MINI-PROJECT REPORT

DATABASE

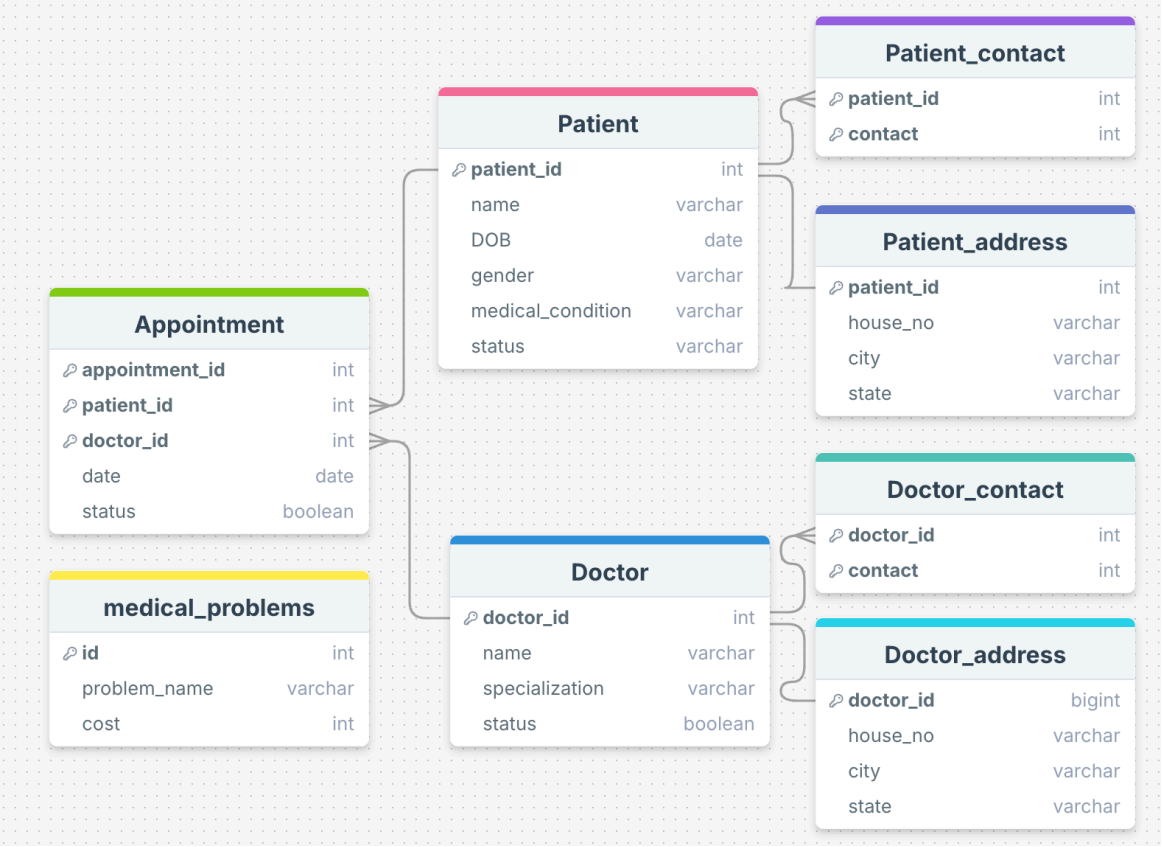
TITLE : HOSPITAL MANAGEMENT SYSTEM

MOHIT R - 3122235002072
Pravin kumar S - 3122235002310

Project Aim

To develop a **simplified hospital management system** that streamlines key administrative and clinical processes, including patient registration, appointment scheduling, medical record management, billing, doctors management, updating any changes through mail and soon. This system aims to improve efficiency, data accessibility, and accuracy for hospital staff and facilitate seamless interactions for patients.

Relation Schema:



List of modules :

> Patient Management
> Doctor Management
> Appointment Scheduling
> Medical Records Management
> Billing and Invoice

> Sending mail (for any updates to patients)
> Database operation (DML, DQL)

DDL Code (tablecreation2.sql) :

The **DDL** code is responsible for creating the tables and defining the structure of your database. This code will set up the necessary tables for the **Patient**, **Doctor**, and **Appointment** entities, establishing relationships between them.

```
— Creating the Patient table
CREATE TABLE PATIENT (
    PATIENT_ID VARCHAR2(30) PRIMARY KEY,
    NAME VARCHAR2(100) NOT NULL,
    DOB DATE NOT NULL,
    GENDER VARCHAR2(10) NOT NULL,
    MEDICAL_CONDITION VARCHAR2(255)
);

ALTER TABLE PATIENT
    ADD STATUS VARCHAR2(
        20
    ) CHECK (
        STATUS IN ('Y', 'N')
    );

-- Y = Active, N = appointment completed

-- Creating the Patient_contact table with ON DELETE CASCADE
CREATE TABLE PATIENT_CONTACT (
    PATIENT_ID VARCHAR2(30),
    CONTACT VARCHAR2(50),
    PRIMARY KEY (PATIENT_ID, CONTACT),
    FOREIGN KEY (PATIENT_ID) REFERENCES
PATIENT(PATIENT_ID) ON DELETE CASCADE
);

-- Creating the Patient_address table with ON DELETE CASCADE
CREATE TABLE PATIENT_ADDRESS (
    PATIENT_ID VARCHAR2(30) PRIMARY KEY,
    HOUSE_NO VARCHAR2(50),
```

```
    CITY VARCHAR2(50),
    STATE VARCHAR2(50),
    FOREIGN KEY (PATIENT_ID) REFERENCES
PATIENT(PATIENT_ID) ON DELETE CASCADE
);

-- Creating the Doctor table
CREATE TABLE DOCTOR (
    DOCTOR_ID VARCHAR2(30) PRIMARY KEY,
    NAME VARCHAR2(100) NOT NULL,
    SPECIALIZATION VARCHAR2(100) NOT NULL,
    STATUS VARCHAR2(20) NOT NULL
);

-- Adding a check constraint on the STATUS column in the Doctor
table
ALTER TABLE DOCTOR
    ADD CONSTRAINT CHK_STATUS CHECK (
        STATUS IN ('A', 'NA', 'W')
    );

-- Creating the Doctor_contact table with ON DELETE CASCADE
CREATE TABLE DOCTOR_CONTACT (
    DOCTOR_ID VARCHAR2(30),
    CONTACT VARCHAR2(50),
    PRIMARY KEY (DOCTOR_ID, CONTACT),
    FOREIGN KEY (DOCTOR_ID) REFERENCES
DOCTOR(DOCTOR_ID) ON DELETE CASCADE
);

-- Creating the Doctor_address table with ON DELETE CASCADE
CREATE TABLE DOCTOR_ADDRESS (
    DOCTOR_ID VARCHAR2(30) PRIMARY KEY,
    HOUSE_NO VARCHAR2(50),
    CITY VARCHAR2(50),
    STATE VARCHAR2(50),
    FOREIGN KEY (DOCTOR_ID) REFERENCES
DOCTOR(DOCTOR_ID) ON DELETE CASCADE
);

-- Creating the Appointment table with ON DELETE CASCADE on
patient and doctor IDs
CREATE TABLE APPOINTMENT (
```

```
    APPOINTMENT_ID VARCHAR2(30) PRIMARY KEY,
    PATIENT_ID VARCHAR2(30) NOT NULL,
    DOCTOR_ID VARCHAR2(30) NOT NULL,
    APPOINTMENT_DATE DATE NOT NULL,
    STATUS CHAR(1) CHECK (STATUS IN ('Y', 'N')), -- Y = Active, N =
Inactive
    FOREIGN KEY (PATIENT_ID) REFERENCES
PATIENT(PATIENT_ID) ON DELETE CASCADE,
    FOREIGN KEY (DOCTOR_ID) REFERENCES
DOCTOR(DOCTOR_ID) ON DELETE CASCADE
);

--============================================
-- auto increment all table's primary key
--============================================

-- Creating a sequence for Patient table
CREATE SEQUENCE PATIENT_SEQ START WITH 1 INCREMENT
BY 1;

-- Creating a trigger for Patient table
CREATE OR REPLACE TRIGGER TRG_PATIENT_ID BEFORE
    INSERT ON PATIENT FOR EACH ROW
BEGIN
    SELECT
        'P'
        || PATIENT_SEQ.NEXTVAL INTO :NEW.PATIENT_ID
    FROM
        DUAL;
END;
/

-- Creating a sequence for Doctor table
CREATE SEQUENCE DOCTOR_SEQ START WITH 1 INCREMENT
BY 1;

-- Creating a trigger for Doctor table
CREATE OR REPLACE TRIGGER TRG_DOCTOR_ID BEFORE
    INSERT ON DOCTOR FOR EACH ROW
BEGIN
    SELECT
        'D'
        || DOCTOR_SEQ.NEXTVAL INTO :NEW.DOCTOR_ID
```

```
    FROM
        DUAL;
END;
/

--DROP SEQUENCE DOCTOR_SEQ;
--DROP TRIGGER TRG_DOCTOR_ID;

-- Creating a sequence for Appointment table
CREATE SEQUENCE APPOINTMENT_SEQ START WITH 1
INCREMENT BY 1;

-- Creating a trigger for Appointment table
CREATE OR REPLACE TRIGGER TRG_APPOINTMENT_ID
BEFORE
    INSERT ON APPOINTMENT FOR EACH ROW
BEGIN
    SELECT
        'A'
        || APPOINTMENT_SEQ.NEXTVAL
INTO :NEW.APPOINTMENT_ID
    FROM
        DUAL;
END;
/

--=========================================
-- Functions
--=========================================

CREATE OR REPLACE FUNCTION GET_APPOINTMENT_DAYS(
    P_APPOINTMENT_ID VARCHAR2
) RETURN NUMBER IS
    DAYS_DIFFERENCE NUMBER;
BEGIN

    -- Fetch the difference in days
    SELECT
        TRUNC(SYSDATE - APPOINTMENT_DATE) INTO
DAYS_DIFFERENCE
    FROM
        APPOINTMENT
    WHERE
```

```
        APPOINTMENT_ID = P_APPOINTMENT_ID;
    RETURN DAYS_DIFFERENCE + 1;
EXCEPTION
    WHEN NO_DATA_FOUND THEN

        -- Handle the case where no appointment is found
        RETURN NULL; -- Or some other meaningful default value
    WHEN TOO_MANY_ROWS THEN

        -- Handle the case where multiple appointments are found
(unexpected)
        RAISE_APPLICATION_ERROR(-20001, 'Multiple appointments
found with the same ID');
END GET_APPOINTMENT_DAYS;
/
```

DML AND DQL COMMANDS CODE :

```
# database_manager.py

import oracledb

class DatabaseManager:
    _instance = None  # Singleton instance placeholder

    @staticmethod
    def get_instance(user=None, password=None, dsn=None):
        '''Retrieve or create the singleton instance of DatabaseManager'''
        if DatabaseManager._instance is None:
            DatabaseManager._instance = DatabaseManager(user,
password, dsn)
        return DatabaseManager._instance

    def __init__(self, user, password, dsn):
        if DatabaseManager._instance is not None:
            raise Exception("This class is a singleton! Use `get_instance()`
to get an instance.")
        self.user = user
        self.password = password
        self.dsn = dsn
        self.connection = None

    def __enter__(self):
```

```
        '''Context manager enter method to ensure connection is
established'''
        self.connect()
        return self

    def __exit__(self, exc_type, exc_value, traceback):
        '''Context manager exit method to ensure connection is closed'''
        self.close()

    def connect(self):
        '''Establish a database connection if not already connected'''
        if not self.connection:
            self.connection = oracledb.connect(
                user=self.user,
                password=self.password,
                dsn=self.dsn
            )
            print("Database connection established.")

    def execute_query(self, query, commit=False):
        '''Executes a query and returns results for SELECT queries'''
        self.connect()
        cursor = self.connection.cursor()
        try:
            cursor.execute(query)
            if commit:
                self.connection.commit()
            else:
                return cursor.fetchall()
        except oracledb.Error as e:
            print(f"Error executing query: {e}")
            if commit:
                self.connection.rollback()
        finally:
            cursor.close()

    def close(self):
        '''Close the database connection if open'''
        if self.connection:
            self.connection.close()
            self.connection = None
            print("Database connection closed.")
```

```
# Command Pattern Implementation
class DatabaseCommand:
    '''Abstract base class for database commands'''
    def execute(self, db_manager):
        raise NotImplementedError("Each command must implement an
execute method")

class SelectCommand(DatabaseCommand):
    def __init__(self, query):
        self.query = query

    def execute(self, db_manager):
        return db_manager.execute_query(self.query)

class InsertCommand(DatabaseCommand):
    def __init__(self, query):
        self.query = query

    def execute(self, db_manager):
        db_manager.execute_query(self.query, commit=True)

class UpdateCommand(DatabaseCommand):
    def __init__(self, query):
        self.query = query

    def execute(self, db_manager):
        db_manager.execute_query(self.query, commit=True)

class DeleteCommand(DatabaseCommand):
    def __init__(self, query):
        self.query = query

    def execute(self, db_manager):
        db_manager.execute_query(self.query, commit=True)
```

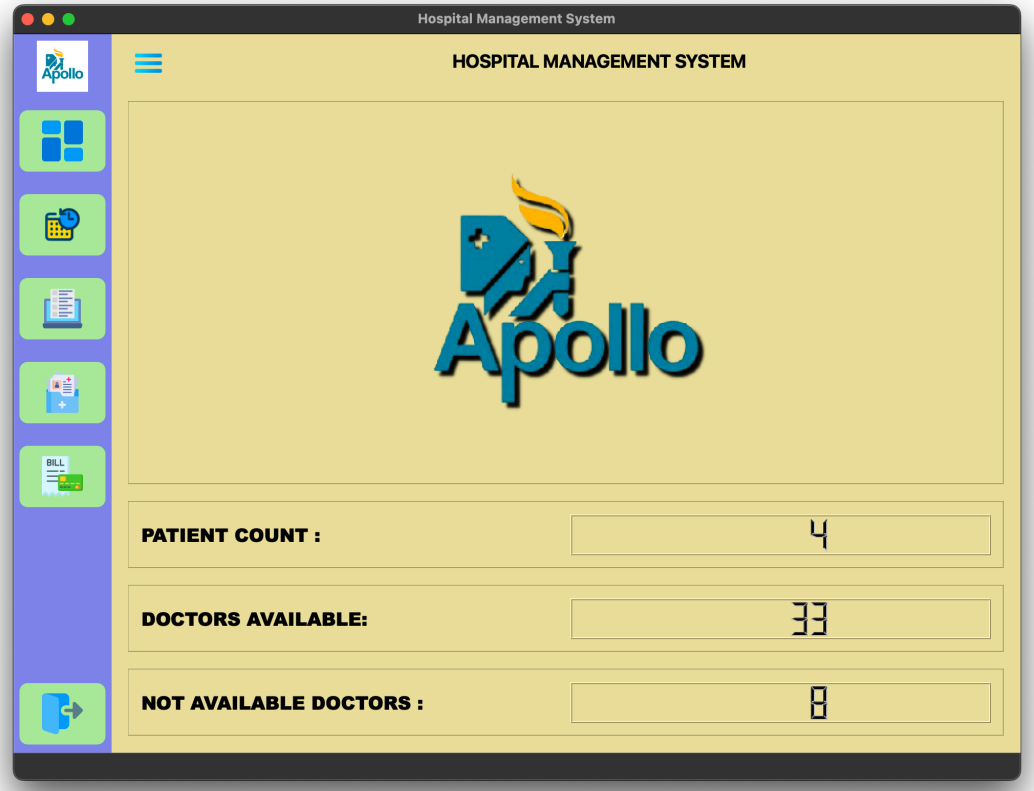
Example DML, DQL Usage :

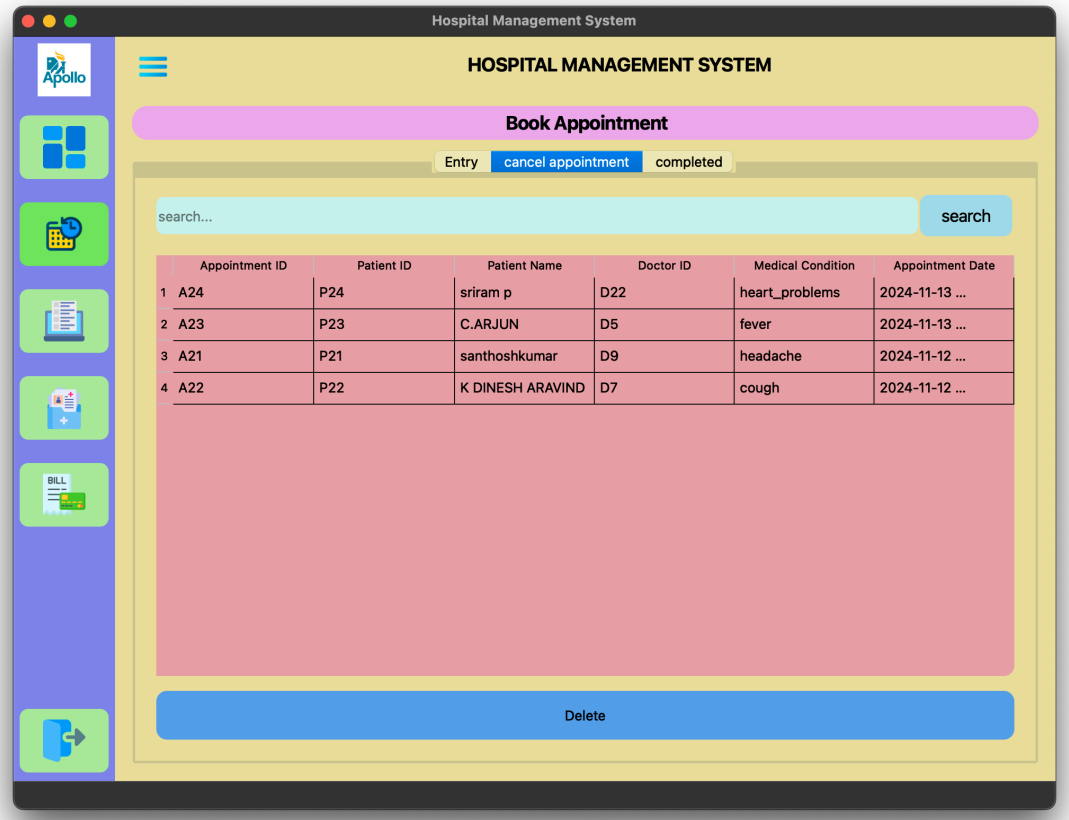
```
active_user_command = SelectCommand(f"select count(*) from
patient where status = 'Y'")
active_users = active_user_command.execute(self.db_manager)[0][0]

insert_into_patient_contact = InsertCommand(f"insert into
patient_contact values (‘{id}','{email}')")
```

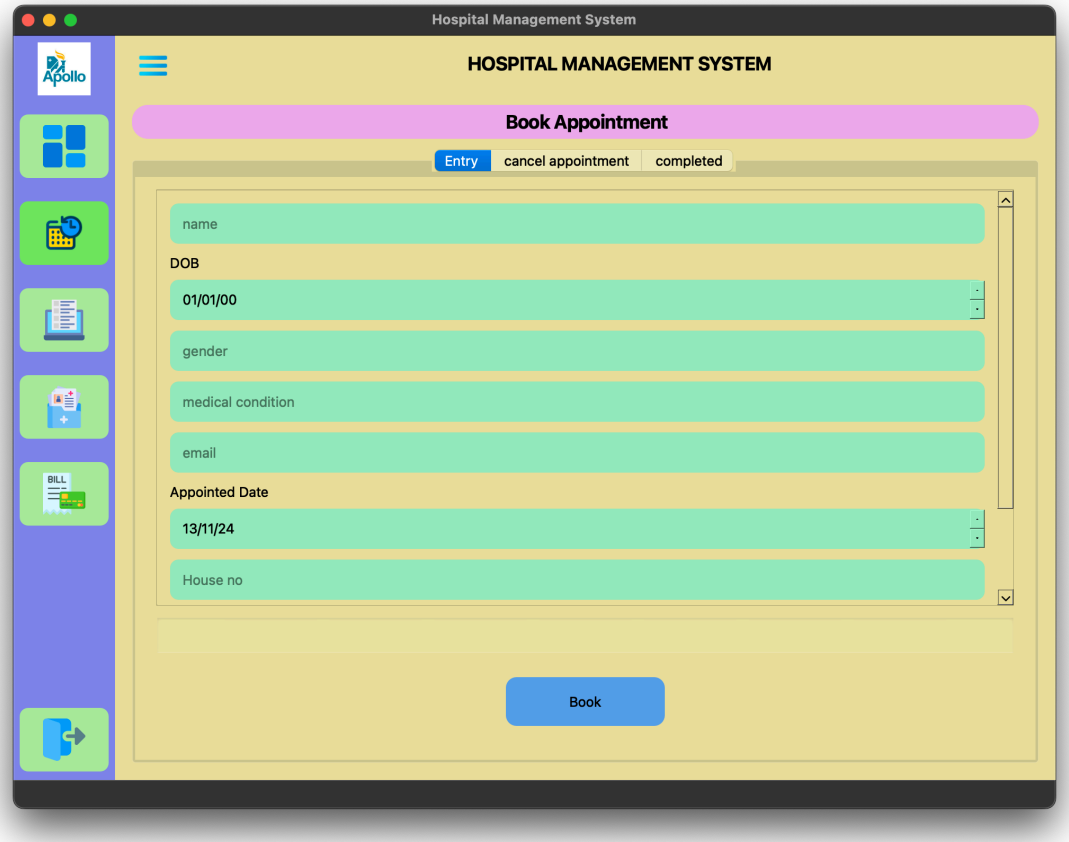
```
insert_into_patient_contact.execute(db_manager)


if condition:
      docselect = f"""
       SELECT d.doctor_id, d.Name, d.specialization, dc.contact
       FROM doctor d
       INNER JOIN doctor_contact dc ON d.doctor_id = dc.doctor_id
       WHERE d.status = '{condition}'
       """
else:
      docselect = f"""
       SELECT d.doctor_id, d.Name, d.specialization, dc.contact
       FROM doctor d
       INNER JOIN doctor_contact dc ON d.doctor_id = dc.doctor_id
       """
selectcommand = SelectCommand(docselect)
doctor_records = selectcommand.execute(self.db_manager)

query = f'''
          SELECT d.doctor_id, d.name, d.specialization, dc.contact
          FROM doctor d
          INNER JOIN doctor_contact dc ON d.doctor_id = dc.doctor_id
          WHERE (LOWER(d.name) LIKE '%' || '{search_text}' || '%'
             OR LOWER(d.doctor_id) LIKE '%' || '{search_text}' || '%')
      '''

select_command = SelectCommand(query)
doctor_records = select_command.execute(self.db_manager)

if condition is None:
   patientselect = '''
       SELECT
             p.PATIENT_ID,
             p.NAME AS PATIENT_NAME,
             p.MEDICAL_CONDITION,
             d.NAME AS ALLOTTED_DOCTOR
          FROM
             PATIENT p
          JOIN
            APPOINTMENT a ON p.PATIENT_ID = a.PATIENT_ID
          JOIN
             DOCTOR d ON a.DOCTOR_ID = d.DOCTOR_ID
```

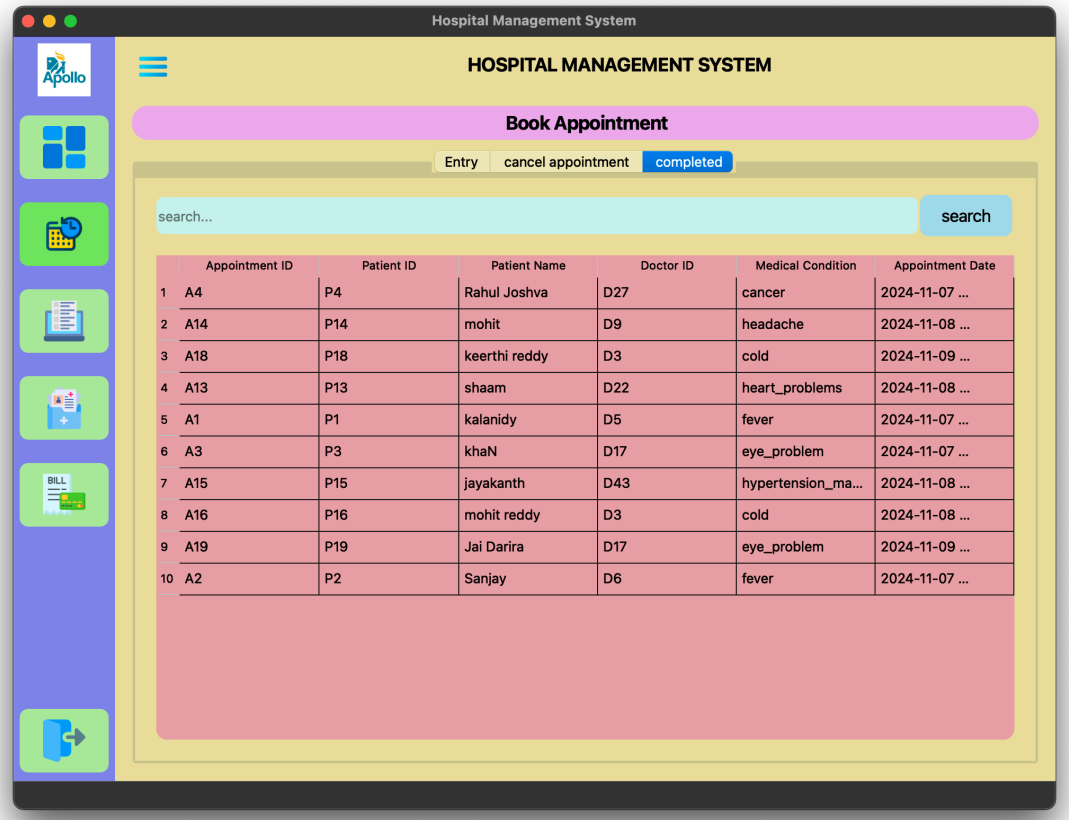
```
            WHERE
                a.STATUS = 'Y'
            '''

else:
    patientselect = f'''
        SELECT
                p.PATIENT_ID,
                p.NAME AS PATIENT_NAME,
                p.MEDICAL_CONDITION,
                d.NAME AS ALLOTTED_DOCTOR
            FROM
                PATIENT p
            JOIN
                APPOINTMENT a ON p.PATIENT_ID = a.PATIENT_ID
            JOIN
                DOCTOR d ON a.DOCTOR_ID = d.DOCTOR_ID
            WHERE
                a.STATUS = '{condition}' and
                p.STATUS = '{condition}'
            '''

selectcommand = SelectCommand(patientselect)
patient_records = selectcommand.execute(self.db_manager)
```

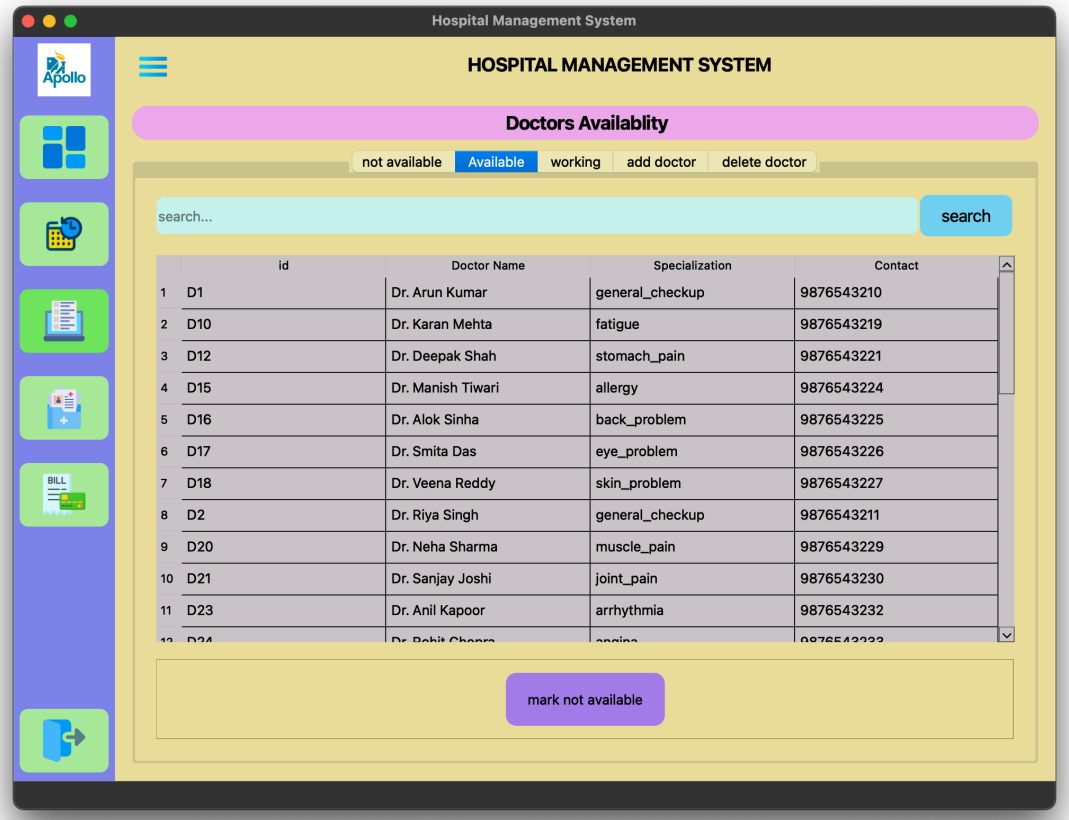
UI OUTPUT SCREENSHOTS :

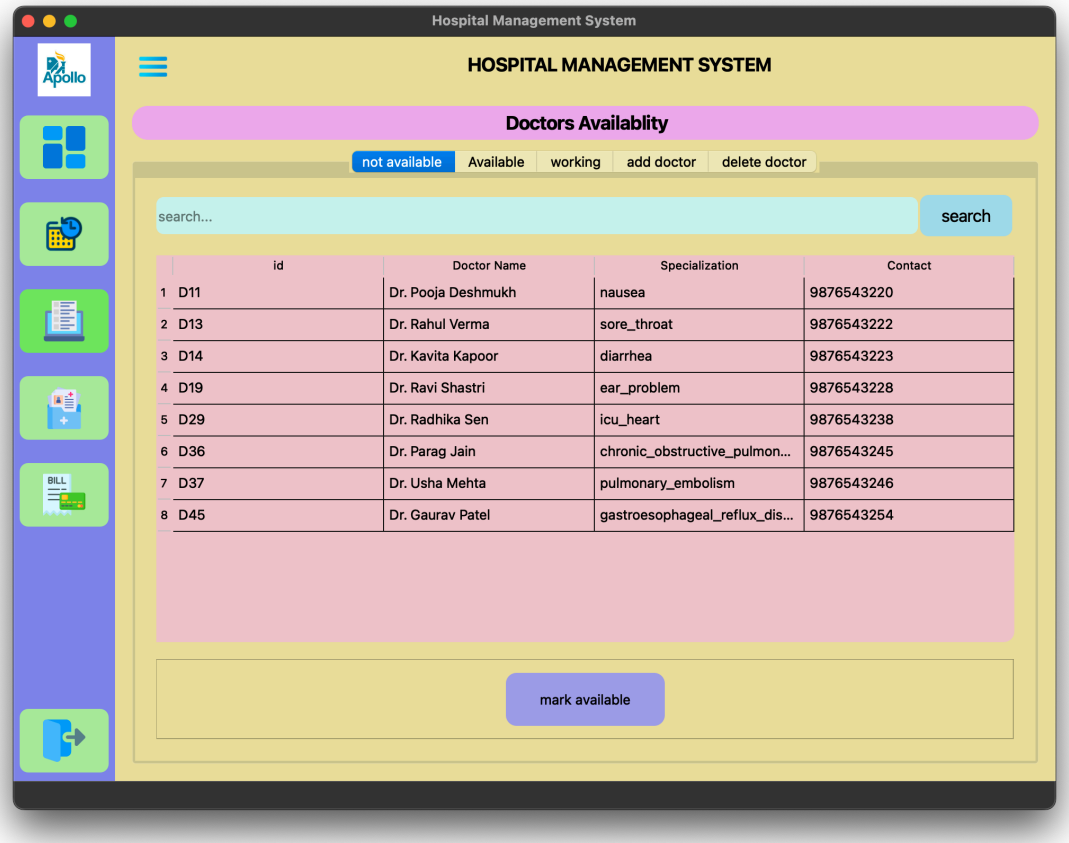


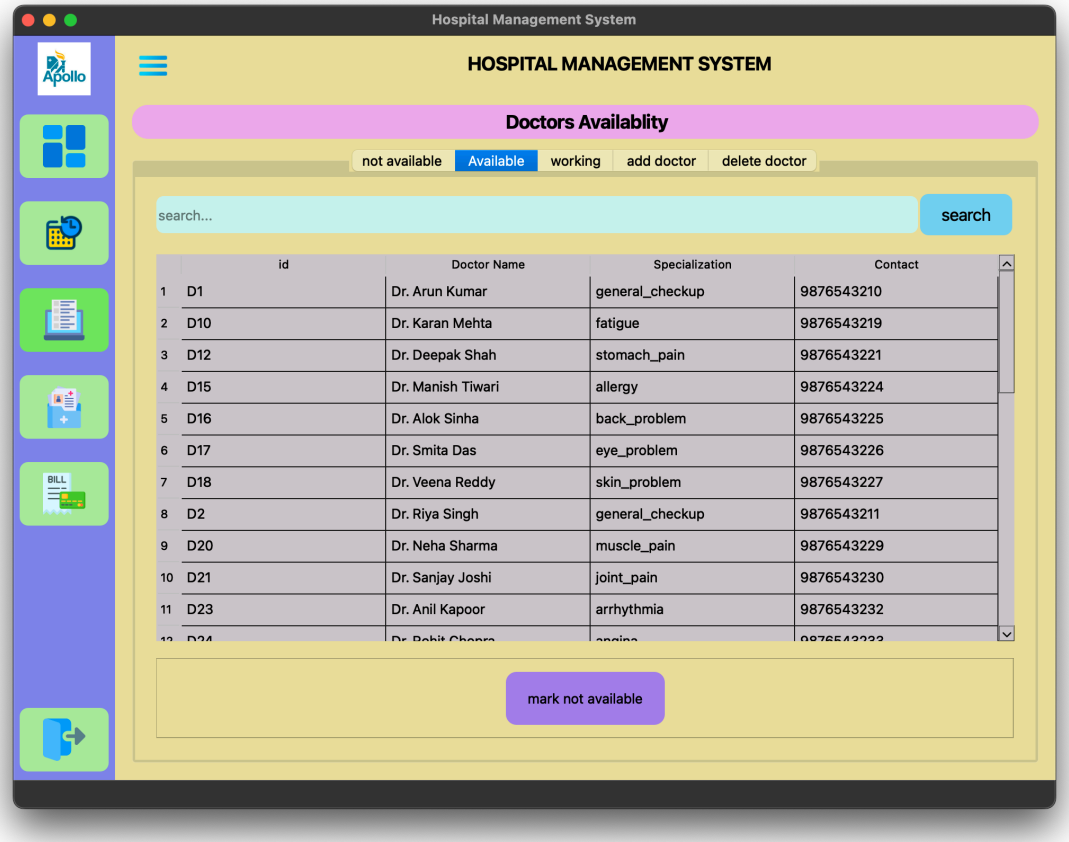

Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Book Appointment
Entry
cancel appointment
completed
search...
search
Appointment ID
Patient ID
Patient Name
Doctor ID
Medical Condition
Appointment Date
1 A24 P24 sriram p D22 heart_problems 2024-11-13 ...
2 A23 P23 C.ARJUN D5 fever 2024-11-13 ...
3 A21 P21 santhoshkumar D9 headache 2024-11-12 ...
4 A22 P22 K DINESH ARAVIND D7 cough 2024-11-12 ...
Delete

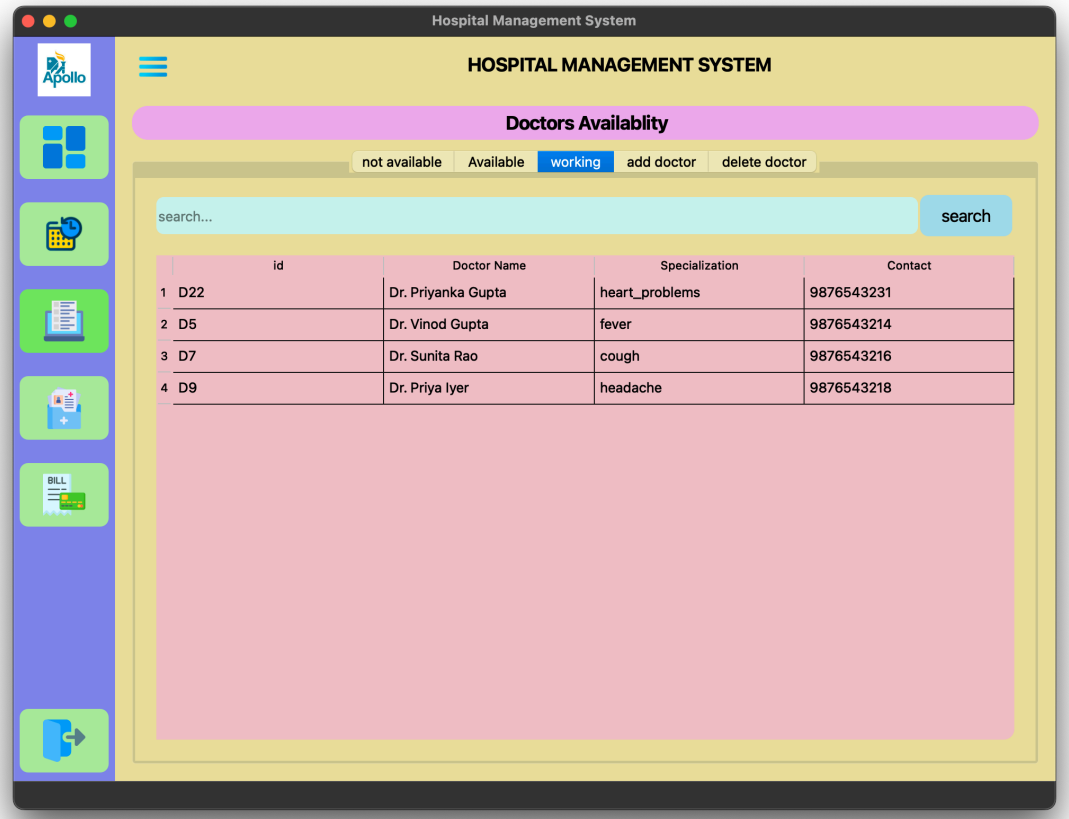

Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Book Appointment
Entry
cancel appointment
completed
name
DOB
01/01/00
gender
medical condition
email
Appointed Date
13/11/24
House no
Book

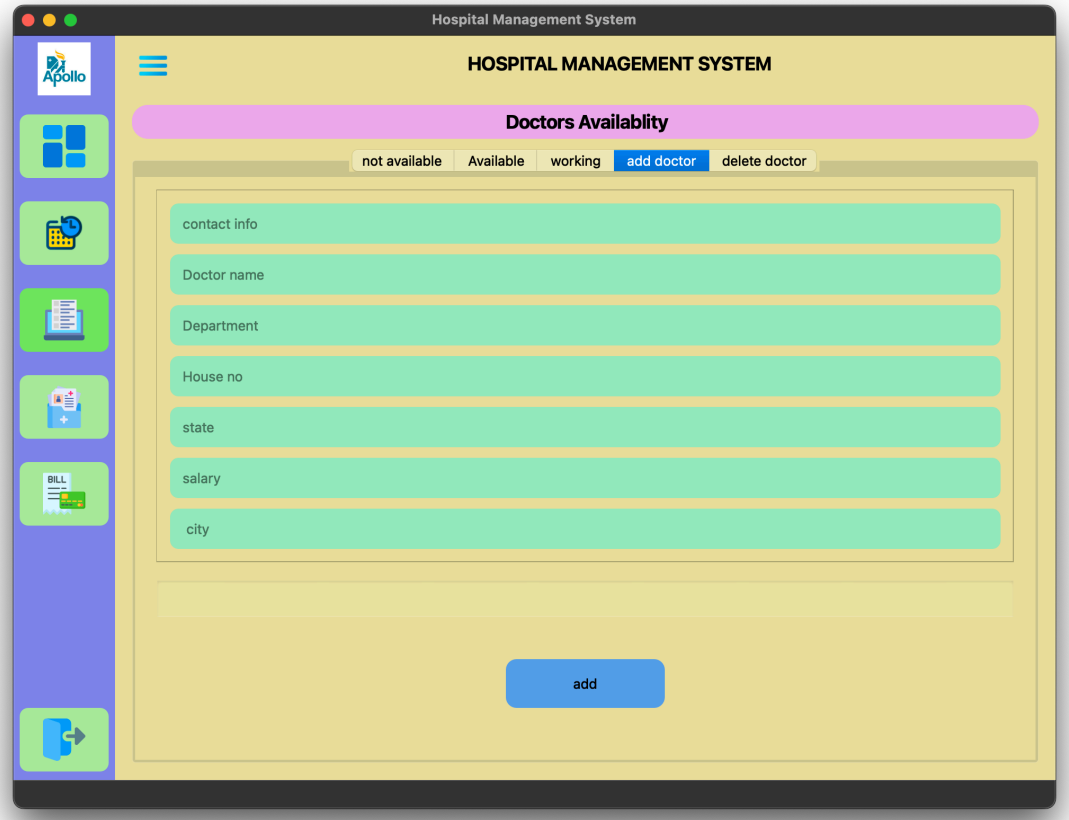

Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Book Appointment
Entry
cancel appointment
completed
search...
search
Appointment ID
Patient ID
Patient Name
Doctor ID
Medical Condition
Appointment Date
1 A4 P4 Rahul Joshva D27 cancer 2024-11-07 ...
2 A14 P14 mohit D9 headache 2024-11-08 ...
3 A18 P18 keerthi reddy D3 cold 2024-11-09 ...
4 A13 P13 shaam D22 heart_problems 2024-11-08 ...
5 A1 P1 kalanidy D5 fever 2024-11-07 ...
6 A3 P3 khaN D17 eye_problem 2024-11-07 ...
7 A15 P15 jayakanth D43 hypertension_ma... 2024-11-08 ...
8 A16 P16 mohit reddy D3 cold 2024-11-08 ...
9 A19 P19 Jai Darira D17 eye_problem 2024-11-09 ...
10 A2 P2 Sanjay D6 fever 2024-11-07 ...


Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Doctors Availablity
not available
Available
working
add doctor
delete doctor
search...
search
id
Doctor Name
Specialization
Contact
1 D1 Dr. Arun Kumar general_checkup 9876543210
2 D10 Dr. Karan Mehta fatigue 9876543219
3 D12 Dr. Deepak Shah stomach_pain 9876543221
4 D15 Dr. Manish Tiwari allergy 9876543224
5 D16 Dr. Alok Sinha back_problem 9876543225
6 D17 Dr. Smita Das eye_problem 9876543226
7 D18 Dr. Veena Reddy skin_problem 9876543227
8 D2 Dr. Riya Singh general_checkup 9876543211
9 D20 Dr. Neha Sharma muscle_pain 9876543229
10 D21 Dr. Sanjay Joshi joint_pain 9876543230
11 D23 Dr. Anil Kapoor arrhythmia 9876543232
mark not available

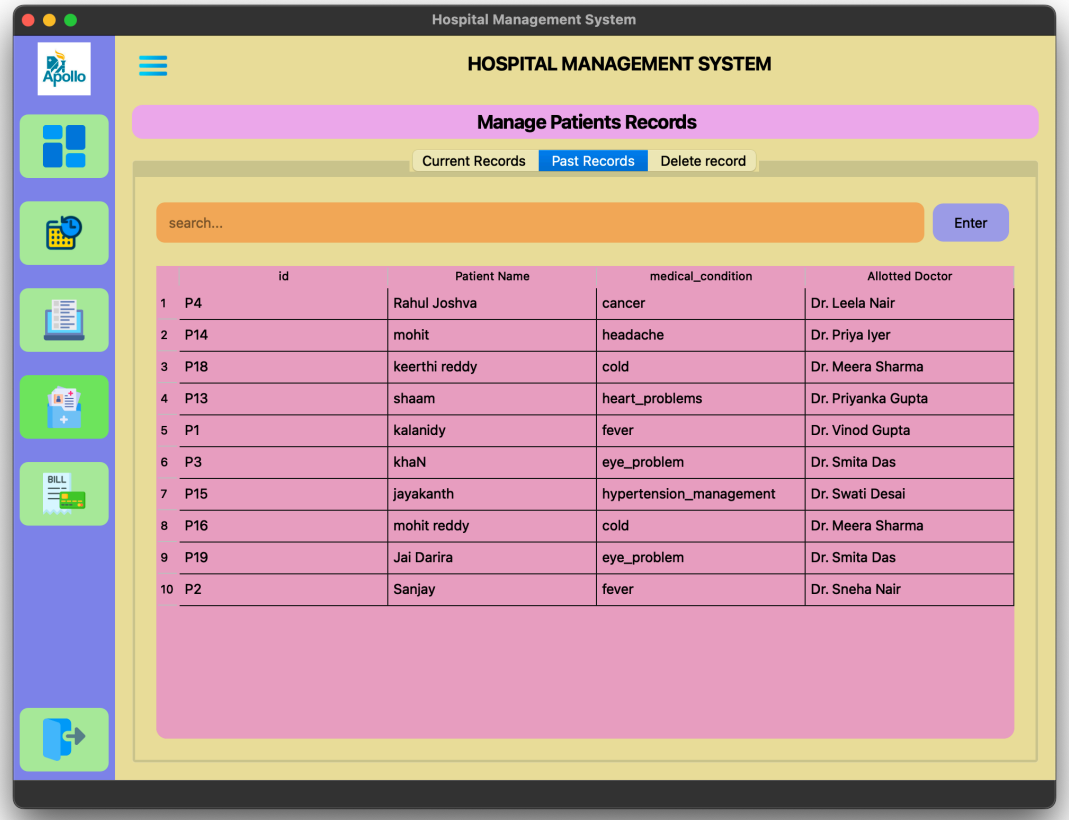

Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Doctors Availablity
not available
Available
working
add doctor
delete doctor
search...
search
id
Doctor Name
Specialization
Contact
1 D11 Dr. Pooja Deshmukh nausea 9876543220
2 D13 Dr. Rahul Verma sore_throat 9876543222
3 D14 Dr. Kavita Kapoor diarrhea 9876543223
4 D19 Dr. Ravi Shastri ear_problem 9876543228
5 D29 Dr. Radhika Sen icu_heart 9876543238
6 D36 Dr. Parag Jain chronic_obstructive_pulmon... 9876543245
7 D37 Dr. Usha Mehta pulmonary_embolism 9876543246
8 D45 Dr. Gaurav Patel gastroesophageal_reflux_dis... 9876543254
mark available


Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Doctors Availablity
not available
Available
working
add doctor
delete doctor
search...
search
id
Doctor Name
Specialization
Contact
1 D1 Dr. Arun Kumar general_checkup 9876543210
2 D10 Dr. Karan Mehta fatigue 9876543219
3 D12 Dr. Deepak Shah stomach_pain 9876543221
4 D15 Dr. Manish Tiwari allergy 9876543224
5 D16 Dr. Alok Sinha back_problem 9876543225
6 D17 Dr. Smita Das eye_problem 9876543226
7 D18 Dr. Veena Reddy skin_problem 9876543227
8 D2 Dr. Riya Singh general_checkup 9876543211
9 D20 Dr. Neha Sharma muscle_pain 9876543229
10 D21 Dr. Sanjay Joshi joint_pain 9876543230
11 D23 Dr. Anil Kapoor arrhythmia 9876543232
mark not available

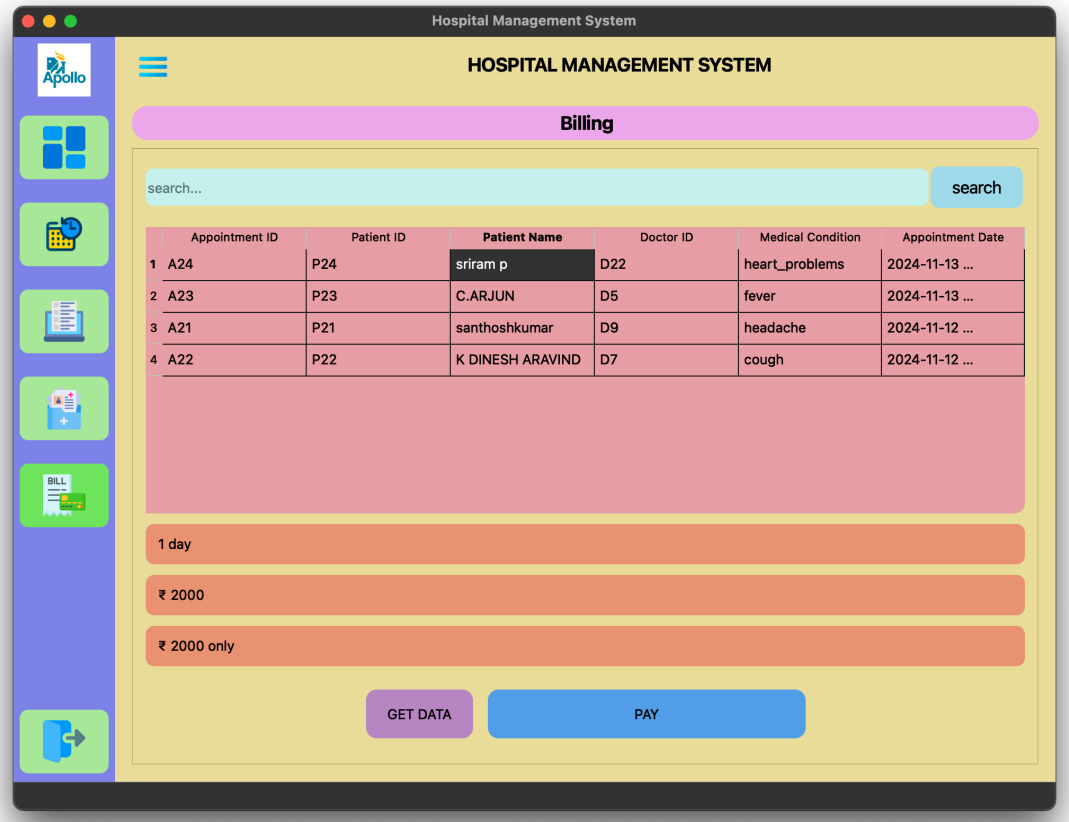

Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Doctors Availablity
not available
Available
working
add doctor
delete doctor
search...
search
id
Doctor Name
Specialization
Contact
1 D22 Dr. Priyanka Gupta heart_problems 9876543231
2 D5 Dr. Vinod Gupta fever 9876543214
3 D7 Dr. Sunita Rao cough 9876543216
4 D9 Dr. Priya Iyer headache 9876543218


Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Doctors Availablity
not available
Available
working
add doctor
delete doctor
contact info
Doctor name
Department
House no
state
salary
city
add

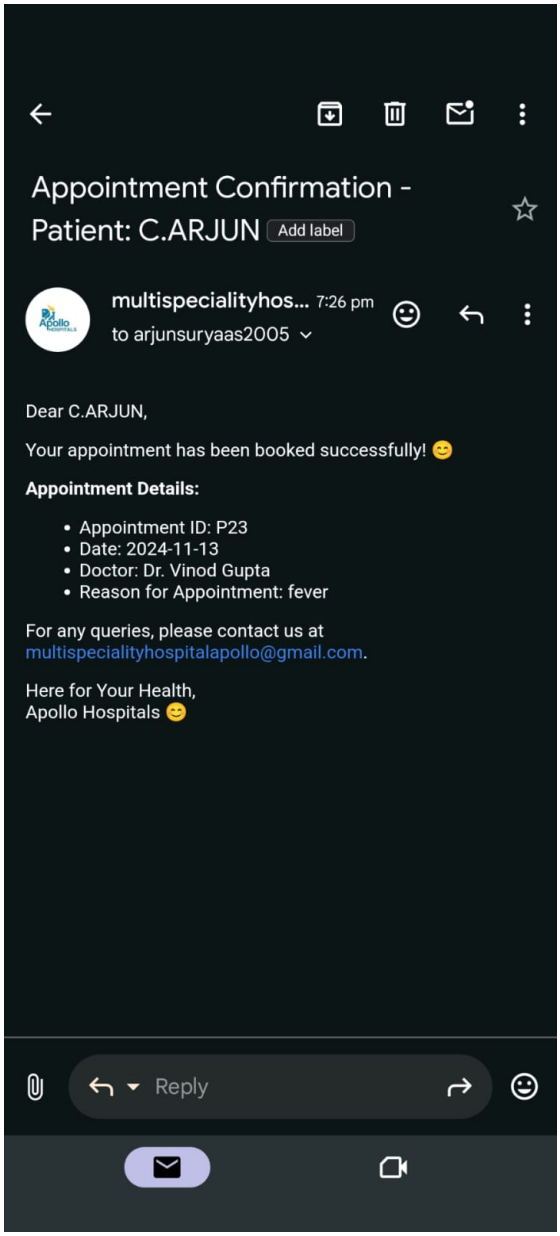

Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Manage Patients Records
Current Records
Past Records
Delete record
search...
Enter
id
Patient Name
medical_condition
Allotted Doctor
1 P24 sriram p heart_problems Dr. Priyanka Gupta
2 P23 C.ARJUN fever Dr. Vinod Gupta
3 P21 santhoshkumar headache Dr. Priya Iyer
4 P22 K DINESH ARAVIND cough Dr. Sunita Rao

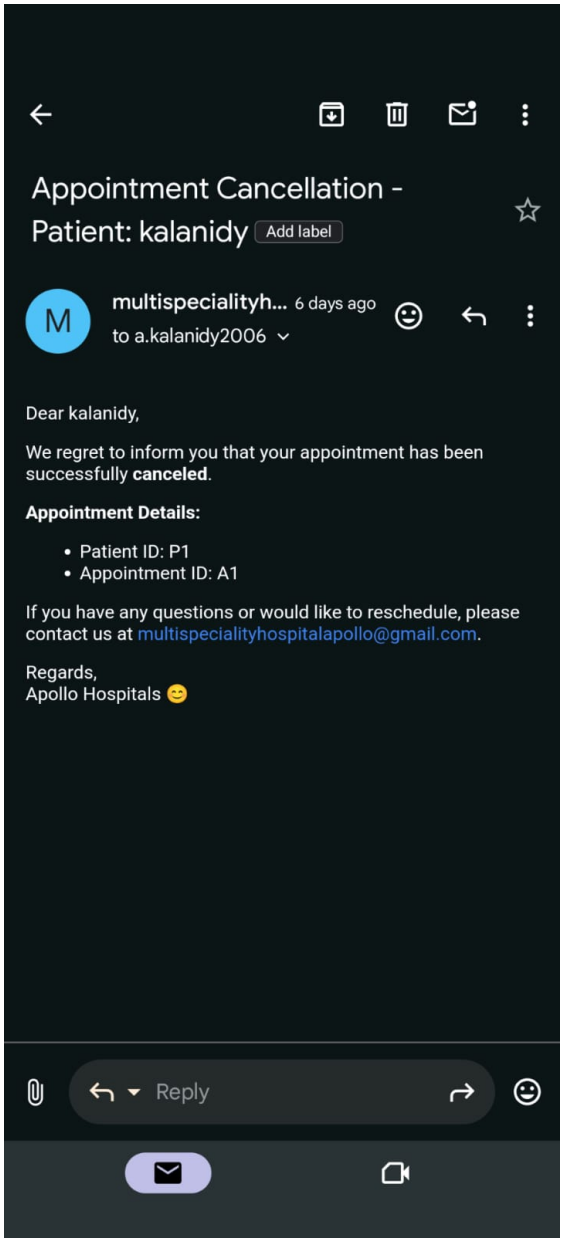

Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Manage Patients Records
Current Records
Past Records
Delete record
search...
Enter
id
Patient Name
medical_condition
Allotted Doctor
1 P4 Rahul Joshva cancer Dr. Leela Nair
2 P14 mohit headache Dr. Priya Iyer
3 P18 keerthi reddy cold Dr. Meera Sharma
4 P13 shaam heart_problems Dr. Priyanka Gupta
5 P1 kalanidy fever Dr. Vinod Gupta
6 P3 khaN eye_problem Dr. Smita Das
7 P15 jayakanth hypertension_management Dr. Swati Desai
8 P16 mohit reddy cold Dr. Meera Sharma
9 P19 Jai Darira eye_problem Dr. Smita Das
10 P2 Sanjay fever Dr. Sneha Nair


Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Manage Patients Records
Current Records
Past Records
Delete record
id
Patient Name
medical_condition
Allotted Doctor
1 P24 sriram p heart_problems Dr. Priyanka Gupta
2 P23 C.ARJUN fever Dr. Vinod Gupta
3 P21 santhoshkumar headache Dr. Priya Iyer
4 P22 K DINESH ARAVIND cough Dr. Sunita Rao
search...
search
Delete Record

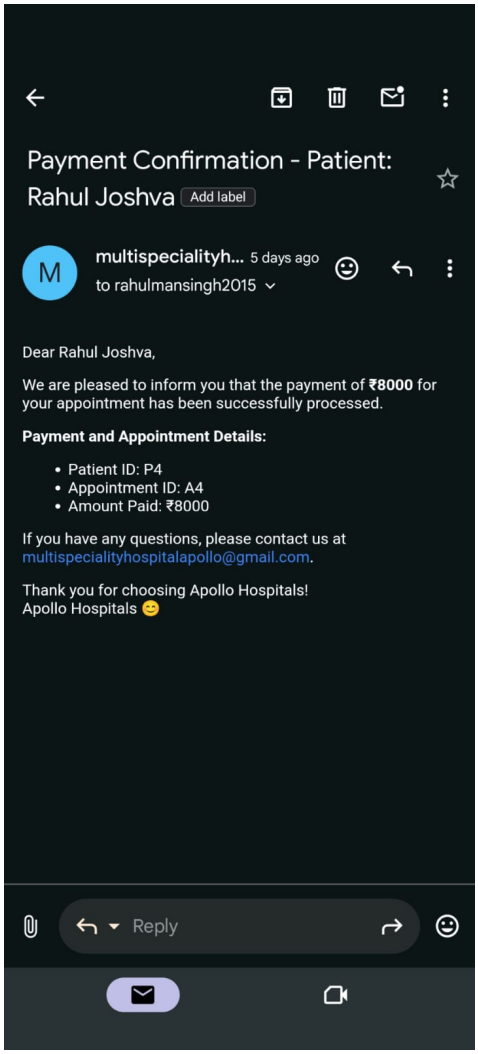

Hospital Management System
HOSPITAL MANAGEMENT SYSTEM
Billing
search...
search
Appointment ID
Patient ID
Patient Name
Doctor ID
Medical Condition
Appointment Date
1 A24 P24 sriram p D22 heart_problems 2024-11-13 ...
2 A23 P23 C.ARJUN D5 fever 2024-11-13 ...
3 A21 P21 santhoshkumar D9 headache 2024-11-12 ...
4 A22 P22 K DINESH ARAVIND D7 cough 2024-11-12 ...
1 day
₹ 2000
₹ 2000 only
GET DATA
PAY

APPOINTMENT CONFIRMATION AND CANCELLATION





PAYMENT PART :



Result :

The Hospital Management System (HMS) will provide an organized and efficient way to manage and track hospital data, including patient records, doctor information, and appointment scheduling.