Sprint – 1

Date	04.11.2022
Team ID	PNT2022TMID20641
Project Name	Visualizing And Predicting Heart Diseases With An Interactive Dash Board

Login.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Login</title>
   <link href='https://fonts.googleapis.com/css?family=Fjalla One' rel='stylesheet'>
   <link href='https://fonts.googleapis.com/css?family=Saira Condensed' rel='stylesheet'>
   <link href='https://fonts.googleapis.com/css?family=Russo One' rel='stylesheet'>
   <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">


   <style>
      body{
         background-color: rgb(16, 12, 12);
      }
      .loginCntnt{
         display: flex;
         margin-top: 12%;
         margin-left: 20%;
      }
      .loginPge{
         background-color: rgb(68, 180, 208);
         width: 400px;
         height: 350px;
      }
      .loginTtle{
         font-family:'Fjalla One';
         font-size: 30px;
         font-weight: bold;
         letter-spacing: 2px;
         margin-left: 40%;
         margin-top: 12px;
      }
      .loginForm{
         position:relative;
         padding: 10px;
```

```
}
.loginForm input{
    width: 50%;
    height:35px;
    border: 2px solid;
    border-radius: 10px;
    border-color: rgb(6, 186, 192);
    padding: 0 10px;
    margin-left: 20%;
    margin-top: 2%;
    background: transparent;
    font-family:'Saira Condensed';
    color: white;
    font-weight: bold;
    letter-spacing: 1.2px;
}


::placeholder{
    color:white;
    opacity: 0.6;
}


.loginForm button{
    margin-left: 40%;
    margin-top: 10%;
    /* width: 150px; */
    width: fit-content;
    height:30px;
    border-radius: 10px;
    border-color: transparent;
    background-color: black;
    color: aqua;
    font-family: 'Russo One';
}


.loginForm button:hover{
    background-color: rgba(255, 255, 255, 0.719);
    color: black;
}


a{
    text-decoration: none;
    color: black;
}
```

```
.signup{
    margin-left: 40%;
    margin-top: 1%;
    font-family: 'Russo One';
    color: rgb(0, 0, 0);
    width: fit-content;
}


.loginPicture{
    width: 550px;
    height: 350px;
}


#ErrorMsg{
    margin-left: 14%;



}
@media only screen and (max-width: 900px) {
    .loginPicture{
        width: 500px;
        height: 350px;
        transition: 0.5s;
    }
    .loginCntnt {
        flex-direction: column;
    }
    .loginPge{
        margin-left: 12%;
    }



}
@media only screen and (max-width: 890px) {
    .loginPge{
        margin-left: 0.1%;
        width: 500px;
        transition: 0.9s;
    }
}
@media only screen and (max-width: 500px) {
    .loginPge{
        margin-left: 15%;
        width: 380px;
        transition: 0.8s;
```

```
        }
    }



  </style>


</head>
<body>
  <div class="loginCntnt">
    <div class="loginImage">
      <img class="loginPicture" src="{{loginpic}}" alt="loginimg">
    </div>
    <div class="loginPge">
      <h2 class="loginTtle">LOGIN</h2>
      <form action="{{ url_for('login') }}" class="loginForm" method="POST">
        <input type="text" name="usrname" id="usrname" placeholder="user name" required ><br>
        <input type="password" name="password" id="password" placeholder="password" required><br>
        <p id="ErrorMsg">{{errorMsg}}</p>
        <button type="submit">LOGIN</button>
      </form>
      <!-- <button class="signup" onclick="{{url_for('register')}}">Sign up</button> -->
      <h4 ><a class="signup" href="{{url_for('register')}}">Sign up</a></h4>
    </div>
  </div>
</body>
</html>
```

Register.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <link href='https://fonts.googleapis.com/css?family=Fjalla One' rel='stylesheet'>
    <link href='https://fonts.googleapis.com/css?family=Saira Condensed'
rel='stylesheet'>
    <link href='https://fonts.googleapis.com/css?family=Russo One' rel='stylesheet'>


    <style>
```

```css
body{
    background-color: black;
}
.rgstrCntnt{
    display: flex;
    margin-top: 12%;
    margin-left: 20%;
}
.rgstrPge{
    margin-top: -4%;
    background-color: rgb(7, 141, 175);
    width: 400px;
    height: 400px;
}
.rgstrTtle{
    font-family:'Fjalla One';
    font-size: 30px;
    font-weight: bold;
    letter-spacing: 2px;
    margin-left: 34%;
    margin-top: 12px;
}
.rgstrForm{
    position:relative;
    padding: 10px;
    margin-top: -3%;
}


.rgstrForm input{
    width: 50%;
    height:35px;
    border: 2px solid;
    border-radius: 10px;
    border-color: rgb(6, 186, 192);
    padding: 0 10px;
    margin-left: 20%;
    margin-top: 2%;
    background: transparent;
    font-family:'Saira Condensed';
    color: white;
    font-weight: bold;
    letter-spacing: 1.2px;
}
```

```
        ::placeholder{
            color:white;
            opacity: 0.6;
        }


        .rgstrForm button{
            margin-left: 36%;
            margin-top: 3%;
            width: 100px;
            height:30px;
            border-radius: 10px;
            border-color: transparent;
            background-color: black;
            color: aqua;
            font-family: 'Russo One';
        }


        .rgstrForm button:hover{
            background-color: rgba(255, 255, 255, 0.719);
            color: black;
        }


        a{
            text-decoration: none;
            color: black;
        }
        .signin{
            margin-left: 43%;
            margin-top: -1%;
        }
        .rgImge{
            width: 550px;
            height: 400px;
        }
    </style>


</head>
<body>
    <div class="rgstrCntnt">
```

```
        <div class="rgstrImage">
            <img class="rgImge" src="{{registerImg}}" alt="Car resale value"
 style="width: 550px;height:300px;">
        </div>
        <div class="rgstrPge">
            <h2 class="rgstrTtle">REGISTER</h2>
            <form action="{{ url_for('register') }}" class="rgstrForm" method="POST">
                <input type="text" id="usrname" name="usrname" placeholder="user
 name" required><br>
                <input type="email" id="email" name="email"
 placeholder="example@gmail.com" required><br>
                <input type="password" id="password" name="password"
 placeholder="password" required><br>
                <input type="password" id="rpassword" placeholder="retype-password"
 required><br>
                <p id="ErrorMsg">{{errorMsg}}</p>
                <button type="submit">REGISTER</button>
            </form>
            <h4 class="signin"><a href="{{url_for('login')}}">Sign in</a></h4>
        </div>
    </div>
 </body>
 </html>
```

App.py:

```
import ssl
import smtplib
import pandas as pd
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from turtle import st
from flask import Flask, render_template, request, redirect, url_for, session
from markupsafe import escape

import ibm_db
conn = ibm_db.connect(
    "DATABASE=bludb;HOSTNAME=764264db-9824-4b7c-82df-
40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=32536;SECURITY=SSL;SSL
ServerCertificate=DigiCertGlobalRootCA.crt;UID=xqh16737;PWD=MOou4YIDP7R6pvIA", '', '')
# email.mime subclasses
# The pandas library is only for generating the current date, which is not necessary
for sending emails

app = Flask(__name__)

var_list = []
app.secret_key = 'your secret key'


@app.route('/')
def home():
    if not session.get("name"):
        return render_template('home.html')
    return render_template('home.html', session=session)
```

```
@app.route('/register')
def new_student():
    return render_template('Register.html')


@app.route('/addrec', methods=['POST', 'GET'])
def addrec():
    if request.method == 'POST':
        fname = request.form['fname']
        lname = request.form['lname']
        cname = request.form['cname']
        state = request.form['state']
        city = request.form['city']
        mobileno = request.form['mobileno']
        emailid = request.form['emailid']
        password = request.form['password']
        pincode = request.form['pincode']

        sql = "SELECT * FROM Users WHERE EMAILID =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, emailid)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)

        if account:
            users = []
            sql = "SELECT * FROM Users"
            stmt = ibm_db.exec_immediate(conn, sql)
            dictionary = ibm_db.fetch_both(stmt)
            while dictionary != False:
                # print ("The Name is : ",  dictionary)
                users.append(dictionary)
                dictionary = ibm_db.fetch_both(stmt)
            return render_template('list.html', msg="You are already a member, please
login using your details", users=users)
        else:

            var_list.append(fname)
            var_list.append(lname)
            var_list.append(cname)
            var_list.append(state)
            var_list.append(city)
            var_list.append(mobileno)
            var_list.append(emailid)
            var_list.append(password)
            var_list.append(pincode)

            bodytemp = r'D:\IBM\GUIDED PROJECT\INVENTORY MANAGEMENT SYSTEM FOR
RETAILERS\templates\email.html'
            with open(bodytemp, "r", encoding='utf-8') as f:
                html = f.read()

            # Set up the email addresses and password. Please replace below with your
email address and password
            email_from = 'padhu10a@gmail.com'
            epassword = 'rbjibzkssszsbrjo'
            email_to = emailid
```

```
            # Generate today's date to be included in the email Subject
            date_str = pd.Timestamp.today().strftime('%Y-%m-%d')

            # Create a MIMEMultipart class, and set up the From, To, Subject fields
            email_message = MIMEMultipart()
            email_message['From'] = email_from
            email_message['To'] = email_to
            email_message['Subject'] = f'Report email - {date_str}'

            # Attach the html doc defined earlier, as a MIMEText html content type to
the MIME message
            email_message.attach(MIMEText(html, "html"))
            # Convert it as a string
            email_string = email_message.as_string()

            # Connect to the Gmail SMTP server and Send Email
            context = ssl.create_default_context()
            with smtplib.SMTP_SSL("smtp.gmail.com", 465, context=context) as server:
                server.login(email_from, epassword)
                server.sendmail(email_from, email_to, email_string)
            return render_template('notify.html')


@app.route('/confirm')
def confirnation():
    insert_sql = "INSERT INTO Users (FIRSTNAME, LASTNAME, COMPANYNAME, STATE, CITY,
MOBILENO, EMAILID, PASSWORD, PINCODE)  VALUES (?,?,?,?,?,?,?,?,?)"
    prep_stmt = ibm_db.prepare(conn, insert_sql)
    ibm_db.bind_param(prep_stmt, 1, var_list[0])
    ibm_db.bind_param(prep_stmt, 2, var_list[1])
    ibm_db.bind_param(prep_stmt, 3, var_list[2])
    ibm_db.bind_param(prep_stmt, 4, var_list[3])
    ibm_db.bind_param(prep_stmt, 5, var_list[4])
    ibm_db.bind_param(prep_stmt, 6, var_list[5])
    ibm_db.bind_param(prep_stmt, 7, var_list[6])
    ibm_db.bind_param(prep_stmt, 8, var_list[7])
    ibm_db.bind_param(prep_stmt, 9, var_list[8])
    ibm_db.execute(prep_stmt)
    return render_template('confirm.html')


@app.route('/login', methods=['POST', 'GET'])
def login():
    msg = ''
    if request.method == 'POST' and 'email' in request.form and 'password' in
request.form:
        email = request.form['email']
        password = request.form['password']

        sql = "SELECT * FROM Users WHERE EMAILID =? AND PASSWORD =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.bind_param(stmt, 2, password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)

        if account:
            session['loggedin'] = True
            session['id'] = account['ID']
            session['email'] = account['EMAILID']
```

```
            session['name'] = account['FIRSTNAME']
            msg = 'Logged in successfully !'
            return render_template('home.html', msg=msg)
        else:
            msg = 'Incorrect email / password !'
    return render_template('login.html', msg=msg)


@app.route('/logout')
def logout():
    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('email', None)
    session.pop('name', None)
    return redirect(url_for('home'))


@app.route('/list')
def list():
    users = []
    sql = "SELECT * FROM Users"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
        # print ("The Name is : ",  dictionary)
        users.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)

    if users:
        return render_template("list.html", users=users, session=session)

    return "No users..."
```

Output:

To run the webpage, run it in the flask.