Coding Challenge

Hospital Mangement

K. S. Pooja

1. Create SQL Schema from the following classes class, use the class attributes for table column name

```
create DATABASE Hospital_Management;

use Hospital_Management;

CREATE TABLE Patient(
   patientId varchar(5) PRIMARY KEY,
   firstName VARCHAR(50) NOT NULL,
   lastName VARCHAR(50) NOT NULL,
   dateOfBirth DATE NOT NULL,
   gender VARCHAR(10) NOT NULL,
   contactNumber VARCHAR(15) NOT NULL,
   address VARCHAR(255) NOT NULL);


CREATE TABLE Doctor (
   doctorId varchar(5) PRIMARY KEY,
   firstName VARCHAR(50) NOT NULL,
   lastName VARCHAR(50) NOT NULL,
   specialization VARCHAR(50) NOT NULL,
   contactNumber VARCHAR(15) NOT NULL);


create table Appointment(
    appointmentId int primary key,
    patientId varchar(5) not null,
    doctorId varchar(5) not null,
     appointmentDate DATETIME not null,
      description varchar(50),
     FOREIGN KEY(patientId) REFerences Patient(patientId),
     FOREIGN KEY (doctorId) References Doctor(DoctorId)
);
```

```sql
INSERT INTO Patient(patientId, firstName, lastName, dateOfBirth, gender,
contactNumber, address)
VALUES
('P1', 'Aarav', 'Kapoor', '1972-09-27', 'Male', '911166676', 'Kurnool'),
('P2', 'Mira', 'Patel', '2002-01-12', 'Female', '9756443310', 'Chennai'),
('P3', 'Ishaan', 'Verma', '1985-04-15', 'Male', '7654387663', 'Allahabad'),
('P4', 'Riya', 'Singh', '2015-05-29', 'Female', '7765423435', 'Hyderabad'),
('P5', 'Karan', 'Malhotra', '1984-11-22', 'Male', '7865431211', 'Mumbai'),
('P6', 'Anaya', 'Deshmukh', '1999-07-04', 'Female', '5556667797', 'Kadapa'),
('P7', 'Rohan', 'Bhatia', '2019-02-19', 'Male', '3378654390', 'Chittor'),
('P8', 'Tara', 'Mehta', '2020-07-22', 'Female', '7654445567', 'Chennai'),
('P9', 'Vikram', 'Nair', '2001-03-17', 'Male', '2765465434', 'Kurnool'),
('P10', 'Sara', 'Shah', '2005-10-01', 'Female', '7265223339', 'Kadapa');


INSERT INTO Doctor(doctorId, firstName, lastName, specialization, contactNumber)
VALUES
('D01', 'Arjun', 'Mehta', 'Cardiologist', '9123456780'),
('D02', 'Neha', 'Kohli', 'Neurologist', '4329087654'),
('D03', 'Raghav', 'Sharma', 'Surgeon', '9876654654'),
('D04', 'Simran', 'Bose', 'Pediatrician', '8769806547'),
('D05', 'Dev', 'Chatterjee', 'Dermatologist', '9876543210'),
('D06', 'Ishita', 'Desai', 'Oncologist', '8907654762'),
('D07', 'Kabir', 'Nayak', 'Dermatologist', '5764789432'),
('D08', 'Anika', 'Kapoor', 'Rheumatologist', '9876867869'),
('D09', 'Rohan', 'Sen', 'Gastroenterologist', '8769806598'),
('D10', 'Aisha', 'Malik', 'Endocrinologist', '7869087651');
Insert into
appointment(appointmentId,patientId,doctorId,appointmentDate,description)
Values
(1,'P10','D07','2025-01-21 10:30','Headache'),
(10,'P7','D08','2024-11-13 11:00','Hair Loss'),
(2,'P8','D09','2024-12-02 10:30','Migrane'),
(3,'P1','D10','2024-12-17 9:30','diabetes'),
(4,'P3','D03','2025-02-11 12:00','Fever'),
(5,'P4','D07','2024-11-29 11:30','Stomach Pain'),
(6,'P5','D02','2024-11-03 10:30','Allergy'),
(7,'P6','D01','2024-12-01 11:00','Hyper Tension'),
(8,'P9','D07','2024-12-30 12:00','Hair loss'),
(9,'P2','D04','2024-11-12 12:00','Allergy');
```

```
select * from Patient;
select * from Doctor;
select * from Appointment;
```

1. Create the following model/entity classes within package entity with variables declared private, constructors(default and parametrized,getters,setters and toString())

1. Define **Patient** class with the following confidential attributes:

   a. patientId
   b. firstName
   c. lastName
   d. dateOfBirth
   e. gender
   f. contactNumber
   g. address

entity/patient.py

```
class Patient:
    def __init__(self, patientId,firstName,lastName,dateOfBirth,gender,contactNumber,address):
        self.patientId = patientId
        self.firstName = firstName
        self.lastName = lastName
        self.dateOfBirth = dateOfBirth
        self.gender = gender
        self.contactNumber = contactNumber
        self.address = address

    #setter methods


    def set_patientId(self,patientId):
        self.patientId = patientId
```

```
def set_firstName(self,firstName):
    self.firstName = firstName
def set_lastName(self,lastName):
    self.lastName = lastName
def set_dateOfBirth(self,dateOfBirth):
    self.dateOfBirth = dateOfBirth
def set_gender(self,gender):
    self.gender = gender
def set_contactNumber(self,contactNumber):
    self.contactNumber = contactNumber
def set_address(self,address):
    self.address = address

#getter methods

def get_patientId(self):
    return self.patientId
def get_firstName(self):
    return self.firstName
def get_lastName(self):
    return self.lastName
def get_dateOfBirth(self):
    return self.dateOfBirth
def get_gender(self):
    return self.gender
def get_contactNumber(self):
    return self.contactNumber
def get_address(self):
    return self.address

def __str__(self):
    return f"Patient ID: {self.patientId()}, Name: {self.firstName} {self.lastName}, " \
        f"DOB: {self.dateOfBirth}, Gender: {self.gender}, Contact: {self.contactNumber}, " \
        f"Address: {self.address}"
```

2. Define **Doctor** class with the following confidential attributes:

a. doctorId
b. firstName
c. lastName
d. specialization
e. contactNumber

entity/doctor.py

```
class Doctor:
    def __init__(self,doctorId,firstName,lastName,specialization,contactNumber):
        self.doctorId = doctorId
        self.firstName = firstName
        self.lastName = lastName
        self.specialization = specialization
        self.contactNumber = contactNumber

    #Setter methods

    def set_doctorId(self,doctorId):
        self.doctorId = doctorId
    def set_firstName(self,firstName):
        self.firstName = firstName
    def set_lastName(self,lastName):
        self.lastName = lastName
    def set_specialization(self,specialization):
        self.specialization = specialization
    def set_contactNumber(self,contactNumber):
        self.contactNumber = contactNumber

    #Getter methods

    def get_doctorId(self):
        return self.doctorId
    def get_firstName(self):
```

```
        return self.firstName
    def get_lastName(self):
        return self.lastName
    def get_specialization(self):
        return self.specialization
    def get_contactNumber(self):
        return self.contactNumber

    def __str__(self):
        return f"Doctor ID: {self.doctorId}, Name: {self.firstName} {self.lastName}, " \
               f"Specialization: {self.specialization}, Contact: {self.contactNumber}"
```

3. Appointment Class:

a. appointmentId
b. patientId
c. doctorId
d. appointmentDate
e. description

entity/appointment.py

```
class Appointment:
    def __init__(self,appointmentId,patientId,doctorId,appointmentDate,description):
        self.patientId = patientId
        self.doctorId = doctorId
        self.appointmentId = appointmentId
        self.appointmentDate = appointmentDate
        self.description = description

#Setter methods

    def set_appointmentId(self,appointmentId):
```

```
        self.appointmentId = appointmentId
    def set_patientId(self,patientId):
        self.patientId = patientId
    def set_doctorId(self,doctorId):
        self.doctorId = doctorId
    def set_appointmentDate(self,appointmentDate):
        self.appointmentDate = appointmentDate
    def set_description(self,description):
        self.description = description

    #Getter methods

    def get_appointmentId(self):
        return self.appointmentId
    def get_patientId(self):
        return self.patientId
    def get_doctorId(self):
        return self.doctorId
    def get_appointmentDate(self):
        return self.appointmentDate
    def get_description(self):
        return self.description

    def __str__(self):
        return f"Appointment ID: {self.appointmentId}, Patient ID: {self.patientId}, Doctor ID:
{self.doctorId}, " \
               f"Date: {self.appointmentDate}, Description: {self.description}"
```

Define IHospitalService interface/abstract class with following methods to interact with database Keep the interfaces and implementation classes in package dao

a. getAppointmentById()

i. Parameters: appointmentId

ii. ReturnType: Appointment object

b. getAppointmentsForPatient()

i. Parameters: patientId

ii. ReturnType: List of Appointment objects

c. getAppointmentsForDoctor()

i. Parameters: doctorId

ii. ReturnType: List of Appointment objects

d. scheduleAppointment()

i. Parameters: Appointment Object

ii. ReturnType: Boolean

e. updateAppointment()

i. Parameters: Appointment Object

ii. ReturnType: Boolean

f. cancelAppointment()

i. Parameters: AppointmentId

ii. ReturnType: Boolean

dao/IHospitalService.py

```
from abc import ABC, abstractmethod
from entity.appointment import Appointment
from typing import List

class IHospitalService(ABC):
    @abstractmethod
    def getAppointmentById(self, appointmentId) -> Appointment:
        pass

    @abstractmethod
    def getAppointmentsForPatient(self, patientId) -> List[Appointment]:
        pass

    @abstractmethod
    def getAppointmentsForDoctor(self, doctorId) -> List[Appointment]:
        pass
```

```
    @abstractmethod
    def scheduleAppointment(self, appointment: Appointment) -> bool:
        pass
    @abstractmethod
    def updateAppointment(self, appointment: Appointment) -> bool:
        pass

    @abstractmethod
    def cancelAppointment(self, appointmentId) -> bool:
        pass
```

Define HospitalServiceImpl class and implement all the methods IHospitalServiceImpl

dao/HospitalServiceImpl.py

```
from dao.IHospitalService import IHospitalService
from entity.appointment import Appointment
from exception.PatientNumberNotFound import PatientNumberNotFoundException
from util.DBConnection import DBConnection
from  tabulate import tabulate

class HospitalServiceImpl(IHospitalService):

    def getAppointmentById(self, appointmentId):
        conn = DBConnection.get_connection()
        cursor=conn.cursor()
        try:
            query = "SELECT * FROM Appointment WHERE appointmentId = ?"
            cursor.execute(query, (appointmentId,))
            appointment = cursor.fetchone()

            if appointment:
                appointment_details=[
```

```
            ['Appointment ID',appointment[0]],
            ["Patient ID",appointment[1]],
            ["Doctor ID",appointment[2]],
            ["Appointment Date",appointment[3]],
            ["Description",appointment[4]],
        ]
        print("-----------Appointment  Details----------")
        print(tabulate(appointment_details,tablefmt="grid"))

    else:
        print("------------Appointment  Not  Found---------- ")
  except Exception as e:
    print(f"Error in fetching appointment: {e}")
    return None
  finally:
    cursor.close()


def getAppointmentsForPatient(self, patientId):
  conn = DBConnection.get_connection()
  cursor=conn.cursor()
  try:
    query = "SELECT * FROM Appointment WHERE patientId = ?"
    cursor.execute(query,(patientId,))
    appointments = []
    for row in cursor.fetchall():
      appointments.append(Appointment(
        appointmentId=row[0],
        patientId=row[1],
        doctorId=row[2],
        appointmentDate=row[3],
        description=row[4]
      ))
```

```
            return appointments
        except PatientNumberNotFoundException as e:
            print(e)
            return []
        finally:
            cursor.close()
    def getAppointmentsForDoctor(self, doctorId):
        conn = DBConnection.get_connection()
        cursor=conn.cursor()
        try:
            query = "SELECT * FROM Appointment WHERE doctorId = ?"
            cursor.execute(query, (doctorId,))
            appointments = []
            for row in cursor.fetchall():
                appointments.append(Appointment(
                    appointmentId=row[0],
                    patientId=row[1],
                    doctorId=row[2],
                    appointmentDate=row[3],
                    description=row[4]
                ))
            return appointments
        except Exception as e:
            print(f"Error in fetching appointments for doctor: {e}")
            return []
    def doctor_exists(self, doctorId):
        conn = DBConnection.get_connection()
        cursor=conn.cursor()
        try:
            query = "SELECT count(*) FROM Doctor WHERE doctorId = ?"
            cursor.execute(query, (doctorId,))
            count = cursor.fetchone()[0]
            return count > 0 #If doctor exists
        except Exception as e:
```

```
            print(f"Error in doctor exists: {e}")
            return False
        finally:
            cursor.close()


    def patient_exists(self, patientId):
        conn = DBConnection.get_connection()
        cursor=conn.cursor()
        try:
            query = "SELECT count(*) FROM Patient WHERE patientId = ?"
            cursor.execute(query, (patientId,))
            count = cursor.fetchone()[0]
            return count > 0
        except Exception as e:
            print(f"Error in patient exists: {e}")
            return False
        finally:
            cursor.close()


    def get_next_appointmentId(self):
        conn = DBConnection.get_connection()
        cursor=conn.cursor()
        try:
            cursor.execute("SELECT Max(appointmentId) FROM Appointment")
            max_id = cursor.fetchone()[0]
            return (max_id+1) if max_id is not None else 1
        except Exception as e:
            print(f"Error in get_next_appointment: {e}")
            return 1
        finally:
            cursor.close()

    def scheduleAppointment(self, appointment):
```

```
        conn = DBConnection.get_connection()
        cursor=conn.cursor()
        try:
            cursor.execute("SELECT COUNT(*) FROM Patient where
patientId=?",(appointment.get_patientId(),))
            patient_exists = cursor.fetchone()[0]
            if not patient_exists:
                print(f"Patient ID {appointment.get_patientId()} does not exist")
                return False

            cursor.execute("Select Count(*) From Doctor where
doctorId=?",(appointment.get_doctorId(),))
            doctor_exists = cursor.fetchone()[0]
            if not doctor_exists:
                print(f"Doctor ID {appointment.get_doctorId()} does not exist")
                return False

            cursor.execute("INSERT INTO Appointment (appointmentId, patientid, doctorId,
appointmentDate, description) VALUES (?,?,?,?,?)",
            (appointment.get_appointmentId(),appointment.get_patientId(),
appointment.get_doctorId(), appointment.get_appointmentDate(), appointment.get_description()))
            conn.commit()
            print("Appointment Scheduled")
            return True
        except Exception as e:
            print(f"Error in scheduleAppointment: {e}")
    def updateAppointment(self, appointment):
        conn = DBConnection.get_connection()
        cursor=conn.cursor()
        try:
            cursor.execute("Select count(*) from Appointment where
appointmentId=?",(appointment.appointmentId,))
            count = cursor.fetchone()[0]
            if not count:
```

```
            print("---------Appointment Not Found---------- ")
            return False

        cursor.execute("UPDATE Appointment SET patientId=?,doctorId=?, appointmentDate = ?,
description = ? WHERE appointmentId =
?",(appointment.get_patientId(),appointment.get_doctorId(),appointment.get_appointmentDate(),a
ppointment.get_description(),appointment.appointmentId))
        conn.commit()
        return True
    except Exception as e:
        print(f"Error in updateAppointment: {e}")
        return False
    finally:
        cursor.close()

  def cancelAppointment(self, appointmentId):
    conn = DBConnection.get_connection()
    cursor=conn.cursor()
    try:
        cursor.execute("Select count(*) from Appointment where
appointmentId=?",(appointmentId,))
        count = cursor.fetchone()[0]
        if count==0:
            print("---------Appointment Not Found---------- ")
            return False

        cursor.execute("DELETE FROM Appointment WHERE appointmentId = ?",(appointmentId,))
        conn.commit()
        return True
    except Exception as e:
        print(f"Error in cancelAppointment: {e}")
        return False
    finally:
        cursor.close()
```

Create a utility class DBConnection in a package util with a static variable connection of Type Connection and a static method getConnection() which returns connection. Connection properties supplied in the connection string should be read from a property file

util/DBConnection.py

```
import sys
import os
sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(__file__))))
import pyodbc
from util.PropertyUtil import PropertyUtil  # Adjust import based on your package structure
class DBConnection:
    connection = None
@staticmethod
    def get_connection():
        if DBConnection.connection is None:
            try:
                conn_str = PropertyUtil.get_property_string()
                DBConnection.connection = pyodbc.connect(conn_str)
                print("Connected Successfully")
            except Exception as e:
                print(f"Connection failed: {e}")
        return DBConnection.connection
    @staticmethod
```

```
def test_connection():
    # This method will only test the connection without executing any queries
    connection = DBConnection.get_connection()
    if connection:
        print("Database connection is successful.")
    else:
        print("Failed to connect to the database.")

if __name__ == "__main__":
    DBConnection.test_connection()
```

Create a utility class PropertyUtil which contains a static method named getPropertyString() which reads a property fie containing connection details like hostname, dbname, username, password, port number and returns a connection string

util/PropertyUtil.py

```
class PropertyUtil:
    @staticmethod
    def get_property_string():
        hostname = "DESKTOP\SQLEXPRESS"  # Your SQL Server instance name
        dbname = "Hospital_Management"          # Your database name

        connection_string = (
            f"Driver={{ODBC Driver 17 for SQL Server}};"
            f"Server={hostname};"
            f"Database={dbname};"
            "Trusted_Connection=yes;"
        )
        return connection_string
```

Create the exceptions in package myexceptions Define the following custom exceptions and throw them in methods whenever needed. Handle all the exceptions in main method,

1. PatientNumberNotFoundException :throw this exception when user enters an invalid patient number which doesn't

exception/PatientNumberNotFound.py

```
class PatientNumberNotFoundException(Exception):
    def __init__(self,patientId):
        super().__init__(f"Patient with Id {patientId} not found.")
```

Create class named MainModule with main method in package mainmodule. Trigger all the methods in service implementation class.

main/main.py

```
from dao.HospitalServiceImpl import HospitalServiceImpl
from entity.appointment import Appointment
```

```
from tabulate import tabulate


class MainModule:
    def __init__(self):
        self.hospital_service = HospitalServiceImpl()

    def menu(self):
        global appointment
        print("*" * 40)
        print("Welcome to Hospital Management System")
        print("*" * 40)
        while True:
            menu = [
                ["1.", "Get Appointment Details by ID"],
                ["2.", "Get Appointments for Patient"],
                ["3.", "Get Appointments for Doctor"],
                ["4.", "Schedule Appointment"],
                ["5.", "Update Appointment"],
                ["6.", "Cancel Appointment"],
                ["7.", "Exit"]
            ]
            # Print the menu using tabulate
            print(tabulate(menu, headers=["Option", "Description"], tablefmt="grid"))

            choice = input("Enter your choice: ")

            if choice == '1':
                appointment_id=int(input("Enter Appointment ID: "))
                try:
                    appointment=self.hospital_service.getAppointmentById(appointment_id)
                    if appointment is None:
                        print("")
                except ValueError :
                    print("Invalid input. Please enter a valid number.")
```

```
            except Exception as e:
                print(f"Error: {e}")
        elif choice == '2':
            patient_id=input("Enter Patient ID to fetch appointment: ")
            appointments=self.hospital_service.getAppointmentsForPatient(patient_id)
            if appointments:
                print(f"Appointments for Patient: {patient_id}")
                table_data = [[appointment.appointmentId, appointment.doctorId,
appointment.appointmentDate,
                               appointment.description]
                              for appointment in appointments]
                headers = ["Appointment Id", "Doctor Id","Appointment Date", "Appointment
Description"]
                print(tabulate(table_data, headers=headers,tablefmt="grid"))
            else :
                print("Patient not found")

        elif choice == '3':
            doctor_id=input("Enter Doctor ID to fetch appointments: ")
            if not self.hospital_service.doctor_exists(doctor_id):
                print(f"*****The doctor Id {doctor_id} doesnot exists******")
                continue
            appointments=self.hospital_service.getAppointmentsForDoctor(doctor_id)
            if appointments:
                print(f"Appointments for Doctor: {doctor_id}")
                table_data = [[appointment.appointmentId, appointment.patientId,
appointment.appointmentDate,
                               appointment.description]
                              for appointment in appointments]
                headers=["Appointment Id", "Patient Id", "Appointment Date", "Appointment
Description"]
                print(tabulate(table_data, headers=headers,tablefmt="grid"))
            else:
                print(f"--------No appointments for Doctor {doctor_id}-------- ")
```

```
elif choice == '4':
    patient_id=input("Enter Patient Id: ")
    doctor_id=input("Enter Doctor ID: ")
    appointment_date=input("Enter Appointment Date(YYYY-MM-DD HH:MM): ")
    description=input("Enter Appointment Description: ")
    appointment_id=self.hospital_service.get_next_appointmentId()
    if appointment_id is None:
        print("Failed to get next appointment.")
    else:
        appointment = Appointment(
            appointmentId=appointment_id,  # Use the generated ID
            patientId=patient_id,
            doctorId=doctor_id,
            appointmentDate=appointment_date,
            description=description
        )
    if self.hospital_service.scheduleAppointment(appointment):
        print("Appointment scheduled successfully.")
    else:
        print("Unable to schedule appointment.")

elif choice == '5':
    appointment_id=int(input("Enter Appointment ID to update: "))
    new_patient_id=input("Enter New Patient ID: ")
    if not self.hospital_service.patient_exists(new_patient_id):
        print("The specified patient Id does not exist.")
        continue
    new_doctor_id=input("Enter New Doctor ID: ")
    new_appointment_date=input("Enter New Appointment Date(YYYY-MM-DD HH:MM): ")
    new_description=input("Enter New Appointment Description: ")
    appointment=Appointment(
        appointmentId=appointment_id,
        patientId=new_patient_id,
        doctorId=new_doctor_id,
```

```
                appointmentDate=new_appointment_date,
                description=new_description
            )
            if self.hospital_service.updateAppointment(appointment):
                print("-------------Appointment updated successfully------------")
            else:
                print("Unable to update appointment.")
        elif choice == '6':
            appointment_id=int(input("Enter Appointment ID to cancel: "))
            if self.hospital_service.cancelAppointment(appointment_id):
                print("Appointment cancelled successfully.")
            else:
                print("Appointment id does not exist")
        elif choice == '7':
            print("Existing the system")
            break
        else:
            print("Invalid choice. Please try again.")

if __name___== "__main__":
    main_module = MainModule()
    main_module.menu()
```

Outputs:

```
select * from Patient;
```

100 %

Results | Messages

	patientId	firstName	lastName	dateOfBirth	gender	contactNumber	address
1	P1	Aarav	Kapoor	1972-09-27	Male	911166676	Kurnool
2	P10	Sara	Shah	2005-10-01	Female	7265223339	Kadapa
3	P2	Mira	Patel	2002-01-12	Female	9756443310	Chennai
4	P3	Ishaan	Verma	1985-04-15	Male	7654387663	Allahabad
5	P4	Riya	Singh	2015-05-29	Female	7765423435	Hyderabad
6	P5	Karan	Malhotra	1984-11-22	Male	7865431211	Mumbai
7	P6	Anaya	Deshmukh	1999-07-04	Female	5556667797	Kadapa
8	P7	Rohan	Bhatia	2019-02-19	Male	3378654390	Chittor
9	P8	Tara	Mehta	2020-07-22	Female	7654445567	Chennai
10	P9	Vikram	Nair	2001-03-17	Male	2765465434	Kurnool

```
select * from Doctor;
```

Results | Messages

	doctorId	firstName	lastName	specialization	contactNumber
1	D01	Arjun	Mehta	Cardiologist	9123456780
2	D02	Neha	Kohli	Neurologist	4329087654
3	D03	Raghav	Sharma	Surgeon	9876654654
4	D04	Simran	Bose	Pediatrician	8769806547
5	D05	Dev	Chatterjee	Dermatologist	9876543210
6	D06	Ishita	Desai	Oncologist	8907654762
7	D07	Kabir	Nayak	Dermatologist	5764789432
8	D08	Anika	Kapoor	Rheumatologist	9876867869
9	D09	Rohan	Sen	Gastroenterologist	8769806598
10	D10	Aisha	Malik	Endocrinologist	7869087651

```
select * from Appointment;
```

100 %

Results | Messages

	appointmentId	patientId	doctorId	appointmentDate	description
1	1	P10	D07	2025-01-21 10:30:00.000	Headache
2	2	P8	D09	2024-12-02 10:30:00.000	Migrane
3	3	P1	D10	2024-12-17 09:30:00.000	diabetes
4	4	P3	D03	2025-02-11 12:00:00.000	Fever
5	5	P4	D07	2024-11-29 11:30:00.000	Stomach Pain
6	6	P5	D02	2024-11-03 10:30:00.000	Allergy
7	7	P6	D01	2024-12-01 11:00:00.000	Hyper Tension
8	8	P9	D07	2024-12-30 12:00:00.000	Hair loss
9	9	P2	D04	2024-11-12 12:00:00.000	Allergy
10	10	P7	D08	2024-11-13 11:00:00.000	Hair Loss

```
****************************************
Welcome to Hospital Management System
****************************************
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 1
Enter Appointment ID: 4
Connected Successfully
-----------Appointment Details----------
+------------------+---------------------+
| Appointment ID   | 4                   |
+------------------+---------------------+
| Patient ID       | P3                  |
+------------------+---------------------+
| Doctor ID        | D03                 |
+------------------+---------------------+
| Appointment Date | 2025-02-11 12:00:00 |
+------------------+---------------------+
| Description      | Fever               |
+------------------+---------------------+
```

```
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 1
Enter Appointment ID: 15
------------Appointment Not Found----------
```

```
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 2
Enter Patient ID to fetch appointment: P5
Appointments for Patient: P5
+------------------+-------------+---------------------+--------------------------+
|   Appointment Id | Doctor Id   | Appointment Date    | Appointment Description  |
+==================+=============+=====================+==========================+
|                6 | D02         | 2024-11-03 10:30:00 | Allergy                  |
+------------------+-------------+---------------------+--------------------------+
```

```
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 2
Enter Patient ID to fetch appointment: P17
Patient not found
```

```
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 3
Enter Doctor ID to fetch appointments: D02
Appointments for Doctor: D02
+----------------+-------------+--------------------+--------------------------+
|   Appointment Id | Patient Id   | Appointment Date    | Appointment Description   |
+================+=============+====================+==========================+
|              6 | P5          | 2024-11-03 10:30:00 | Allergy                  |
+----------------+-------------+--------------------+--------------------------+
```

```
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 3
Enter Doctor ID to fetch appointments: D33
Connected Successfully
*****The doctor Id D33 doesnot exists******
```

```
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 4
Enter Patient Id: P3
Enter Doctor ID: D07
Enter Appointment Date(YYYY-MM-DD HH:MM): 2025-01-01
Enter Appointment Description: Hair loss
Appointment Scheduled
Appointment scheduled successfully.
```

Results | Messages

	appointmentId	patientId	doctorId	appointmentDate	description
1	1	P10	D07	2025-01-21 10:30:00.000	Headache
2	2	P8	D09	2024-12-02 10:30:00.000	Migrane
3	3	P1	D10	2024-12-17 09:30:00.000	diabetes
4	4	P3	D03	2025-02-11 12:00:00.000	Fever
5	5	P4	D07	2024-11-29 11:30:00.000	Stomach Pain
6	6	P5	D02	2024-11-03 10:30:00.000	Allergy
7	7	P6	D01	2024-12-01 11:00:00.000	Hyper Tension
8	8	P9	D07	2024-12-30 12:00:00.000	Hair loss
9	9	P2	D04	2024-11-12 12:00:00.000	Allergy
10	10	P7	D08	2024-11-13 11:00:00.000	Hair Loss
11	11	P3	D07	2025-01-01 00:00:00.000	Hair loss

```
+----------+--------------------------------+
|   Option | Description                    |
+==========+================================+
|        1 | Get Appointment Details by ID  |
+----------+--------------------------------+
|        2 | Get Appointments for Patient   |
+----------+--------------------------------+
|        3 | Get Appointments for Doctor    |
+----------+--------------------------------+
|        4 | Schedule Appointment           |
+----------+--------------------------------+
|        5 | Update Appointment             |
+----------+--------------------------------+
|        6 | Cancel Appointment             |
+----------+--------------------------------+
|        7 | Exit                           |
+----------+--------------------------------+
Enter your choice: 4
Enter Patient Id: P13
Enter Doctor ID: D08
Enter Appointment Date(YYYY-MM-DD HH:MM): 2014-12-12
Enter Appointment Description: Teeth ache
Patient ID P13 does not exist
Unable to schedule appointment.
```

```
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 5
Enter Appointment ID to update: 2
Enter New Patient ID: P4
Enter New Doctor ID: D06
Enter New Appointment Date(YYYY-MM-DD HH:MM): 2025-01-05
Enter New Appointment Description: Fever
------------Appointment updated successfully------------
```

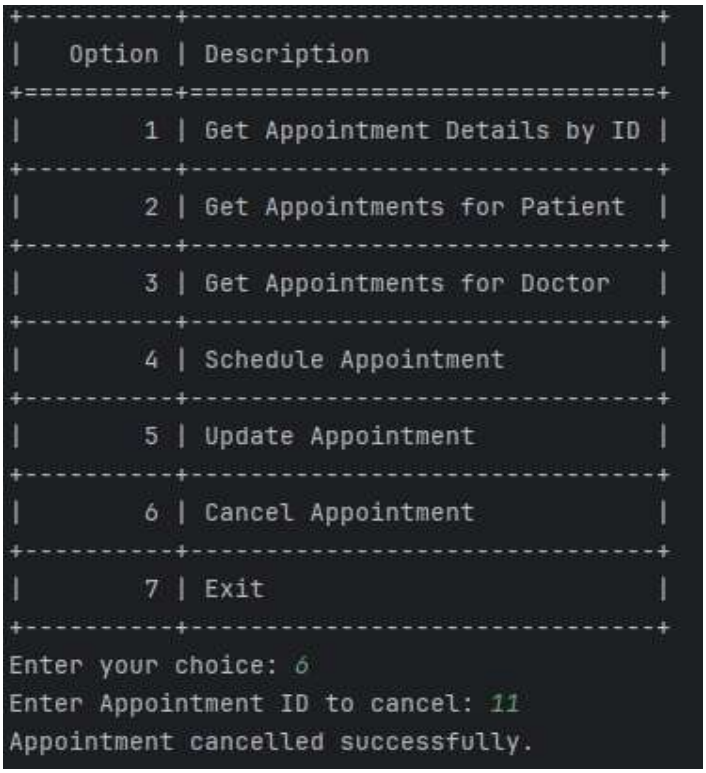
```
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 6
Enter Appointment ID to cancel: 6
Appointment cancelled successfully.
```

100 %

Results | Messages

	appointmentId	patientId	doctorId	appointmentDate	description
1	1	P10	D07	2025-01-21 10:30:00.000	Headache
2	2	P4	D06	2025-01-05 00:00:00.000	Fever
3	3	P1	D10	2024-12-17 09:30:00.000	diabetes
4	4	P3	D03	2025-02-11 12:00:00.000	Fever
5	5	P4	D07	2024-11-29 11:30:00.000	Stomach Pain
6	7	P6	D01	2024-12-01 11:00:00.000	Hyper Tension
7	8	P9	D07	2024-12-30 12:00:00.000	Hair loss
8	9	P2	D04	2024-11-12 12:00:00.000	Allergy
9	10	P7	D08	2024-11-13 11:00:00.000	Hair Loss
10	11	P3	D07	2025-01-01 00:00:00.000	Hair loss

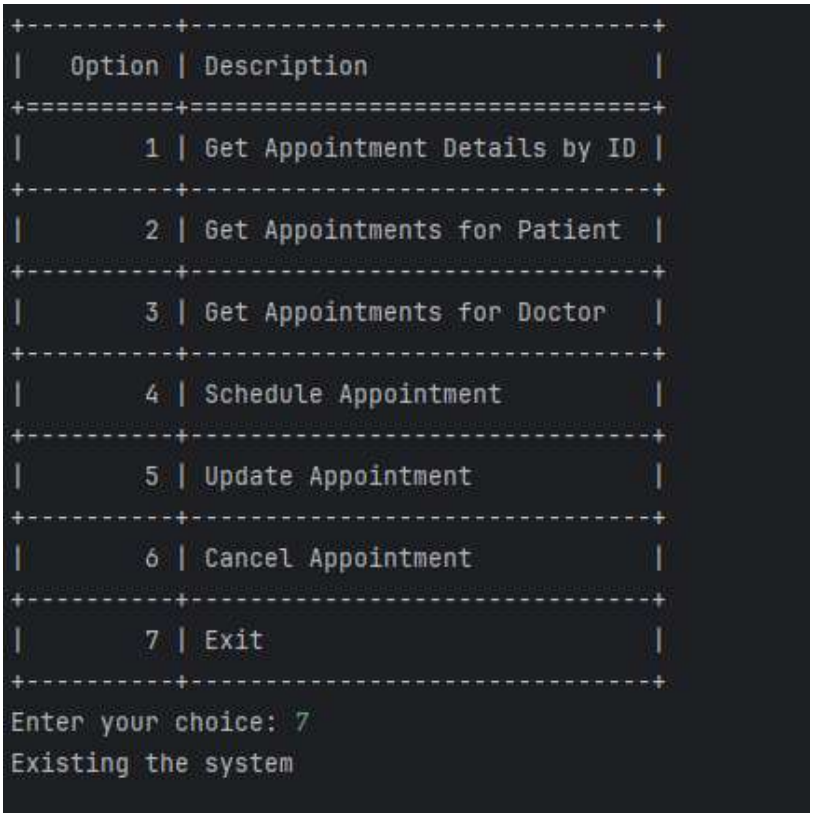
```
+----------+------------------------------+
|   Option | Description                  |
+==========+==============================+
|        1 | Get Appointment Details by ID |
+----------+------------------------------+
|        2 | Get Appointments for Patient  |
+----------+------------------------------+
|        3 | Get Appointments for Doctor   |
+----------+------------------------------+
|        4 | Schedule Appointment          |
+----------+------------------------------+
|        5 | Update Appointment            |
+----------+------------------------------+
|        6 | Cancel Appointment            |
+----------+------------------------------+
|        7 | Exit                          |
+----------+------------------------------+
Enter your choice: 6
Enter Appointment ID to cancel: 20
---------Appointment Not Found----------
Appointment id does not exist
```



```
+----------+--------------------------------+
|   Option | Description                    |
+==========+================================+
|        1 | Get Appointment Details by ID  |
+----------+--------------------------------+
|        2 | Get Appointments for Patient   |
+----------+--------------------------------+
|        3 | Get Appointments for Doctor    |
+----------+--------------------------------+
|        4 | Schedule Appointment           |
+----------+--------------------------------+
|        5 | Update Appointment             |
+----------+--------------------------------+
|        6 | Cancel Appointment             |
+----------+--------------------------------+
|        7 | Exit                           |
+----------+--------------------------------+
Enter your choice: 6
Enter Appointment ID to cancel: 11
Appointment cancelled successfully.
```

Results Messages

	appointmentId	patientId	doctorId	appointmentDate	description
1	1	P10	D07	2024-10-28 10:30:00.000	Hair loss
2	2	P8	D09	2024-11-02 10:30:00.000	Stomach Ache
3	3	P1	D10	2024-10-17 09:30:00.000	diabetes
4	5	P10	D08	2024-11-02 12:30:00.000	Arthritis
5	6	P5	D02	2024-11-03 10:30:00.000	Migrane
6	7	P6	D01	2024-11-01 11:00:00.000	Hyper Tension
7	8	P9	D07	2024-10-30 12:00:00.000	Hair loss
8	9	P2	D04	2024-10-12 12:00:00.000	Allergy
9	10	P7	D08	2024-10-13 11:00:00.000	Arthritis


+----------+------------------------------+
| Option | Description |
+==========+==============================+
| 1 | Get Appointment Details by ID |
+----------+------------------------------+
| 2 | Get Appointments for Patient |
+----------+------------------------------+
| 3 | Get Appointments for Doctor |
+----------+------------------------------+
| 4 | Schedule Appointment |
+----------+------------------------------+
| 5 | Update Appointment |
+----------+------------------------------+
| 6 | Cancel Appointment |
+----------+------------------------------+
| 7 | Exit |
+----------+------------------------------+
Enter your choice: 7
Existing the system