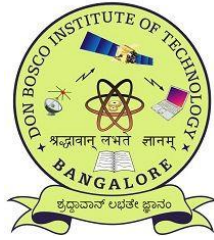


DON BOSCO INSTITUTE OF TECHNOLOGY



VISVESVARAYA TECHNOLOGICAL UNIVERSITY JNANA
SANGAMA, BELAGAVI – 590014



MINI PROJECT REPORT ON BRANDWISE MOBILE DATABASE

Submitted in partial fulfillment for the requirement of 5th semester for the

Degree of Bachelor of Engineering in
COMPUTER SCIENCE & ENGINEERING

For the Academic Year 2021-22

SUBMITTED BY:

CHETHAN S [1DB19CS035]

GS SUDEEP [1DB19CS035]

Under the guidance of:

Mrs. Sumathi M

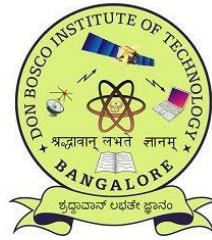
Asst. Professor

Dept. Of CSE

DON BOSCO INSTITUTE OF TECHNOLOGY, BENGALURU-560074

DON BOSCO INSTITUTE OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING

Kumbalagodu, Bengaluru-560074



CERTIFICATE

This is to certify that the Project Report entitled **BRANDWISE MOBILE DATABASE** is a Bonafide Project work carried out by **CHETHAN S (1DB19CS035)** and **GS SUDEEP (1DB19CS049)**, in partial fulfillment of '5th semester for the Degree of Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi, during the academic year 2020-21. It is certified that all corrections/suggestions indicated for Internal Assessments have been incorporated with the degree mentioned.

Signature of Guide

Mrs. Sumathi M
Assistant Professor,
Dept. of CSE,
DBIT, Bengaluru.

Signature of HOD

Prof. B.S UMASHANKAR
Professor & Head,
Dept. of CSE,
DBIT, Bengaluru.

External Viva

Name of the Examiners

1. _____

2. _____

Signature with Date

DON BOSCO INSTITUTE OF TECHNOLOGY

Kumbalagodu, Bengaluru -560074



DECLARATION

I, CHETHAN S and GS SUDEEP, student of fifth semester B.E, Department of Computer Science and Engineering, Don Bosco Institute of Technology, Kumbalagodu, Bengaluru, declare, that the Project Work entitled BRANDWISE MOBILE DATABASE has been carried out by and submitted in partial fulfillment of the requirement of 5th Semester Aug 2021-Jan 2022. The matter embodied in this report has been submitted to any university or institute for the award of any other degree or diploma.

Place: Bengaluru

Chethan S

1DB19CS035

Date:31/01/2022

GS Sudeep

1DB19CS049

ACKNOWLEDGEMENT

At the various stages in making the mini project, a number of people have given me invaluable comments on the manuscript. I take this opportunity to express my deepest gratitude and appreciation to all those who helped me directly or indirectly towards the successful completion of this project.

I would like to thank our Principal Dr. HEMADRI NAIDU T, Don Bosco Institute of Technology for his support throughout this project.

I express my whole hearted gratitude to Prof. B.S. UMASHANKAR, who is our respectable Head of Dept. of Computer Science. I wish to acknowledge his valuable help and encouragement.

In this regard I owe a heartfelt gratitude to my guide Mrs. Sumathi M, Assistant Professor of Department of Computer Science and Engineering, for her timely advice on the project and regular assistance throughout the project work. I would also like to thank the teaching and non-teaching staff members of the Department of Computer Science and Engineering for their corporation.

ABSTRACT

We will use technologies like MySQL, HTML, CSS and Python to build this website which would help us acquire skills of web development. Psychiatrist Database Management System provides the benefits of streamlined operations, enhanced administration & control, superior patient care, strict cost control and improved profitability. PDMS is powerful, flexible, and easy to use and is designed and developed to deliver real conceivable benefits to patients and doctors. More importantly it is backed by reliable and dependable support. Psychiatrist Database Management System is custom built to meet the specific requirement of the mid and large size hospitals across the globe. All the required modules and features have been particularly built to just fit in to your requirement. Not stopping only to this but they are highly satisfied and appreciating. Entire application is web based and built on 3 tier architecture using the latest technologies. The sound database of the application makes it more users friendly and expandable. The package is highly customizable and can be modified as per the needs and requirements of our clients.

Table of Contents

Sl.No	Topic	Page No.
1	Introduction	
	1.1 Overview	1
	1.2 Background and Motivation	1
	1.3 Methodology	2
2	Scope of the Project	3
3	Requirements	4
4	Data Directory	5
	4.1 E-R Diagram	6
	4.2 Detailed Schema Diagram	7
5	Relational Database Design	8
6	Graphical User Interface	9
7	Source Code	10
	7.1 Snapshots	31
8	Conclusion	36
9	References	37

CHAPTER 1

INTRODUCTION

1.1 Overview

In the current world where every application finds a place in the online platform, Psychiatry Hospital System would be one of them. A psychiatry hospital system includes several functionalities such as maintaining patient records, Doctor records, availability of doctors, appointments with doctors or medication prescribed for patients. Understanding these fundamental functionalities helps one to understand how hospital system works and helps to build modern trends in online hospital systems.

Keeping in mind the functionalities and the need of present generation we have developed a database management system for psychiatry hospital/department for the patients and doctors working in the hospital. It involves the database that hospital maintains for all doctors and patient's functionality. It keeps the day-by-day records of patients and doctors involving information of patient appointments with doctors, patient profiles and doctor profiles.

It removes the need of physical movement to a bank to carry out basic services offered at a hospital.

This project involves various concepts of database and web development. Our website has a user-friendly interface and it is being created using Xampp control Panel wherein MYSQL has been used for database management to store and retrieve data from the database. In addition, we have created multiple pages linked to each other. These pages have been developed using HTML and CSS. Python is used as a tool for connectivity between the front-end and the backend.

1.2 Background and Motivation

Background!!

In this section of our project, we will like to mention background or the prerequisites skills that we need to have in order to carry out this project. The main area of knowledge in this project is the creation of databases. Thorough knowledge of ER diagram, functional dependencies, normalization etc. For the completion of the project, we should have knowledge of the subjects listed-

1. HTML, CSS and Python for Front-end development
2. MYSQL in XAMPP SERVER for backend
3. SQL for preparing the database

Motivation!!

Our major motivation of carrying out this project is that amidst the ongoing digital hospital system environment, the need for an efficient digital hospital management solution is strongly felt. These systems are crucial for efficient time management for both doctors and patients. This also provides multidisciplinary approach to the management as well as the governance of the hospital related activities. These systems make the best use of computer systems as well as software for helping hospital and patients to create account and book appointments easily. The project makes a sincere effort to provide all the mentioned features to meet the requirements of the hospital in terms of both the hospital staffs and the patients. These factors piqued our interests to work on such project.

Objective: -

Need of Psychiatry database management

- 1) Convenience
- 2) Types of doctors
- 3) Availability of doctors
- 4) Crowd control

1.3 Methodology

To implement the above goals, the following methodology needs to be followed:

1. Specifying the Application and various components of the Architecture.
2. Specifying the bindings between the tasks and the resources either manually or by the design Tools.
3. Specifying the port interconnections between the resources.

CHAPTER 2

SCOPE OF PROJECT

The scope of this project is clear to give a simple and attractive user interface to simplify the work as well as to reduce the human efforts.

In this website, patients will be able to book appointments with doctors and doctors can view it accordingly without being physically present in the hospital.

CHAPTER 3

REQUIRMENTS

Hardware Requirement:

Processor: Pentium 5 or above

Ram: 2GB or more

Storage: 1GB or more

Software Requirement:

Operating System: Windows XP or above

Frontend tool: HTML, CSS

Backend tool: MySQL, Python

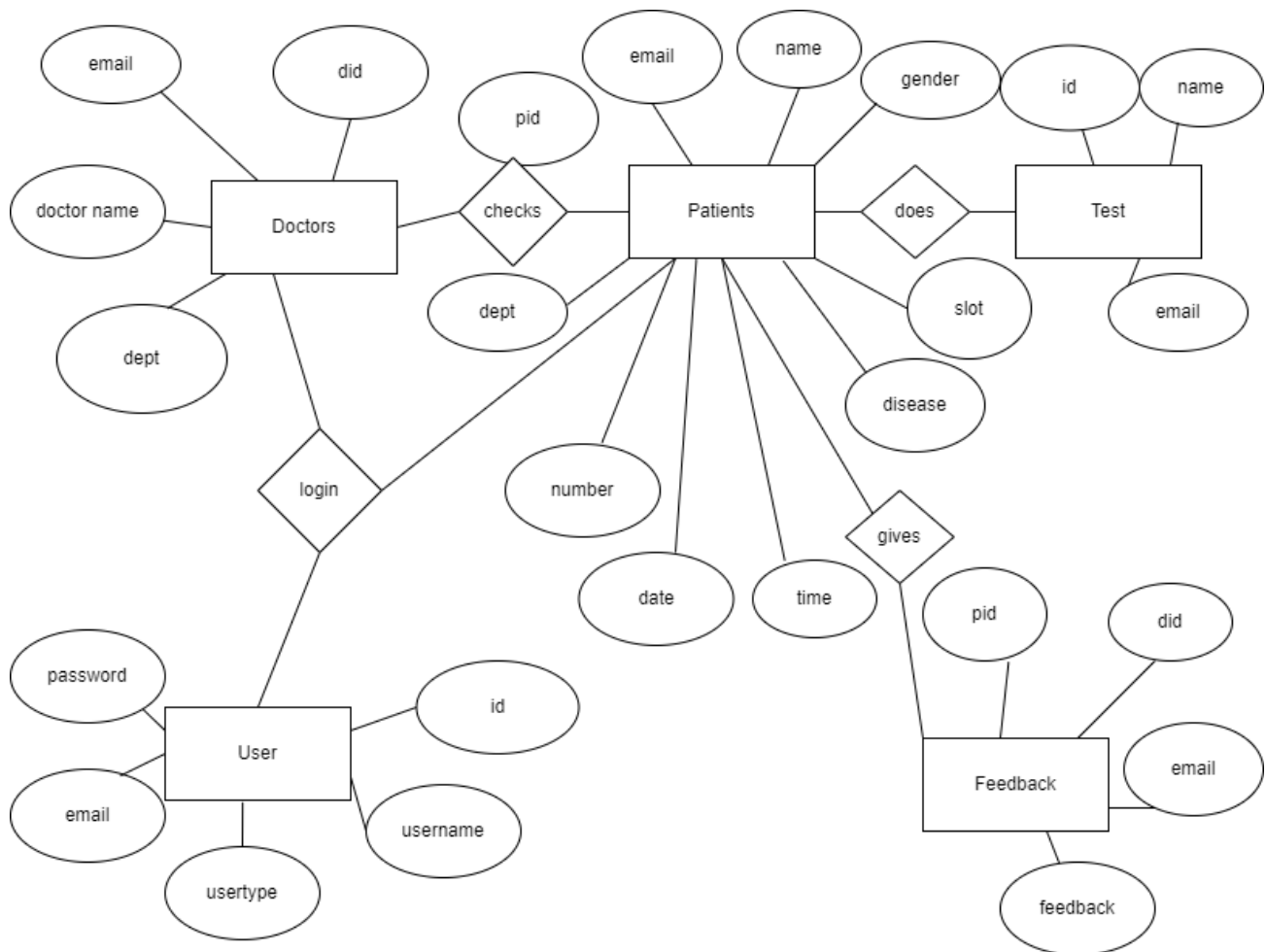
CHAPTER 4

DATA DIRECTORY

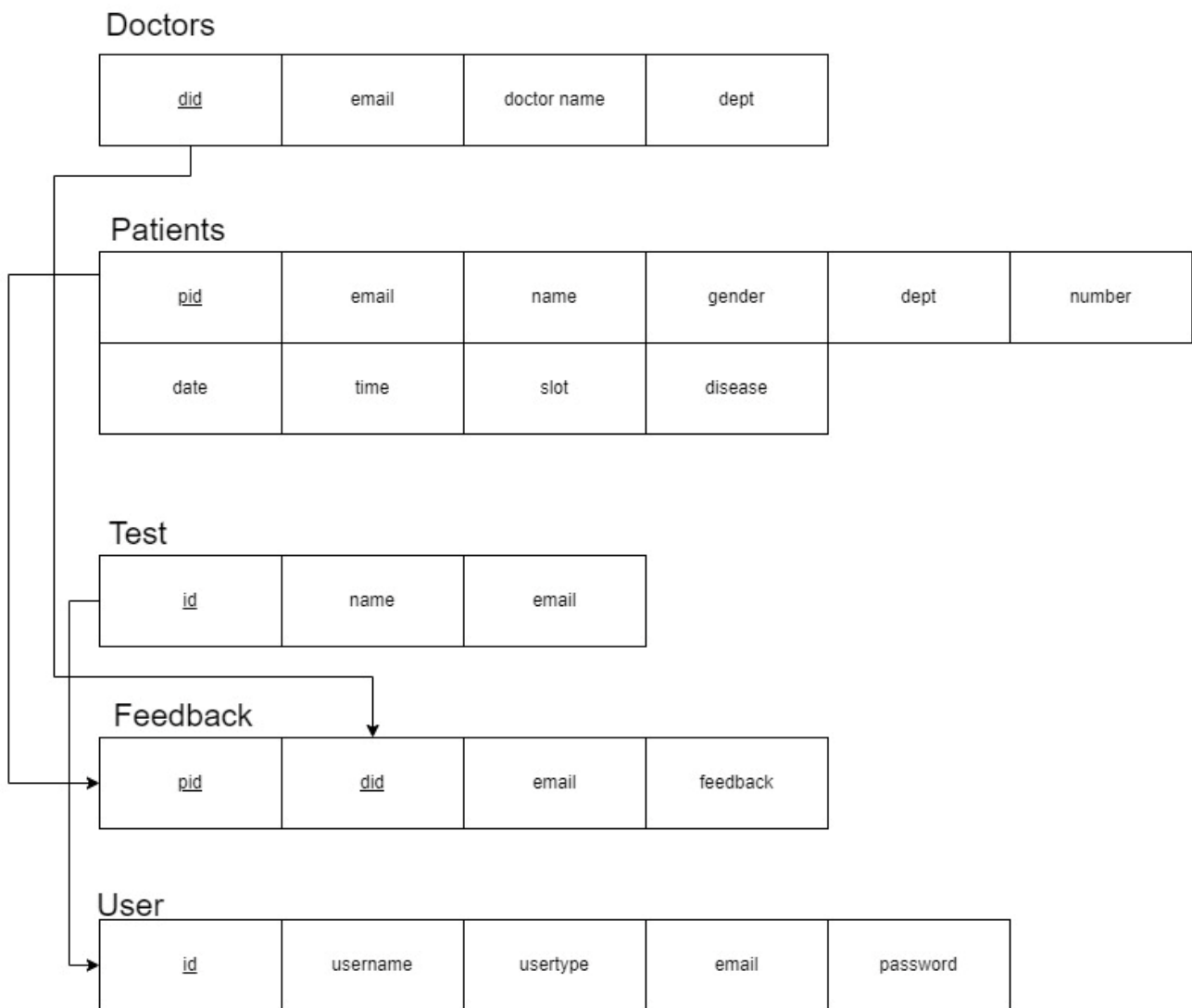
Table name:- User

```
(`id`,`username`,`usertype`,`email`,`password`) VALUES  
(13, 'anees', 'Doctor', 'anees@gmail.com',  
'pbkdf2:sha256:150000$xAKZCiJG$4c7a7e704708f86659d730565ff92e8327b01be5c49a6b1ef8af  
df1c531fa5c3'),  
(14, 'aneeqah', 'Patient', 'aneeqah@gmail.com',  
'pbkdf2:sha256:150000$Yf51ilDC$028cff81a536ed9d477f9e45efcd9e53a9717d0ab5171d75334c  
397716d581b8'),  
(16, 'Chethan S', 'Patient', 'chethans2001@gmail.com',  
'pbkdf2:sha256:260000$1qOZG950XnKEqE9f$0a3ad2ab18cf4e2cb8897e66d4daa168a7aa92967bc  
5fb58f1da95636864bb68'),  
(17, 'Chethan', 'Doctor', 'qwerty@gmail.com',  
'pbkdf2:sha256:260000$1Ij0F3OYkYzUj7HS$35b23cd79f6f4f25485a9fbc952e9ee60e4136cb5e7a  
8add6c4dd904117264a6');
```

4.1 E-R Diagram



4.2 Schema Diagram



CHAPTER 5

RELATIONAL DATA BASE DESIGN

1] DOCTOR :-

did, email, doctor_name, dept

did	email	Doctor_name	dept
-----	-------	-------------	------

2] PATIENTS:-

pid, email, name, gender, dept, number, date, time, slot, disease

pid	Email	name	gender	dept	number	date	time	slot	disease
-----	-------	------	--------	------	--------	------	------	------	---------

3] TEST :-

Id, name, email

id	name	email
----	------	-------

4] FEEDBACK :-

Pid, did, email, feedback

Pid	did	email	feedback
-----	-----	-------	----------

5] USER :-

Id, username, usertype, email, password

id	username	usertype	email	password
----	----------	----------	-------	----------

CHAPTER 6

GRAPHICAL USER INTERFACE

Application is very user friendly and uses a GUI interface implemented in Python and HTML to Communicate with the user. Various features are self-explanatory. Forms are easy to fill in and data can be added, removed or updated very easily. The application includes tool-tip hints to give a brief description of the particular input field.

List boxes are used to display all the components at once so that user can see all the components of a particular type at once.

Features:

1. Clean separation of various components to facilitate easy modification and revision
2. All the data is maintained in separate files to facilitate easy modification.
3. All the data required for different operations is kept in a separate file
4. Quick and easy saving and loading of database file.

CHAPTER 7

SOURCECODE

main.py: -

```
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
```

```
with open('config.json','r') as c:
    params = json.load(c)["params"]
```

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

```
# 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=params['gmail-user'],
    MAIL_PASSWORD=params['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_URL']='mysql://username:password@localhost/databas_table_name'
app.config['SQLALCHEMY_DATABASE_URI']='mysql://root:@localhost/pdbms'
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))

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))
```

```
# here we will pass endpoints and run the fuction
```

```
@app.route('/')
```

```
def index():
```

```
    a=params['gmail-user']
```

```
    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 ('{email}','{doctorname}','{dept}'))")
        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`")
```

```
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`,`disease`,`time`,`date`,`dept`,`number`) VALUES
        ('{email}','{name}','{gender}','{slot}','{disease}','{time}','{date}','{dept}','{number}')")
```

```
# mail starts from here
```

```
    # mail.send_message(subject, sender=params['gmail-user'],
recipients=[email],body=f"YOUR bOOKING IS CONFIRMED THANKS FOR
CHOOSING US \nYour Entered Details are :\nName: {name}\nSlot: {slot}")
```

```
flash("Booking Confirmed","info")
```

```
return render_template('patient.html',doct=doct)
```

```
@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` = '{gender}', `slot` = '{slot}', `disease` = '{disease}', `time` =
'{time}', `date` = '{date}', `dept` = '{dept}', `number` = '{number}' WHERE
`patients`.`pid` = {pid}")
        flash("Slot is Updates","success")
        return redirect('/bookings')

    return render_template('edit.html',posts=posts)
```

```
@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`) VALUES
('{username}','{usertype}','{email}','{encpassword}')")

# this is method 2 to save data in db
# newuser=User(username=username,email=email,password=encpassword)
# db.session.add(newuser)
# db.session.commit()
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():
    logout_user()
    flash("Logout SuccessFul","warning")
    return redirect(url_for('login'))

@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('triggers.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)
```

base.html

```
<!doctype html>
<html lang="en">

<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- Bootstrap CSS -->
  <link rel="stylesheet" href="static/css/virtualregister.css">
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css"
integrity="sha384-
TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2"
crossorigin="anonymous">
  { % block css % }
  { % endblock css % }

  <title>

  { % block title % }

  { % endblock title % }

</title>
</head>

<body>
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
    <a class="navbar-brand" href="/">P.D.M.S</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav mr-auto">
        <li class="nav-item active">
```

```

    <a class="nav-link" href="/">Home <span class="sr-
only">(current)</span></a>
  </li>

```

```

{% if current_user.usertype=="Doctor" %}
  <li class="nav-item">
    <a class="nav-link" href="/doctors">Doctors</a>
  </li>

```

```

    <li class="nav-item">
      <a class="nav-link" href="/bookings">Booking Details</a>
    </li>

```

```

    <li class="nav-item">
      <a class="nav-link" href="/details">Patients Details</a>
    </li>

```

```

{% else %}

```

```

    <li class="nav-item">
      <a class="nav-link" href="/patients">Patients Booking</a>
    </li>

```

```

    <li class="nav-item">
      <a class="nav-link" href="/bookings">Booking Details</a>
    </li>

```

```

{% endif %}

```

```

  {% if current_user.is_authenticated %}
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
        Welcome {{ current_user.username }}
      </a>
      <div class="dropdown-menu" aria-labelledby="navbarDropdown">

        <a class="dropdown-item" href="/logout">Logout</a>
      </div>
    </li>

```

```

{% else %}

```

```

<li class="nav-item dropdown">

```



```

    <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
    role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">

```

```

    Authentication

```

```

    </a>

```

```

    <div class="dropdown-menu" aria-labelledby="navbarDropdown">

```

```

        <a class="dropdown-item" href="/signup">Signup</a>

```

```

        <a class="dropdown-item" href="/login">Login </a>

```

```

    </div>

```

```

</li>

```

```

{% endif %}

```

```

</ul>

```

```

<form class="form-inline my-2 my-lg-0" action="/search" method="post">

```

```

    <input class="form-control mr-sm-2" type="search"
placeholder="Department" name="search" aria-label="Search">

```

```

    <button class="btn btn-outline-light my-2 my-sm-0"
type="submit">Search</button>

```

```

</form>

```

```

</div>

```

```

</nav>

```

```

{% block body %}

```

```

{% endblock body %}

```

```

<div class="card text-white bg-dark">

```

```

    <div class="card-header">Done By:</div>

```

```

    <div class="card-body">

```

```

        <h5 class="card-title">Chethan S</h5>

```

```

        <h5 class="card-title">G S Sudeep</h5>

```

```

        <p class="card-text">from DBIT, Bangalore.</p>

```

```

    </div>

```

```

</div>

```

```

<!-- Optional JavaScript; choose one of the two! -->

```

```

<!-- Option 1: jQuery and Bootstrap Bundle (includes Popper) -->
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkf
j" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZD
yx" crossorigin="anonymous"></script>

</body>

</html>

```

Booking.html

```

{% extends 'base.html' %}

{% block title %}
Booking
{% endblock title %}

{% block body %}
{% with messages=get_flashed_messages(with_categories=true) %}
{% if messages %}
{% for category, message in messages %}

<div class="alert alert-{ { category} } alert-dismissible fade show" role="alert">
  { { message} }

  <button type="button" class="close" data-dismiss="alert" aria-label="Close">
    <span aria-hidden="true">&times;</span>
  </button>
</div>

{% endfor %}
{% endif %}
{% endwith %}
<table class="table">
<thead class="thead-light">
<tr>
  <th scope="col">PID</th>

```

```

    <th scope="col">EMAIL</th>
    <th scope="col">NAME</th>
    <th scope="col">GENDER</th>
    <th scope="col">SLOT</th>
    <th scope="col">DISEASE</th>
    <th scope="col">DATE</th>
    <th scope="col">TIME</th>
    <th scope="col">D.DEPARTMENT</th>
    <th scope="col">PHONE NUMBER</th>
    <th scope="col">EDIT</th>
    <th scope="col">DELETE</th>

</tr>
</thead>
<tbody>
{% for post in query %}
<tr>
    <th scope="row">{{ post.pid }}</th>
    <td>{{ post.email }}</td>
    <td>{{ post.name }}</td>
    <td>{{ post.gender }}</td>
    <td>{{ post.slot }}</td>
    <td>{{ post.disease }}</td>
    <td>{{ post.date }}</td>
    <td>{{ post.time }}</td>
    <td>{{ post.dept }}</td>
    <td>{{ post.number }}</td>
    <td><a href="/edit/{{ post.pid }}"><button class="btn btn-success">Edit </button>
</a> </td>
    <td><a href="/delete/{{ post.pid }}"><button onclick="return confirm('Are you
sure to Delete data');" class="btn btn-success">Delete </button> </a> </td>
</tr>
{% endfor %}

</tbody>
</table>

{% endblock body %}

```

Doctor.html

```
{% extends 'base.html' % }

{% block title % }
Doctors
{% endblock title % }

{% block body % }

<div class="container">

<div class="row">

<div class="col-md-4"></div>
<div class="col-md-4">
{% with messages=get_flashed_messages(with_categories=true) % }
{% if messages % }
{% for category, message in messages % }

<div class="alert alert-{{ category }} alert-dismissible fade show" role="alert">
    {{ message }}

    <button type="button" class="close" data-dismiss="alert" aria-label="Close">
        <span aria-hidden="true">&times;</span>
    </button>
</div>

    {% endfor % }
    {% endif % }
    {% endwith % }
<br>
<h2 class="text-center text-white bg-dark">Doctors Booking</h2>

<br>
<form action="/doctors" method="post" class="jumbotron">

<div class="form-group">

<input type="email" class="form-control" name="email"
value={{ current_user.email }} required>

</div>
```

```
<div class="form-group">
```

```
<input type="text" class="form-control" name="doctorname" placeholder="Doctor  
Name" required>
```

```
</div>
```

```
<div class="form-group">
```

```
<input type="text" class="form-control" name="dept" placeholder="Doctor  
Department" required>
```

```
</div>
```

```
<button type="submit" class="btn btn-dark btn-sm btn-block">Book</button>
```

```
</form>
```

```
</div>
```

```
<div class="col-md-4"></div>
```

```
</div>
```

```
</div>
```

```
{% endblock body %}
```

Index.html

```
{% extends 'base.html' %}
```

```
{% block title %}
```

```
Doctors
```

```
{% endblock title %}
```

```
{% block body %}
```

```
<div class="container">
```

```
<div class="row">
```

```
<div class="col-md-4"></div>
```

```
<div class="col-md-4">
  {% with messages=get_flashed_messages(with_categories=true) %}
  {% if messages %}
  {% for category, message in messages %}

<div class="alert alert-{{ category }} alert-dismissible fade show" role="alert">
  {{ message }}

  <button type="button" class="close" data-dismiss="alert" aria-label="Close">
    <span aria-hidden="true">&times;</span>
  </button>
</div>

  {% endfor %}
  {% endif %}
  {% endwith %}
<br>
<h2 class="text-center text-white bg-dark">Doctors Booking</h2>

<br>
<form action="/doctors" method="post" class="jumbotron">

<div class="form-group">

<input type="email" class="form-control" name="email"
value={{ current_user.email }} required>

</div>

<div class="form-group">

<input type="text" class="form-control" name="doctorname" placeholder="Doctor
Name" required>

</div>

<div class="form-group">

<input type="text" class="form-control" name="dept" placeholder="Doctor
Department" required>

</div>
```

```

<button type="submit" class="btn btn-dark btn-sm btn-block">Book</button>

</form>

</div>
<div class="col-md-4"></div>

</div>

</div>

{% endblock body %}

```

patient.html

```

{% extends 'base.html' %}

{% block title %}
Patients Booking
{% endblock title %}

{% block body %}

<div class="container mt-3">
<div class="row">

<div class="col-md-5">
<div class="card" style="width: 18rem;">
  
  <div class="card-body">
    <h5 class="card-title">HOSPITAL DOCTORS</h5>
    <p class="card-text">Doctors Names</p>
  </div>
  <ul class="list-group list-group-flush">
    <li class="list-group-item">Cras justo odio</li>
    <li class="list-group-item">Dapibus ac facilisis in</li>
    <li class="list-group-item">Vestibulum at eros</li>
  </ul>
  <div class="card-body">
    <a href="#" class="card-link">Contact Us</a>
    <a href="#" class="card-link">About US</a>
  </div>

```

</div>

</div>

<div class="col-md-5">

<h4 class="text-center bg-dark text-white">Book Your Slot</h4>

{% with messages=get_flashed_messages(with_categories=true) %}

{% if messages %}

{% for category, message in messages %}

<div class="alert alert-{{ category }} alert-dismissible fade show" role="alert">
 {{ message }}

 <button type="button" class="close" data-dismiss="alert" aria-label="Close">
 ×

 </button>

</div>

{% endfor %}

{% endif %}

{% endwith %}

<form action="/patients" method="post" class="jumbotron">

<div class="form-group">

 <input type="email" class="form-control" id="email"
value={{ current_user.email }} name="email" required>

</div>

<div class="form-group">

 <input type="text" class="form-control" id="name" name="name"
placeholder="Full Name" required>

</div>

<div class="form-group">

<select class="form-control" id="gender" name="gender" required>

<option selected>Gender</option>

<option value="Male">Male</option>

<option value="Female">Female</option>

<option value="Others">Others</option>

</select>

</div>

<div class="form-group">

<select class="form-control" id="slot" name="slot" required>

<option selected>Slot</option>

<option value="morning">Morning</option>

<option value="evening">Evening</option>

<option value="night">Night</option>

</select>

</div>

<div class="form-group">

<input type="time" class="form-control" name="time" placeholder="Time" required>

</div>

<div class="form-group">

<input type="date" class="form-control" name="date" placeholder="date" required>

</div>

<div class="form-group">

```
<input type="text" class="form-control" id="disease" name="disease"
placeholder="Disease" required>

</div>

<div class="form-group">

<select class="form-control" id="dept" name="dept" required>
    <option selected>Select Doctor Department</option>
    {% for d in doct %}
    <option value="{{d.dept}}">{{d.dept}}</option>
    {% endfor %}
</select>
</div>

<div class="form-group">

    <input type="number" class="form-control" id="number" name="number"
placeholder="Phone Number" required>

</div>

<button type="submit" id="btn" class="btn btn-dark btn-sm btn-
block">Book</button>
</form>

</div>

</div>

</div>

{% endblock body %}
```

Signup.html

```
{% extends 'base.html' % }

{% block title % }
Signup
{% endblock title % }

{% block body % }
<div class="container mt-3">

<div class="row">

<div class="col-md-4">

</div>

<div class="col-md-4 jumbotron">
<h4 class="text-center bg-light text-dark">Sign Up Here</h4>
<br>

{% with messages=get_flashed_messages(with_categories=true) % }
{% if messages % }
{% for category, message in messages % }

<div class="alert alert-{{ category }} alert-dismissible fade show" role="alert">
    {{ message }}

    <button type="button" class="close" data-dismiss="alert" aria-label="Close">
        <span aria-hidden="true">&times;</span>
    </button>
</div>

{% endfor % }
{% endif % }
{% endwith % }

<form action="/signup" method="POST">

<div class="form-group">
```

```
<label for="username">UserName</label>
<input type="text" class="form-control" id="username" name="username"
required >
</div>

<div class="form-group">

<label for="usertype">Select UserType</label>

<select class="form-control" id="usertype" name="usertype" required>
  <option selected>Select</option>
  <option value="Patient">Patient</option>
  <option value="Doctor">Doctor</option>
</select>
</div>

<div class="form-group">
  <label for="email">Email Address</label>
  <input type="email" class="form-control" id="email" name="email" required>
  <small id="emailHelp" class="form-text text-muted">We'll never share your email
with anyone else.</small>
</div>

<div class="form-group">
  <label for="password">Password</label>
  <input type="password" class="form-control" id="password" name="password"
required>
</div>

  <button type="submit" class="btn btn-dark btn-sm btn-block">Signup</button>
</form>
<br>
Already a User <a href="/login">Login</a>

</div>

<div class="col-md-4">

</div>

</div>
```

</div>

{% endblock body %}

7.1 Snapshots of the application

The Home page is shown in Fig7.1.1

The signup page is shown in Fig 7.1.2

The login page is shown in Fig 7.1.3

The Patient booking page is shown in Fig 7.1.4

The Booking details of patient's side page is shown in Fig 7.1.5

The Doctor's page is shown in Fig 7.1.6.

The Booking Details of doctor's side page is shown in Fig 7.1.7.

The Patients Details page is shown in Fig 7.1.8.

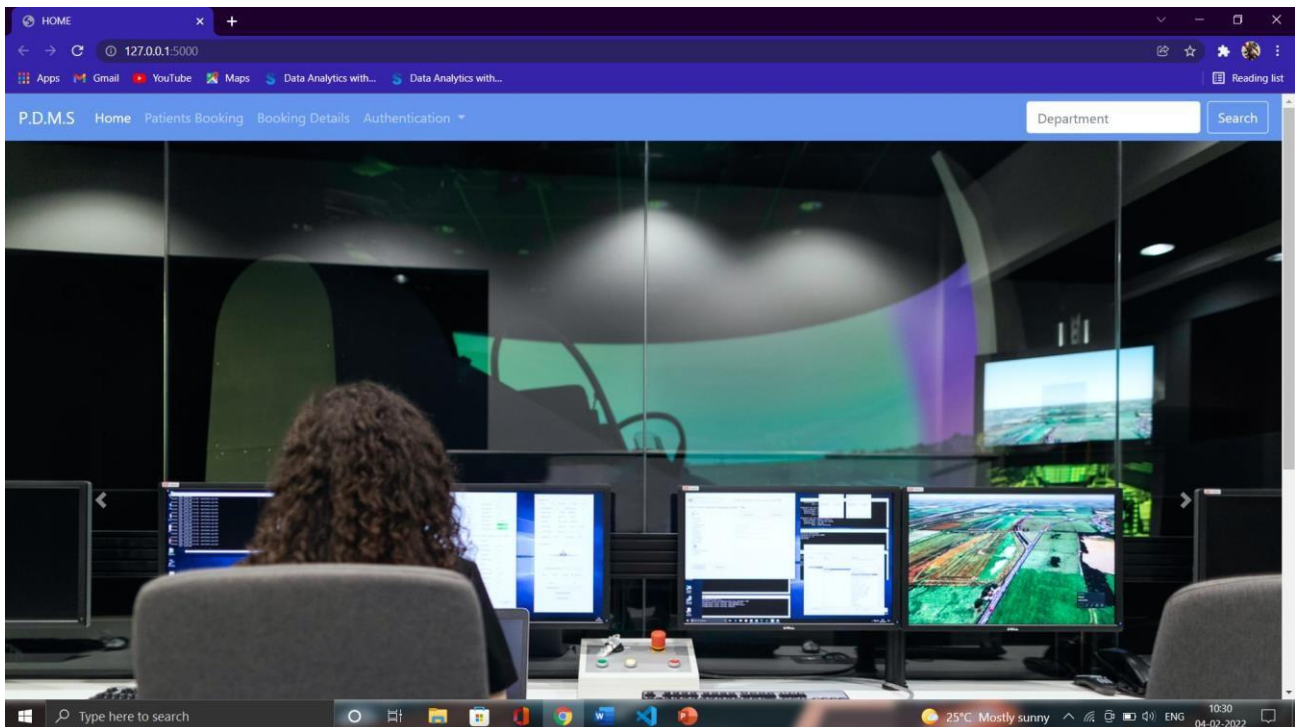


Fig7.1.1: Home page

The screenshot shows a web browser window with the URL `127.0.0.1:5000/signup`. The page has a blue header with the navigation menu: P.D.M.S, Home, Patients Booking, Booking Details, and Authentication. A search bar with the placeholder 'Department' and a 'Search' button is on the right. The main content area features a blue box with the title 'Sign Up Here'. Inside this box, there are four input fields: 'UserName', 'Email Address', and 'Password', each with a green border. Between the 'Email Address' and 'Password' fields, there is a small text note: 'We'll never share your email with anyone else.' Below the 'Password' field is a 'Signup' button. The browser's taskbar at the bottom shows the Windows search bar, task icons, and system tray information including '25°C Mostly sunny', '10:30', and '04-02-2022'.

Fig7.1.2: Sign Up

The screenshot shows a web browser window with the URL `127.0.0.1:5000/login`. The page has a blue header with the navigation menu: P.D.M.S, Home, Patients Booking, Booking Details, and Authentication. A search bar with the placeholder 'Department' and a 'Search' button is on the right. The main content area features a blue box with the title 'Login'. Inside this box, there are two input fields: 'Email Address' and 'Password', each with a green border. Below the 'Password' field is a 'Login' button. At the bottom of the blue box, there is a link: 'Not a User [Signup](#)'. The browser's taskbar at the bottom shows the Windows search bar, task icons, and system tray information including '25°C Mostly sunny', '10:31', and '04-02-2022'.

Fig7.1.3: Login

Patients Booking

127.0.0.1:5000/patients

P.D.M.S Home Patients Booking Booking Details Welcome Chethan S

Department Search

Book Your Slot

chethans2001@gmail.com

Full Name

Gender

Slot

--:--

dd-mm-yyyy

Disease

Select Doctor Department

Phone Number

HOSPITAL DOCTORS

Doctors Names

Cras justo odio

Dapibus ac facilisis in

Vestibulum at eros

Contact Us About US

Fig7.1.4: Patients Booking

Booking

127.0.0.1:5000/bookings

P.D.M.S Home Patients Booking Booking Details Welcome Chethan S

Department Search

PID	EMAIL	NAME	GENDER	SLOT	DISEASE	DATE	TIME	D.DEPARTMENT	PHONE NUMBER	EDIT	DELETE
18	chethans2001@gmail.com	Chethan S	Male	morning	asddfam	2022-02-05	13:35:00	Endocrinologists	0	Edit	Delete

Done By:

Chethan S
G S Sudeep
from DBIT, Bangalore.

Fig7.1.5: Booking details of patient's side

Doctors Booking

asdfg@gmail.com

Doctor Name

Doctor Department

Book

Done By:

Chethan S

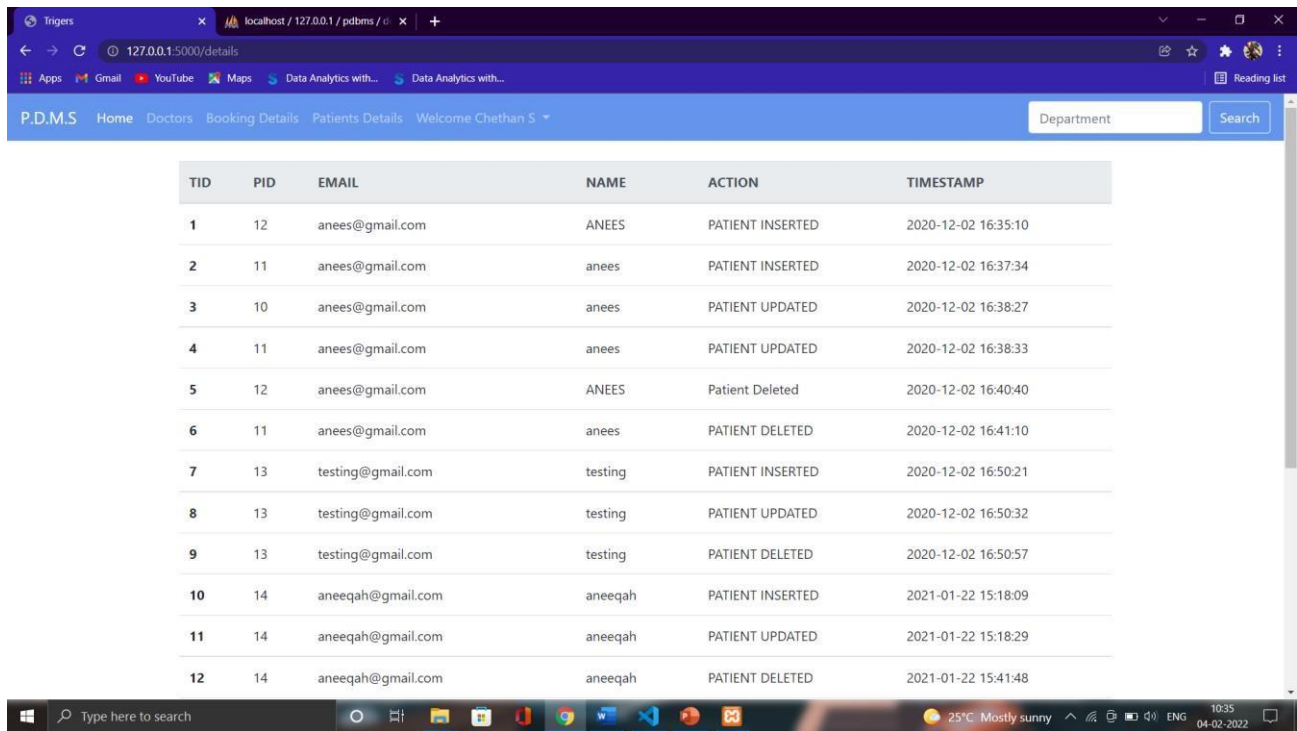
G S Sudeep

Fig7.1.6: Doctors

PID	EMAIL	NAME	GENDER	SLOT	DISEASE	DATE	TIME	D.DEPARTMENT	PHONE NUMBER	EDIT	DELETE
2	anees1@gmail.com	anees1 rehman khan	Male	evening1	cold1	2020-02-02	21:20:00	ortho11predict	9874561110	Edit	Delete
5	patient@gmail.com	patien	Male	morning	fevr	2020-11-18	18:06:00	Cardiologists	9874563210	Edit	Delete
7	patient@gmail.com	anees	Male	evening	cold	2020-11-05	22:18:00	Dermatologists	9874563210	Edit	Delete
8	patient@gmail.com	anees	Male	evening	cold	2020-11-05	22:18:00	Dermatologists	9874563210	Edit	Delete
9	aneesurrehman423@gmail.com	anees	Male	morning	cold	2020-11-26	17:27:00	Anesthesiologists	9874563210	Edit	Delete
10	anees@gmail.com	anees	Male	evening	fever	2020-12-09	16:25:00	Cardiologists	9874589654	Edit	Delete
17	aneeqah@gmail.com	aneeqah	Female	evening	fever	2021-01-23	15:48:00	Endocrinologists	9874563210	Edit	Delete
18	chethans2001@gmail.com	Chethan S	Male	morning	asddfam	2022-02-05	13:35:00	Endocrinologists	0	Edit	Delete

Done By:

Fig7.1.7: Booking details of doctor's side



TID	PID	EMAIL	NAME	ACTION	TIMESTAMP
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

Fig7.1.8: Patients Details

CONCLUSION

Thus, we have successfully implemented online psychiatry database management system which helps patients and doctors of the hospital to easily book and view appointments without being physically present in hospital.

We have successfully implemented various functionalities of database and python and created fully functional database management system for psychiatry management system.

REFERENCES

1. Ramez Elmasri, Shamikant B. Navathe; Fundamentals of Database Management Systems; 5th Edition; Pearson; 2017
2. Raghuramakrishnan, and Gehrke; Database Management Systems; 3rd edition; McGraw Hill; 2014
3. <https://www.w3schools.com/sql/default.asp>
4. <https://www.youtube.com/CodeWithHarry/>
5. <https://www.w3schools.com/css/default.asp>