Hospital Management System

22.12.2022

Certificate

Dec 22, 2022

We are pleased to present this Certificate of Achievement to Tanuku Chidroop, in recognition of their outstanding work on the project report on "Hospital Management System."

They have demonstrated exceptional dedication, hard work, and technical expertise in the completion of this project. Their research, analysis, and communication skills were of the highest caliber, and the final project report was a testament to their commitment to excellence.

We are proud to acknowledge their achievement and look forward to their continued success in the future.

Supervisor,

Ms. Inturi Anitha Rani

Assistant Professor

SRM University - AP

Acknowledgement

We would like to express our sincere gratitude to [Name of Instructor or Institution] for providing us with the opportunity to work on this project. Their guidance and support have been invaluable in helping us to complete this report.

We would also like to thank our seniors for their assistance and support. Without their help, this project would not have been possible.

Finally, we would like to thank our friends and family for their encouragement and support throughout the duration of this project.

Sincerely,

Tanuku Chidroop - AP20110010154

Abstract

Hospital Management System is an organized computerized system designed and programmed to deal with day to day operations and management of the hospital activities. The program can look after inpatients, outpatients, records, database treatments, status illness, billings in the pharmacy and labs. It also maintains hospital information such as ward id, doctors in charge and department administering. The major problem for the patient nowadays is to get a report after consultation. Many hospitals manage reports in their system but it's not available to the patient when he / she is outside. In this project we are going to provide the extra facility to store the report in the database and make it available from anywhere in the world.

Table of Contents

Introduction

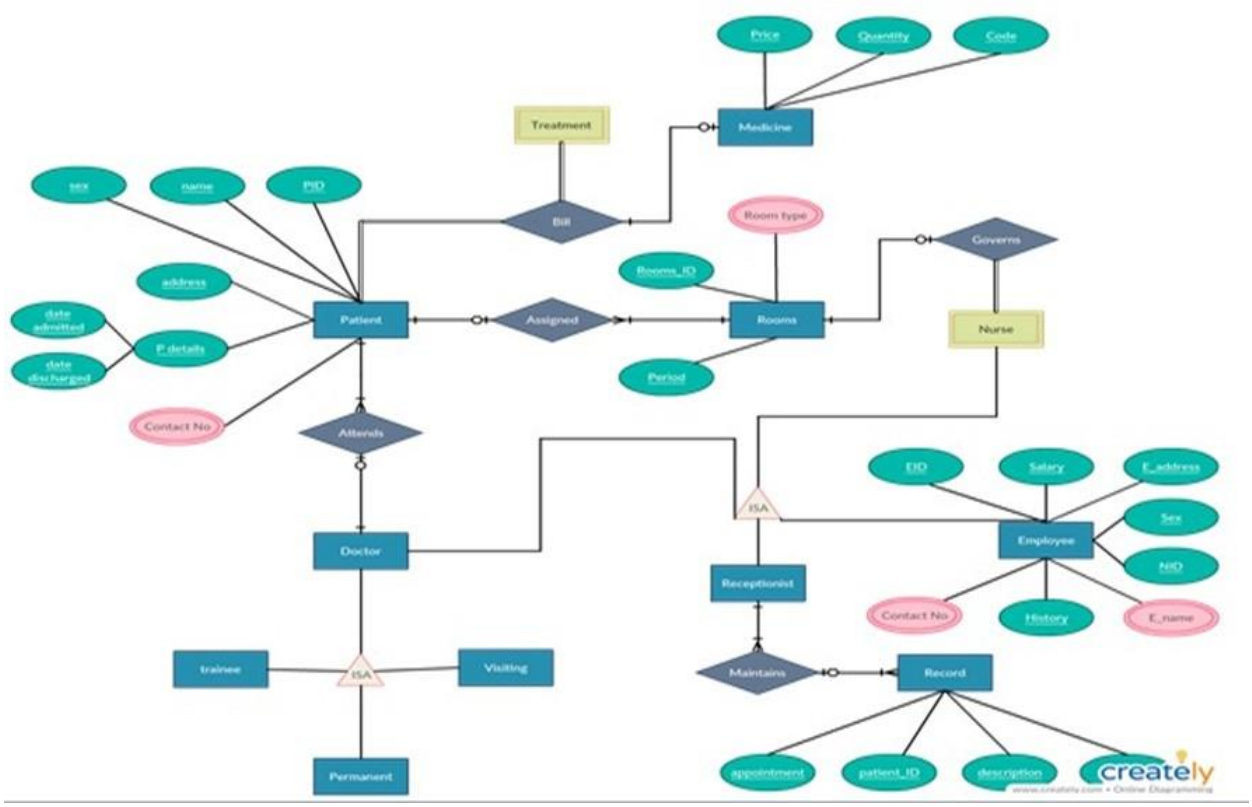
The project Hospital Management system includes registration of patients, storing their details into the system, and also computerized billing in the pharmacy, and labs. The software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically. It includes a search facility to know the current status of each room. User can search availability of a doctor and the details of a patient using the id.

The Hospital Management System can be entered using a username and password. It is accessible either by an administrator or receptionist. Only they can add data into the database. The data can be retrieved easily. The interface is very user-friendly. The data are well protected for personal use and makes the data processing very fast.

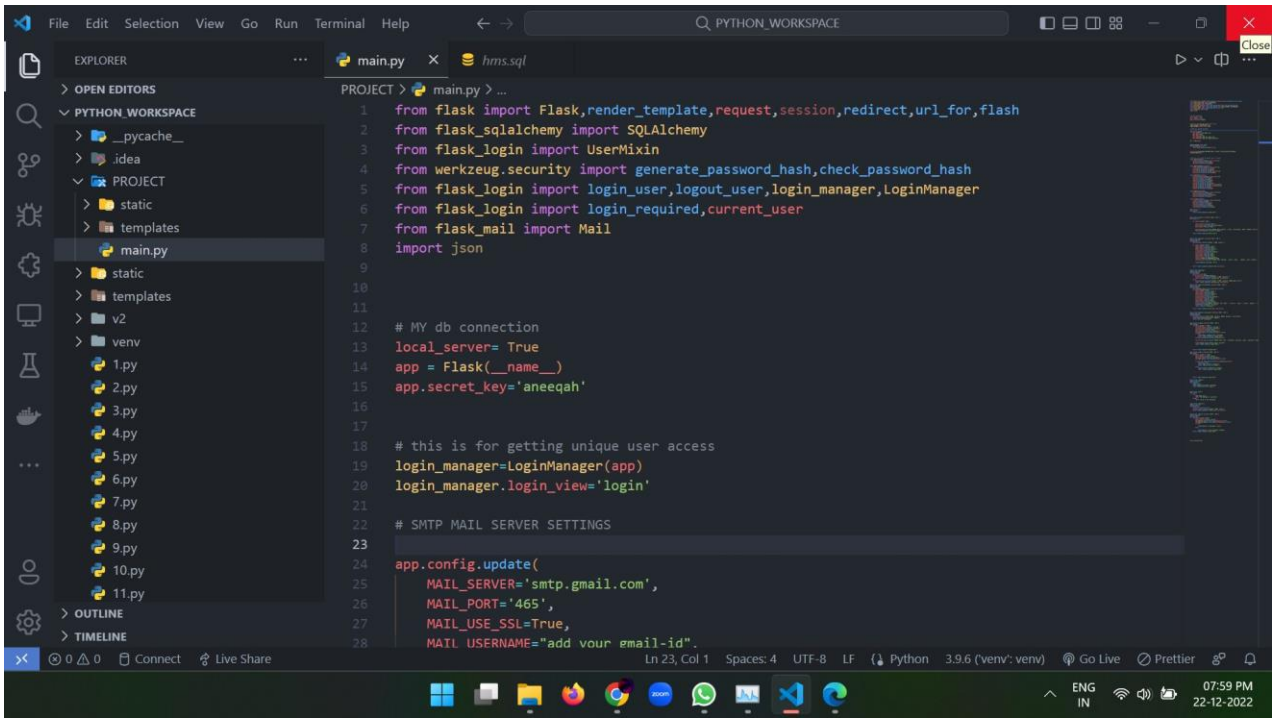
Hospital Management System is powerful, flexible, and easy to use and is designed and developed to deliver real conceivable benefits to hospitals. Hospital Management System is designed for multispeciality hospitals, to cover a wide range of hospital administration and management processes. It is an integrated end-to end Hospital Management System that provides relevant information across the hospital to support effective decision making for patient care, hospital administration and critical financial accounting, in a seamless flow.

Hospital Management System is a software product suite designed to improve the quality and management of hospital management in the areas of clinical process analysis and activity-based costing. Hospital Management System enables you to develop your organization and improve its effectiveness and quality of work. Managing the key processes efficiently is critical to the success of the hospital helps you manage your processes.

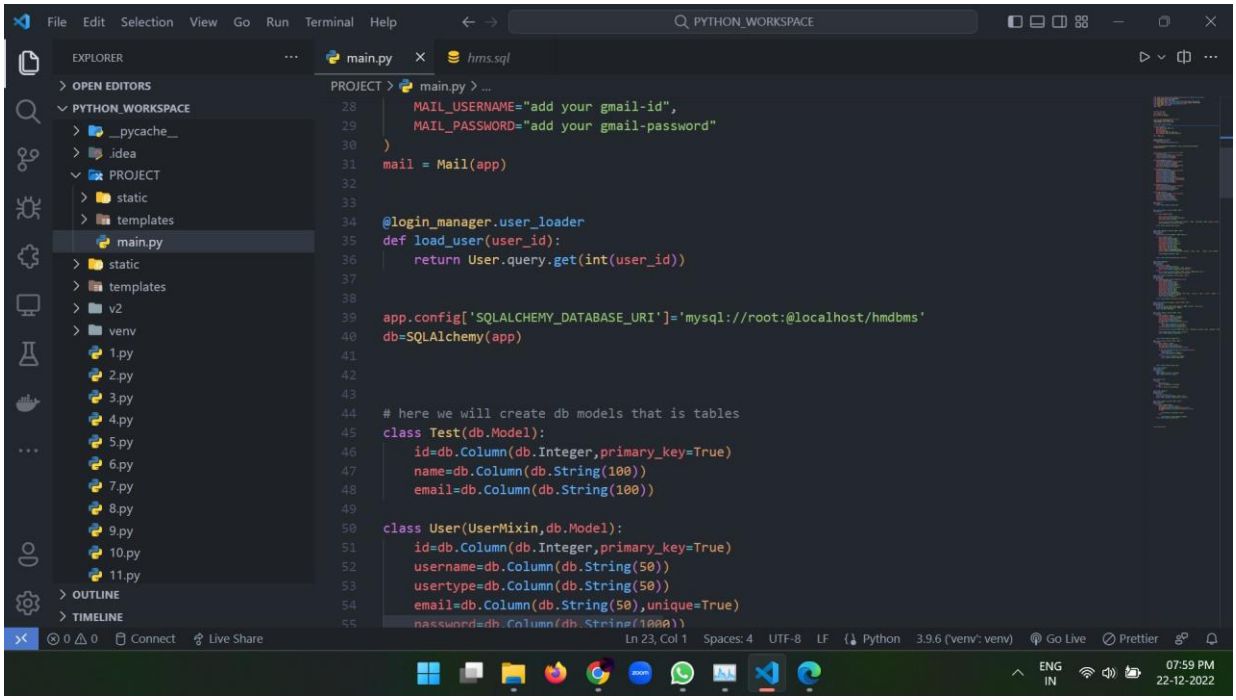
ER Diagram


Price
Quantity
Code
Medicine
Treatment
sex
name
PID
Bill
Room type
Governs
Rooms_ID
address
Patient
Assigned
Rooms
Nurse
date admitted
P.details
date discharged
Period
Contact No
Attends
EID
Salary
E_address
ISA
Sex
Employee
Doctor
Receptionist
NID
Contact No
History
E_name
trainee
ISA
Visiting
Maintains
Record
Permanent
appointment
patient_ID
description
creately

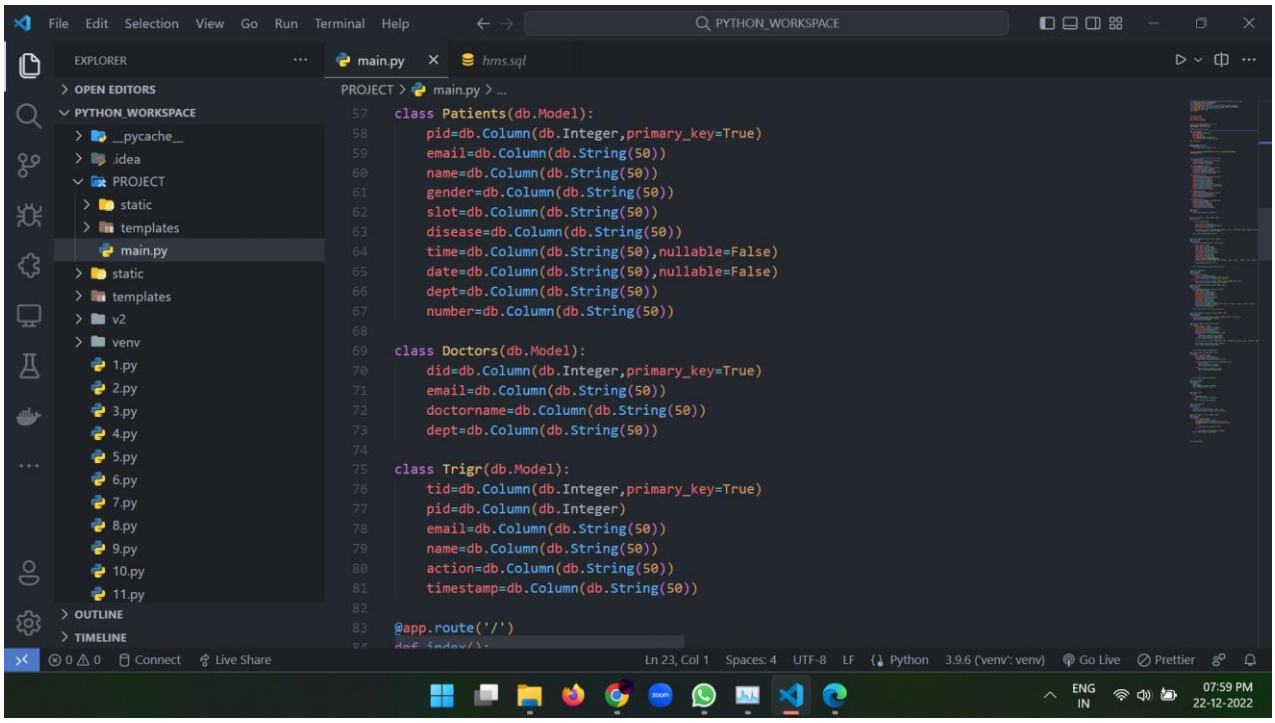
Code

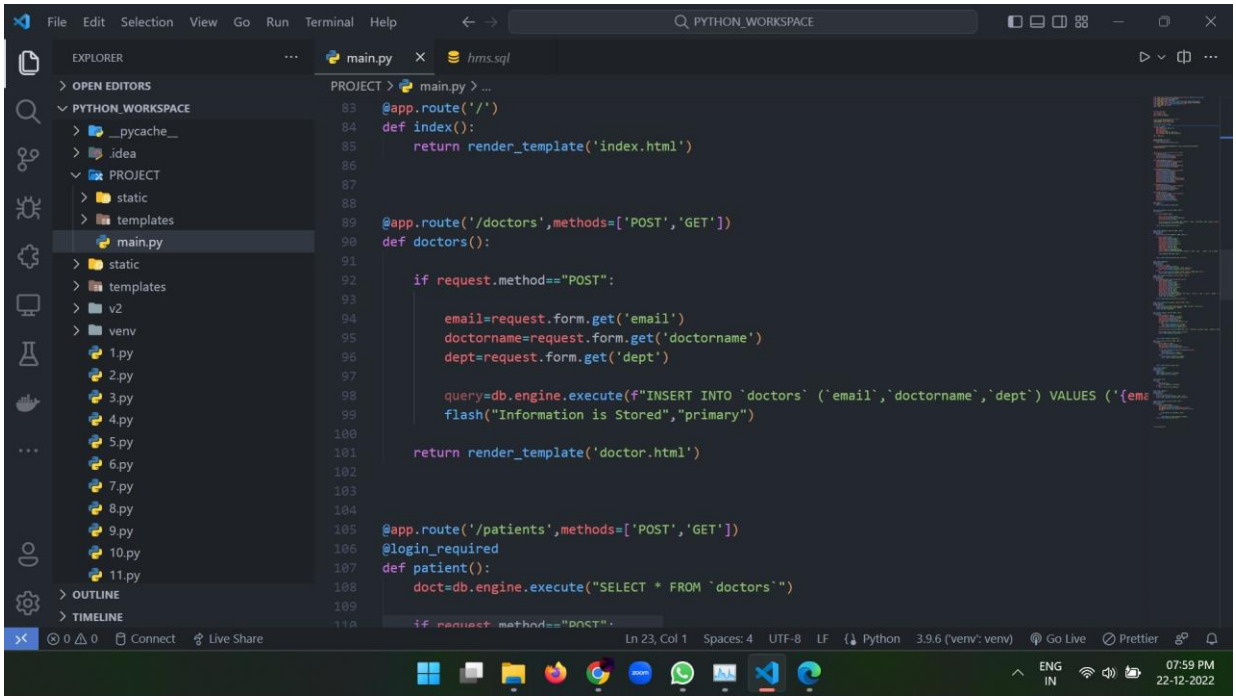


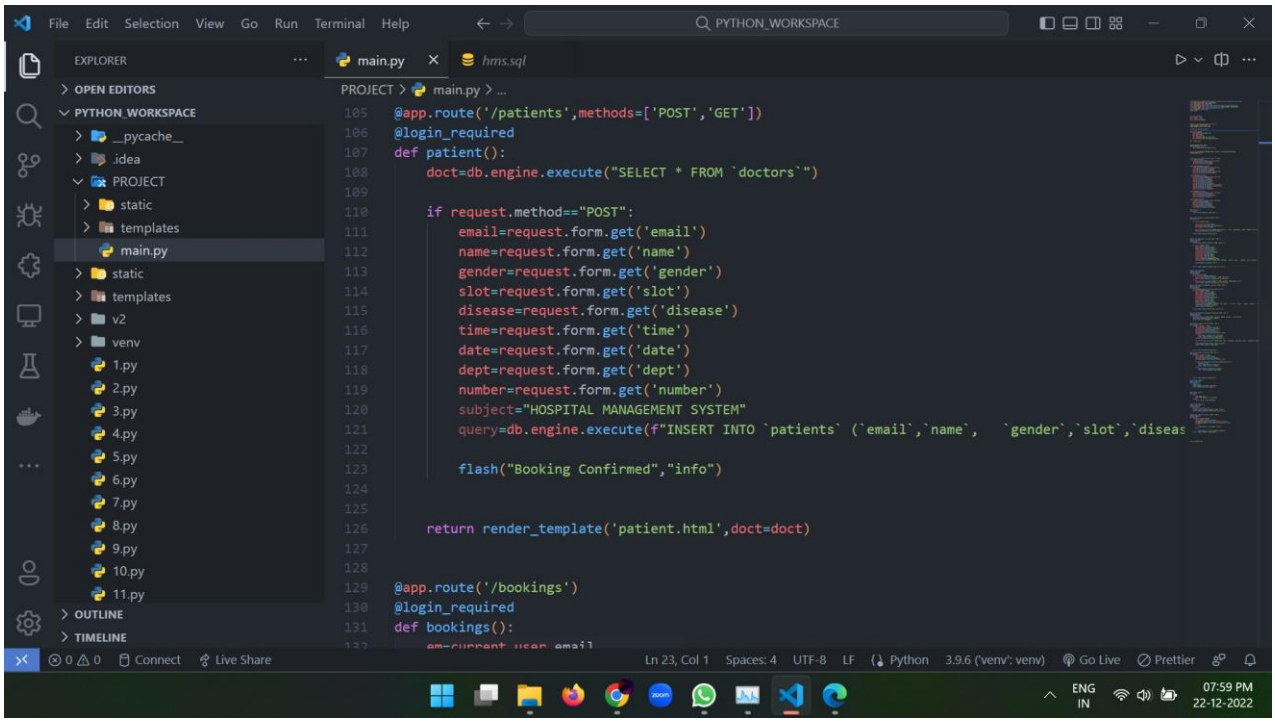
```
from flask import Flask,render_template,request,session,redirect,url_for,flash
from flask_sqlalchemy import SQLAlchemy
from flask_login import UserMixin
from werkzeug.security import generate_password_hash,check_password_hash
from flask_login import login_user,logout_user,login_manager,LoginManager
from flask_login import login_required,current_user
from flask_mail import Mail
import json



# MY db connection
local_server= True
app = Flask(__name__)
app.secret_key='aneeqah'


# this is for getting unique user access
login_manager=LoginManager(app)
login_manager.login_view='login'

# SMTP MAIL SERVER SETTINGS

app.config.update(
    MAIL_SERVER='smtp.gmail.com',
    MAIL_PORT='465',
    MAIL_USE_SSL=True,
    MAIL_USERNAME="add your gmail-id",
```

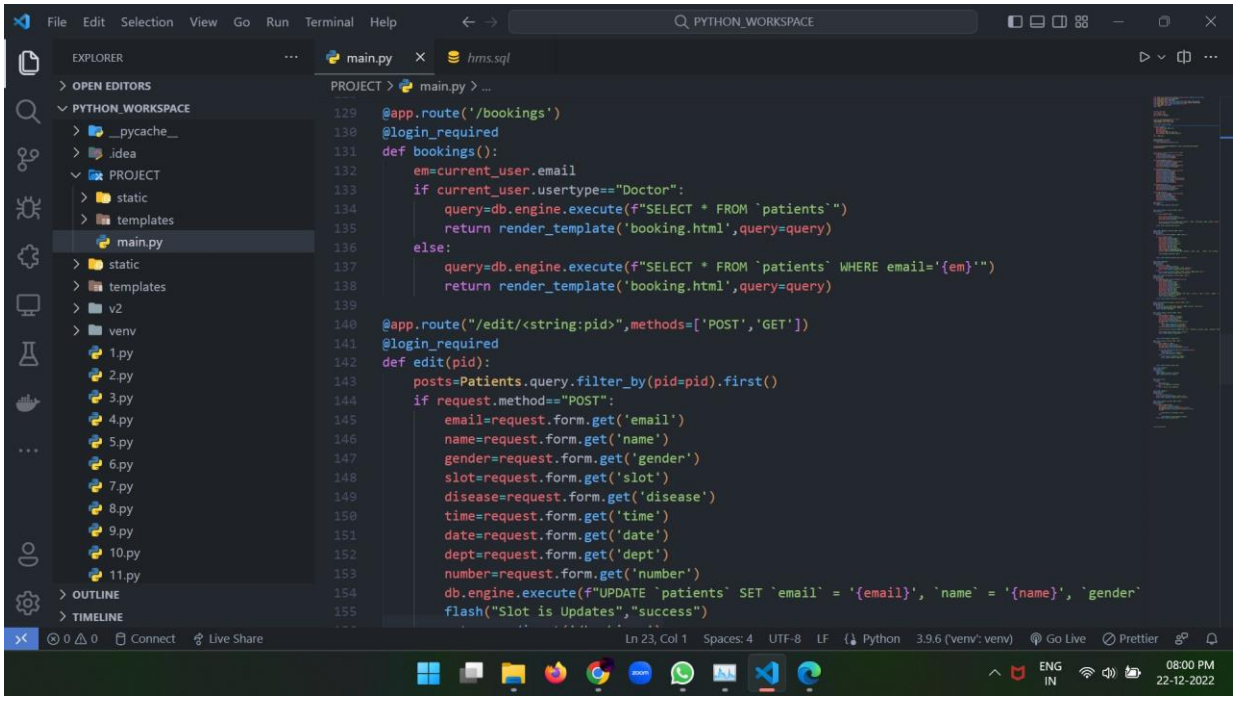


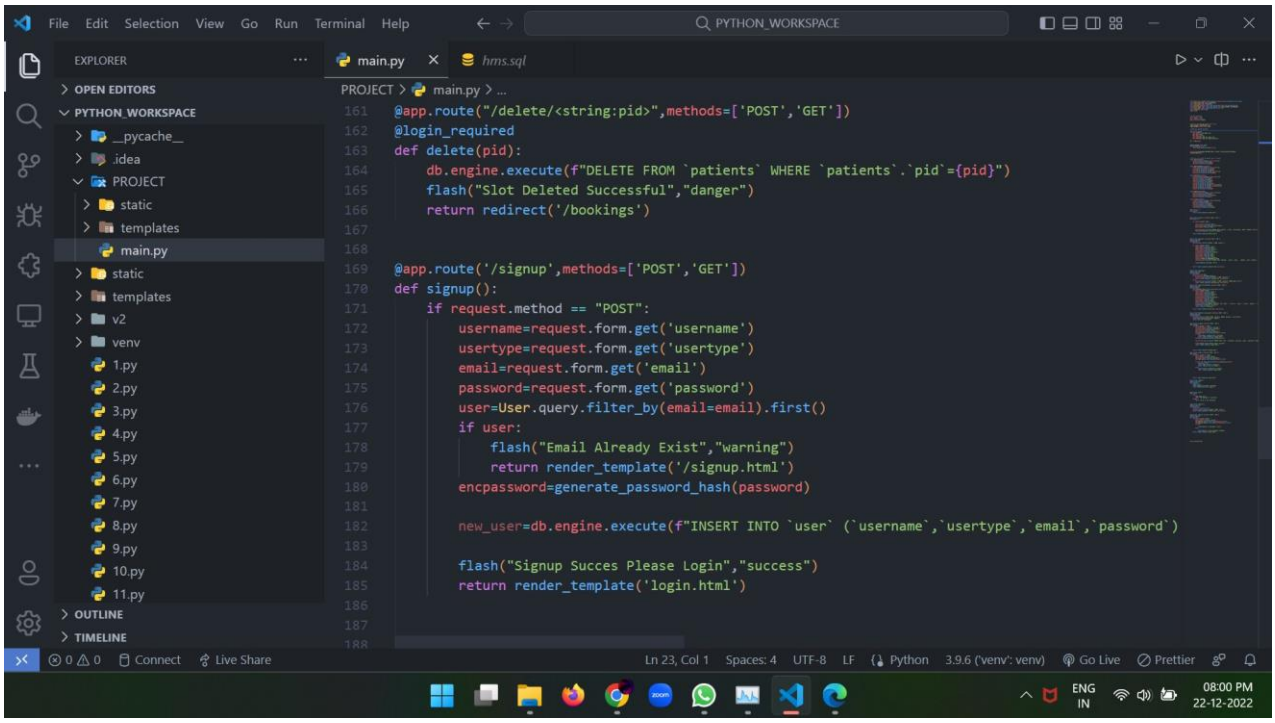
```
    MAIL_USERNAME="add your gmail-id",
    MAIL_PASSWORD="add your gmail-password"
)
mail = Mail(app)


@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))


app.config['SQLALCHEMY_DATABASE_URI']='mysql://root:@localhost/hmdbms'
db=SQLAlchemy(app)



# here we will create db models that is tables
class Test(db.Model):
    id=db.Column(db.Integer,primary_key=True)
    name=db.Column(db.String(100))
    email=db.Column(db.String(100))

class User(UserMixin,db.Model):
    id=db.Column(db.Integer,primary_key=True)
    username=db.Column(db.String(50))
    usertype=db.Column(db.String(50))
    email=db.Column(db.String(50),unique=True)
    password=db.Column(db.String(1000))
```


File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
PROJECT > main.py > ...
OPEN EDITORS
PYTHON_WORKSPACE
__pycache__
.idea
PROJECT
static
templates
main.py
static
templates
v2
venv
1.py
2.py
3.py
4.py
5.py
6.py
7.py
8.py
9.py
10.py
11.py
OUTLINE
TIMELINE
class Patients(db.Model):
pid=db.Column(db.Integer,primary_key=True)
email=db.Column(db.String(50))
name=db.Column(db.String(50))
gender=db.Column(db.String(50))
slot=db.Column(db.String(50))
disease=db.Column(db.String(50))
time=db.Column(db.String(50),nullable=False)
date=db.Column(db.String(50),nullable=False)
dept=db.Column(db.String(50))
number=db.Column(db.String(50))
class Doctors(db.Model):
did=db.Column(db.Integer,primary_key=True)
email=db.Column(db.String(50))
doctorname=db.Column(db.String(50))
dept=db.Column(db.String(50))
class Trigr(db.Model):
tid=db.Column(db.Integer,primary_key=True)
pid=db.Column(db.Integer)
email=db.Column(db.String(50))
name=db.Column(db.String(50))
action=db.Column(db.String(50))
timestamp=db.Column(db.String(50))
@app.route('/')
Connect
Live Share
Ln 23, Col 1
Spaces: 4
UTF-8
LF
Python
3.9.6 ('venv': venv)
Go Live
Prettier
ENG
IN
07:59 PM
22-12-2022

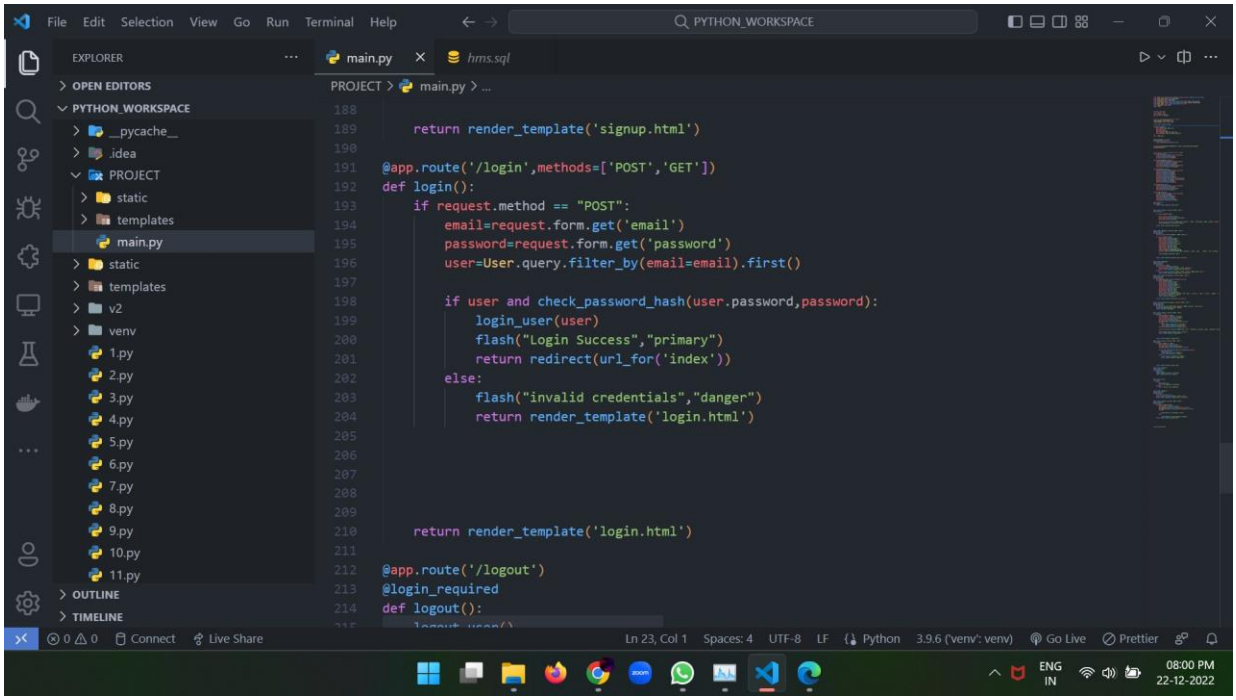

File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
PROJECT > main.py > ...
@app.route('/')
def index():
return render_template('index.html')
@app.route('/doctors',methods=['POST','GET'])
def doctors():
if request.method=="POST":
email=request.form.get('email')
doctorname=request.form.get('doctorname')
dept=request.form.get('dept')
query=db.engine.execute(f"INSERT INTO `doctors` (`email`,`doctorname`,`dept`) VALUES ('{ema
flash("Information is Stored","primary")
return render_template('doctor.html')
@app.route('/patients',methods=['POST','GET'])
@login_required
def patient():
doct=db.engine.execute("SELECT * FROM `doctors`")
Connect
Live Share
Ln 23, Col 1
Spaces: 4
UTF-8
LF
Python
3.9.6 ('venv': venv)
Go Live
Prettier
ENG
IN
07:59 PM
22-12-2022


File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
PROJECT > main.py > ...
OPEN EDITORS
PYTHON_WORKSPACE
__pycache__
.idea
PROJECT
static
templates
main.py
static
templates
v2
venv
1.py
2.py
3.py
4.py
5.py
6.py
7.py
8.py
9.py
10.py
11.py
OUTLINE
TIMELINE
@app.route('/patients',methods=['POST','GET'])
@login_required
def patient():
doct=db.engine.execute("SELECT * FROM `doctors`")
if request.method=="POST":
email=request.form.get('email')
name=request.form.get('name')
gender=request.form.get('gender')
slot=request.form.get('slot')
disease=request.form.get('disease')
time=request.form.get('time')
date=request.form.get('date')
dept=request.form.get('dept')
number=request.form.get('number')
subject="HOSPITAL MANAGEMENT SYSTEM"
query=db.engine.execute(f"INSERT INTO `patients` (`email`,`name`, `gender`,`slot`,`diseas
flash("Booking Confirmed","info")
return render_template('patient.html',doct=doct)
@app.route('/bookings')
@login_required
def bookings():
Ln 23, Col 1 Spaces: 4 UTF-8 LF Python 3.9.6 ('venv': venv) Go Live Prettier
ENG IN
07:59 PM 22-12-2022

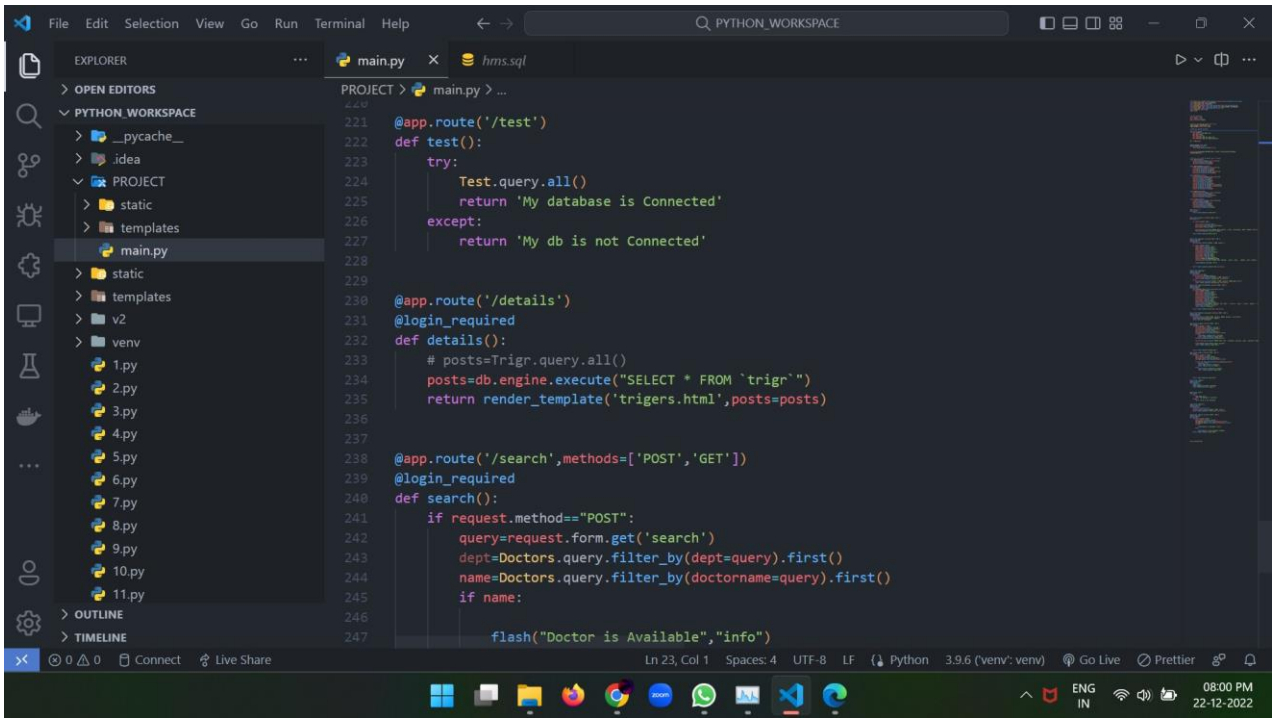

File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
PROJECT > main.py > ...
@app.route('/bookings')
@login_required
def bookings():
em=current_user.email
if current_user.usertype=="Doctor":
query=db.engine.execute(f"SELECT * FROM `patients`")
return render_template('booking.html',query=query)
else:
query=db.engine.execute(f"SELECT * FROM `patients` WHERE email='{em}'")
return render_template('booking.html',query=query)
@app.route("/edit/<string:pid>",methods=['POST','GET'])
@login_required
def edit(pid):
posts=Patients.query.filter_by(pid=pid).first()
if request.method=="POST":
email=request.form.get('email')
name=request.form.get('name')
gender=request.form.get('gender')
slot=request.form.get('slot')
disease=request.form.get('disease')
time=request.form.get('time')
date=request.form.get('date')
dept=request.form.get('dept')
number=request.form.get('number')
db.engine.execute(f"UPDATE `patients` SET `email` = '{email}', `name` = '{name}', `gender`
flash("Slot is Updates","success")
Ln 23, Col 1 Spaces: 4 UTF-8 LF Python 3.9.6 ('venv': venv) Go Live Prettier
ENG IN
08:00 PM 22-12-2022

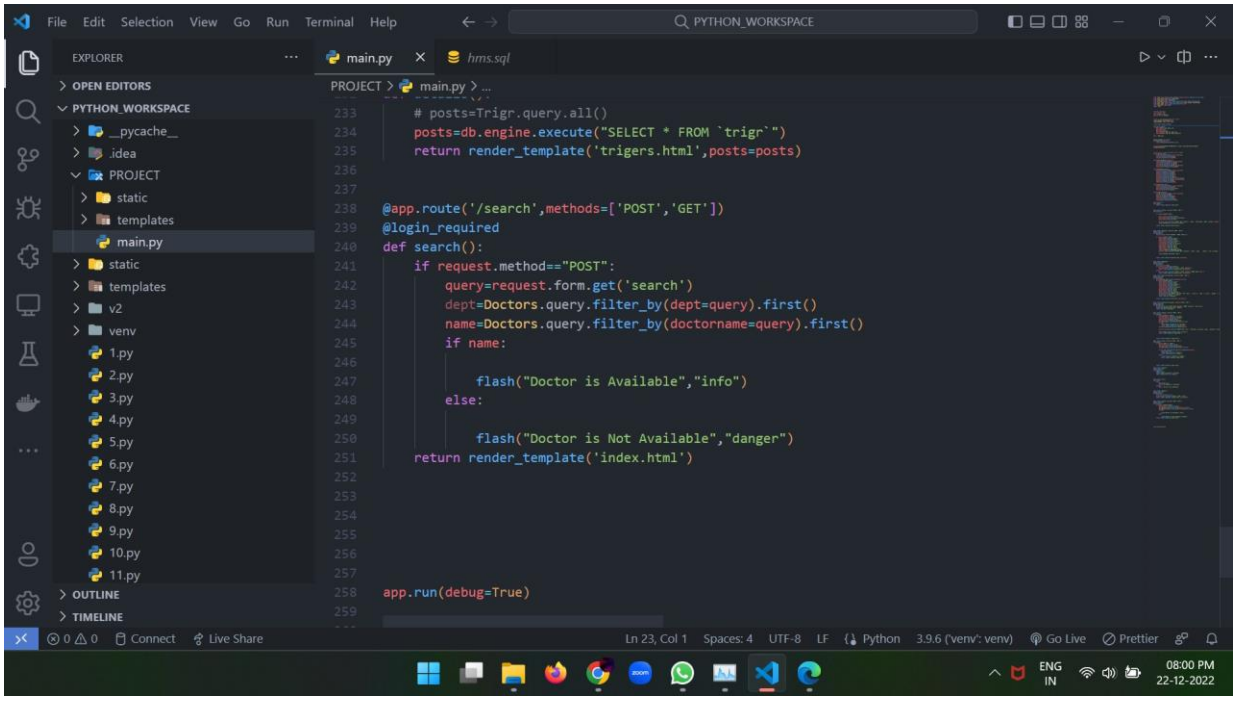


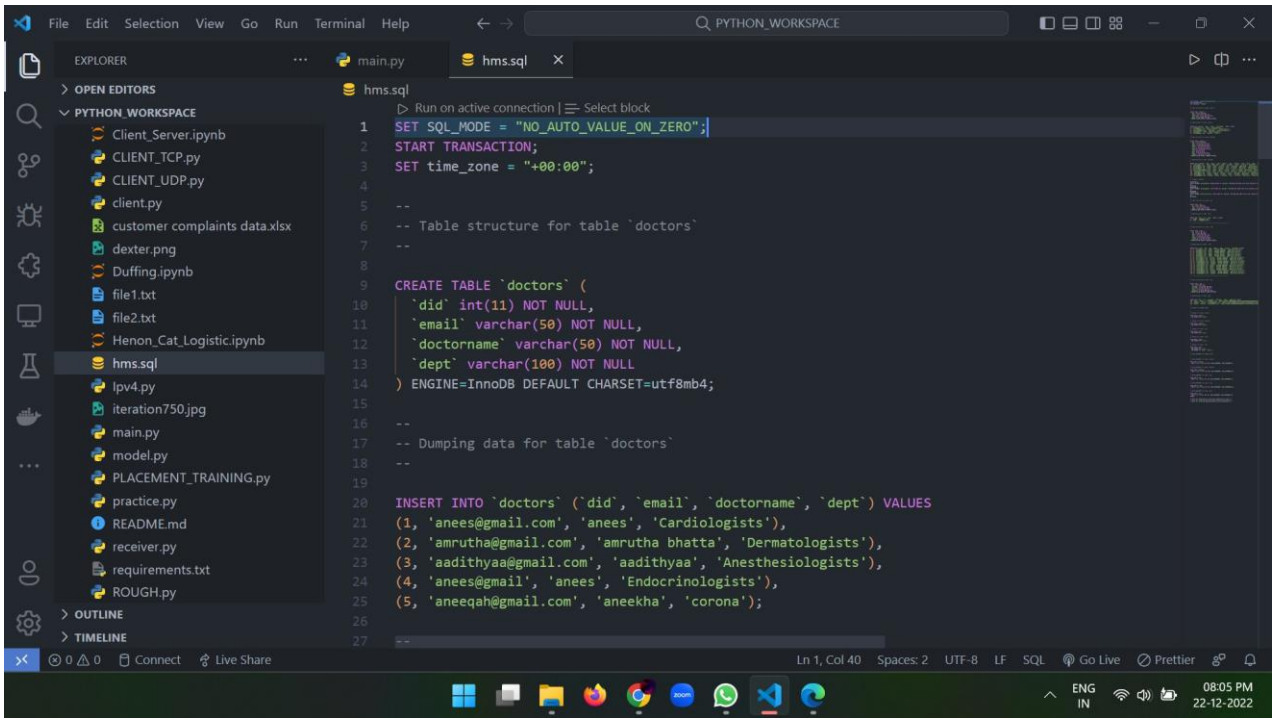
```
@app.route("/delete/<string:pid>",methods=['POST','GET'])
@login_required
def delete(pid):
    db.engine.execute(f"DELETE FROM `patients` WHERE `patients`.`pid`={pid}")
    flash("Slot Deleted Successful","danger")
    return redirect('/bookings')


@app.route('/signup',methods=['POST','GET'])
def signup():
    if request.method == "POST":
        username=request.form.get('username')
        usertype=request.form.get('usertype')
        email=request.form.get('email')
        password=request.form.get('password')
        user=User.query.filter_by(email=email).first()
        if user:
            flash("Email Already Exist","warning")
            return render_template('/signup.html')
        encpassword=generate_password_hash(password)

        new_user=db.engine.execute(f"INSERT INTO `user` (`username`,`usertype`,`email`,`password`)

        flash("Signup Succes Please Login","success")
        return render_template('login.html')
```

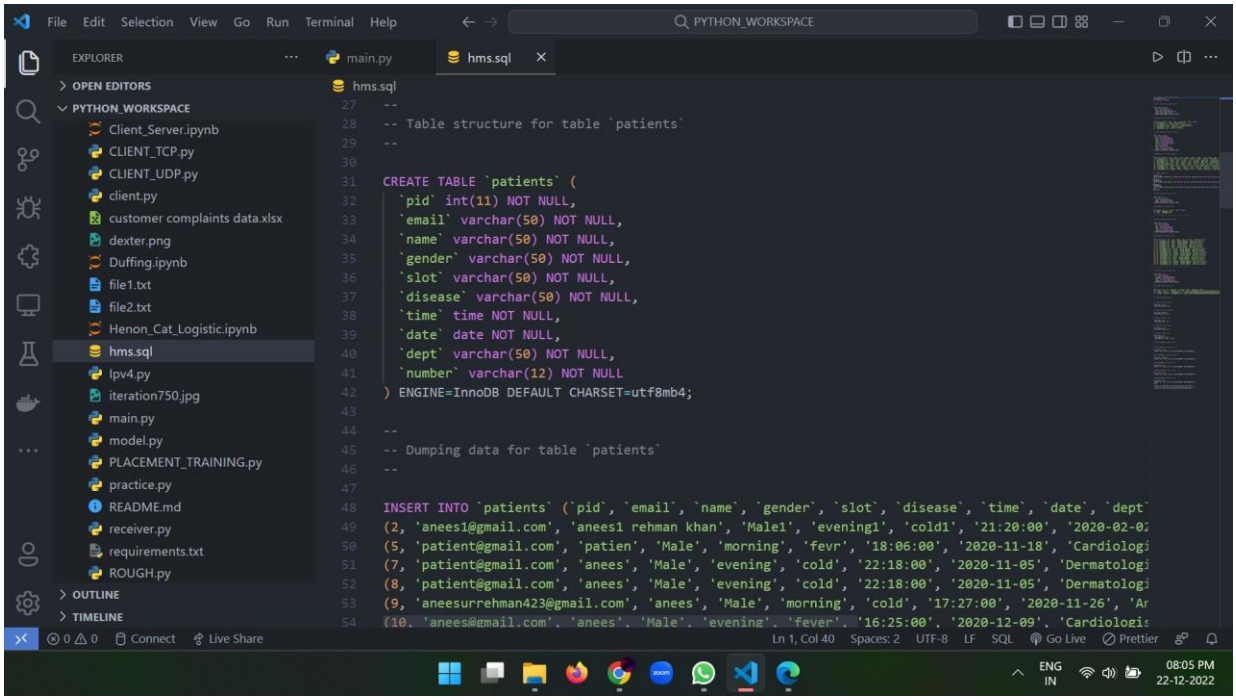


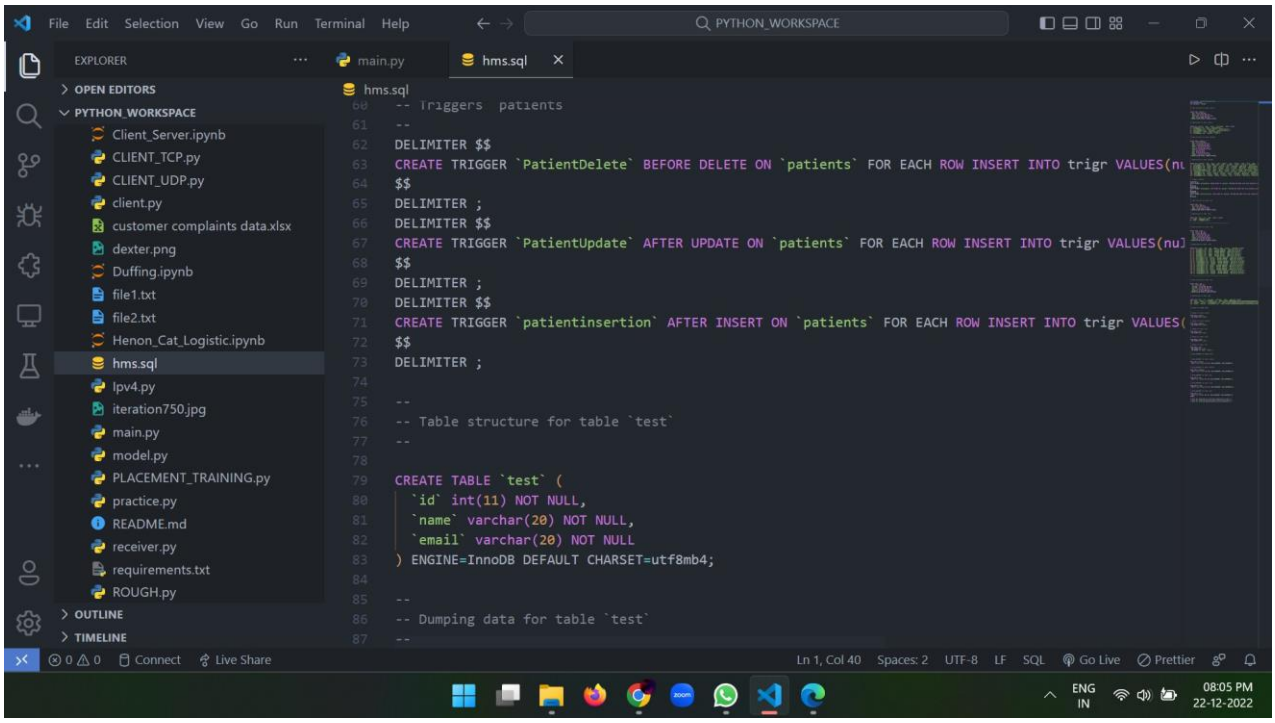
```
    return render_template('signup.html')

@app.route('/login',methods=['POST','GET'])
def login():
    if request.method == "POST":
        email=request.form.get('email')
        password=request.form.get('password')
        user=User.query.filter_by(email=email).first()

        if user and check_password_hash(user.password,password):
            login_user(user)
            flash("Login Success","primary")
            return redirect(url_for('index'))
        else:
            flash("invalid credentials","danger")
            return render_template('login.html')




    return render_template('login.html')

@app.route('/logout')
@login_required
def logout():
```

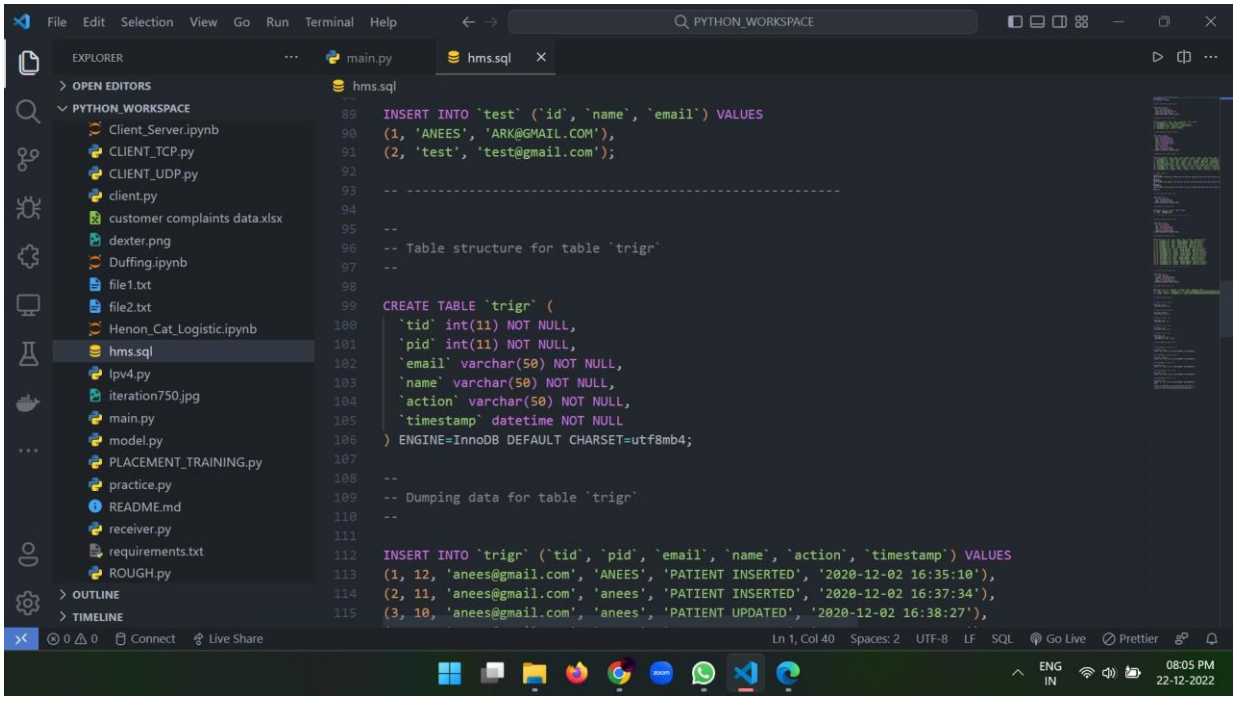

File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
PROJECT > main.py > ...
OPEN EDITORS
PYTHON_WORKSPACE
__pycache__
.idea
PROJECT
static
templates
main.py
static
templates
v2
venv
1.py
2.py
3.py
4.py
5.py
6.py
7.py
8.py
9.py
10.py
11.py
OUTLINE
TIMELINE
@app.route('/test')
def test():
try:
Test.query.all()
return 'My database is Connected'
except:
return 'My db is not Connected'
@app.route('/details')
@login_required
def details():
# posts=Trigr.query.all()
posts=db.engine.execute("SELECT * FROM `trigr`")
return render_template('trigers.html',posts=posts)
@app.route('/search',methods=['POST','GET'])
@login_required
def search():
if request.method=="POST":
query=request.form.get('search')
dept=Doctors.query.filter_by(dept=query).first()
name=Doctors.query.filter_by(doctorname=query).first()
if name:
flash("Doctor is Available","info")
Ln 23, Col 1 Spaces: 4 UTF-8 LF Python 3.9.6 ('venv': venv) Go Live Prettier
ENG IN
08:00 PM
22-12-2022

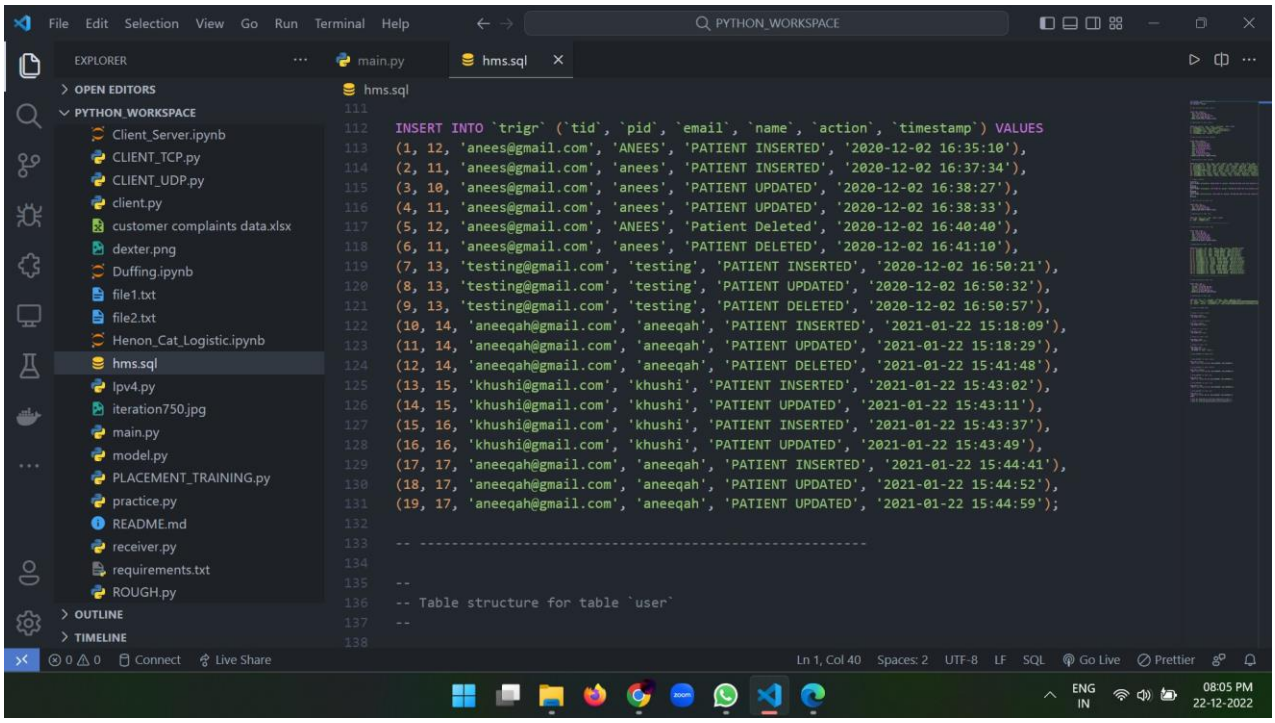

File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
PROJECT > main.py > ...
OPEN EDITORS
PYTHON_WORKSPACE
__pycache__
.idea
PROJECT
static
templates
main.py
static
templates
v2
venv
1.py
2.py
3.py
4.py
5.py
6.py
7.py
8.py
9.py
10.py
11.py
OUTLINE
TIMELINE
# posts=Trigr.query.all()
posts=db.engine.execute("SELECT * FROM `trigr`")
return render_template('trigers.html',posts=posts)
@app.route('/search',methods=['POST','GET'])
@login_required
def search():
if request.method=="POST":
query=request.form.get('search')
dept=Doctors.query.filter_by(dept=query).first()
name=Doctors.query.filter_by(doctorname=query).first()
if name:
flash("Doctor is Available","info")
else:
flash("Doctor is Not Available","danger")
return render_template('index.html')
app.run(debug=True)
Ln 23, Col 1 Spaces: 4 UTF-8 LF Python 3.9.6 ('venv': venv) Go Live Prettier
ENG IN
08:00 PM
22-12-2022

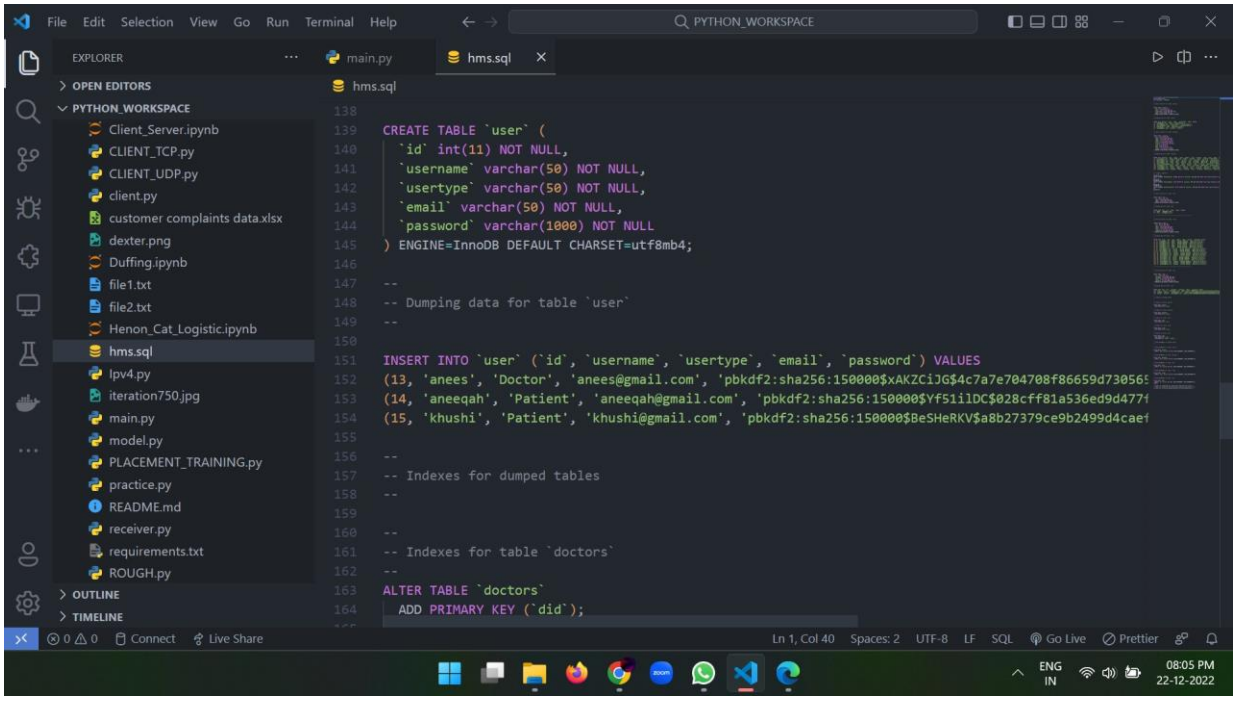

File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
OPEN EDITORS
PYTHON_WORKSPACE
Client_Server.ipynb
CLIENT_TCP.py
CLIENT_UDP.py
client.py
customer complaints data.xlsx
dexter.png
Duffing.ipynb
file1.txt
file2.txt
Henon_Cat_Logistic.ipynb
hms.sql
Ipv4.py
iteration750.jpg
main.py
model.py
PLACEMENT_TRAINING.py
practice.py
README.md
receiver.py
requirements.txt
ROUGH.py
OUTLINE
TIMELINE
Run on active connection | Select block
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";
--
-- Table structure for table `doctors`
--
CREATE TABLE `doctors` (
`did` int(11) NOT NULL,
`email` varchar(50) NOT NULL,
`doctorname` varchar(50) NOT NULL,
`dept` varchar(100) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `doctors`
--
INSERT INTO `doctors` (`did`, `email`, `doctorname`, `dept`) VALUES
(1, 'anees@gmail.com', 'anees', 'Cardiologists'),
(2, 'amrutha@gmail.com', 'amrutha bhatta', 'Dermatologists'),
(3, 'aadithyaa@gmail.com', 'aadithyaa', 'Anesthesiologists'),
(4, 'anees@gmail', 'anees', 'Endocrinologists'),
(5, 'aneeqah@gmail.com', 'aneekha', 'corona');
Ln 1, Col 40 Spaces: 2 UTF-8 LF SQL Go Live Prettier
Connect Live Share
ENG IN
08:05 PM 22-12-2022

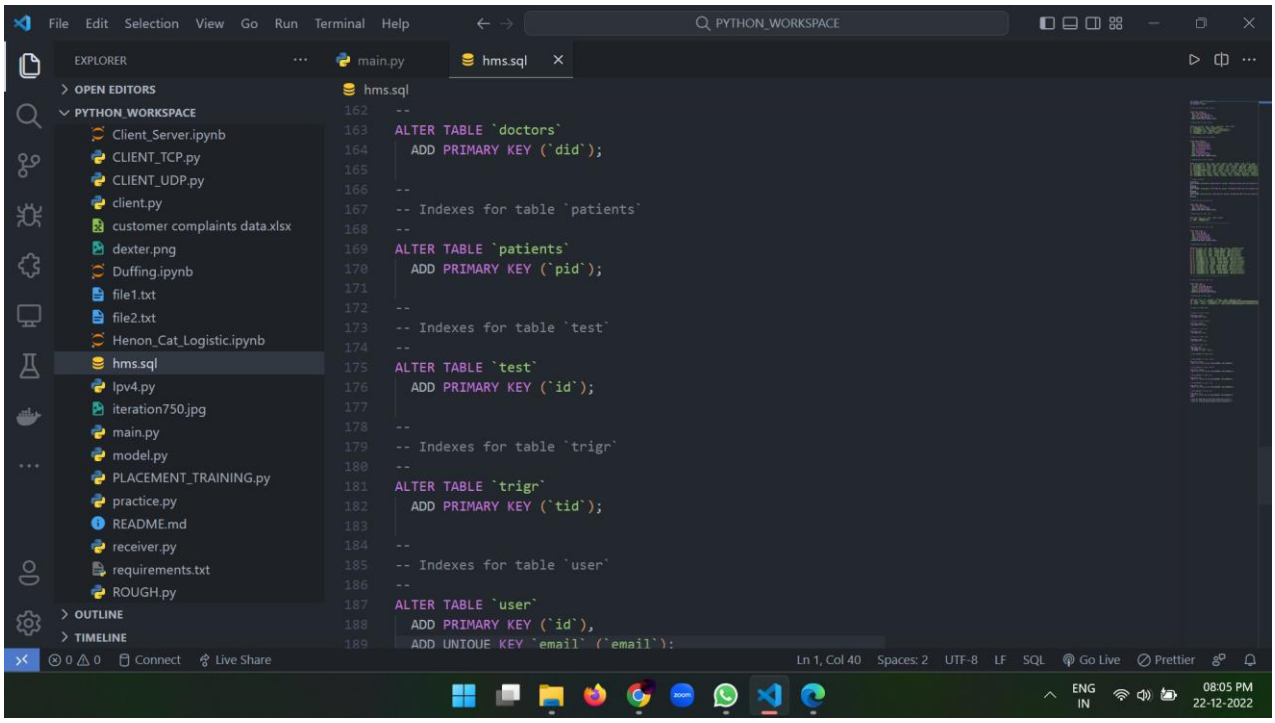

File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
OPEN EDITORS
PYTHON_WORKSPACE
Client_Server.ipynb
CLIENT_TCP.py
CLIENT_UDP.py
client.py
customer complaints data.xlsx
dexter.png
Duffing.ipynb
file1.txt
file2.txt
Henon_Cat_Logistic.ipynb
hms.sql
Ipv4.py
iteration750.jpg
main.py
model.py
PLACEMENT_TRAINING.py
practice.py
README.md
receiver.py
requirements.txt
ROUGH.py
OUTLINE
TIMELINE
--
-- Table structure for table `patients`
--
CREATE TABLE `patients` (
`pid` int(11) NOT NULL,
`email` varchar(50) NOT NULL,
`name` varchar(50) NOT NULL,
`gender` varchar(50) NOT NULL,
`slot` varchar(50) NOT NULL,
`disease` varchar(50) NOT NULL,
`time` time NOT NULL,
`date` date NOT NULL,
`dept` varchar(50) NOT NULL,
`number` varchar(12) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `patients`
--
INSERT INTO `patients` (`pid`, `email`, `name`, `gender`, `slot`, `disease`, `time`, `date`, `dept`
(2, 'anees1@gmail.com', 'anees1 rehman khan', 'Male1', 'evening1', 'cold1', '21:20:00', '2020-02-0
(5, 'patient@gmail.com', 'patien', 'Male', 'morning', 'fevr', '18:06:00', '2020-11-18', 'Cardiologi
(7, 'patient@gmail.com', 'anees', 'Male', 'evening', 'cold', '22:18:00', '2020-11-05', 'Dermatologi
(8, 'patient@gmail.com', 'anees', 'Male', 'evening', 'cold', '22:18:00', '2020-11-05', 'Dermatologi
(9, 'aneesurrehman423@gmail.com', 'anees', 'Male', 'morning', 'cold', '17:27:00', '2020-11-26', 'Ar
(10, 'anees@gmail.com', 'anees', 'Male', 'evening', 'fever', '16:25:00', '2020-12-09', 'Cardiologis
Ln 1, Col 40 Spaces: 2 UTF-8 LF SQL Go Live Prettier
Connect Live Share
ENG IN
08:05 PM 22-12-2022

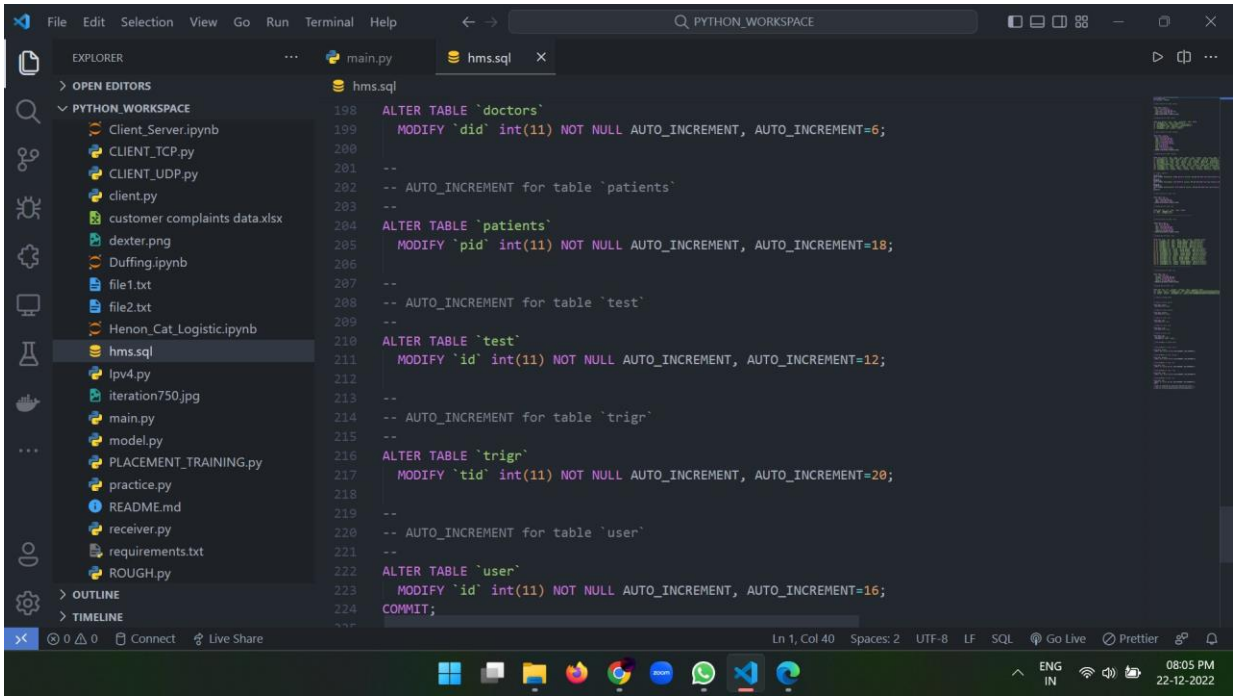

File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
OPEN EDITORS
PYTHON_WORKSPACE
Client_Server.ipynb
CLIENT_TCP.py
CLIENT_UDP.py
client.py
customer complaints data.xlsx
dexter.png
Duffing.ipynb
file1.txt
file2.txt
Henon_Cat_Logistic.ipynb
hms.sql
Ipv4.py
iteration750.jpg
main.py
model.py
PLACEMENT_TRAINING.py
practice.py
README.md
receiver.py
requirements.txt
ROUGH.py
OUTLINE
TIMELINE
-- Triggers patients
--
DELIMITER $$
CREATE TRIGGER `PatientDelete` BEFORE DELETE ON `patients` FOR EACH ROW INSERT INTO trigr VALUES(nu
$$
DELIMITER ;
DELIMITER $$
CREATE TRIGGER `PatientUpdate` AFTER UPDATE ON `patients` FOR EACH ROW INSERT INTO trigr VALUES(nu
$$
DELIMITER ;
DELIMITER $$
CREATE TRIGGER `patientinsertion` AFTER INSERT ON `patients` FOR EACH ROW INSERT INTO trigr VALUES(
$$
DELIMITER ;
--
-- Table structure for table `test`
--
CREATE TABLE `test` (
`id` int(11) NOT NULL,
`name` varchar(20) NOT NULL,
`email` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `test`
--
Connect
Live Share
Ln 1, Col 40
Spaces: 2
UTF-8
LF
SQL
Go Live
Prettier
ENG
IN
08:05 PM
22-12-2022


File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
OPEN EDITORS
PYTHON_WORKSPACE
Client_Server.ipynb
CLIENT_TCP.py
CLIENT_UDP.py
client.py
customer complaints data.xlsx
dexter.png
Duffing.ipynb
file1.txt
file2.txt
Henon_Cat_Logistic.ipynb
hms.sql
Ipv4.py
iteration750.jpg
main.py
model.py
PLACEMENT_TRAINING.py
practice.py
README.md
receiver.py
requirements.txt
ROUGH.py
OUTLINE
TIMELINE
INSERT INTO `test` (`id`, `name`, `email`) VALUES
(1, 'ANEES', 'ARK@GMAIL.COM'),
(2, 'test', 'test@gmail.com');
--
-- Table structure for table `trigr`
--
CREATE TABLE `trigr` (
`tid` int(11) NOT NULL,
`pid` int(11) NOT NULL,
`email` varchar(50) NOT NULL,
`name` varchar(50) NOT NULL,
`action` varchar(50) NOT NULL,
`timestamp` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `trigr`
--
INSERT INTO `trigr` (`tid`, `pid`, `email`, `name`, `action`, `timestamp`) VALUES
(1, 12, 'anees@gmail.com', 'ANEES', 'PATIENT INSERTED', '2020-12-02 16:35:10'),
(2, 11, 'anees@gmail.com', 'anees', 'PATIENT INSERTED', '2020-12-02 16:37:34'),
(3, 10, 'anees@gmail.com', 'anees', 'PATIENT UPDATED', '2020-12-02 16:38:27'),
Connect
Live Share
Ln 1, Col 40
Spaces: 2
UTF-8
LF
SQL
Go Live
Prettier
ENG
IN
08:05 PM
22-12-2022


File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
OPEN EDITORS
PYTHON_WORKSPACE
Client_Server.ipynb
CLIENT_TCP.py
CLIENT_UDP.py
client.py
customer complaints data.xlsx
dexter.png
Duffing.ipynb
file1.txt
file2.txt
Henon_Cat_Logistic.ipynb
hms.sql
Ipv4.py
iteration750.jpg
main.py
model.py
PLACEMENT_TRAINING.py
practice.py
README.md
receiver.py
requirements.txt
ROUGH.py
OUTLINE
TIMELINE
INSERT INTO `trigr` (`tid`, `pid`, `email`, `name`, `action`, `timestamp`) VALUES
(1, 12, 'anees@gmail.com', 'ANEES', 'PATIENT INSERTED', '2020-12-02 16:35:10'),
(2, 11, 'anees@gmail.com', 'anees', 'PATIENT INSERTED', '2020-12-02 16:37:34'),
(3, 10, 'anees@gmail.com', 'anees', 'PATIENT UPDATED', '2020-12-02 16:38:27'),
(4, 11, 'anees@gmail.com', 'anees', 'PATIENT UPDATED', '2020-12-02 16:38:33'),
(5, 12, 'anees@gmail.com', 'ANEES', 'Patient Deleted', '2020-12-02 16:40:40'),
(6, 11, 'anees@gmail.com', 'anees', 'PATIENT DELETED', '2020-12-02 16:41:10'),
(7, 13, 'testing@gmail.com', 'testing', 'PATIENT INSERTED', '2020-12-02 16:50:21'),
(8, 13, 'testing@gmail.com', 'testing', 'PATIENT UPDATED', '2020-12-02 16:50:32'),
(9, 13, 'testing@gmail.com', 'testing', 'PATIENT DELETED', '2020-12-02 16:50:57'),
(10, 14, 'aneeqah@gmail.com', 'aneeqah', 'PATIENT INSERTED', '2021-01-22 15:18:09'),
(11, 14, 'aneeqah@gmail.com', 'aneeqah', 'PATIENT UPDATED', '2021-01-22 15:18:29'),
(12, 14, 'aneeqah@gmail.com', 'aneeqah', 'PATIENT DELETED', '2021-01-22 15:41:48'),
(13, 15, 'khushi@gmail.com', 'khushi', 'PATIENT INSERTED', '2021-01-22 15:43:02'),
(14, 15, 'khushi@gmail.com', 'khushi', 'PATIENT UPDATED', '2021-01-22 15:43:11'),
(15, 16, 'khushi@gmail.com', 'khushi', 'PATIENT INSERTED', '2021-01-22 15:43:37'),
(16, 16, 'khushi@gmail.com', 'khushi', 'PATIENT UPDATED', '2021-01-22 15:43:49'),
(17, 17, 'aneeqah@gmail.com', 'aneeqah', 'PATIENT INSERTED', '2021-01-22 15:44:41'),
(18, 17, 'aneeqah@gmail.com', 'aneeqah', 'PATIENT UPDATED', '2021-01-22 15:44:52'),
(19, 17, 'aneeqah@gmail.com', 'aneeqah', 'PATIENT UPDATED', '2021-01-22 15:44:59');
-- --
--
-- Table structure for table `user`
--
Ln 1, Col 40 Spaces: 2 UTF-8 LF SQL Go Live Prettier
ENG IN
08:05 PM 22-12-2022

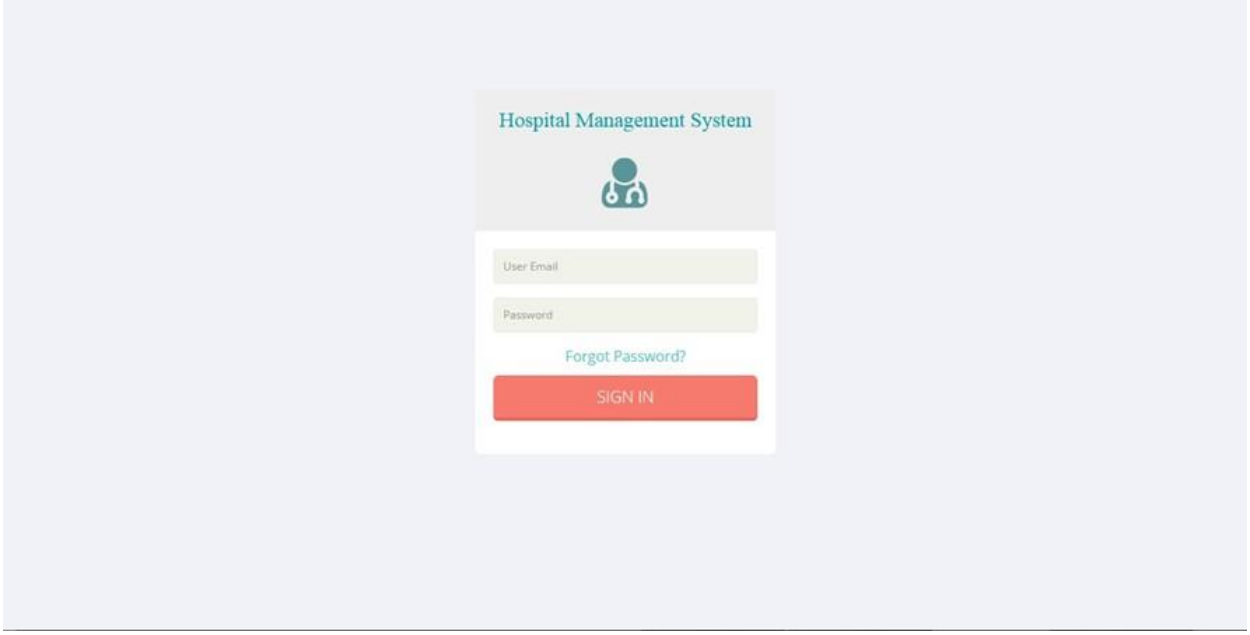

File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
OPEN EDITORS
PYTHON_WORKSPACE
Client_Server.ipynb
CLIENT_TCP.py
CLIENT_UDP.py
client.py
customer complaints data.xlsx
dexter.png
Duffing.ipynb
file1.txt
file2.txt
Henon_Cat_Logistic.ipynb
hms.sql
Ipv4.py
iteration750.jpg
main.py
model.py
PLACEMENT_TRAINING.py
practice.py
README.md
receiver.py
requirements.txt
ROUGH.py
OUTLINE
TIMELINE
CREATE TABLE `user` (
`id` int(11) NOT NULL,
`username` varchar(50) NOT NULL,
`usertype` varchar(50) NOT NULL,
`email` varchar(50) NOT NULL,
`password` varchar(1000) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `user`
--
INSERT INTO `user` (`id`, `username`, `usertype`, `email`, `password`) VALUES
(13, 'anees', 'Doctor', 'anees@gmail.com', 'pbkdf2:sha256:150000$xAKZCiJG$4c7a7e704708f86659d730565
(14, 'aneeqah', 'Patient', 'aneeqah@gmail.com', 'pbkdf2:sha256:150000$Yf51ilDC$028cff81a536ed9d477f
(15, 'khushi', 'Patient', 'khushi@gmail.com', 'pbkdf2:sha256:150000$BeSHeRKV$a8b27379ce9b2499d4caef
--
-- Indexes for dumped tables
--
--
-- Indexes for table `doctors`
--
ALTER TABLE `doctors`
ADD PRIMARY KEY (`did`);
Ln 1, Col 40 Spaces: 2 UTF-8 LF SQL Go Live Prettier
ENG IN
08:05 PM 22-12-2022


File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
OPEN EDITORS
PYTHON_WORKSPACE
Client_Server.ipynb
CLIENT_TCP.py
CLIENT_UDP.py
client.py
customer complaints data.xlsx
dexter.png
Duffing.ipynb
file1.txt
file2.txt
Henon_Cat_Logistic.ipynb
hms.sql
Ipv4.py
iteration750.jpg
main.py
model.py
PLACEMENT_TRAINING.py
practice.py
README.md
receiver.py
requirements.txt
ROUGH.py
OUTLINE
TIMELINE
-- ALTER TABLE `doctors` ADD PRIMARY KEY (`did`);
-- Indexes for table `patients`
-- ALTER TABLE `patients` ADD PRIMARY KEY (`pid`);
-- Indexes for table `test`
-- ALTER TABLE `test` ADD PRIMARY KEY (`id`);
-- Indexes for table `trigr`
-- ALTER TABLE `trigr` ADD PRIMARY KEY (`tid`);
-- Indexes for table `user`
-- ALTER TABLE `user` ADD PRIMARY KEY (`id`), ADD UNIQUE KEY `email` (`email`);
Connect
Live Share
Ln 1, Col 40
Spaces: 2
UTF-8
LF
SQL
Go Live
Prettier
ENG
IN
08:05 PM
22-12-2022

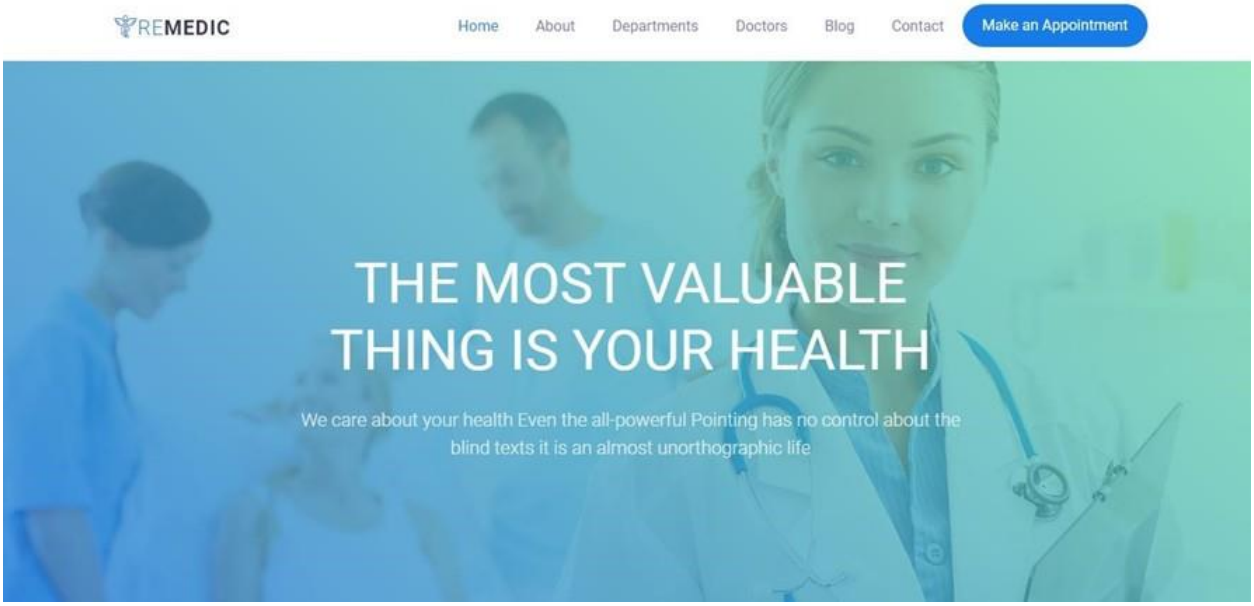

File Edit Selection View Go Run Terminal Help
PYTHON_WORKSPACE
EXPLORER
main.py
hms.sql
OPEN EDITORS
PYTHON_WORKSPACE
Client_Server.ipynb
CLIENT_TCP.py
CLIENT_UDP.py
client.py
customer complaints data.xlsx
dexter.png
Duffing.ipynb
file1.txt
file2.txt
Henon_Cat_Logistic.ipynb
hms.sql
Ipv4.py
iteration750.jpg
main.py
model.py
PLACEMENT_TRAINING.py
practice.py
README.md
receiver.py
requirements.txt
ROUGH.py
OUTLINE
TIMELINE
ALTER TABLE `doctors` MODIFY `did` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=6;
-- AUTO_INCREMENT for table `patients`
ALTER TABLE `patients` MODIFY `pid` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=18;
-- AUTO_INCREMENT for table `test`
ALTER TABLE `test` MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=12;
-- AUTO_INCREMENT for table `trigr`
ALTER TABLE `trigr` MODIFY `tid` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=20;
-- AUTO_INCREMENT for table `user`
ALTER TABLE `user` MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=16;
COMMIT;
Connect
Live Share
Ln 1, Col 40
Spaces: 2
UTF-8
LF
SQL
Go Live
Prettier
ENG
IN
08:05 PM
22-12-2022

Site Snapshots

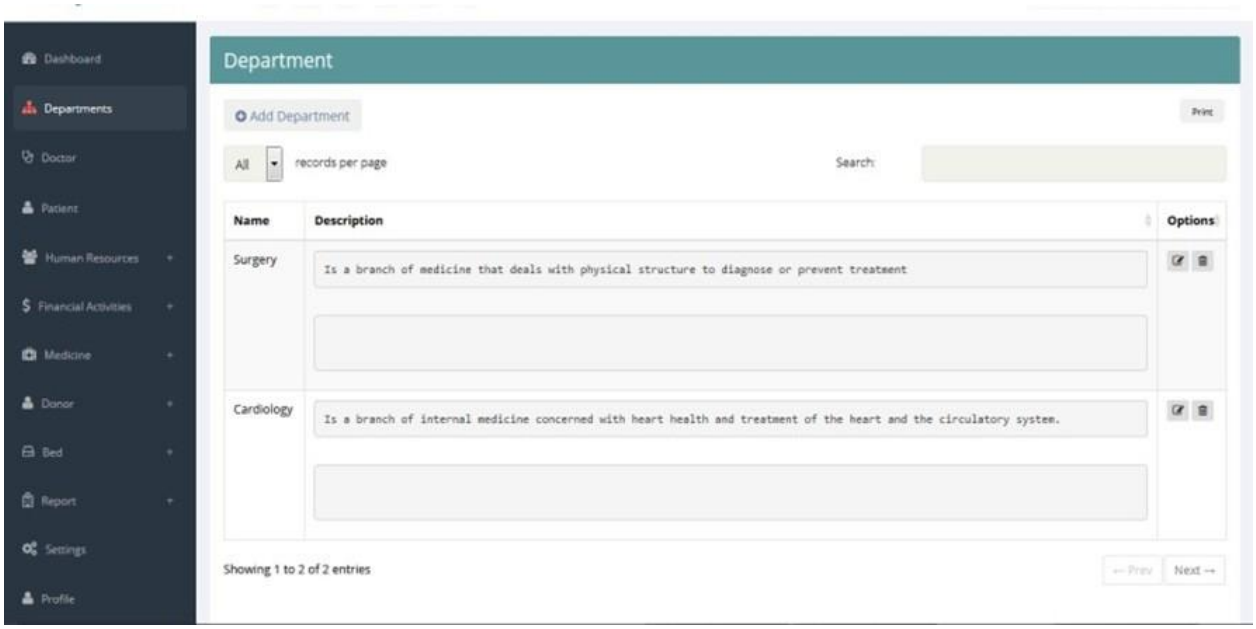
LOGIN PAGE



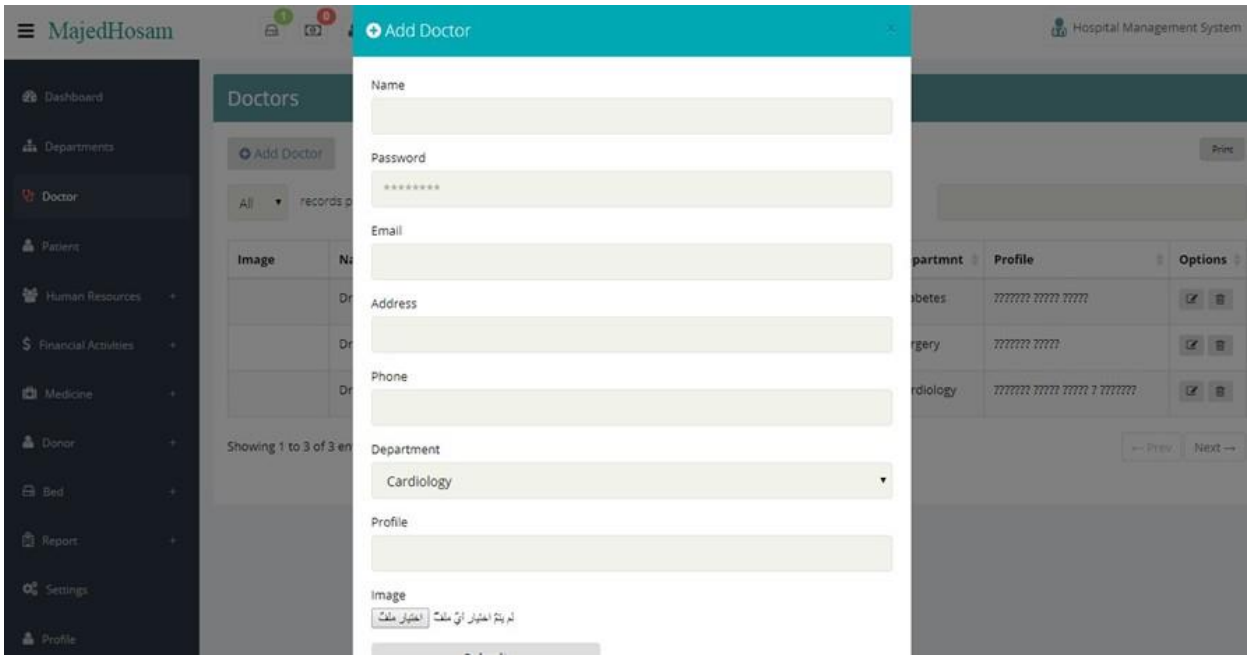
MAIN PAGE



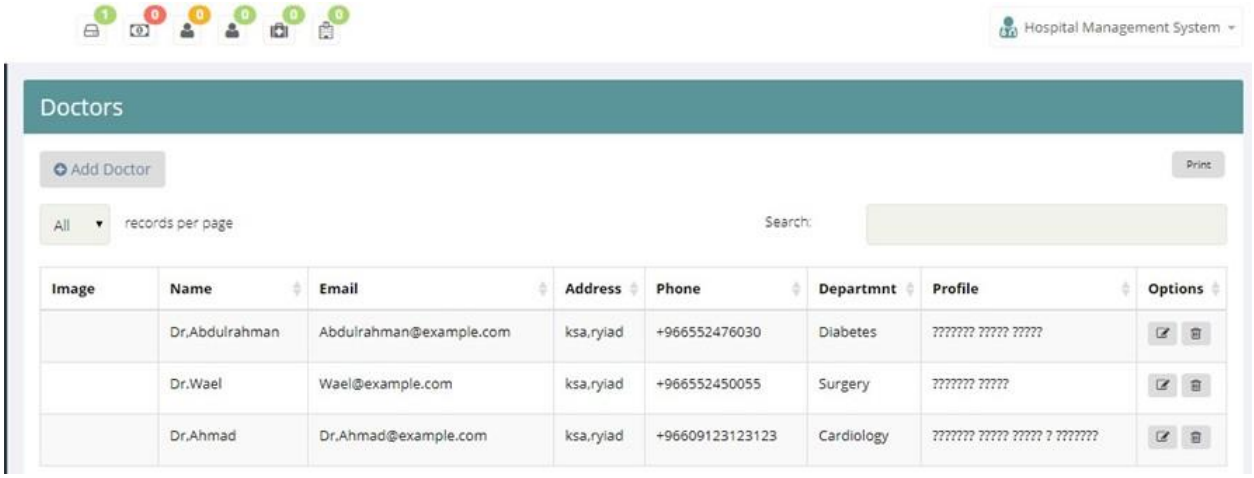
DEPARTMENTS



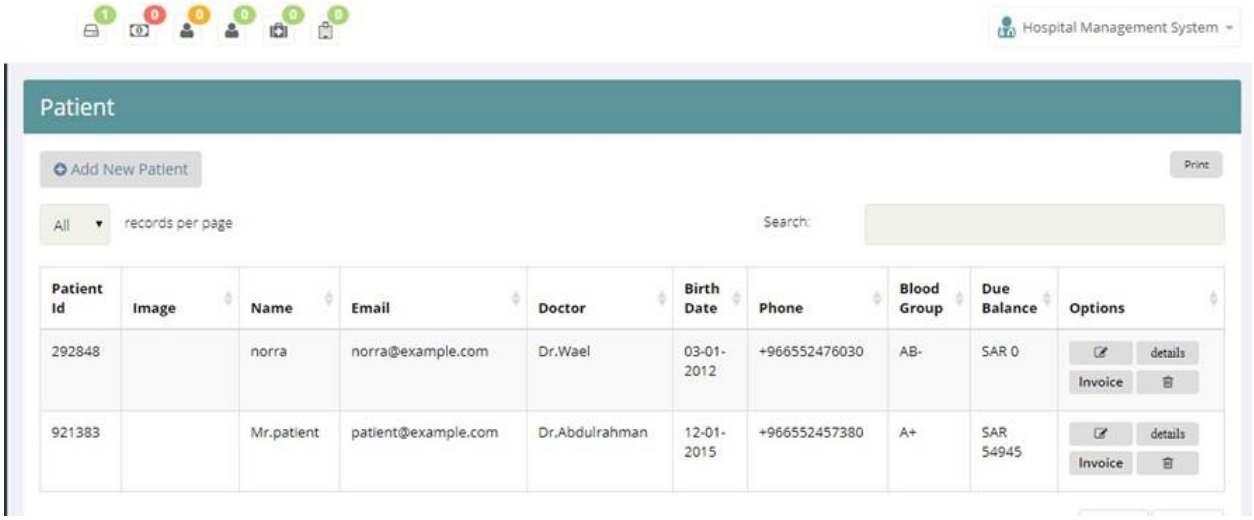
NEW EMPLOYEE DOCTOR



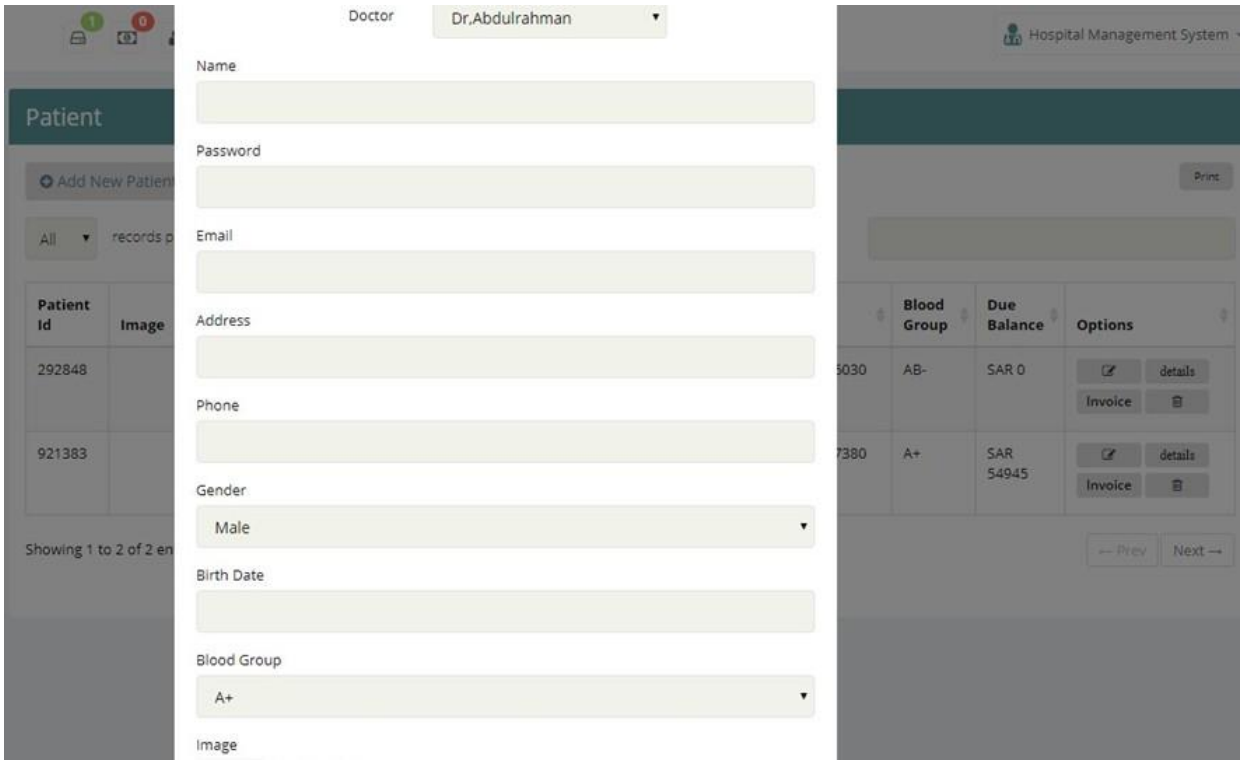
DOCTOR MODULE



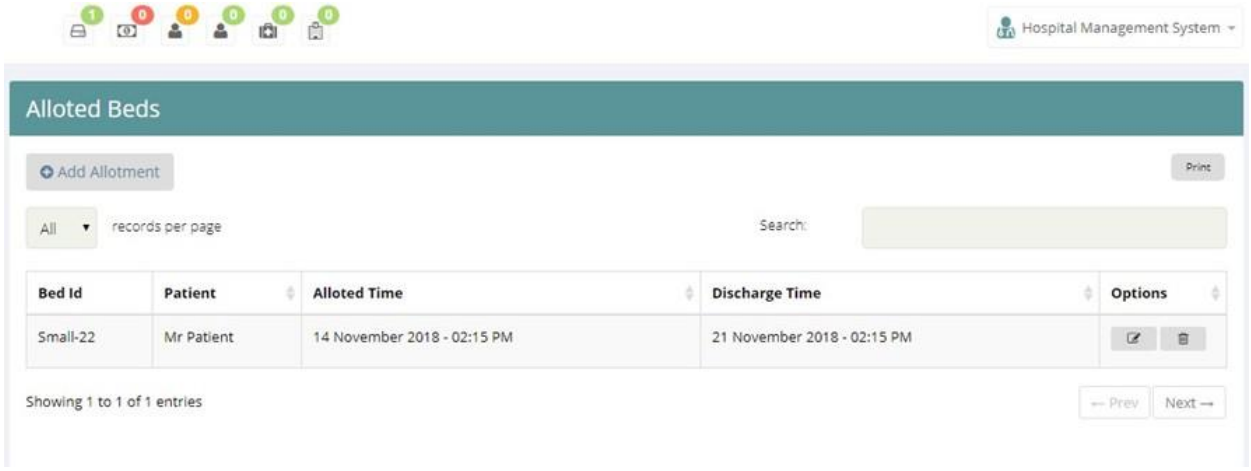
OPERATION DETAILS



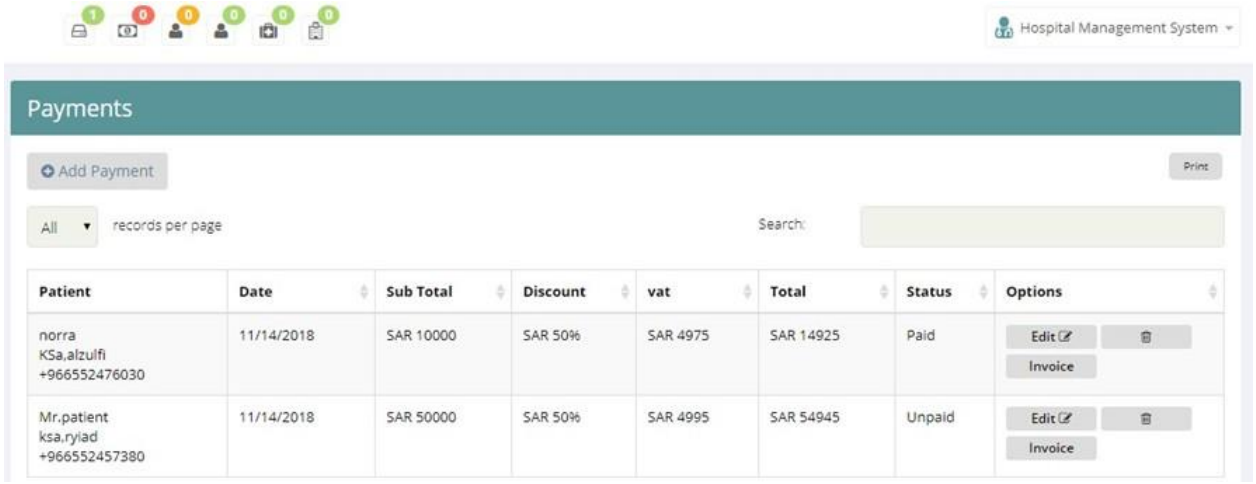
CREATE OPERATION DETAILS



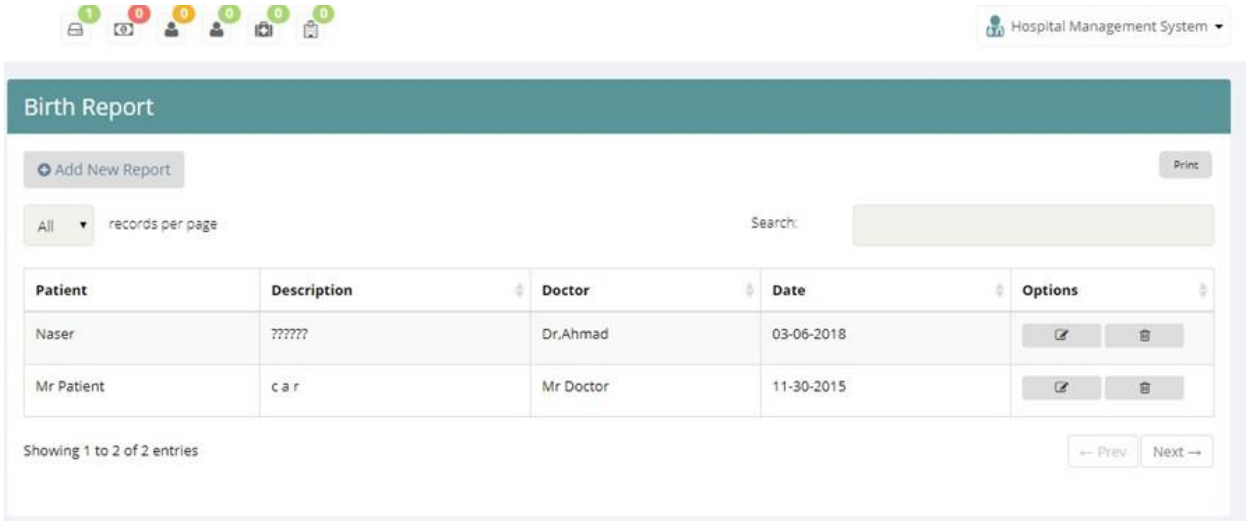
BED ALLOTMENT



PAYMENT DETAILS



BIRTH REPORT



OPERATION REPORT