Computer Science Project
MediCheck

Kendriya Vidyalaya Sector-5, Dwarka
Class: 12th A

Submitted By:

Kunal Kumar

Akshat Singh

Priya Jha

Table of Contents

Certificate

This is to certify that Kunal Kumar, Akshat Singh and Priya Jha of Class XII A of Kendriya Vidyalaya Sector-5, Dwarka have done their project on MediCheck: Pharmacy Management System under the supervision of Mr. Rahul Lakra. They have taken interest and have shown utmost sincerity in the completion of this project.

I certify that this project is as per the guidelines issued by the CBSE Board.

Internal Examiner **External Examiner**

Acknowledgment

It is with great pleasure that we acknowledge our sincere gratitude to our teacher, Mr. Rahul Lakra who taught and undertook the responsibility of teaching us the subject of Computer Science. We have greatly benefited from his classes.

We are especially indebted to our Principal Mr. Satish Singh who has always been a source of encouragement and support and without whose inspiration this project would not have been successful.

Finally, we would like to express our sincere appreciation for all the other students of our batch for their support & help in completing the project.

Introduction

MediCheck is a comprehensive Pharmacy Management System designed to streamline and enhance various aspects of pharmaceutical operations. It serves as a centralized platform for managing medicines, stocks, and orders efficiently. The system provides a user-friendly interface, making it accessible for pharmacy staff to perform tasks such as adding new medicines, updating stock information, and processing customer orders.

Uses Of MediCheck

1. **Inventory Management:** One of the primary uses of MediCheck is to maintain an organized record of medicines and their corresponding stocks. This ensures that pharmacies can easily track available inventory, expiration dates, and other critical information.

2. **Order Processing:** The system facilitates the seamless processing of customer orders. Pharmacists can quickly check the availability of medicines, manage order details, and provide accurate information to customers.

3. **Data Organization:** With a structured database, MediCheck enables pharmacies to organize and retrieve essential information efficiently. This proves crucial for regulatory compliance, audits, and overall business management.

4. **Customer Service:** The system enhances customer service by enabling pharmacists to access real-time information on medicines, their prices, and availability. This ensures that customers receive accurate and prompt assistance.

5. **Reporting and Analytics:** MediCheck comes equipped with reporting features that allow pharmacies to generate insights into sales trends, stock

levels, and order history. This data-driven approach empowers pharmacies to make informed business decisions.

How MediCheck Helps People?

- **Patients:** Patients benefit from an organized and efficient pharmacy system as it ensures that medicines are readily available, reducing waiting times. Accurate order processing and inventory management contribute to an overall positive customer experience.

- **Pharmacy Staff:** Pharmacy staff experience improved workflow and reduced manual workload. The system automates many processes, minimizing the risk of errors and allowing staff to focus on delivering high-quality service.

- **Business Owners:** For pharmacy owners, MediCheck offers a comprehensive solution for managing business operations. The reporting and analytics features provide valuable insights for strategic decision-making, leading to improved profitability and growth.

Software Development Life Cycle

The development of the MediCheck Pharmacy Management System followed a structured 3-Step Software Development Life Cycle (SDLC) to ensure a systematic and efficient approach to the project. The SDLC used can be outlined as follows:

1. **Planning:** In this phase, the project requirements were gathered and analyzed. The team defined the scope, objectives, and constraints of the system. A project plan was created, outlining the tasks, timelines, and resource requirements.

2. **Design:** The design phase involved creating a detailed blueprint of the system architecture, database schema, and user interface. The team collaborated to define the data models, functionalities, and user workflows. This phase laid the foundation for the subsequent implementation.

3. **Implementation:** The actual coding of the system took place in the implementation phase. We used the Python programming language to write the application code, and MySQL was chosen as the database management system. Regular code reviews and collaboration ensured the quality and consistency of the codebase.

The SDLC adopted for the development of MediCheck ensured a systematic and well-organized approach, resulting in a reliable and feature-rich Pharmacy Management System.

Showcase

System Requirements

Hardware Requirements:

- **Processor:** Dual-core processor or higher
- **RAM:** 4 GB or higher
- **Storage:** 20 MB or higher free disk space

Software Requirements:

- **Operating System:** Linux, macOS or Windows
- **Software Programs:** Python 3.7+, MySQL 8.0+
- **Dependencies:** mysql-connector-python, pickle, os

Overview

MediCheck is a Python-based application that leverages MySQL as its underlying database management system. The code comprises three main modules: Medicines, Stocks, and Orders. It facilitates database connections, table creation, data insertion, retrieval, and updating. Let's break down the main components and functionalities in simple terms:

- **Initialization:** The code starts by initializing the necessary components, including establishing a connection to the MySQL database and creating tables for Medicines, Stocks, and Orders.

- **Functions:** The code defines functions for adding, searching, displaying, editing, and deleting records in the database. These functions encapsulate the core functionalities of the system, ensuring modularity and ease of maintenance.

- **Main Menu:** The main menu serves as the user interface, allowing users to navigate through different sections, including Medicines, Stocks, Orders, and an option to exit the application.

- **Menu Options:** Within each section, users can perform various operations such as viewing, searching, updating, adding, and deleting records. The code provides a structured and interactive menu for users to choose their desired actions.

- **Database Operations:** The code interacts with the MySQL database to execute SQL queries for tasks such as retrieving data, updating records, and adding new entries. This ensures the real-time synchronization of the system with the underlying database.

- **User Input Handling:** The code incorporates user input handling to gather information for operations like adding new medicines, updating stock details, or searching for specific records. This makes the system interactive and user-friendly.

Code:

```
# Imports
import mysql.connector as con
import pickle
import os


# Setup
# Create credentials for MySQL connection
if not os.path.isfile('./credentials.dat'):
    f = open('credentials.dat', 'wb+')
    print("Credentials File Not Found... Creating New File...")
    username = input("Enter MySQL User: ")
    password = input("Enter Password: ")
    credentials = (username, password)
    pickle.dump(credentials, f)
    f.close()
    print("Credentials Creating Completed. Exiting...")
    print("Rerun Application to Continue.")
    exit(0)
```

```
# Create MySQL connection
print("Initializing SQL Connection...")
f = open('credentials.dat', 'rb')
credentials = pickle.load(f)
db = con.connect(host='localhost', user=credentials[0], password=credentials[1])
cur = db.cursor()
if not db.is_connected():
    print("Connection Failed. Exiting...")
    exit(0)
else:
    print("Connection Successful.\n")
f.close()

# Create Databases & Tables
print('Initializing Database and Tables...')
cur.execute('CREATE DATABASE IF NOT EXISTS MediCheck;')
db.database = 'MediCheck'
cur.execute(
'CREATE TABLE IF NOT EXISTS Medicines(MedID INTEGER PRIMARY KEY, Name VARCHAR(50) NOT NULL, T
ype VARCHAR(50) NOT NULL, Manufacture_Company VARCHAR(50) NOT NULL, Description VARCHAR(300),
Side_Effects VARCHAR(300) NOT NULL, Price DECIMAL(6,2) NOT NULL, Available BOOLEAN NOT NULL,
Latest_Batch_Date DATE);'
)
cur.execute(
'CREATE TABLE IF NOT EXISTS Stocks(StockID INTEGER PRIMARY KEY, MedID INTEGER NOT NULL, Box_C
ount INTEGER NOT NULL, Arrival_Date DATE NOT NULL, Manufacture_Date DATE NOT NULL, Expiration
_Date DATE NOT NULL, Price DECIMAL(7,2) NOT NULL, Shelf_No CHAR(4) NOT NULL, Finished BOOLEAN
NOT NULL, FOREIGN KEY (MedID) REFERENCES Medicines(MedID) ON DELETE CASCADE);'
)
cur.execute(
'CREATE TABLE IF NOT EXISTS Orders(OrderID INTEGER PRIMARY KEY, Name VARCHAR(50) NOT NULL, Ad
dress VARCHAR(150) NOT NULL, Phone_No INTEGER NOT NULL, Order_Charge DECIMAL(7,2) NOT NULL, O
rder_Date DATE NOT NULL, Medicines VARCHAR(100) NOT NULL, Med_Count VARCHAR(100) NOT NULL, Co
mpleted BOOLEAN NOT NULL, Completion_Date DATE);'
)
print("Initialization Completed. Starting Application...\n")
```

```
# Functions

# Add data to tables

def add(table: str, item_count=1):

    if table == 'Medicines':

        for i in range(item_count):

            if item_count == 1:

                print("\nEnter Details of the Medicine:")

            else:

                print(f"\nEnter Details of Medicine {i+1}:")

            med_id = int(input("MedID: "))

            med_name = input("Medicine Name: ")

            med_type = input("Medicine Type: ")

            company = input("Manufacturing Company: ")

            description = input("Description: ")

            side_effects = input("Side-Effects(separated by ','): ")

            price = float(input("Medicine Price: "))

            available = bool(int(input("Medicine Available(1 = yes, 0 = no): ")))

            latest_date = input("Newest Batch Date(YYYY-MM-DD): ")
```

```
        if latest_date == '':

            latest_date = '1970-01-01'

        print('\nAdding to Database...')

        cur.execute(f"INSERT INTO {table} VALUES({med_id}, '{med_name}', '{med_type}', '{
company}', '{description}', '{side_effects}', {price}, {available}, '{latest_date}');")

    db.commit()

    print(f"{item_count} Medicines Added to Database.")

elif table == 'Stocks':

    for i in range(item_count):

        if item_count == 1:

            print("\nEnter Details of the Stock:")

        else:

            print(f"\nEnter Details of Stock {i+1}:")

        stock_id = int(input("StockID: "))

        med_id = int(input("MedID of Stock Medicine: "))

        box_count = int(input("No. of Boxes: "))

        arrival_date = input("Arrival Date(YYYY-MM-DD): ")

        manufacture_date = input("Manufacture Date(YYYY-MM-DD): ")

        expiration_date = input("Expiration Date(YYYY-MM-DD): ")

        stock_price = float(input('Stock Price: '))

        shelf_no = input("Shelf No of Stock(Ex-A124): ")

        print('\nAdding to Database...')
```

```
        cur.execute(f"INSERT INTO {table} VALUES({stock_id}, {med_id}, {box_count}, '{
arrival_date}', '{manufacture_date}', '{expiration_date}', {stock_price}, '{shelf_no}
', False);")
    db.commit()
    print(f"{item_count} Stocks Added to Database.")
elif table == "Orders":
    print("\nEnter Order Details:")
    order_id = int(input("OrderID: "))
    client_name = input("Client Name: ")
    client_address = input("Client Address: ")
    phone_no = input("Client Phone No.: ")
    order_charge = float(input("Order Charge"))
    order_date = input("Order Date(YYYY-MM-DD): ")
    medicines = input("Medicines(MedID separated by ','): ")
    med_count = input("Medicine Amount(with respect to above entry separated by ','): ")
    print('\nAdding to Database...')
    cur.execute(f"INSERT INTO {table} VALUES({order_id}, '{client_name}', '{client_address}
', {phone_no}, {order_charge}, '{order_date}', '{medicines}', '{med_count}
', False, '1970-01-01');")
    db.commit()
    print(f"Order Added to Database.")
```

```
# Search through the tables in different modes for different filters
def search(table: str, search_mode: int, search_kw='', search_count=0):
    if search_mode == 0:
        print('Fetching Full Data...')
        cur.execute(f'SELECT * FROM {table};')
    elif table == 'Medicines':
        print(f'Fetching Data...')
        if search_mode == 1:
            cur.execute(f'SELECT * FROM {table} WHERE Name = "{search_kw}";')
        elif search_mode == 2:
            cur.execute(f'SELECT * FROM {table} WHERE Type = "{search_kw}";')
        elif search_mode == 3:
            cur.execute(f'SELECT * FROM {table} WHERE Manufacture_Company = "{search_kw}";')
        elif search_mode == 4:
            cur.execute(f'SELECT * FROM {table} WHERE Available = {search_kw};')
        elif search_mode == 5:
            cur.execute(f'SELECT * FROM {table} ORDER BY Latest_Batch_Date DESC;')
    elif table == 'Stocks':
        print(f'Fetching Data...')
        if search_mode == 1:
            cur.execute(f'SELECT * FROM {table} WHERE MedID = {search_kw};')
```

```
    elif search_mode == 2:
        cur.execute(f'SELECT * FROM {table} WHERE Arrival_Date = "{search_kw}";')
    elif search_mode == 3:
        cur.execute(f'SELECT * FROM {table} WHERE Expiration_Date = "{search_kw}
" AND Finished = 0;')
    elif search_mode == 4:
        cur.execute(f'SELECT * FROM {table} WHERE Shelf_No = "{search_kw}";')
    elif search_mode == 5:
        cur.execute(f'SELECT * FROM {table} WHERE Finished = {search_kw};')
elif table == 'Orders':
    print(f'Fetching Data...')
    if search_mode == 1:
        cur.execute(f'SELECT * FROM {table} WHERE Name = "{search_kw}";')
    elif search_mode == 2:
        cur.execute(f'SELECT * FROM {table} WHERE Order_Date = "{search_kw}";')
    elif search_mode == 3:
        cur.execute(f'SELECT * FROM {table} WHERE Completed = {search_kw};')
    elif search_mode == 4:
        cur.execute(f'SELECT * FROM {table} WHERE Completion_Date = "{search_kw}";')
if search_count == 0:
    return cur.fetchall()
else:
    return cur.fetchmany(search_count)
```

```
# Display the fetched data in a tabular form
def display(table:str, cursor_list: list, display_list: list):
    if table == 'Medicines':
        index_dict = {'MedID':0, 'Name':1, 'Type':2, 'Manufacture_Company':3, 'Description':4
, 'Side_Effects':5, 'Price':6, 'Available':7, 'Latest_Batch_Date':8}
    elif table == 'Stocks':
        index_dict = {'StockID':0, 'MedID':1, 'Box_Count':2, 'Arrival_Date':3,
'Manufacture_Date':4, 'Expiration_Date':5, 'Price':6, 'Shelf_No':7, 'Finished':8}
    elif table == 'Orders':
        index_dict = {'OrderID':0, 'Name':1, 'Address':2, 'Phone_No':3, 'Order_Charge':4,
'Order_Date':5, 'Medicines':6, 'Med_Count':7, 'Completed':8, 'Completion_Date':9}

    print('Displaying Fetched Data...')
    for i in display_list:
        print(f'| {i} |', end='')
    print('\n\n')
    for i in cursor_list:
        for j in display_list:
            if len(str(i[index_dict[j]])) > 15:
                print(f'| {str(i[index_dict[j]])[:15]}... |', end='')
            else:
                print(f'| {i[index_dict[j]]} |', end='')
        print('\n')
```

```
# Edit existing data
def edit(table: str, item_id: int):
    if table == 'Medicines':
        cur.execute(f'SELECT * FROM {table} WHERE MedID = {item_id};')
        data = cur.fetchone()
        print('\nEnter Details of Medicine to edit, leave blank to keep existing data:')
        med_name = input(f"Medicine Name[{data[1]}]: ")
        med_type = input(f"Medicine Type[{data[2]}]: ")
        company = input(f"Manufacturing Company[{data[3]}]: ")
        description = input(f"Description[{data[4]}]: ")
        side_effects = input(f"Side-Effects[{data[5]}]: ")
        price = input(f"Medicine Price[{data[6]}]: ")
        available = input(f"Medicine Available[{data[7]}]: ")
        latest_date = input(f"Newest Batch Date[{data[8]}]: ")
        edit_lst =
 [med_name, med_type, company, description, side_effects, price, available, latest_date]

        print('Editing Medicine Record...')
        q_str = f'UPDATE {table} SET'
        for i in edit_lst:
            if i =≠ '':
```

```
            if edit_lst.index(i) == 0:
                q_str += f" Name = '{i}',"
            elif edit_lst.index(i) == 1:
                q_str += f" Type = '{i}',"
            elif edit_lst.index(i) == 2:
                q_str += f" Manufacture_Company = '{i}',"
            elif edit_lst.index(i) == 3:
                q_str += f" Description = '{i}',"
            elif edit_lst.index(i) == 4:
                q_str += f" Side_Effects = '{i}',"
            elif edit_lst.index(i) == 5:
                q_str += f" Price = {float(i)},"
            elif edit_lst.index(i) == 6:
                q_str += f" Available = {bool(int(i))},"
            elif edit_lst.index(i) == 7:
                q_str += f" Latest_Batch_Date = '{i}'"
    cur.execute(q_str.strip(',') + f' WHERE MedID = {item_id};')
    db.commit()
    print('Done.')
elif table == 'Stocks':
    cur.execute(f'SELECT * FROM {table} WHERE StockID = {item_id};')
```

```
data = cur.fetchone()
print('\nEnter Details of Stock to edit, leave blank to keep existing data:')
med_id = input(f"MedID of Stock Medicine[{data[1]}]: ")
box_count = input(f"No. of Boxes[{data[2]}]: ")
arrival_date = input(f"Arrival Date[{data[3]}]: ")
manufacture_date = input(f"Manufacture Date[{data[4]}]: ")
expiration_date = input(f"Expiration Date[{data[5]}]: ")
stock_price = input(f'Stock Price[{data[6]}]: ')
shelf_no = input(f"Shelf No of Stock[{data[7]}]: ")
finished = input(f"Stock Finished[{data[8]}]: ")
edit_lst =
 [med_id, box_count, arrival_date, manufacture_date, expiration_date, stock_price, shelf_no,
finished]

print('Editing Stock Record...')
q_str = f'UPDATE {table} SET'
for i in edit_lst:
    if i != '':
        if edit_lst.index(i) == 0:
            q_str += f' MedID = {int(i)},'
        elif edit_lst.index(i) == 1:
            q_str += f' Box_Count = {int(i)},'
        elif edit_lst.index(i) == 2:
```

```
                q_str += f' Arrival_Date = "{i}",'
            elif edit_lst.index(i) == 3:
                q_str += f' Manufacture_Date = "{i}",'
            elif edit_lst.index(i) == 4:
                q_str += f' Expiration_Date = "{i}",'
            elif edit_lst.index(i) == 5:
                q_str += f' Price = {float(i)},'
            elif edit_lst.index(i) == 6:
                q_str += f' Shelf_No = "{i}",'
            elif edit_lst.index(i) == 7:
                q_str += f' Finished = {bool(int(i))}'
    cur.execute(q_str.strip(',') + f' WHERE StockID = {item_id};')
    db.commit()
    print('Done.')
elif table == 'Orders':
    cur.execute(f'SELECT * FROM {table} WHERE OrderID = {item_id};')
    data = cur.fetchone()
    print('\nEnter Details of Order to edit, leave blank to keep existing data:')
    client_name = input(f"Client Name[{data[1]}]: ")
    client_address = input(f"Client Address[{data[2]}]: ")
    phone_no = input(f"Client Phone No.[{data[3]}]: ")
```

```
order_charge = input(f"Order Charge[{data[4]}]: ")
order_date = input(f"Order Date[{data[5]}]: ")
medicines = input(f"Medicines[{data[6]}]: ")
med_count = input(f"Medicine Amount[{data[7]}]: ")
completed = input(f"Order Completed[{data[8]}]: ")
completion_date = input(f"Order Completion Date[{data[9]}]: ")
edit_lst =
 [client_name, client_address, phone_no, order_charge, order_date, medicines, med_count, comp
leted, completion_date]

print('Editing Order Details...')
q_str = f'UPDATE {table} SET'
for i in edit_lst:
    if i != '':
        if edit_lst.index(i) == 0:
            q_str += f' Name = "{i}",'
        elif edit_lst.index(i) == 1:
            q_str += f' Address = "{i}",'
        elif edit_lst.index(i) == 2:
            q_str += f' Phone_No = {int(i)},'
        elif edit_lst.index(i) == 3:
            q_str += f' Order_Charge = {float(i)},'
        elif edit_lst.index(i) == 4:
```

```
            q_str += f' Order_Date = "{i}",'
        elif edit_lst.index(i) == 5:
            q_str += f' Medicines = "{i}",'
        elif edit_lst.index(i) == 6:
            q_str += f' Med_Count = "{i}",'
        elif edit_lst.index(i) == 7:
            q_str += f' Completed = {bool(int(i))},'
        elif edit_lst.index(i) == 8:
            q_str += f' Completion_Date = "{i}"'
cur.execute(q_str.strip(',') + f' WHERE OrderID = {item_id};')
db.commit()
print('Done.')
```

```
# Delete unwanted data
def delete(table: str, item_id: int):
    if table == 'Medicines':
        print('\nDeleting Medicine Record...')
        cur.execute(f'DELETE FROM {table} WHERE MedID = {item_id};')
        db.commit()
        print('Done.')
    elif table == 'Stocks':
        print('\nDeleting Stocks Record...')
        cur.execute(f'DELETE FROM {table} WHERE StockID = {item_id};')
        db.commit()
        print('Done.')
    elif table == 'Orders':
        print('\nDeleting Order Record...')
        cur.execute(f'DELETE FROM {table} WHERE OrderID = {item_id};')
        db.commit()
        print('Done.')
```

```
# Main Menu of the application
print('WELCOME TO MEDICHECK!'.center(110,' '))
while True:
    print("""~~~~~~~~~~~~~~~~Menu~~~~~~~~~~~~~~~~
1. Medicines
2. Stocks
3. Orders
4. Exit""")
    tab = input('\nSelect Tab or Exit [1/2/3/4]: ')

    if tab == "1":
        table = 'Medicines'
        while True:
            print("""\n~~~~~~~~~~Medicines Menu~~~~~~~~~~
1. Show All Medicines
2. Search For Medicines
3. Update A Medicine Record
4. Add New Medicines
5. Delete A Medicine Record
6. Exit""")
            choice = input('\nSelect Option or Exit Menu [1/2/3/4/5/6]: ')
```

```
            display_lst = ['MedID', 'Name', 'Type', 'Manufacture_Company', 'Description',
'Side_Effects', 'Price', 'Available', 'Latest_Batch_Date']


            if choice == '1':
                print("Showing All Data...")
                data = search(table, 0)
                count = len(data)
                display(table, data, display_lst)
                print(f'\nDisplaying {count} Rows.')
                print('Done.')
            elif choice == '2' or choice == '3' or choice == '5':
                print("""Search On The Basis Of:
1. Name
2. Type
3. Company
4. Available
5. Newly Available
6. Not Available""")
                s_choice = input('Choose [1/2/3/4/5/6]: ')
                s_count = int(input('How many results should be gathered(0 for all): '))
                keyword = ''
                result = []
```

```
            if s_choice == '1':
                keyword = input('Search: ')
                result = search(table, 1, keyword, s_count)
                count = len(result)
                display(table, result, display_lst)
                print(f'\nDisplay {count} Records.')
            elif s_choice == '2':
                keyword = input('Search: ')
                result = search(table, 2, keyword, s_count)
                count = len(result)
                display(table, result, display_lst)
                print(f'\nDisplaying {count} Records.')
            elif s_choice == '3':
                keyword = input('Search: ')
                result = search(table, 3, keyword, s_count)
                count = len(result)
                display(table, result, display_lst)
                print(f'\nDisplaying {count} Records.')
            elif s_choice == '4':
                keyword = 1
                result = search(table, 4, keyword, s_count)
```

```
            count = len(result)
            display(table, result, display_lst)
            print(f'\nDisplaying {count} Records.')
        elif s_choice == '5':
            result = search(table, 5, search_count=s_count)
            count = len(result)
            display(table, result, display_lst)
            print(f'\nDisplaying {count} Records.')
        elif s_choice == '6':
            keyword = 0
            result = search(table, 4, keyword, s_count)
            count = len(result)
            display(table, result, display_lst)
            print(f'\nDisplaying {count} Records.')


    if choice == '2':
        m_choice = int(input(f'Choose To Learn More[1-{count} or 0 to Exit]: '))
        if m_choice == 0:
            pass
        else:
            medicine = result[m_choice-1]
```

```
                        available = ''
                        if int(medicine[7]) == 0:
                            available = 'No'
                        elif int(medicine[7]) == 1:
                            available = 'Yes'
                        print(f"""\nMore Info on MedID: {medicine[0]}
Name: {medicine[1]}
Type: {medicine[2]}
Side Effects: {medicine[5]}
Manufactured By: {medicine[3]}
Description: {medicine[4]}
Price: {medicine[6]}
Available: {available}
Latest Stock Date: {medicine[8]}""")
                elif choice == '3':
                    m_choice = int(input(f'Choose To Edit[1-{count} or 0 to Exit]: '))
                    if m_choice == 0:
                        pass
                    else:
                        edit(table, result[m_choice-1][0])
                elif choice == '5':
```

```
                    m_choice = int(input(f'Choose To Delete[1-{count} or 0 to Exit]: '))
                    if m_choice == 0:
                        pass
                    else:
                        delete(table, result[m_choice-1][0])
            elif choice == '4':
                n = int(input('\nHow many medicines to add? '))
                add(table, n)
            elif choice == '6':
                print('\nGoing to Main Menu...\n')
                break
    if tab == "2":
        table = 'Stocks'
        while True:
            print("""\n~~~~~~~~~~~~Stocks Menu~~~~~~~~~~~
1. Show Medicine Stock History
2. Search For Stocks
3. Update A Medicine Stock Record
4. Add Newly Received Stocks
5. Delete A Stock Record
6. Exit""")
```

```
            choice = input('\nSelect Option or Exit Menu [1/2/3/4/5/6]: ')
            display_lst = ['StockID', 'MedID', 'Box_Count', 'Arrival_Date',
'Manufacture_Date', 'Expiration_Date', 'Price', 'Shelf_No', 'Finished']

            if choice == '1':
                print("Showing All Data...")
                data = search(table, 0)
                count = len(data)
                display(table, data, display_lst)
                print(f'\nDisplaying {count} Rows.')
                print('Done.')
            elif choice == '2' or choice == '3' or choice == '5':
                print("""Search On The Basis Of:
1. Medicine
2. Arrival Date
3. Expiration Date
4. Shelf No.
5. Finished
6. Not Finished""")
                s_choice = input('Choose [1/2/3/4/5/6]: ')
                s_count = int(input('How many results should be gathered(0 for all): '))
                keyword = ''
```

```
            result = []


            if s_choice == '1':
                keyword = int(input('Search: '))
                result = search(table, 1, keyword, s_count)
                count = len(result)
                display(table, result, display_lst)
                print(f'\nDisplay {count} Records.')
            elif s_choice == '2':
                keyword = input('Search[YYYY-MM-DD]: ')
                result = search(table, 2, keyword, s_count)
                count = len(result)
                display(table, result, display_lst)
                print(f'\nDisplaying {count} Records.')
            elif s_choice == '3':
                keyword = input('Search[YYYY-MM-DD]: ')
                result = search(table, 3, keyword, s_count)
                count = len(result)
                display(table, result, display_lst)
                print(f'\nDisplaying {count} Records.')
            elif s_choice == '4':
```

```
            keyword = input('Search[Ex-A123]: ')
            result = search(table, 4, keyword, s_count)
            count = len(result)
            display(table, result, display_lst)
            print(f'\nDisplaying {count} Records.')
        elif s_choice == '5':
            keyword = 1
            result = search(table, 5, keyword, s_count)
            count = len(result)
            display(table, result, display_lst)
            print(f'\nDisplaying {count} Records.')
        elif s_choice == '6':
            keyword = 0
            result = search(table, 5, keyword, s_count)
            count = len(result)
            display(table, result, display_lst)
            print(f'\nDisplaying {count} Records.')

        if choice == '2':
            m_choice = int(input(f'Choose To Learn More[1-{count} or 0 to Exit]: '))
            if m_choice == 0:
```

```
                    pass
                else:
                    stock = result[m_choice-1]
                    finished = ''
                    if int(stock[8]) == 0:
                        finished == 'No'
                    elif int(stock[8]) == 1:
                        finished == 'Yes'
                    print(f"""\nMore Info on StockID: {stock[0]}
MedID: {stock[1]}
No. Of Boxes: {stock[2]}
Arrival Date: {stock[3]}
Manufacture Date: {stock[4]}
Expiration Date: {stock[5]}
Price: {stock[6]}
Shelf No.: {stock[7]}
Finished: {finished}""")
            elif choice == '3':
                m_choice = int(input(f'Choose To Edit[1-{count} or 0 to Exit]: '))
                if m_choice == 0:
                    pass
```

```
                    else:
                        edit(table, result[m_choice-1][0])
                elif choice == '5':
                    m_choice = int(input(f'Choose To Delete[1-{count} or 0 to Exit]: '))
                    if m_choice == 0:
                        pass
                    else:
                        delete(table, result[m_choice-1][0])
            elif choice == '4':
                n = int(input('\nHow many stocks to add? '))
                add(table, n)
            elif choice == '6':
                print('\nGoing to Main Menu...\n')
                break
    if tab == "3":
        table = 'Orders'
        while True:
            print("""\n~~~~~~~~~~~~~Orders Menu~~~~~~~~~~~
1. Show Order History
2. Order Search
3. Update Order Details
```

```
4. Place New Orders
5. Delete Order Record
6. Exit""")
            choice = input('\nSelect Option or Exit Menu [1/2/3/4/5/6]: ')
            display_lst = ['OrderID', 'Name', 'Address', 'Phone_No', 'Order_Charge',
'Order_Date', 'Medicines', 'Med_Count', 'Completed', 'Completion_Date']


            if choice == '1':
                print("Showing All Data...")
                data = search(table, 0)
                count = len(data)
                display(table, data, display_lst)
                print(f'\nDisplaying {count} Rows.')
                print('Done.')
            elif choice == '2' or choice == '3' or choice == '5':
                print("""Search On The Basis Of:
1. Client Name
2. Order Date
3. Completed
4. Completion Date
5. Not Completed""")
                s_choice = input('Choose [1/2/3/4/5]: ')
```

```
            s_count = int(input('How many results should be gathered(0 for all): '))
            keyword = ''
            result = []

            if s_choice == '1':
                keyword = input('Search: ')
                result = search(table, 1, keyword, s_count)
                count = len(result)
                display(table, result, display_lst)
                print(f'\nDisplay {count} Records.')
            elif s_choice == '2':
                keyword = input('Search[YYYY-MM-DD]: ')
                result = search(table, 2, keyword, s_count)
                count = len(result)
                display(table, result, display_lst)
                print(f'\nDisplaying {count} Records.')
            elif s_choice == '3':
                keyword = 1
                result = search(table, 3, keyword, s_count)
                count = len(result)
                display(table, result, display_lst)
```

```
            print(f'\nDisplaying {count} Records.')
        elif s_choice == '4':
            keyword = input('Search[YYYY-MM-DD]: ')
            result = search(table, 4, keyword, s_count)
            count = len(result)
            display(table, result, display_lst)
            print(f'\nDisplaying {count} Records.')
        elif s_choice == '5':
            keyword = 0
            result = search(table, 3, keyword, s_count)
            count = len(result)
            display(table, result, display_lst)
            print(f'\nDisplaying {count} Records.')
        if choice == '2':
            m_choice = int(input(f'Choose To Learn More[1-{count} or 0 to Exit]: '))
            if m_choice == 0:
                pass
            else:
                order = result[m_choice-1]
                completed = ''
                if int(order[8]) == 0:
```

```
                        completed = 'No'
                    elif int(order[8]) == 1:
                        completed = 'Yes'
                    print(f"""\nMore Info on OrderID: {order[0]}
Client Name: {order[1]}
Client Address: {order[2]}
Client Phone No.: {order[3]}
Order Charge: {order[4]}
Order Date: {order[5]}
Medicines Order: {order[6]}
No. of Medicines(w.r.t Medicine Order): {order[7]}
Completed: {completed}
Completion Date: {order[9]}""")
            elif choice == '3':
                m_choice = int(input(f'Choose To Edit[1-{count} or 0 to Exit]: '))
                if m_choice == 0:
                    pass
                else:
                    edit(table, result[m_choice-1][0])
            elif choice == '5':
                m_choice = int(input(f'Choose To Delete[1-{count} or 0 to Exit]: '))
```

```
                    if m_choice == 0:
                        pass
                    else:
                        delete(table, result[m_choice-1][0])
            elif choice == '4':
                n = int(input('\nHow many orders to add? '))
                add(table, n)
            elif choice == '6':
                print('\nGoing to Main Menu...\n')
                break
    elif tab == "4":
        print('\nExiting Application...')
        break
    else:
        print("\nPlease Enter Correct Option!\n")


# Closing
db.close()
```

Output

```
kunal@fedoralappy ../MediCheck[📦 v0.1.0]
[🐍 v3.11.7( poetry: medicheck_v3.11 )]
➜ python MediCheck.py                                                    🔋 60%
Initializing SQL Connection...
Connection Successful.

Initializing Database and Tables...
Initialization Completed. Starting Application...

                                        WELCOME TO MEDICHECK!
~~~~~~~~~~~~~~~Menu~~~~~~~~~~~~~~~
1. Medicines
2. Stocks
3. Orders
4. Exit

Select Tab or Exit [1/2/3/4]:
```

```
Select Tab or Exit [1/2/3/4]: h

Please Enter Correct Option!

~~~~~~~~~~~~~~~~~Menu~~~~~~~~~~~~~~~~~
1. Medicines
2. Stocks
3. Orders
4. Exit

Select Tab or Exit [1/2/3/4]:
```

```
Select Tab or Exit [1/2/3/4]: 1

~~~~~~~~~~~Medicines Menu~~~~~~~~~~~
1. Show All Medicines
2. Search For Medicines
3. Update A Medicine Record
4. Add New Medicines
5. Delete A Medicine Record
6. Exit

Select Option or Exit Menu [1/2/3/4/5/6]:
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 1
Showing All Data...
Fetching Full Data...
Displaying Fetched Data...
| MedID || Name || Type || Manufacture_Company || Description || Side_Effects || Price || Available || Latest_Batch_Date |


| 2 || Xofluza || Antiviral || Medplus ||  || Vomiting,Diarrh... || 60.00 || 0 || 1970-01-01 |

| 3 || Zovirax || Antiviral || Medplus || It is an antivi... || Dizziness,Drows... || 150.00 || 1 || 2023-12-22 |

| 4 || Hydralazine || Vasodilator || Medline || It is used with... || Severe Tirednes... || 150.00 || 1 || 2024-01-05 |

| 5 || Aclovate Cream || Skin Cream || Medplus ||  || Stretch Marks,S... || 200.00 || 0 || 1970-01-01 |

| 6 || Altexide || Cardovascular D... || Medline || It s used to tr... || Dizziness,Drows... || 200.00 || 1 || 2024-01-05
|

| 7 || Aledryl Elixir || Antihistamine || Cipla || It s an antihis... || Dizziness,Drows... || 100.00 || 0 || 1970-01-01 |


Displaying 6 Rows.
Done.
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 2
Search On The Basis Of:
1. Name
2. Type
3. Company
4. Available
5. Newly Available
6. Not Available
Choose [1/2/3/4/5/6]: 3
How many results should be gathered(0 for all): 2
Search: Medplus
Fetching Data...
Displaying Fetched Data...
| MedID || Name || Type || Manufacture_Company || Description || Side_Effects || Price || Available || Latest_Batch_Date |


| 2 || Xofluza || Antiviral || Medplus ||  || Vomiting,Diarrh... || 60.00 || 0 || 1970-01-01 |

| 3 || Zovirax || Antiviral || Medplus || It is an antivi... || Dizziness,Drows... || 150.00 || 1 || 2023-12-22 |


Displaying 2 Records.
Choose To Learn More[1-2 or 0 to Exit]:
```

```
Displaying 2 Records.
Choose To Learn More[1-2 or 0 to Exit]: 2

More Info on MedID: 3
Name: Zovirax
Type: Antiviral
Side Effects: Dizziness,Drowsiness,Mood Changes,Unsteady Movement,Bloody/Dark Urine
Manufactured By: Medplus
Description: It is an antiviral drug used to treat infections caused by herpes viruses, such as genital herpes, cold sores
, shingles, and chickenpox.
Price: 150.00
Available: Yes
Latest Stock Date: 2023-12-22
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 3
Search On The Basis Of:
1. Name
2. Type
3. Company
4. Available
5. Newly Available
6. Not Available
Choose [1/2/3/4/5/6]: 6
How many results should be gathered(0 for all): 0
Fetching Data...
Displaying Fetched Data...
| MedID || Name || Type || Manufacture_Company || Description || Side_Effects || Price || Available || Latest_Batch_Date |


| 2 || Xofluza || Antiviral || Medplus ||  || Vomiting,Diarrh... || 60.00 || 0 || 1970-01-01 |

| 5 || Aclovate Cream || Skin Cream || Medplus ||  || Stretch Marks,S... || 200.00 || 0 || 1970-01-01 |

| 7 || Aledryl Elixir || Antihistamine || Cipla || It s an antihis... || Dizziness,Drows... || 100.00 || 0 || 1970-01-01 |


Displaying 3 Records.
Choose To Edit[1-3 or 0 to Exit]: 2
```

```
Choose To Edit[1-3 or 0 to Exit]: 1

Enter Details of Medicine to edit, leave blank to keep existing data:
Medicine Name[Xofluza]:
Medicine Type[Antiviral]:
Manufacturing Company[Medplus]:
Description[]: It is an antiviral medication used for treating the flu(influenza).
Side-Effects[Vomiting,Diarrhea,Nausea]:
Medicine Price[60.00]:
Medicine Available[0]: 1
Newest Batch Date[1970-01-01]: 2024-01-05
Editing Medicine Record...
Done.
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 4

How many medicines to add? 2

Enter Details of Medicine 1:
MedID: 1
Medicine Name: Simethicone
Medicine Type: Anti-gastritis
Manufacturing Company: Cipla
Description: It is used to treat symptoms of gas such as uncomfortable or painful pressure, fullness, and bloating.
Side-Effects(separated by ','): No Side-Effects
Medicine Price: 80
Medicine Available(1 = yes, 0 = no): 0
Newest Batch Date(YYYY-MM-DD):

Adding to Database...

Enter Details of Medicine 2:
MedID: 8
Medicine Name: Phenylephrine
Medicine Type: Nasal Decongestant
Manufacturing Company: Medline
Description:
Side-Effects(separated by ','): Nervousness,Dizziness,Sleepiness
Medicine Price: 25
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 5
Search On The Basis Of:
1. Name
2. Type
3. Company
4. Available
5. Newly Available
6. Not Available
Choose [1/2/3/4/5/6]: 3
How many results should be gathered(0 for all): 0
Search: Cipla
Fetching Data...
Displaying Fetched Data...
| MedID || Name || Type || Manufacture_Company || Description || Side_Effects || Price || Available || Latest_Batch_Date |


| 1 || Simethicone || Anti-gastritis || Cipla || It is used to t... || No Side-Effects || 80.00 || 0 || 1970-01-01 |

| 7 || Aledryl Elixir || Antihistamine || Cipla || It s an antihis... || Dizziness,Drows... || 100.00 || 0 || 1970-01-01 |


Displaying 2 Records.
Choose To Delete[1-2 or 0 to Exit]: 1
```

```
~~~~~~~~~~Medicines Menu~~~~~~~~~~
1. Show All Medicines
2. Search For Medicines
3. Update A Medicine Record
4. Add New Medicines
5. Delete A Medicine Record
6. Exit

Select Option or Exit Menu [1/2/3/4/5/6]: 6

Going to Main Menu...
```

```
~~~~~~~~~~~~~~~~Menu~~~~~~~~~~~~~~~~
1. Medicines
2. Stocks
3. Orders
4. Exit

Select Tab or Exit [1/2/3/4]: 2

~~~~~~~~~~Stocks Menu~~~~~~~~~~
1. Show Medicine Stock History
2. Search For Stocks
3. Update A Medicine Stock Record
4. Add Newly Received Stocks
5. Delete A Stock Record
6. Exit

Select Option or Exit Menu [1/2/3/4/5/6]:
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 1
Showing All Data...
Fetching Full Data...
Displaying Fetched Data...
| StockID || MedID || Box_Count || Arrival_Date || Manufacture_Date || Expiration_Date || Price || Shelf_No || Finished |


| 1 || 3 || 12 || 2023-12-22 || 2023-12-20 || 2025-12-20 || 30000.00 || C365 || 0 |

| 2 || 4 || 10 || 2024-01-05 || 2024-01-04 || 2026-01-04 || 50000.00 || A879 || 0 |

| 3 || 6 || 15 || 2024-01-05 || 2023-12-20 || 2025-12-20 || 4000.00 || B679 || 0 |

| 4 || 2 || 21 || 2024-01-26 || 2024-01-20 || 2025-01-20 || 40000.00 || A699 || 0 |


Displaying 4 Rows.
Done.
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 2
Search On The Basis Of:
1. Medicine
2. Arrival Date
3. Expiration Date
4. Shelf No.
5. Finished
6. Not Finished
Choose [1/2/3/4/5/6]: 1
How many results should be gathered(0 for all): 0
Search: 2
Fetching Data...
Displaying Fetched Data...
| StockID || MedID || Box_Count || Arrival_Date || Manufacture_Date || Expiration_Date || Price || Shelf_No || Finished |


| 4 || 2 || 21 || 2024-01-26 || 2024-01-20 || 2025-01-20 || 40000.00 || A699 || 0 |


Display 1 Records.
Choose To Learn More[1-1 or 0 to Exit]:
```

```
Choose To Learn More[1-1 or 0 to Exit]: 1

More Info on StockID: 4
MedID: 2
No. Of Boxes: 21
Arrival Date: 2024-01-26
Manufacture Date: 2024-01-20
Expiration Date: 2025-01-20
Price: 40000.00
Shelf No.: A699
Finished:
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 4

How many stocks to add? 1

Enter Details of the Stock:
StockID: 5
MedID of Stock Medicine: 8
No. of Boxes: 15
Arrival Date(YYYY-MM-DD): 2024-01-28
Manufacture Date(YYYY-MM-DD): 2024-01-20
Expiration Date(YYYY-MM-DD): 2026-01-20
Stock Price: 10000
Shelf No of Stock(Ex-A124): F112

Adding to Database...
1 Stocks Added to Database.
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 5
Search On The Basis Of:
1. Medicine
2. Arrival Date
3. Expiration Date
4. Shelf No.
5. Finished
6. Not Finished
Choose [1/2/3/4/5/6]: 6
How many results should be gathered(0 for all): 0
Fetching Data...
Displaying Fetched Data...
| StockID || MedID || Box_Count || Arrival_Date || Manufacture_Date || Expiration_Date || Price || Shelf_No || Finished |

| 1 || 3 || 12 || 2023-12-22 || 2023-12-20 || 2025-12-20 || 30000.00 || C365 || 0 |

| 2 || 4 || 10 || 2024-01-05 || 2024-01-04 || 2026-01-04 || 50000.00 || A879 || 0 |

| 3 || 6 || 15 || 2024-01-05 || 2023-12-20 || 2025-12-20 || 4000.00 || B679 || 0 |

| 4 || 2 || 21 || 2024-01-26 || 2024-01-20 || 2025-01-20 || 40000.00 || A699 || 0 |

| 5 || 8 || 15 || 2024-01-28 || 2024-01-20 || 2026-01-20 || 10000.00 || F112 || 0 |
```

```
| 1 || 3 || 12 || 2023-12-22 || 2023-12-20 || 2025-12-20 || 30000.00 || C365 || 0 |

| 2 || 4 || 10 || 2024-01-05 || 2024-01-04 || 2026-01-04 || 50000.00 || A879 || 0 |

| 3 || 6 || 15 || 2024-01-05 || 2023-12-20 || 2025-12-20 || 4000.00 || B679 || 0 |

| 4 || 2 || 21 || 2024-01-26 || 2024-01-20 || 2025-01-20 || 40000.00 || A699 || 0 |

| 5 || 8 || 15 || 2024-01-28 || 2024-01-20 || 2026-01-20 || 10000.00 || F112 || 0 |


Displaying 5 Records.
Choose To Delete[1-5 or 0 to Exit]: 0
```

```
~~~~~~~~~~~~Orders Menu~~~~~~~~~~~~
1. Show Order History
2. Order Search
3. Update Order Details
4. Place New Orders
5. Delete Order Record
6. Exit

Select Option or Exit Menu [1/2/3/4/5/6]: 1
Showing All Data...
Fetching Full Data...
Displaying Fetched Data...
| OrderID || Name || Address || Phone_No || Order_Charge || Order_Date || Medicines || Med_Count || Completed || Completio
n_Date |



Displaying 0 Rows.
Done.
```

```
Select Option or Exit Menu [1/2/3/4/5/6]: 4

How many orders to add? 1

Enter Order Details:
OrderID: 1
Client Name: Kunal Kumar
Client Address: RZ/B-92, New Delhi
Client Phone No.: 88103024
Order Charge: 1150
Order Date(YYYY-MM-DD): 2024-01-28
Medicines(MedID separated by ','): 2,4,8
Medicine Amount(with respect to above entry separated by ','): 4,3,2

Adding to Database...
Order Added to Database.
```

```
~~~~~~~~~~~~Orders Menu~~~~~~~~~~~
1. Show Order History
2. Order Search
3. Update Order Details
4. Place New Orders
5. Delete Order Record
6. Exit

Select Option or Exit Menu [1/2/3/4/5/6]: 1
Showing All Data...
Fetching Full Data...
Displaying Fetched Data...
| OrderID || Name || Address || Phone_No || Order_Charge || Order_Date || Medicines || Med_Count || Completed || Completio
n_Date |


| 1 || Kunal Kumar || RZ/B-92, New De... || 88103024 || 1150.00 || 2024-01-28 || 2,4,8 || 4,3,2 || 0 || 1970-01-01 |


Displaying 1 Rows.
Done.
```

```
~~~~~~~~~~~~~Orders Menu~~~~~~~~~~~~
1. Show Order History
2. Order Search
3. Update Order Details
4. Place New Orders
5. Delete Order Record
6. Exit

Select Option or Exit Menu [1/2/3/4/5/6]: 6

Going to Main Menu...

~~~~~~~~~~~~~~~~Menu~~~~~~~~~~~~~~~~
1. Medicines
2. Stocks
3. Orders
4. Exit

Select Tab or Exit [1/2/3/4]: 4

Exiting Application...
```