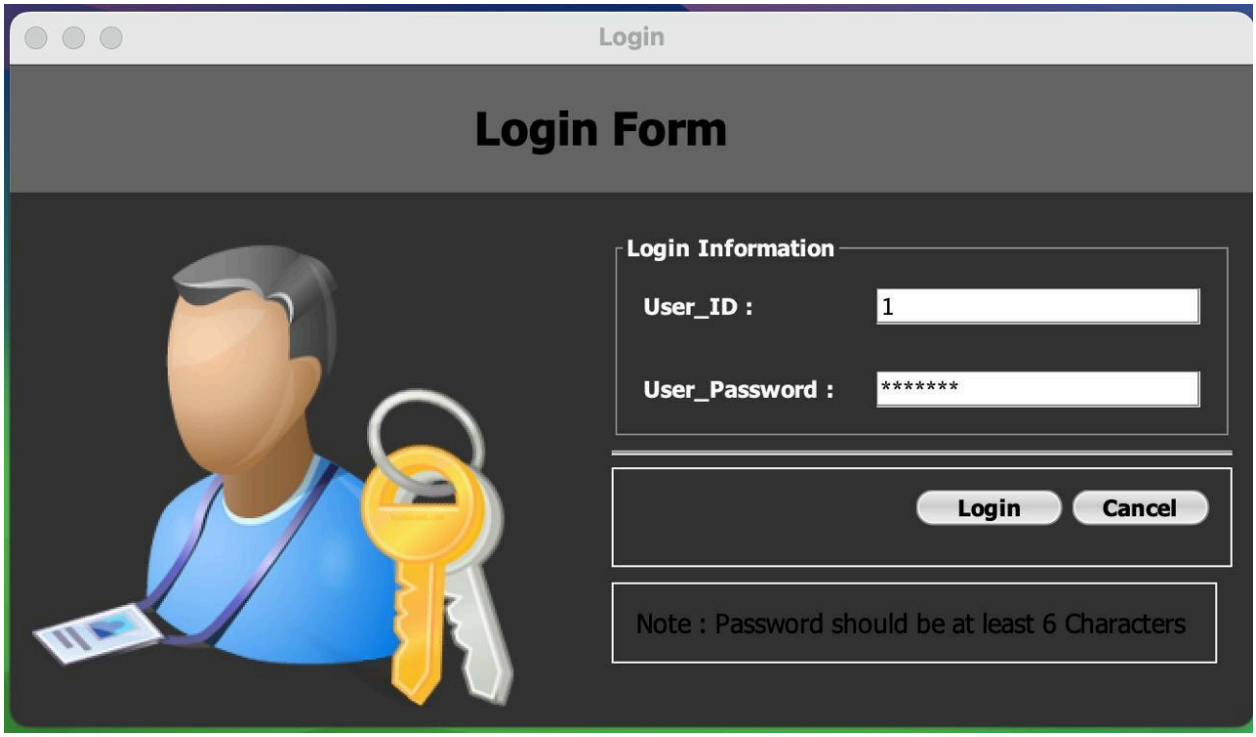


(a) settings

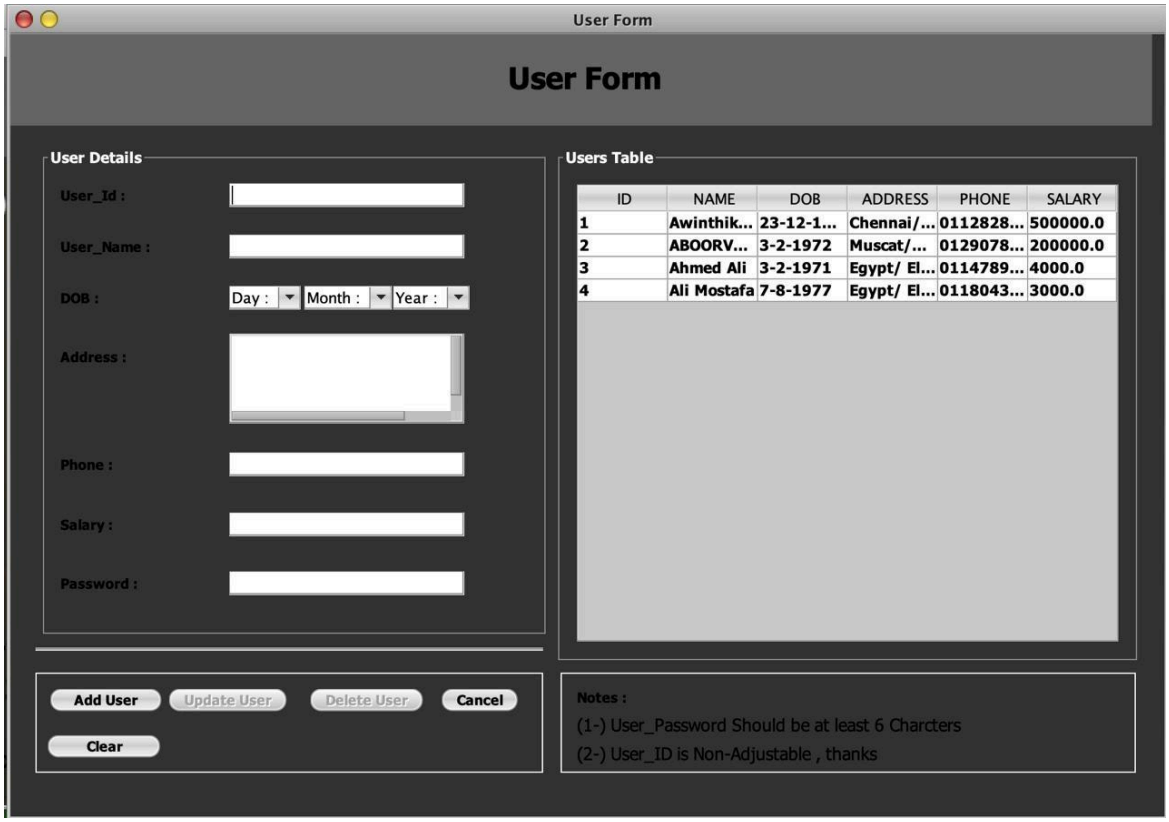


(b) Company form

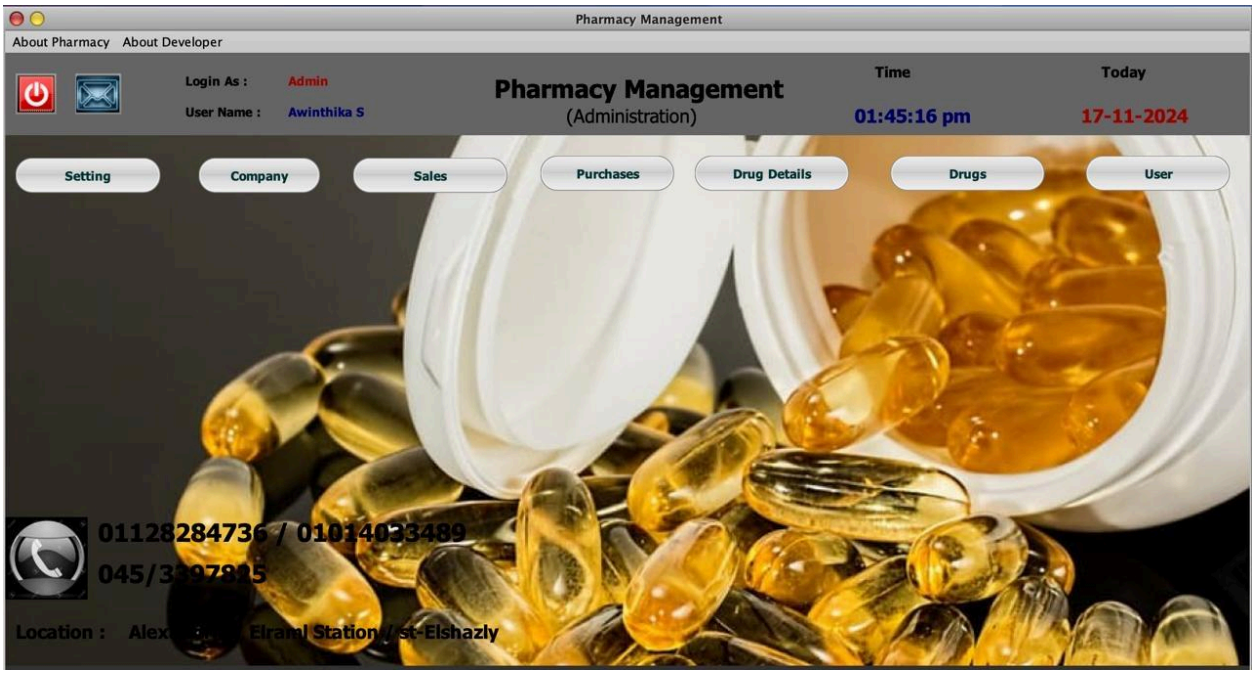
© sales bill form



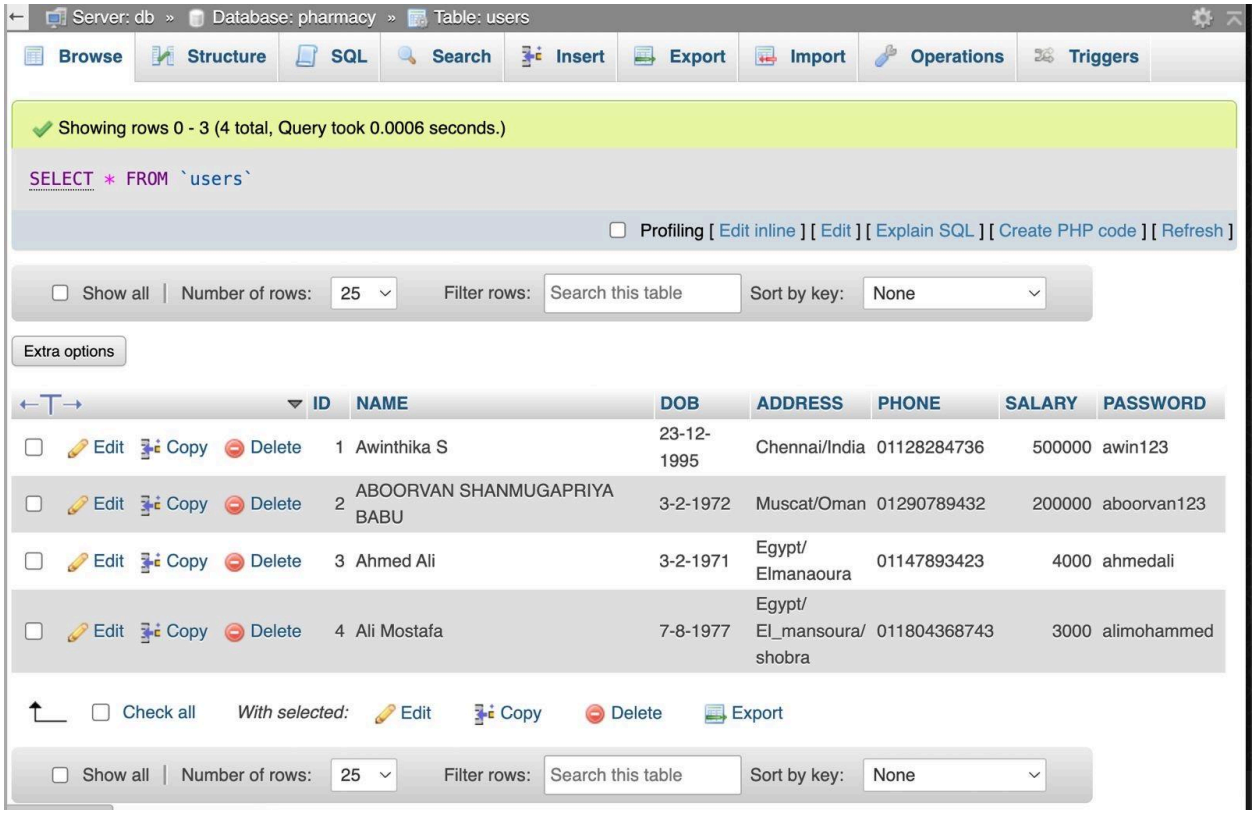
(d) login form



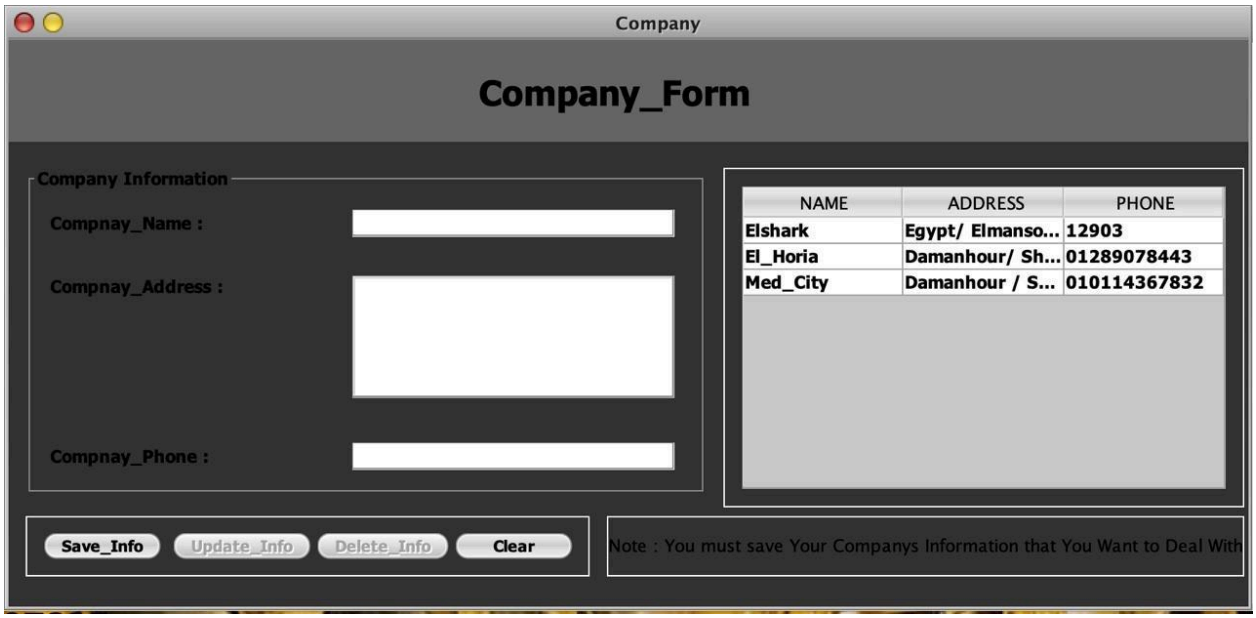
(e) user form



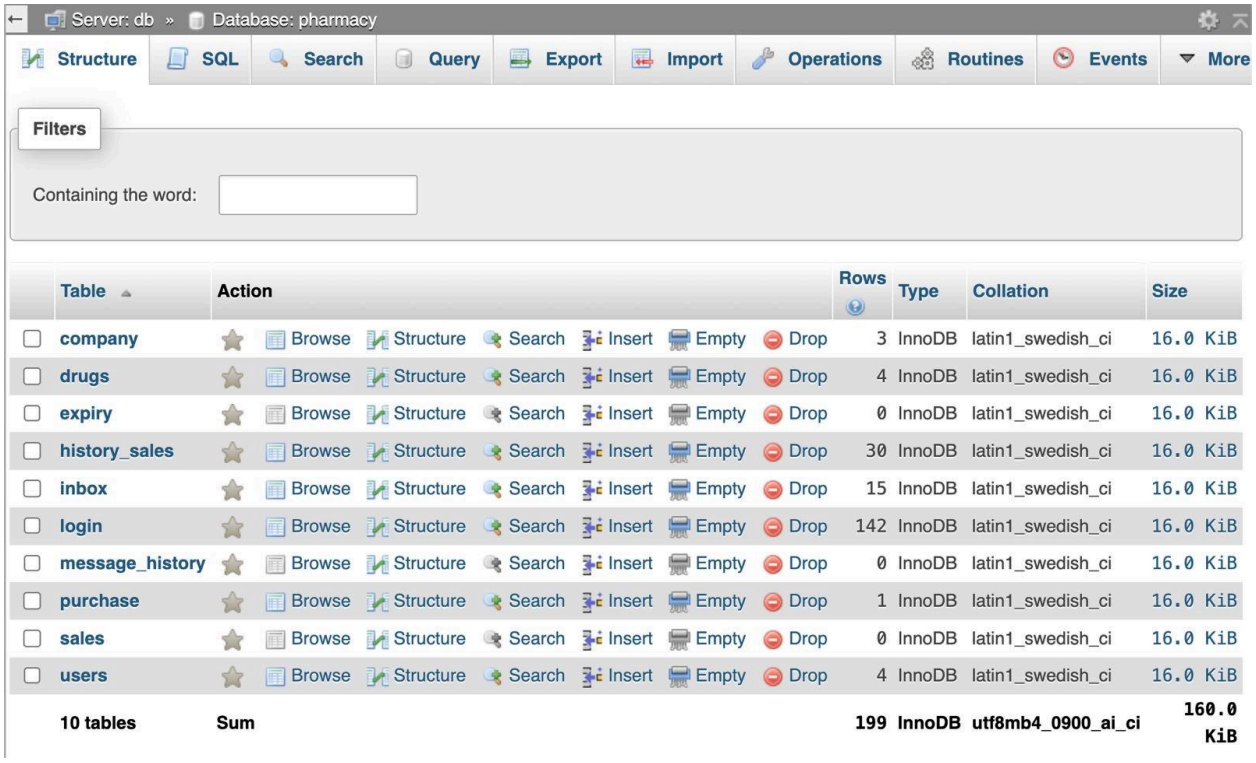
(f) dashboard



(g) users table

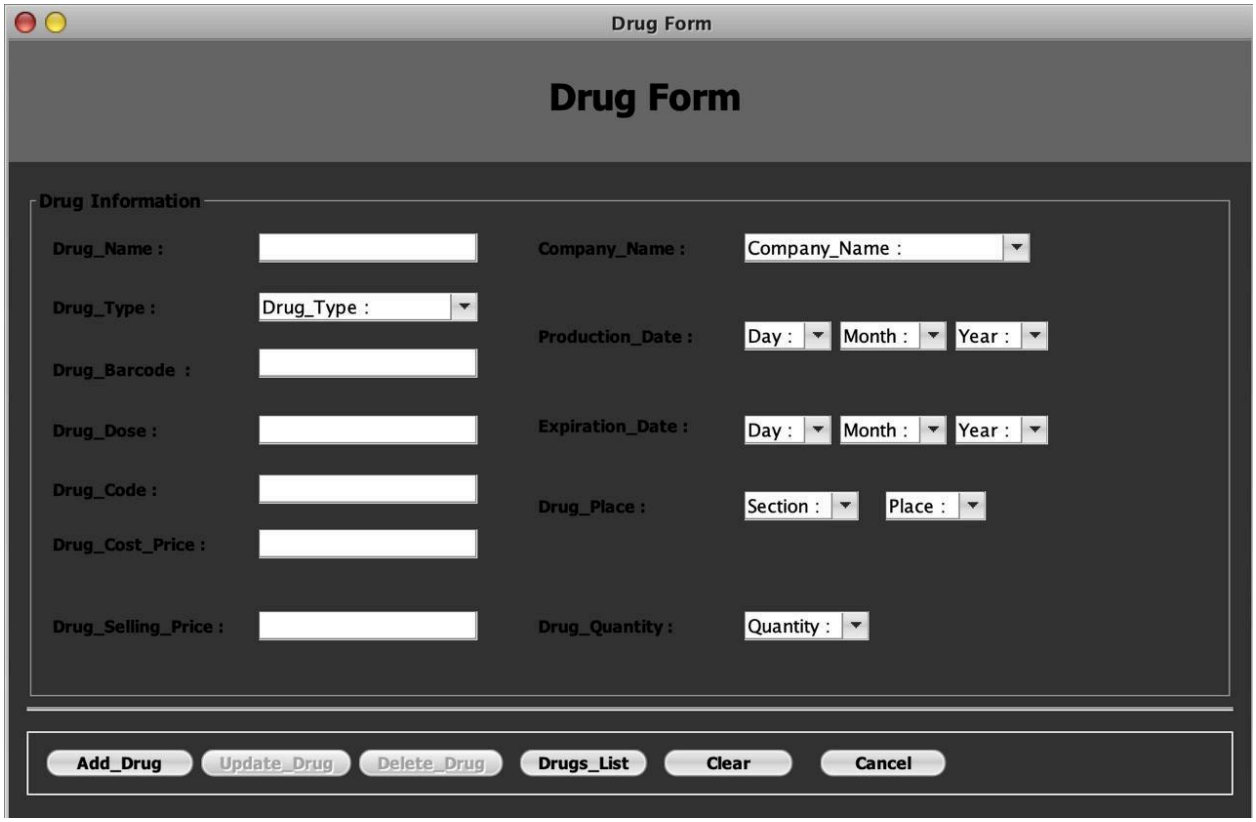


(h) company form



(i) pharmacy DB structure view in myphpadmin

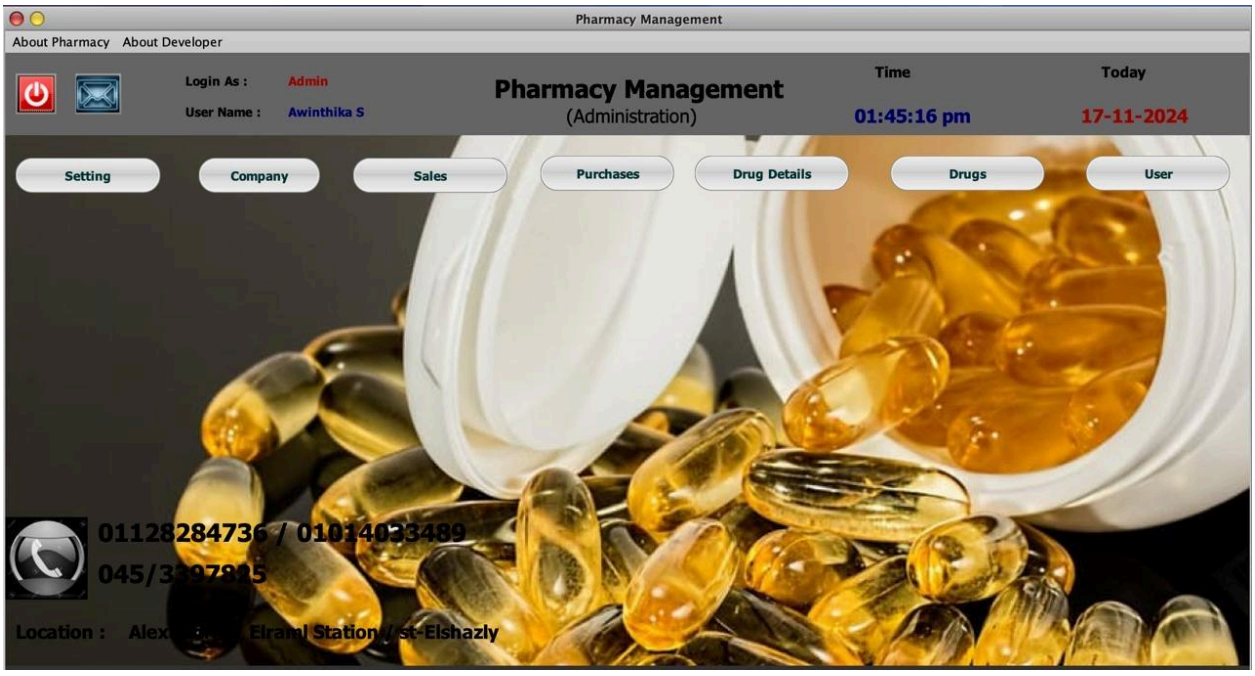
(j) buy drug form



(k) form to add drug

CHAPTER 7 FUTURE SCOPE

Future work for the Pharmacy Management System involves several enhancements aimed at maintaining its relevance and efficiency in an evolving pharmaceutical landscape. Enhancing reporting and analytics will provide detailed insights into sales trends, inventory levels, and customer purchasing patterns, while integration with healthcare systems, such as electronic health records (EHR), will facilitate seamless prescription data sharing and validation. Developing a mobile application will enable on-the-go access for pharmacists and customers, and integrating IoT devices will automate inventory tracking and stock updates. Implementing AI and machine learning will optimize inventory management and personalize marketing efforts, while advanced security measures, including two-factor authentication and biometric verification, will ensure robust protection against vulnerabilities. Expanding CRM features will improve customer interactions and loyalty programs, and integrating diverse payment options will accommodate various transaction methods. Ensuring ongoing compliance with healthcare regulations like GDPR and HIPAA, and providing comprehensive user training and support will enhance system usability. Scalability and performance optimization will cater to larger pharmacy chains, while multilingual and localization support will serve a diverse user base. Additionally, promoting environmental sustainability through digital receipts and responsible disposal of expired medicines will contribute to eco-friendly practices. These future enhancements will ensure the Pharmacy Management System remains a vital, efficient, and secure tool for pharmacies globally.



CHAPTER 8 TESTING

1. 3.1.UNIT TESTING:

Unit testing involves testing individual components of the system to ensure they function correctly. For your system, you should test: Database initialization functions, Functions to add customers, medicines, and sales , Functions to retrieve data from the database.

2. 3.2.Integration Testing

Integration testing ensures that different modules of the system work together as expected. This involves testing the interaction between the database and the application logic.

3. 3.3.Functional Testing

Functional testing ensures that the system's functionality meets the specified requirements. This involves testing all the features of the system to ensure they work as expected.

4. 3.4.Performance Testing

Performance testing ensures that the system performs well under expected load conditions. This involves testing the response times and throughput of the application.

5. 3.5.Regression Testing

Regression testing ensures that new changes or updates to the system do not introduce new bugs. This involves re-running previous test cases to verify that the existing functionality still works as expected.

6. 3.6.Usability Testing

Usability testing ensures that the system is user-friendly and easy to navigate. This involves testing the user interface and user experience.

CHAPTER 9 CONCLUSION

In conclusion, the Pharmacy Management System stands as a robust and comprehensive solution designed to address the multifaceted needs of modern pharmacies. With core functionalities that include managing customer records, maintaining medicine inventories, recording sales transactions, and tracking pharmacists' shifts, the system ensures efficient and streamlined operations. Its implementation with SQLite guarantees data integrity and accessibility, while the intuitive Streamlit-based user interface facilitates user interaction. Rigorous testing across unit, integration, functional, performance, security, usability, and regression

dimensions ensures the system's reliability and security. Looking forward, strategic enhancements such as advanced reporting, integration with healthcare systems, mobile accessibility, IoT integration, AI-driven optimizations, enhanced security measures, expanded CRM features, diverse payment options, regulatory compliance, scalability, multilingual support, and sustainability initiatives will further elevate the system.

REFERENCES

1. H. F. Korth, A. Silberschatz, and S. Sudarshan, Database System Concepts, 6th ed. New York: McGraw-Hill, 2010.
2. R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, 7th ed. Boston: Addison-Wesley, 2016.
3. C. J. Date, An Introduction to Database Systems, 8th ed. Reading, MA: Addison-Wesley, 2003.

4. K. Ullas, P. J. Fatima, and R. Rajesh, "An Efficient Pharmacy Management System Using Relational Database," International Journal of Pharmacy and Technology, vol. 12, no. 1, pp. 1121-1128, Mar. 2020.
5. L. N. Kumar and V. Kumar, "Data Management and Analytics in Healthcare Systems," Journal of Healthcare Engineering, vol. 2018, Article ID 7912820, 10 pages, 2018.
6. M. S. Khan, "Improving Medication Safety through Computerized Systems," Journal of Healthcare Informatics Research, vol. 5, no. 2, pp. 95-105, Apr. 2021.
7. A. Sharma, B. Mehta, and R. Gupta, "Design and Implementation of a Pharmacy Management System Using SQL," in Proc. 2019 International Conference on Innovations in Information Technology, Al Ain, UAE, Dec. 2019, pp. 32-37.
8. P. Nguyen and T. Tran, "Enhancing Pharmacy Inventory Management with Database Solutions," in Proc. 2020 IEEE International Conference on Healthcare Informatics, Beijing, China, Aug. 2020,

pp. 45-50.

9. M. Patel, "Best Practices for Database Management in Pharmacy Systems," [Online]. Available: https://www.pharmacytimes.com/best-practices-for-database-management. [Accessed: May 10, 2024].