DATABASE MANAGEMENT SYSTEMS
ITE1003 - J COMPONENT

Title:

HOSPITAL MANAGEMENT SYSTEM (Patient-Doctor Appointment Management System)

Faculty Name: Tapan Kumar Das

Team Members:

Tanishq Tyagi - 20BIT0192
Priyanka Chowdhury - 20BIT0197
Tanisha Mishra - 20BIT0319

Abstract:

Generally, despite the advance in technology, many government hospitals still use a manual system for the management and maintenance of critical information in the Hospitals. The current system requires numerous paper forms. Often information (on these forms) is incomplete or does not follow proper management standards. Forms are often lost in transit between departments requiring a comprehensive auditing process to ensure that no vital

information is lost. Multiple copies of the same information exist in the hospital which may lead to inconsistencies in data.

A significant part of the operation of any hospital involves the acquisition, management, and timely retrieval of great volumes of information. This information typically involves patient details, doctor details, billing details, medication details, test reports, etc. All of this information must be managed in an efficient and cost-wise fashion so that resources may be effectively utilized. In our project, we will automate the management of the hospital, making it more efficient and error-free.

METHODOLOGY:

In this project, we aim to make a Hospital management System, which will provide data like patient details, doctor details, billing details, medication details, test reports, pharmacy store details of various hospitals. The patient will be assigned a unique patient id that will helps access all their information and records. Similarly, doctors will be assigned a unique doctor id by which their information can be stored in the database. Also, we will be using the Oracle, Flask , comm to perform sql queries database to achieve this goal.

SOFTWARE USED:

1. LiveSql/SQLlite
2. Python
3. Flask
4. comm (python library) to perform sql queries
5. Bootstrap for frontend

EXPECTED RESULT:

We will be developing a Hospital Management System which will provide data like patient details, doctor details and appointment details. In this we plan to automate the management of the hospital, making it more efficient and error-free.

Data Requirements:
Entity types

1)PATIENT:
- pat_id which is unique and not null.
- Name is a composite attribute of first name and last name which should not be null.
- Address stores address of patient which should not be null.
- Phone no. has phone no of patient.
- Insurance which should not be null.

2)DOCTOR:
- doc_id which is unique and not null.
- Name is a composite attribute of first name and last name which should not be null.
- Address stores address of patient which should not be null.
- Phone no. has phone no of patient.
- Insurance which should not be null.

3)APPOINTMENT:
- app_id which is unique and not null.
- pat_id which is an INTEGER NOT NULL,
- doc_id which is an INTEGER NOT NULL,
- appointment_date DATE NOT NULL,
- FOREIGN KEY(pat_id) REFERENCES patient(pat_id),
- FOREIGN KEY(doc_id) REFERENCES doctor(doc_id));

Sql Commands to create tables:

PATIENT TABLE

```
create table patient(
pat_id number not null primary key,
pat_first_name varchar(30) not null,
pat_last_name varchar(30) not null,
pat_insurence_no varchar(30) not null,
pat_ph_no varchar(30) not null,
pat_date timestamp not null,
pat_address varchar(100) not null);
```

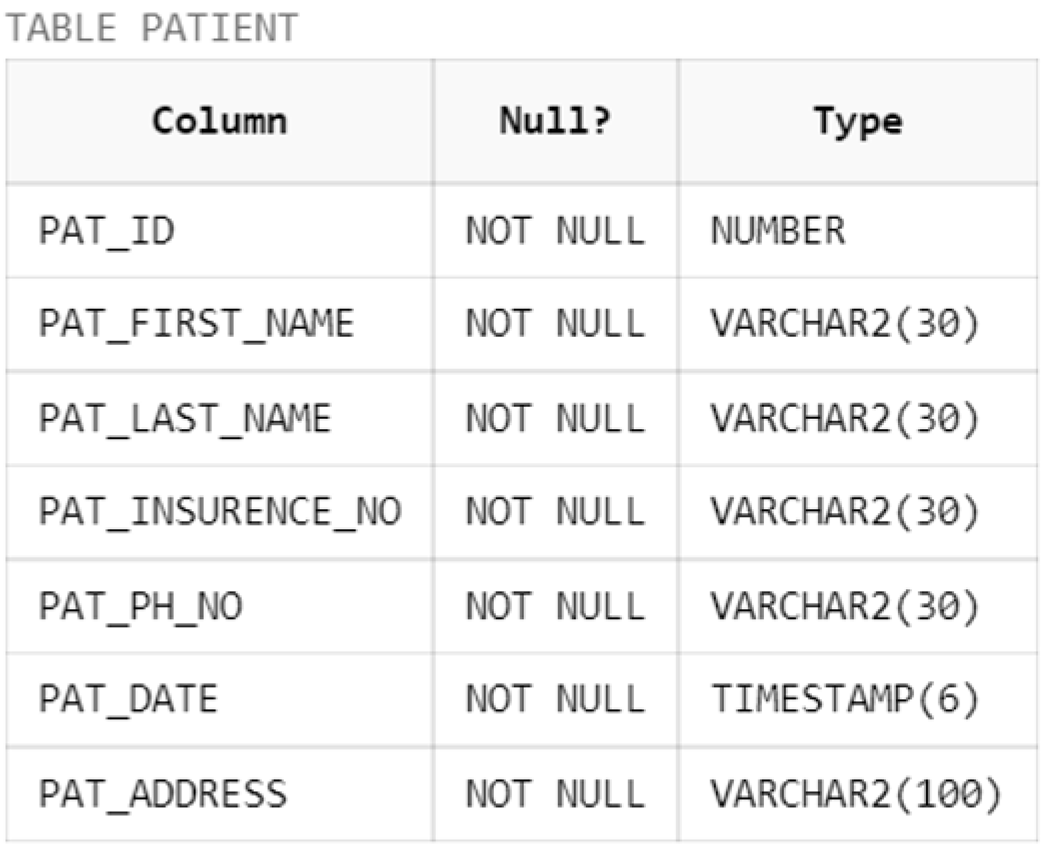


TABLE PATIENT

Column	Null?	Type
PAT_ID	NOT NULL	NUMBER
PAT_FIRST_NAME	NOT NULL	VARCHAR2(30)
PAT_LAST_NAME	NOT NULL	VARCHAR2(30)
PAT_INSURENCE_NO	NOT NULL	VARCHAR2(30)
PAT_PH_NO	NOT NULL	VARCHAR2(30)
PAT_DATE	NOT NULL	TIMESTAMP(6)
PAT_ADDRESS	NOT NULL	VARCHAR2(100)

DOCTOR TABLE

```
create table doctor(
doc_id number not null primary key,
doc_first_name varchar(30) not null,
doc_last_name varchar(30) not null,
doc_ph_no varchar(12) not null,
doc_date timestamp not null,
doc_address varchar(100) not null);
```

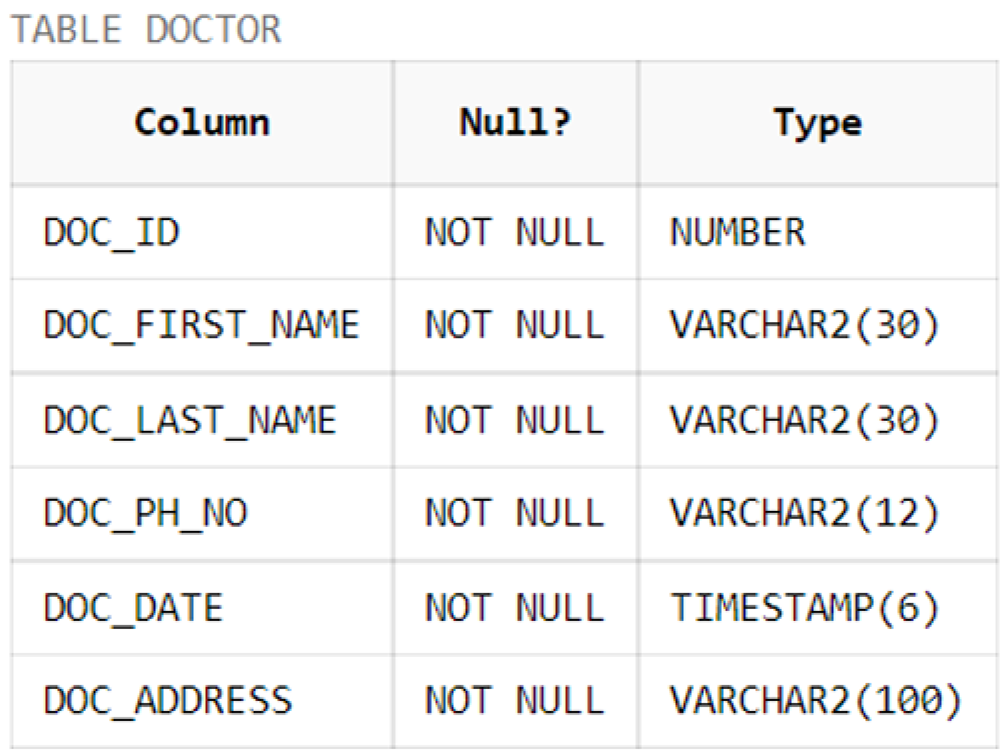


TABLE DOCTOR

Column	Null?	Type
DOC_ID	NOT NULL	NUMBER
DOC_FIRST_NAME	NOT NULL	VARCHAR2(30)
DOC_LAST_NAME	NOT NULL	VARCHAR2(30)
DOC_PH_NO	NOT NULL	VARCHAR2(12)
DOC_DATE	NOT NULL	TIMESTAMP(6)
DOC_ADDRESS	NOT NULL	VARCHAR2(100)

APPOINTMENT TABLE

```
create table appointment(
app_id number not null primary key,
pat_id number not null,
doc_id number not null,
app_date timestamp not null);
```

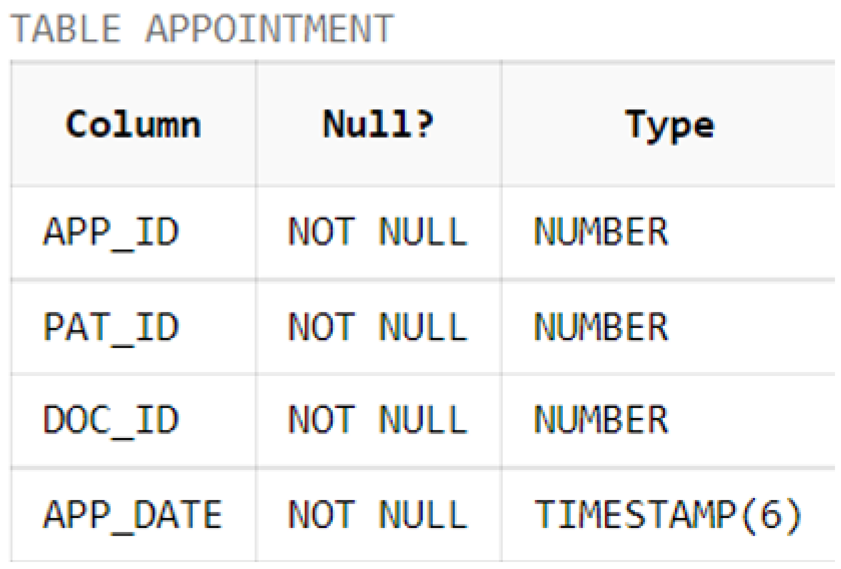


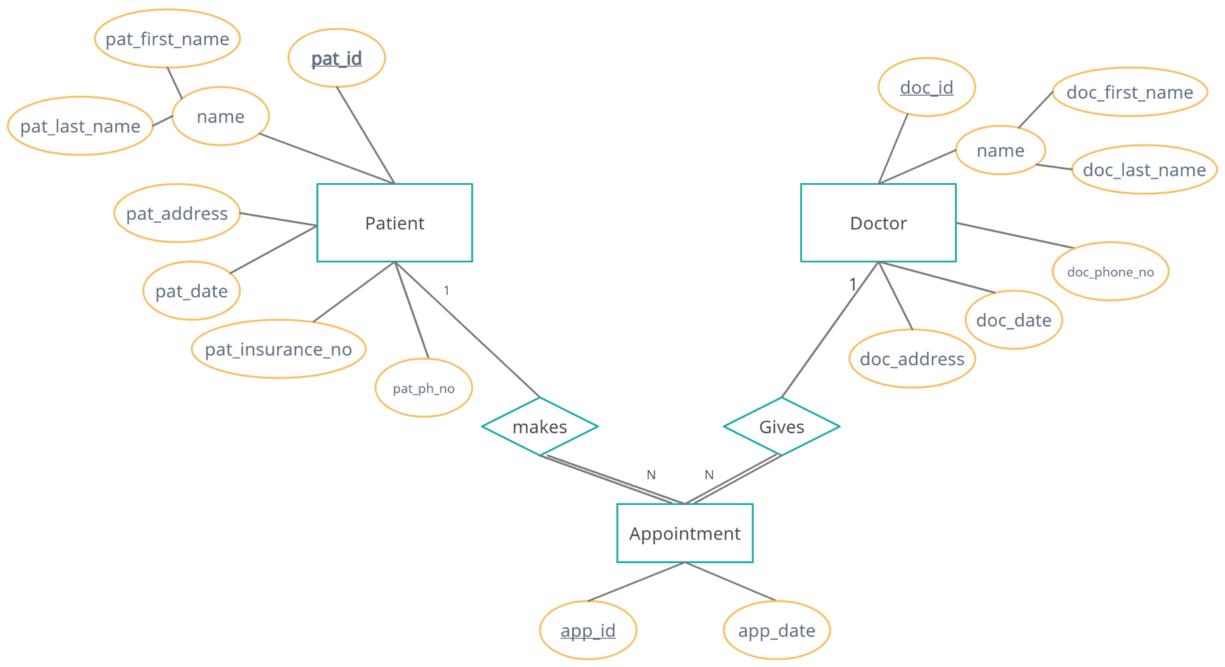
TABLE APPOINTMENT

Column	Null?	Type
APP_ID	NOT NULL	NUMBER
PAT_ID	NOT NULL	NUMBER
DOC_ID	NOT NULL	NUMBER
APP_DATE	NOT NULL	TIMESTAMP(6)

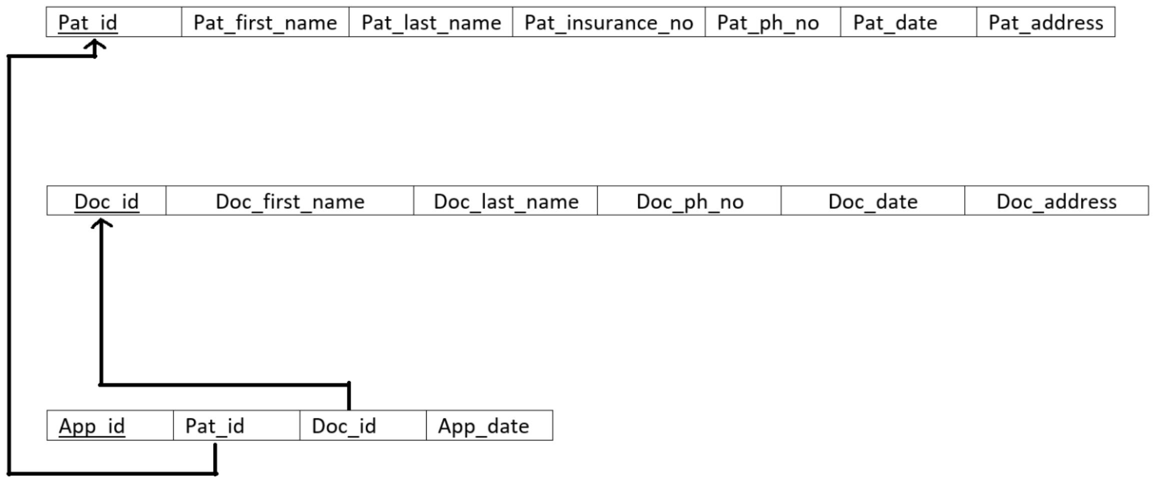
CONSTRAINTS:

```
alter table appointment add foreign key (pat_id) references patient(pat_id);
alter table appointment add foreign key (doc_id) references doctor(doc_id);
```

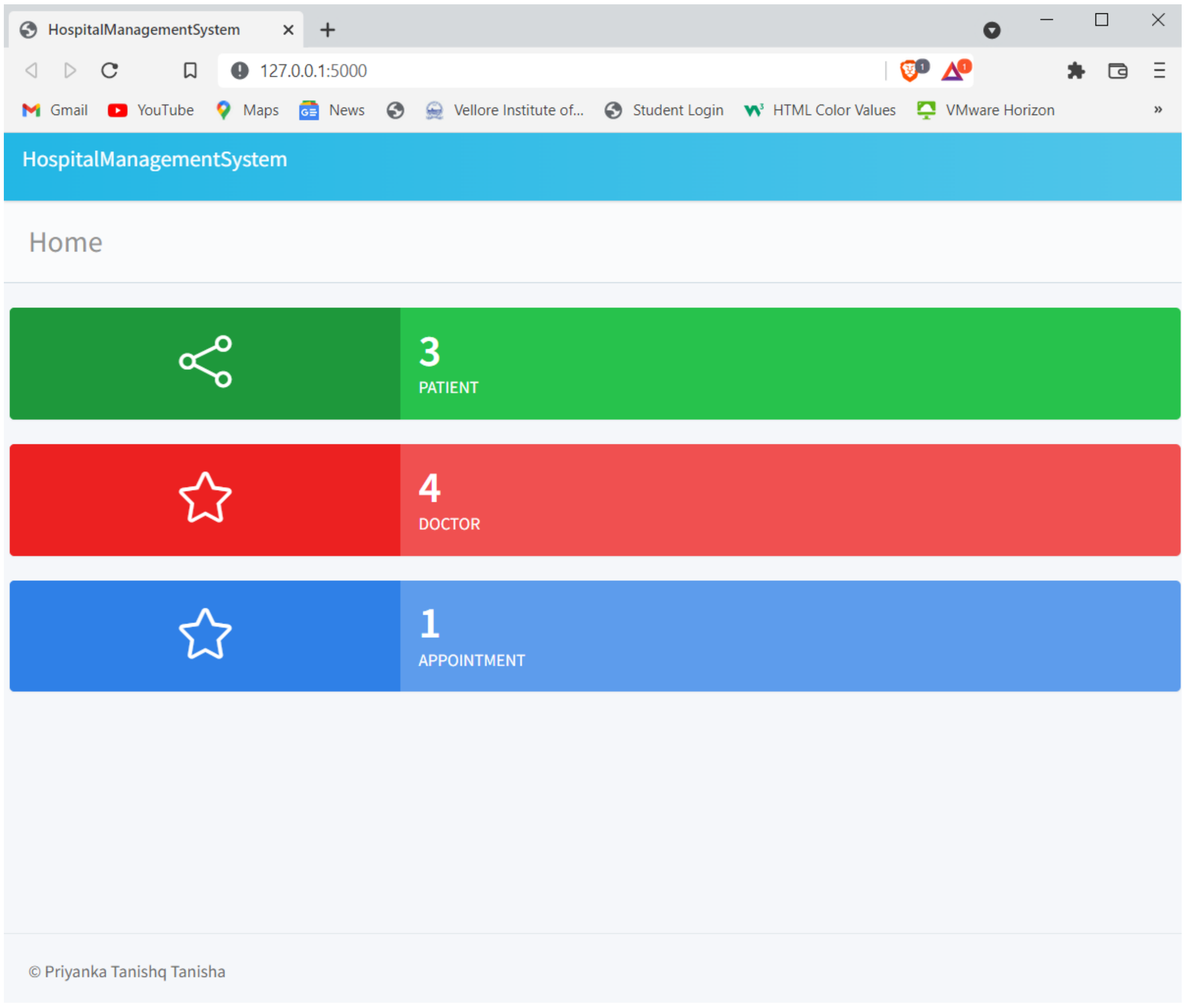
ER Diagram:



Relational Database Schema



WEBSITE SCREENSHOTS



HospitalManagementSystem

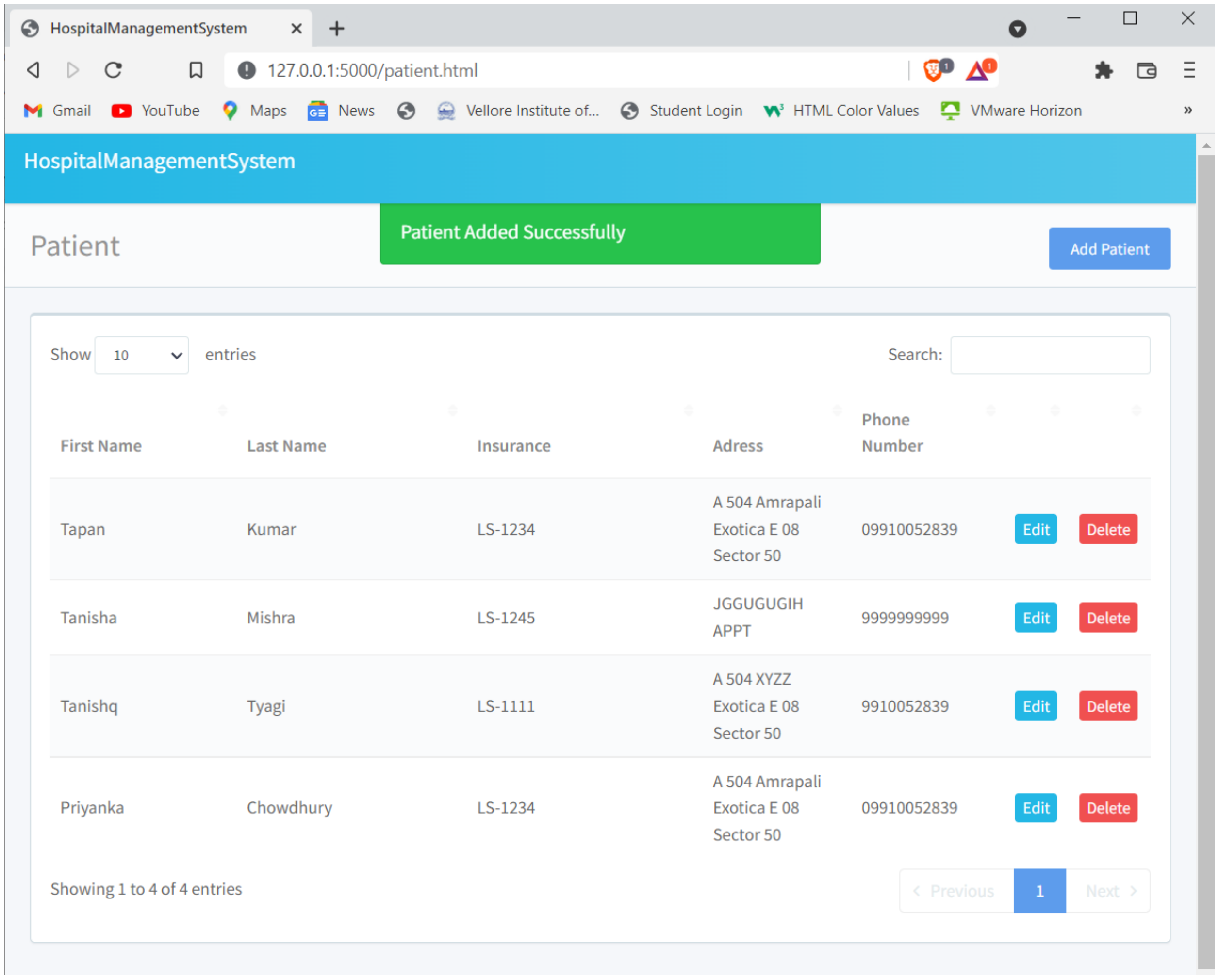
Home

3
PATIENT

4
DOCTOR

1
APPOINTMENT

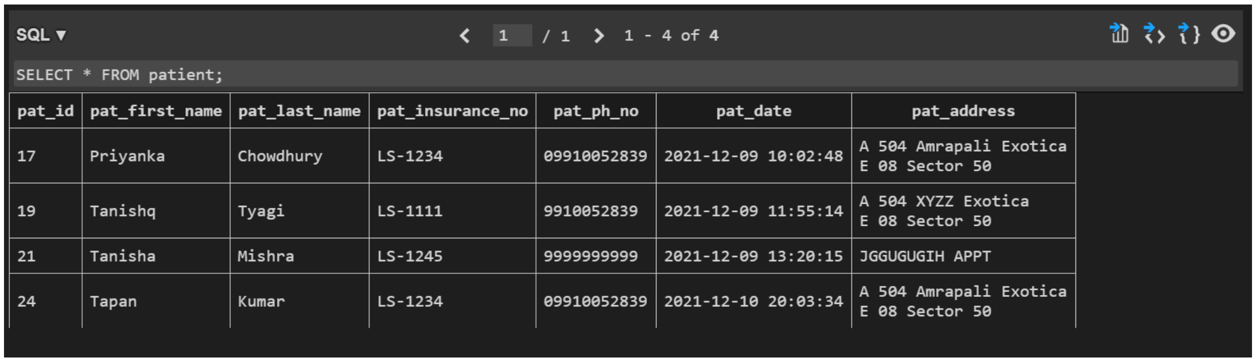
© Priyanka Tanishq Tanisha



Patient Added Successfully

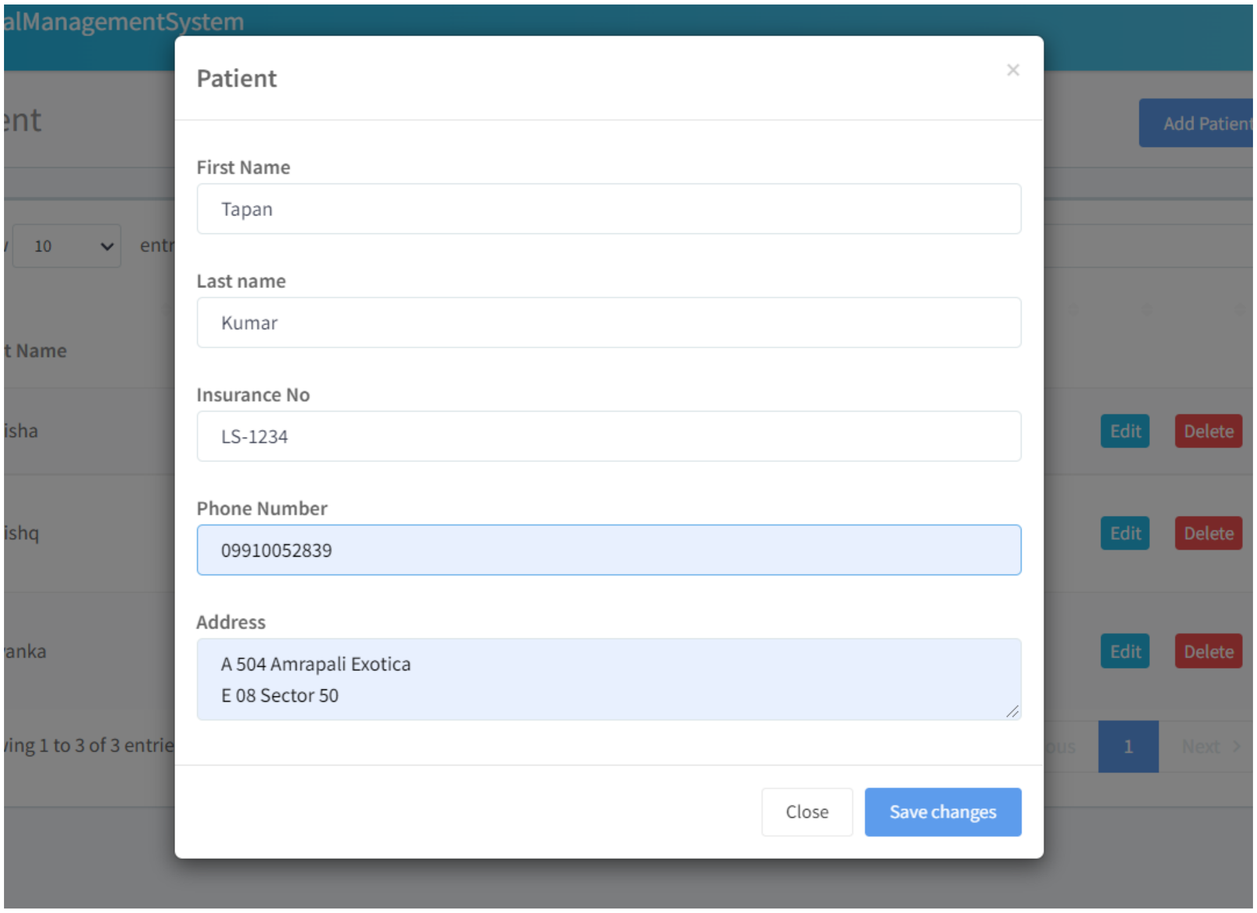
First Name	Last Name	Insurance	Adress	Phone Number		
Tapan	Kumar	LS-1234	A 504 Amrapali Exotica E 08 Sector 50	09910052839	Edit	Delete
Tanisha	Mishra	LS-1245	JGGUGUGIH APPT	9999999999	Edit	Delete
Tanishq	Tyagi	LS-1111	A 504 XYZZ Exotica E 08 Sector 50	9910052839	Edit	Delete
Priyanka	Chowdhury	LS-1234	A 504 Amrapali Exotica E 08 Sector 50	09910052839	Edit	Delete

Showing 1 to 4 of 4 entries



```
SELECT * FROM patient;
```

pat_id	pat_first_name	pat_last_name	pat_insurance_no	pat_ph_no	pat_date	pat_address
17	Priyanka	Chowdhury	LS-1234	09910052839	2021-12-09 10:02:48	A 504 Amrapali Exotica E 08 Sector 50
19	Tanishq	Tyagi	LS-1111	9910052839	2021-12-09 11:55:14	A 504 XYZZ Exotica E 08 Sector 50
21	Tanisha	Mishra	LS-1245	9999999999	2021-12-09 13:20:15	JGGUGUGIH APPT
24	Tapan	Kumar	LS-1234	09910052839	2021-12-10 20:03:34	A 504 Amrapali Exotica E 08 Sector 50



Patient

First Name: Tapan

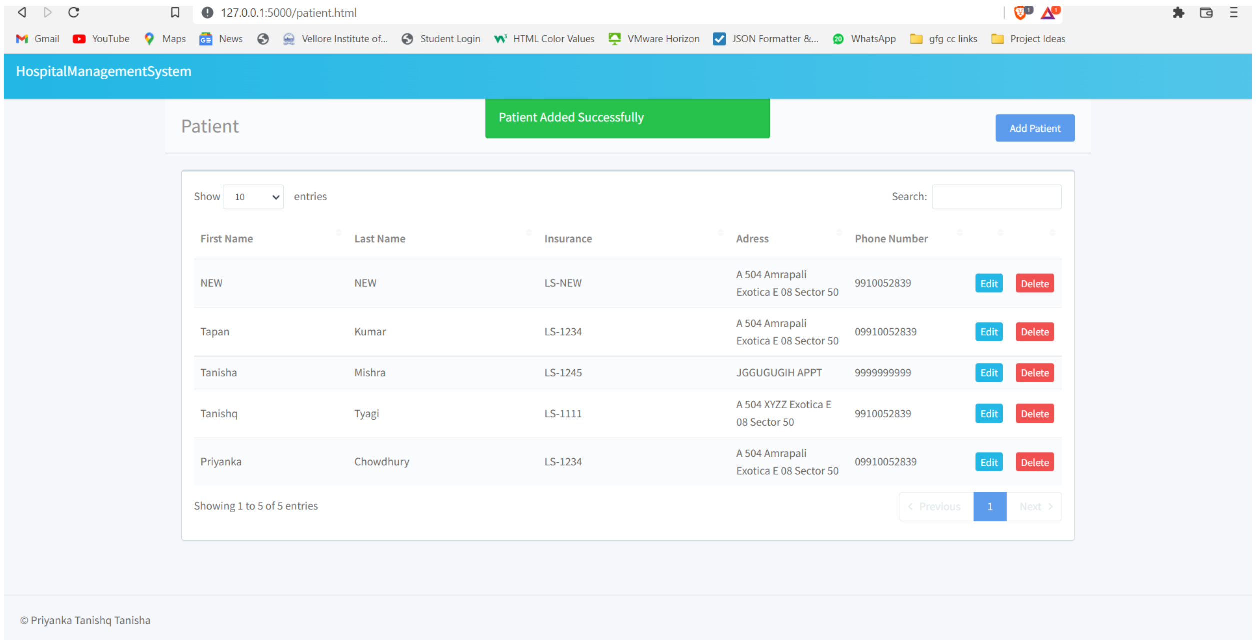
Last name: Kumar

Insurance No: LS-1234

Phone Number: 09910052839

Address: A 504 Amrapali Exotica
E 08 Sector 50

ON ADDING NEW PATIENT



Patient Added Successfully

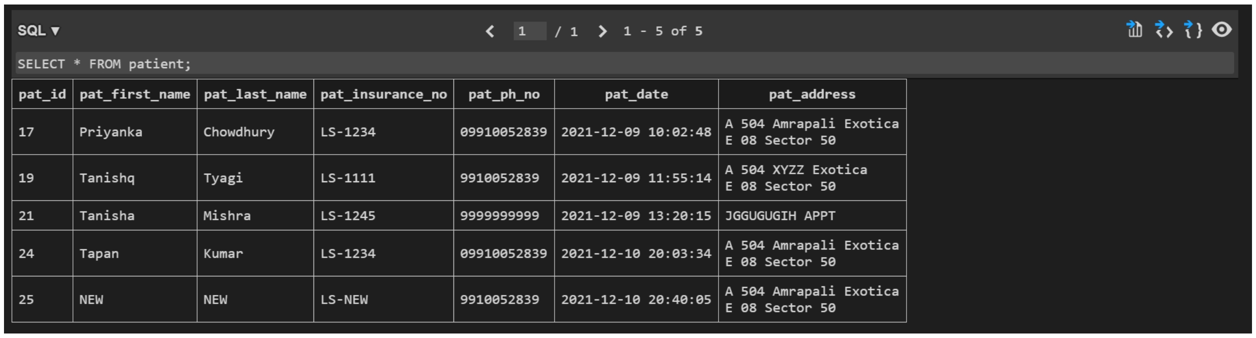
Show 10 entries   Search:

First Name	Last Name	Insurance	Adress	Phone Number		
NEW	NEW	LS-NEW	A 504 Amrapali Exotica E 08 Sector 50	9910052839	Edit	Delete
Tapan	Kumar	LS-1234	A 504 Amrapali Exotica E 08 Sector 50	09910052839	Edit	Delete
Tanisha	Mishra	LS-1245	JGGUGUGIH APPT	9999999999	Edit	Delete
Tanishq	Tyagi	LS-1111	A 504 XYZZ Exotica E 08 Sector 50	9910052839	Edit	Delete
Priyanka	Chowdhury	LS-1234	A 504 Amrapali Exotica E 08 Sector 50	09910052839	Edit	Delete

Showing 1 to 5 of 5 entries

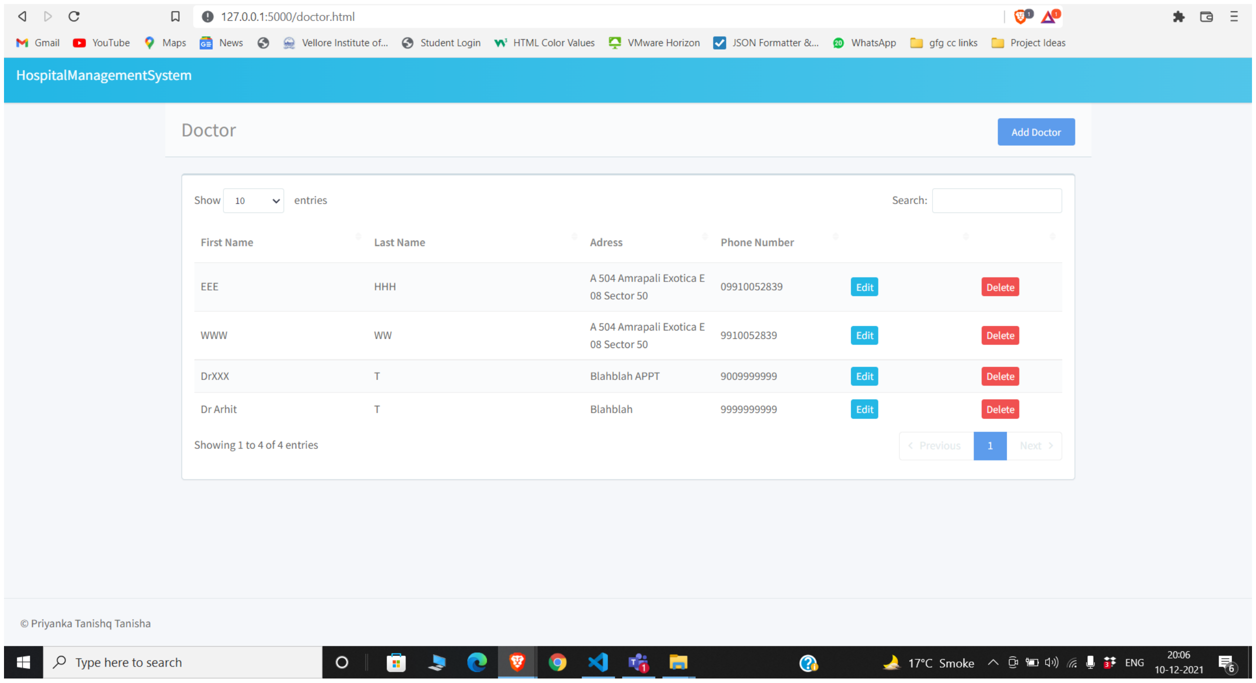
© Priyanka Tanishq Tanisha

NEW ROW ADDED



```
SELECT * FROM patient;
```

pat_id	pat_first_name	pat_last_name	pat_insurance_no	pat_ph_no	pat_date	pat_address
17	Priyanka	Chowdhury	LS-1234	09910052839	2021-12-09 10:02:48	A 504 Amrapali Exotica E 08 Sector 50
19	Tanishq	Tyagi	LS-1111	9910052839	2021-12-09 11:55:14	A 504 XYZZ Exotica E 08 Sector 50
21	Tanisha	Mishra	LS-1245	9999999999	2021-12-09 13:20:15	JGGUGUGIH APPT
24	Tapan	Kumar	LS-1234	09910052839	2021-12-10 20:03:34	A 504 Amrapali Exotica E 08 Sector 50
25	NEW	NEW	LS-NEW	9910052839	2021-12-10 20:40:05	A 504 Amrapali Exotica E 08 Sector 50

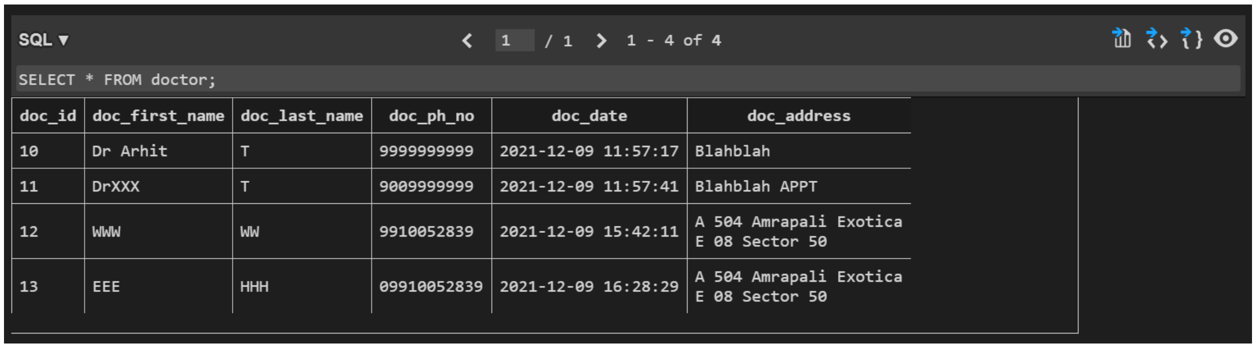


Doctor

First Name	Last Name	Adress	Phone Number		
EEE	HHH	A 504 Amrapali Exotica E 08 Sector 50	09910052839	Edit	Delete
WWW	WW	A 504 Amrapali Exotica E 08 Sector 50	9910052839	Edit	Delete
DrXXX	T	Blahblah APPT	9009999999	Edit	Delete
Dr Arhit	T	Blahblah	9999999999	Edit	Delete

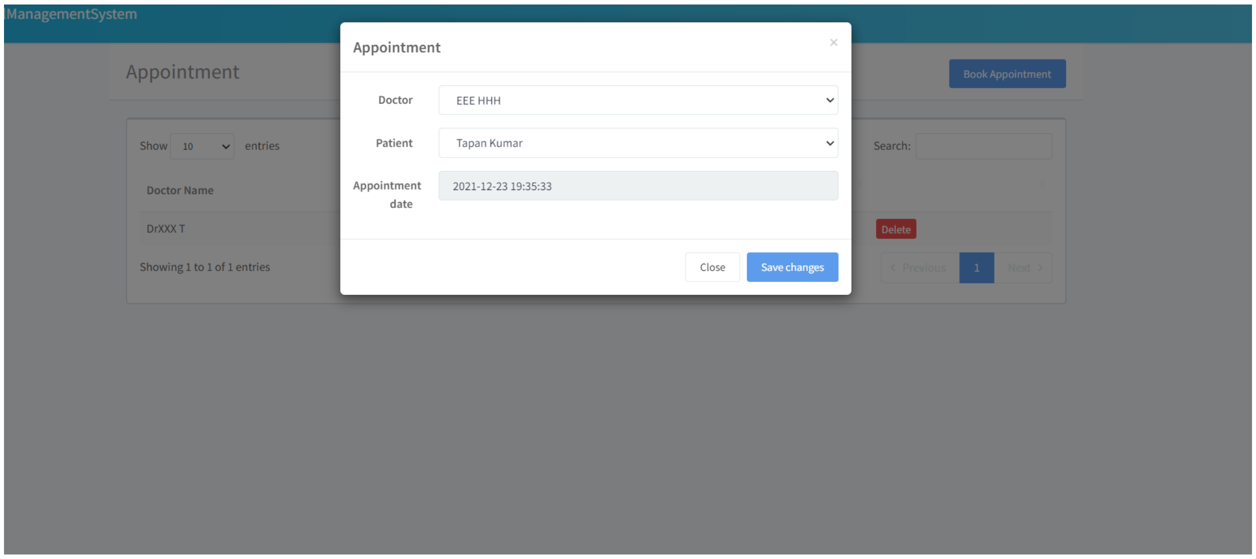
Showing 1 to 4 of 4 entries

© Priyanka Tanishq Tanisha



```
SELECT * FROM doctor;
```

doc_id	doc_first_name	doc_last_name	doc_ph_no	doc_date	doc_address
10	Dr Arhit	T	9999999999	2021-12-09 11:57:17	Blahblah
11	DrXXX	T	9009999999	2021-12-09 11:57:41	Blahblah APPT
12	WWW	WW	9910052839	2021-12-09 15:42:11	A 504 Amrapali Exotica E 08 Sector 50
13	EEE	HHH	09910052839	2021-12-09 16:28:29	A 504 Amrapali Exotica E 08 Sector 50



ManagementSystem

Appointment

Book Appointment

Show 10 entries

Search:

Doctor Name

DrXXX T

Delete

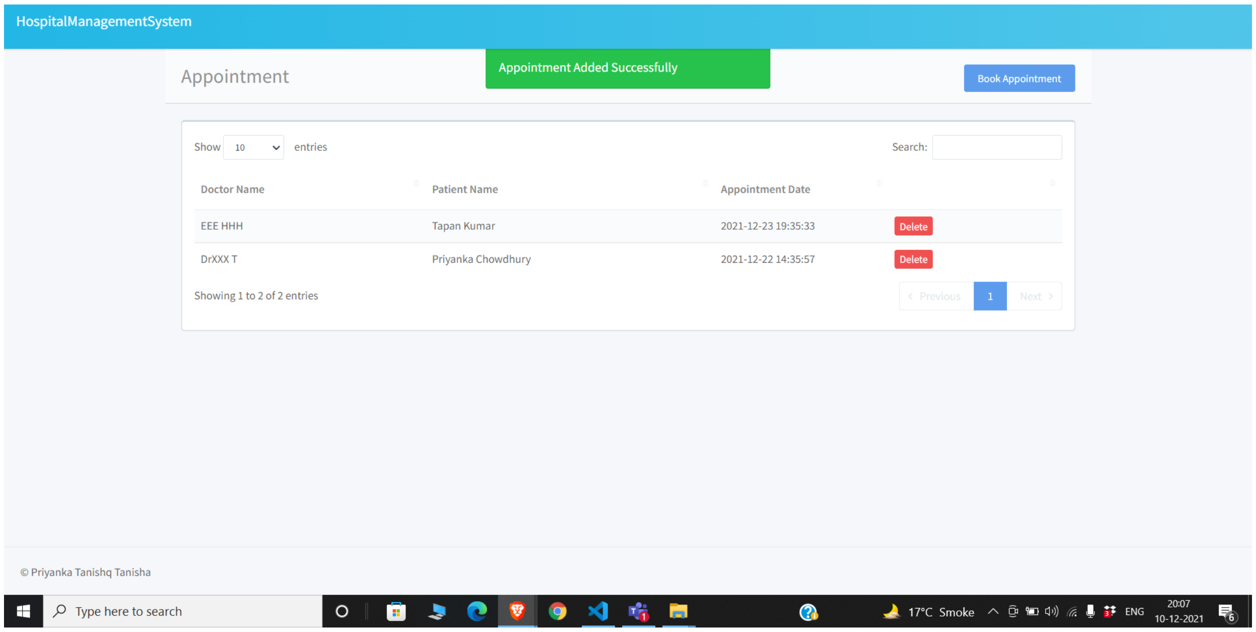
Showing 1 to 1 of 1 entries

Previous 1 Next

Appointment

Doctor	EEE HHH
Patient	Tapan Kumar
Appointment date	2021-12-23 19:35:33

Close Save changes



HospitalManagementSystem

Appointment

Appointment Added Successfully

Book Appointment

Show 10 entries

Search:

Doctor Name	Patient Name	Appointment Date	
EEE HHH	Tapan Kumar	2021-12-23 19:35:33	Delete
DrXXX T	Priyanka Chowdhury	2021-12-22 14:35:57	Delete

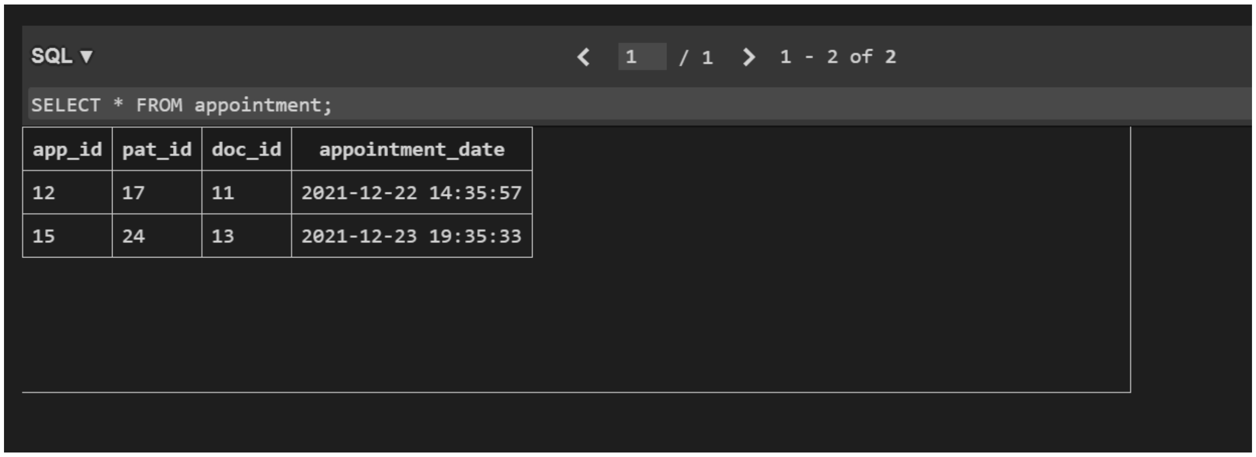
Showing 1 to 2 of 2 entries

Previous 1 Next

© Priyanka Tanishq Tanisha

Type here to search

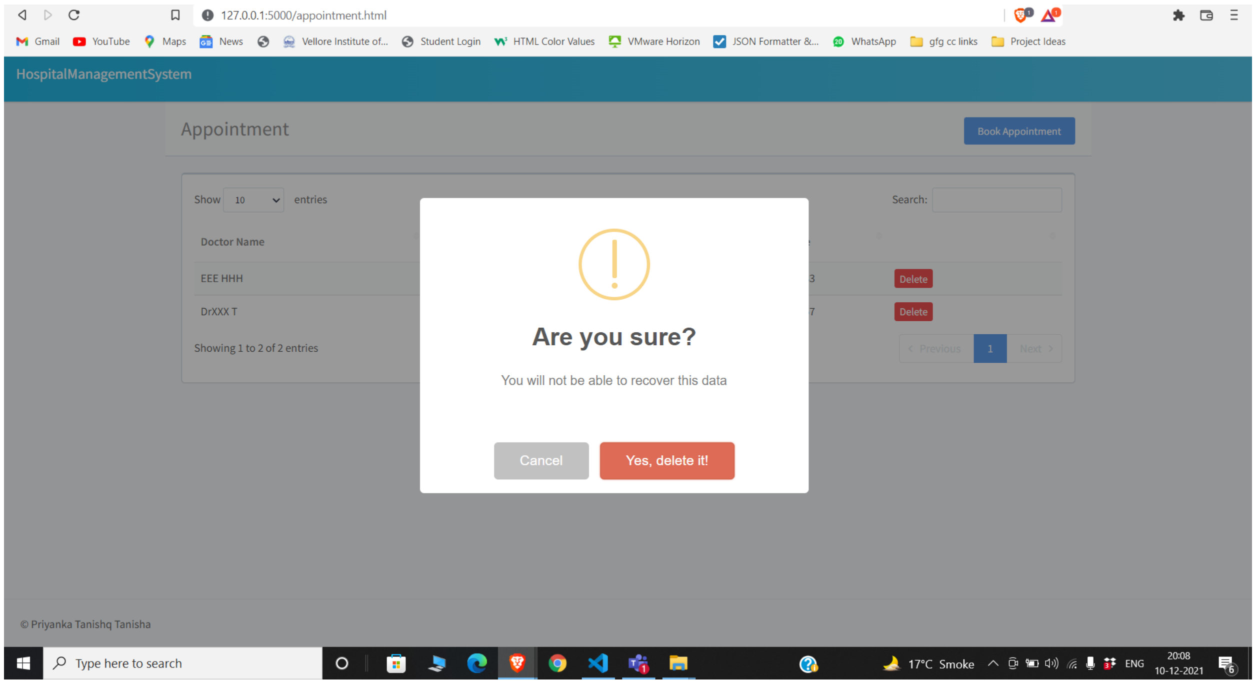
17°C Smoke ENG 20:07 10-12-2021



```
SQL ▼                    1 / 1    1 - 2 of 2
SELECT * FROM appointment;
```

app_id	pat_id	doc_id	appointment_date
12	17	11	2021-12-22 14:35:57
15	24	13	2021-12-23 19:35:33

Deleting data from the table



127.0.0.1:5000/appointment.html

HospitalManagementSystem

Appointment

Book Appointment

Show 10 entries

Search:

Doctor Name		
EEE HHH	3	Delete
DrXXX T	7	Delete

Showing 1 to 2 of 2 entries

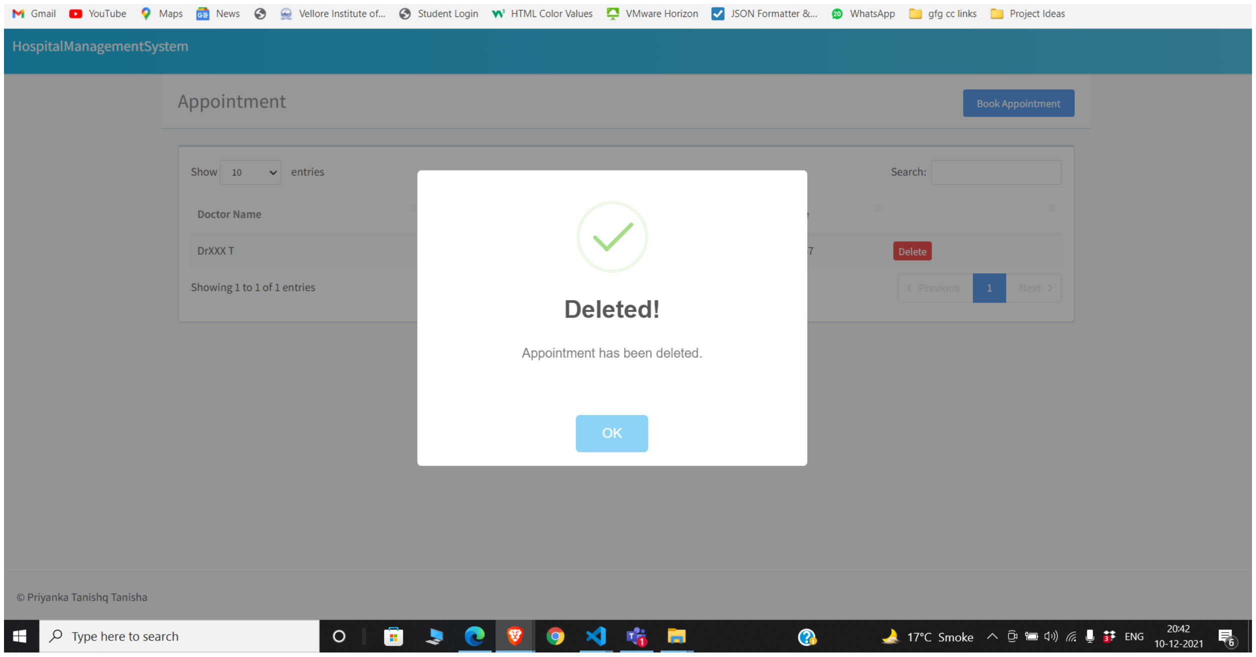
Previous 1 Next

Are you sure?

You will not be able to recover this data

Cancel | Yes, delete it!

© Priyanka Tanishq Tanisha


HospitalManagementSystem

Appointment

Book Appointment

Show 10 entries Search:

Doctor Name	
DrXXX T	Delete

Showing 1 to 1 of 1 entries

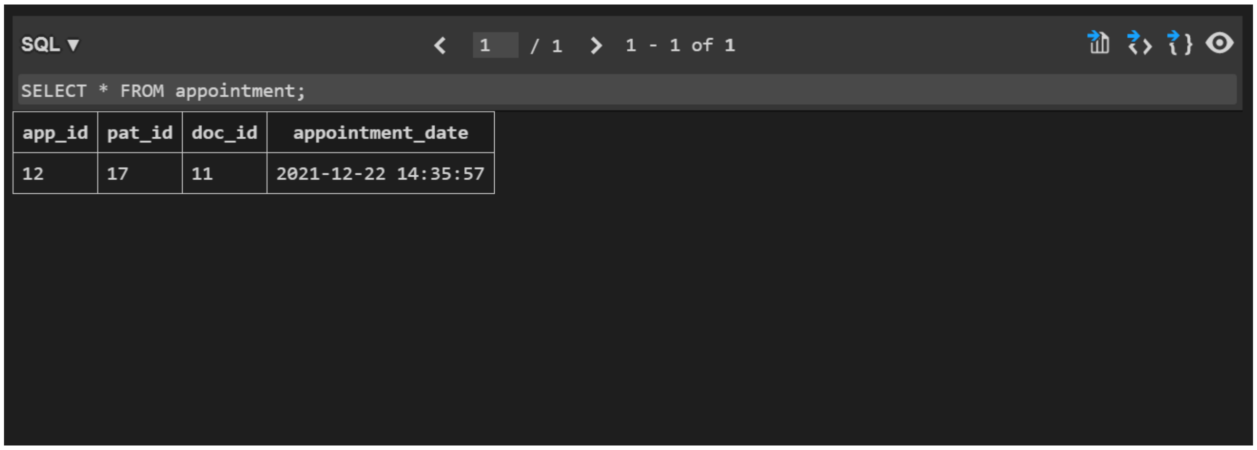
Deleted!

Appointment has been deleted.

OK

© Priyanka Tanishq Tanisha

ROW REMOVED AS APPOINTMENT HAS BEEN DELETED


```
SELECT * FROM appointment;
```

app_id	pat_id	doc_id	appointment_date
12	17	11	2021-12-22 14:35:57

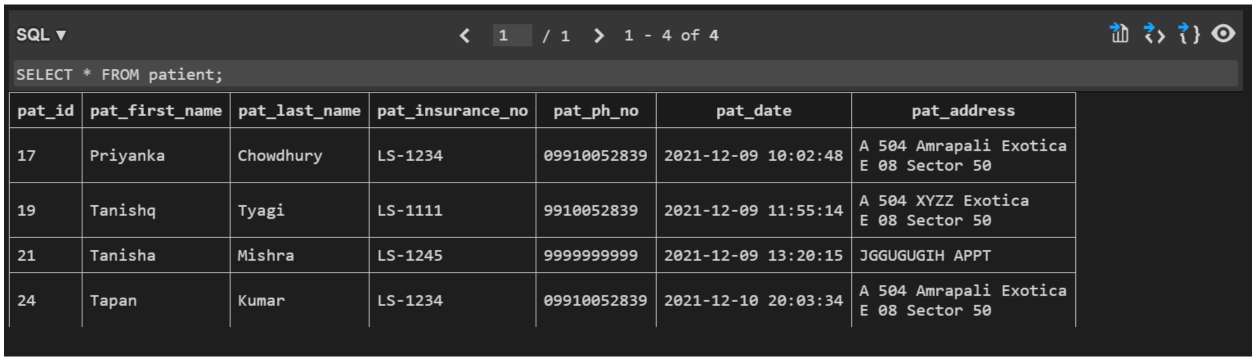
BACKEND

WEBSITE CODE

model.py

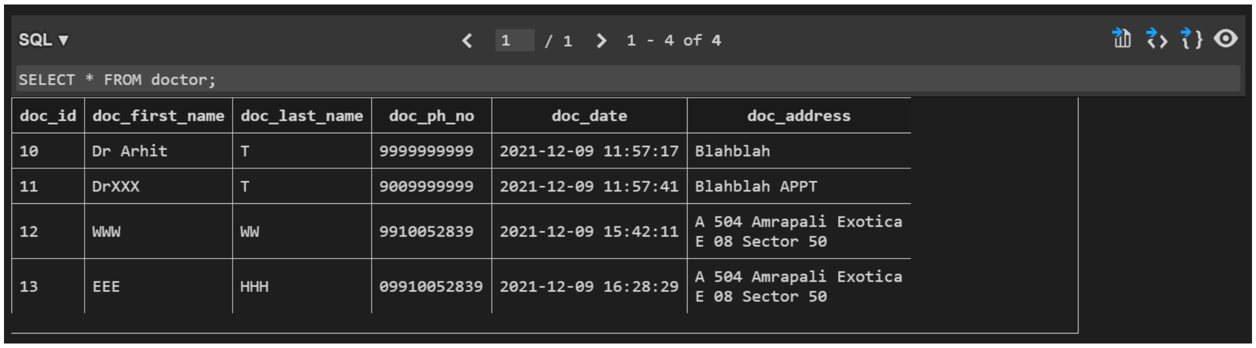
```
import sqlite3
import json
with open('config.json') as data_file:
    config = json.load(data_file)

conn=sqlite3.connect(config['database'], check_same_thread=False)
conn.execute('pragma foreign_keys=ON')
```

```
def dict_factory(cursor, row):
    """This is an function use to format the json when retrieve from the
mysql database"""
    d = {}
    for idx, col in enumerate(cursor.description):
        d[col[0]] = row[idx]
    return d

conn.row_factory = dict_factory

conn.execute('''CREATE TABLE if not exists patient
(pat_id INTEGER PRIMARY KEY AUTOINCREMENT,
pat_first_name TEXT NOT NULL,
pat_last_name TEXT NOT NULL,
pat_insurance_no TEXT NOT NULL,
pat_ph_no TEXT NOT NULL,
pat_date DATE DEFAULT (datetime('now','localtime')),
pat_address TEXT NOT NULL);''')
at
conn.execute('''CREATE TABLE if not exists doctor
(doc_id INTEGER PRIMARY KEY AUTOINCREMENT,
doc_first_name TEXT NOT NULL,
doc_last_name TEXT NOT NULL,
doc_ph_no TEXT NOT NULL,
doc_date DATE DEFAULT (datetime('now','localtime')),
doc_address TEXT NOT NULL);''')

conn.execute('''CREATE TABLE if not exists appointment
(app_id INTEGER PRIMARY KEY AUTOINCREMENT,
pat_id INTEGER NOT NULL,
doc_id INTEGER NOT NULL,
appointment_date DATE NOT NULL,
FOREIGN KEY(pat_id) REFERENCES patient(pat_id),
FOREIGN KEY(doc_id) REFERENCES doctor(doc_id));''')
```



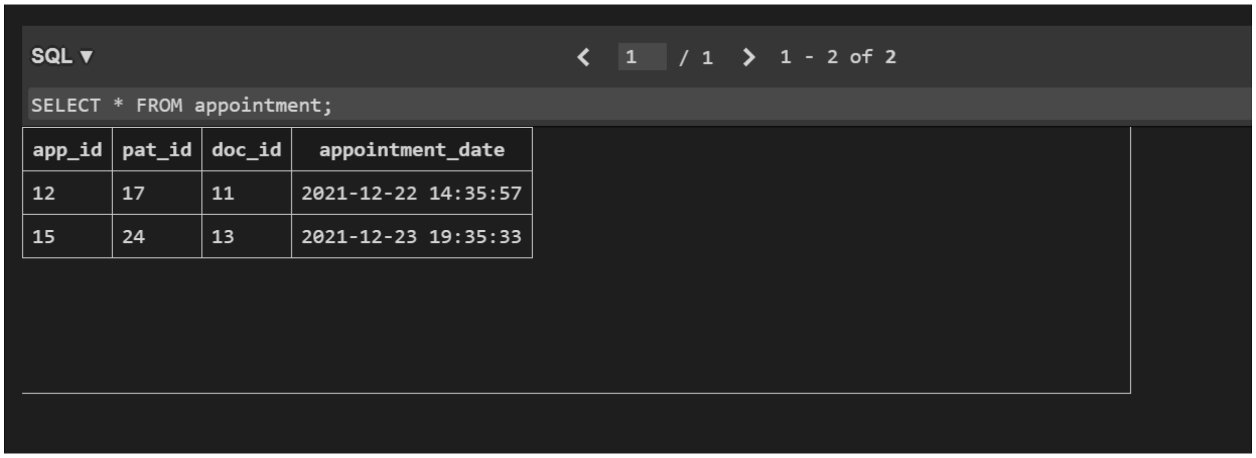
```
SELECT * FROM patient;
```

pat_id	pat_first_name	pat_last_name	pat_insurance_no	pat_ph_no	pat_date	pat_address
17	Priyanka	Chowdhury	LS-1234	09910052839	2021-12-09 10:02:48	A 504 Amrapali Exotica E 08 Sector 50
19	Tanishq	Tyagi	LS-1111	9910052839	2021-12-09 11:55:14	A 504 XYZZ Exotica E 08 Sector 50
21	Tanisha	Mishra	LS-1245	9999999999	2021-12-09 13:20:15	JGGUGUGIH APPT
24	Tapan	Kumar	LS-1234	09910052839	2021-12-10 20:03:34	A 504 Amrapali Exotica E 08 Sector 50



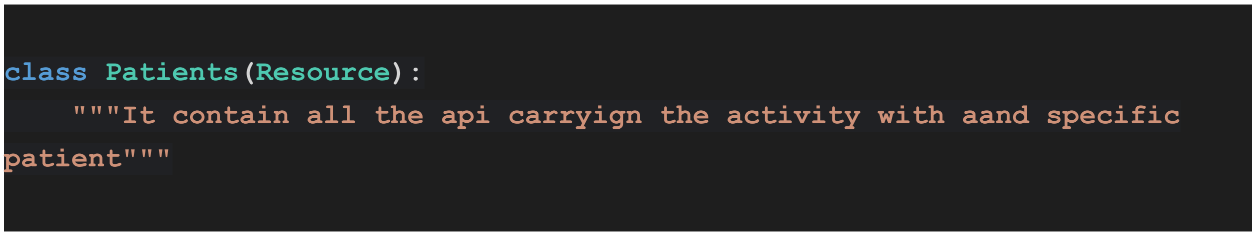
```
SELECT * FROM doctor;
```

doc_id	doc_first_name	doc_last_name	doc_ph_no	doc_date	doc_address
10	Dr Arhit	T	9999999999	2021-12-09 11:57:17	Blahblah
11	DrXXX	T	9009999999	2021-12-09 11:57:41	Blahblah APPT
12	WWW	WW	9910052839	2021-12-09 15:42:11	A 504 Amrapali Exotica E 08 Sector 50
13	EEE	HHH	09910052839	2021-12-09 16:28:29	A 504 Amrapali Exotica E 08 Sector 50



```
SELECT * FROM appointment;
```

app_id	pat_id	doc_id	appointment_date
12	17	11	2021-12-22 14:35:57
15	24	13	2021-12-23 19:35:33



```
class Patients(Resource):
    """It contain all the api carryign the activity with aand specific
patient"""
```

```
    def get(self):
        """Api to retive all the patient from the database"""

        patients = conn.execute("SELECT * FROM patient  ORDER BY pat_date
DESC").fetchall()
        return patients


    def post(self):
        """api to add the patient in the database"""

        patientInput = request.get_json(force=True)
        pat_first_name=patientInput['pat_first_name']
        pat_last_name = patientInput['pat_last_name']
        pat_insurance_no = patientInput['pat_insurance_no']
        pat_ph_no = patientInput['pat_ph_no']
        pat_address = patientInput['pat_address']
        patientInput['pat_id']=conn.execute('''INSERT INTO
patient(pat_first_name,pat_last_name,pat_insurance_no,pat_ph_no,pat_addres
s)
            VALUES(?,?,?,?,?)''', (pat_first_name, pat_last_name,
pat_insurance_no,pat_ph_no,pat_address)).lastrowid
        conn.commit()
        return patientInput

class Patient(Resource):
    """It contains all apis doing activity with the single patient
entity"""

    def get(self,id):
        """api to retrive details of the patient by it id"""

        patient = conn.execute("SELECT * FROM patient WHERE
pat_id=?",(id,)).fetchall()
        return patient

    def delete(self,id):
        """api to delete the patiend by its id"""

        conn.execute("DELETE FROM patient WHERE pat_id=?",(id,))
        conn.commit()
        return {'msg': 'sucessfully deleted'}

    def put(self,id):
        """api to update the patient by it id"""

        patientInput = request.get_json(force=True)
        pat_first_name = patientInput['pat_first_name']
        pat_last_name = patientInput['pat_last_name']
        pat_insurance_no = patientInput['pat_insurance_no']
        pat_ph_no = patientInput['pat_ph_no']
        pat_address = patientInput['pat_address']
        conn.execute("UPDATE patient SET
pat_first_name=?,pat_last_name=?,pat_insurance_no=?,pat_ph_no=?,pat_addres
s=? WHERE pat_id=?",
                     (pat_first_name, pat_last_name,
pat_insurance_no,pat_ph_no,pat_address,id))
        conn.commit()
        return patientInput
```

```
class Doctors(Resource):
    """This contain apis to carry out activity with all doctors"""

    def get(self):
        """Retrive list of all the doctor"""

        doctors = conn.execute("SELECT * FROM doctor ORDER BY doc_date
DESC").fetchall()
        return doctors



    def post(self):
        """Add the new doctor"""

        doctorInput = request.get_json(force=True)
        doc_first_name=doctorInput['doc_first_name']
        doc_last_name = doctorInput['doc_last_name']
        doc_ph_no = doctorInput['doc_ph_no']
        doc_address = doctorInput['doc_address']
        doctorInput['doc_id']=conn.execute('''INSERT INTO
doctor(doc_first_name,doc_last_name,doc_ph_no,doc_address)
            VALUES''', (doc_first_name,
doc_last_name,doc_ph_no,doc_address)).lastrowid
        conn.commit()
        return doctorInput

class Doctor(Resource):
    """It includes all the apis carrying out the activity with the single
doctor"""


    def get(self,id):
        """get the details of the docktor by the doctor id"""

        doctor = conn.execute("SELECT * FROM doctor WHERE
doc_id=?",(id,)).fetchall()
        return doctor

    def delete(self, id):
        """Delete the doctor by its id"""

        conn.execute("DELETE FROM doctor WHERE doc_id=?", (id,))
        conn.commit()
        return {'msg': 'successfully deleted'}

    def put(self,id):
        """Update the doctor by its id"""

        doctorInput = request.get_json(force=True)
        doc_first_name=doctorInput['doc_first_name']
        doc_last_name = doctorInput['doc_last_name']
        doc_ph_no = doctorInput['doc_ph_no']
        doc_address = doctorInput['doc_address']
        conn.execute(
            "UPDATE doctor SET
doc_first_name=?,doc_last_name=?,doc_ph_no=?,doc_address=? WHERE
doc_id=?",
            (doc_first_name, doc_last_name, doc_ph_no, doc_address, id))
        conn.commit()
        return doctorInput
```

```
class Appointment(Resource):
    """This contain all api doing activity with single appointment"""

    def get(self,id):
        """retrive a singe appointment details by its id"""

        appointment = conn.execute("SELECT * FROM appointment WHERE
app_id=?",(id,)).fetchall()
        return appointment


    def delete(self,id):
        """Delete teh appointment by its id"""

        conn.execute("DELETE FROM appointment WHERE app_id=?",(id,))
        conn.commit()
        return {'msg': 'sucessfully deleted'}

    def put(self,id):
        """Update the appointment details by the appointment id"""

        appointment = request.get_json(force=True)
        pat_id = appointment['pat_id']
        doc_id = appointment['doc_id']
        conn.execute("UPDATE appointment SET pat_id=?,doc_id=? WHERE
app_id=?",
                     (pat_id, doc_id, id))
        conn.commit()
        return appointment
```