SOURCE CODE

App.py

```
from flask import Flask, render_template, flash, request, session
from flask import render_template, redirect, url_for, request

importjson
from json2html import *
importrequests

import ibm_db
importpandas
import ibm_db_dbi
fromsqlalchemyimportcreate_engine

engine = create_engine('sqlite://',
                       echo = False)

dsn_hostname = "98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud"
dsn_uid = "tvd24047"
dsn_pwd = "C0fhAXeLsuoQuvel"

dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "BLUDB"
dsn_port = "30875"
dsn_protocol = "TCPIP"
dsn_security = "SSL"

dsn = (
```

```
    "DRIVER={0};"
    "DATABASE={1};"
    "HOSTNAME={2};"
    "PORT={3};"
    "PROTOCOL={4};"
    "UID={5};"
    "PWD={6};"
    "SECURITY={7};").format(dsn_driver, dsn_database, dsn_hostname, dsn_port, dsn_protocol,
dsn_uid, dsn_pwd,dsn_security)



try:
    conn = ibm_db.connect(dsn, "", "")
    print ("Connected to database: ", dsn_database, "as user: ", dsn_uid, "on host: ", dsn_hostname)

except:
    print ("Unable to connect: ", ibm_db.conn_errormsg() )

app = Flask(__name__)
app.config['DEBUG']
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'




@app.route("/")
defhomepage():

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

```
@app.route("/Home")
defHome():
    return render_template('index.html')

@app.route("/AdminLogin")
defAdminLogin():
    return render_template('AdminLogin.html')

@app.route("/NewUser")
defNewUser():
    return render_template('NewUser.html')

@app.route("/NewCompany")
defNewCompany():
    return render_template('NewCompany.html')

@app.route("/UserLogin")
defStudentLogin():
    return render_template('UserLogin.html')

@app.route("/CompanyLogin")
defCompanyLogin():
    return render_template('CompanyLogin.html')


@app.route("/Search")
defSearch():
    return render_template('Search.html')
```

```
@app.route("/AdminHome")
defAdminHome():

    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * from regtb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()
    return render_template('AdminHome.html', data=data)




@app.route("/ACompanyInfo")
defACompanyInfo():

    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * from companytb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()


    return render_template('ACompanyInfo.html', data=data)
```

```
@app.route("/AjobInfo")
defAjobInfo():


    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * from jobtb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()

    return render_template('AjobInfo.html', data=data)


@app.route("/SCompanyInfo")
defSCompanyInfo():


    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * from jobtb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()

    return render_template('SCompanyInfo.html', data=data)

@app.route("/CompanyHome")
```

```
defCompanyHome():
    return render_template('CompanyHome.html')


@app.route("/UserHome")
defUserHome():

    uname=  session['uname']

    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM regtb where Username='"+ uname +"' "
    dataframe = pandas.read_sql(selectQuery, pd_conn)
    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()

    return render_template('UserHome.html', data=data)




@app.route("/CJobInfo")
defCJobInfo():


    cname=  session['cname']


    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM jobtb where Cname='"+ cname +"' "
    dataframe = pandas.read_sql(selectQuery, pd_conn)
    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
```

```
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()



    return render_template('CJobInfo.html', data=data)


@app.route("/adminlogin", methods=['GET', 'POST'])
defadminlogin():
    error = None
    if request.method == 'POST':
      if request.form['uname'] == 'admin'or request.form['password'] == 'admin':




        conn = ibm_db.connect(dsn, "", "")
        pd_conn = ibm_db_dbi.Connection(conn)
        selectQuery = "SELECT * FROM regtb  "
        dataframe = pandas.read_sql(selectQuery, pd_conn)
        dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
        data = engine.execute("SELECT * FROM Employee_Data").fetchall()

        return render_template('AdminHome.html', data=data)

      else:
       return render_template('index.html', error=error)



@app.route("/userlogin", methods=['GET', 'POST'])
defuserlogin():
    error = None
```

```
    if request.method == 'POST':

        username = request.form['uname']
        password = request.form['password']


        conn = ibm_db.connect(dsn, "", "")

        pd_conn = ibm_db_dbi.Connection(conn)

        selectQuery = "SELECT * from regtb where UserName='" + username + "' and password='"
+ password + "'"
        dataframe = pandas.read_sql(selectQuery, pd_conn)

        ifdataframe.empty:
            data1 = 'Username or Password is wrong'
            return render_template('goback.html', data=data1)
        else:
            print("Login")
           selectQuery = "SELECT * from regtb where UserName='" + username + "' and password='"
+ password + "'"
            dataframe = pandas.read_sql(selectQuery, pd_conn)

            dataframe.to_sql('Employee_Data',
                             con=engine,
                             if_exists='append')

            # run a sql query
            print(engine.execute("SELECT * FROM Employee_Data").fetchall())

                return  render_template('UserHome.html',  data=engine.execute("SELECT  *  FROM
Employee_Data").fetchall())
```

```
@app.route("/companylogin", methods=['GET', 'POST'])
defcompanylogin():
    error = None
    if request.method == 'POST':

        uname = request.form['uname']
        password = request.form['password']
        session['cname'] = uname


        conn = ibm_db.connect(dsn, "", "")

        pd_conn = ibm_db_dbi.Connection(conn)

            selectQuery = "SELECT * from companytb where UserName='" + uname + "' and password='" + password + "'"
        dataframe = pandas.read_sql(selectQuery, pd_conn)

        ifdataframe.empty:
            data1 = 'Username or Password is wrong'
            return render_template('goback.html', data=data1)
        else:
            print("Login")
                selectQuery = "SELECT * from companytb where UserName='" + uname + "' and password='" + password + "'"
            dataframe = pandas.read_sql(selectQuery, pd_conn)

            dataframe.to_sql('Employee_Data',
                        con=engine,
```

```
                    if_exists='append')

        # run a sql query
        print(engine.execute("SELECT * FROM Employee_Data").fetchall())

         return render_template('CompanyHome.html', data=engine.execute("SELECT * FROM
Employee_Data").fetchall())

@app.route("/NewStudent1", methods=['GET', 'POST'])
defNewStudent1():
    if request.method == 'POST':

        name = request.form['name']
        gender = request.form['gender']
        Age = request.form['Age']
        email = request.form['email']
        pnumber = request.form['pnumber']
        address = request.form['address']
        Degree = request.form['Degree']
        depat = request.form['depat']
        uname = request.form['uname']
        passw = request.form['passw']

    conn = ibm_db.connect(dsn, "", "")

    insertQuery = "insert into regtb values('" + name + "','" + gender + "','" + Age + "','" + email +
"','" + pnumber + "','" + address + "','" + Degree + "','" + depat + "','" + uname + "','" + passw + "')"
    insert_table = ibm_db.exec_immediate(conn, insertQuery)

    sendmsg(email, "Successfully registered this website")
```

```
    data1 = 'Record Saved!'
    return render_template('goback.html', data=data1)




@app.route("/newcompany", methods=['GET', 'POST'])
defnewcompany():
    if request.method == 'POST':

        cname = request.form['cname']
        regno = request.form['regno']
        mobile = request.form['mobile']

        email = request.form['email']
        Website = request.form['Website']
        address = request.form['address']
        uname = request.form['uname']
        passw = request.form['passw']

        conn = ibm_db.connect(dsn, "", "")

                                    insertQuery = "insert into companytb values('"+cname+"','"+regno+"','"+mobile+"','"+email+"','"+Website+"','"+address+"','"+uname+"','"+passw+"')"
        insert_table = ibm_db.exec_immediate(conn, insertQuery)

        data1 = 'Record Saved!'
        return render_template('goback.html', data=data1)
```

```
@app.route("/newjob", methods=['GET', 'POST'])
defnewjob():
    if request.method == 'POST':
        cnn = session['cname']
        cname = request.form['cname']
        cno = request.form['cno']
        Address = request.form['Address']
        JobLocation = request.form['JobLocation']
        Vacancy = request.form['Vacancy']
        Job = request.form['Job']
        Department = request.form['depat']
        website = request.form['website']

        conn = ibm_db.connect(dsn, "", "")

         insertQuery = "insert into jobtb values('" + cname + "','" + cno + "','" + Address + "','" +
JobLocation + "','" + Vacancy + "','" + Job + "','" + Department + "','" + website + "','"+cnn+"')"
        insert_table = ibm_db.exec_immediate(conn, insertQuery)

        conn = ibm_db.connect(dsn, "", "")
        pd_conn = ibm_db_dbi.Connection(conn)
        selectQuery1 = "SELECT  *  FROM  regtb where  Department='" + Department + "'"
        dataframe = pandas.read_sql(selectQuery1, pd_conn)

        dataframe.to_sql('regtb', con=engine, if_exists='append')
        data1 = engine.execute("SELECT * FROM regtb").fetchall()

        foritem1indata1:
            Mobile = item1[5]
```

```
            Email = item1[4]
            sendmsg(Email,"Jop Title"+Job + " More Info Visit Website")




    data = 'Record Saved!'
    return render_template("goback.html", data=data)




@app.route("/jobsearch", methods=['GET', 'POST'])
defjobsearch():
    if request.method == 'POST':
        jobname = request.form['name']


        url = "https://linkedin-jobs-search.p.rapidapi.com/"

        payload = {
            "search_terms": jobname,
            "location": "india",
            "page": "1"
        }
        headers = {
            "content-type": "application/json",
            "X-RapidAPI-Key": "b045b9af95msha8d7c3160785729p1674cdjsnbdf4adbf9868",
            "X-RapidAPI-Host": "linkedin-jobs-search.p.rapidapi.com"
```

```
        }

        response = requests.request("POST", url, json=payload, headers=headers)

        print(response.text)

        infoFromJson = json.loads(response.text)

        df = pandas.json_normalize(infoFromJson)

        df.to_sql('regtb', con=engine, if_exists='append')
        data1 = engine.execute("SELECT * FROM regtb").fetchall()



        return render_template('Search.html',data=data1)
```

```
#send grid

defsendmsg(Mailid,message):
    importsmtplib
    fromemail.mime.multipartimportMIMEMultipart
    fromemail.mime.textimportMIMEText
    fromemail.mime.baseimportMIMEBase
    fromemailimportencoders
```

```
fromaddr = "sampletest685@gmail.com"
toaddr = Mailid

# instance of MIMEMultipart
msg = MIMEMultipart()

# storing the senders email address
msg['From'] = fromaddr

# storing the receivers email address
msg['To'] = toaddr

# storing the subject
msg['Subject'] = "Alert"

# string to store the body of the mail
body = message

# attach the body with the msg instance
msg.attach(MIMEText(body, 'plain'))

# creates SMTP session
s = smtplib.SMTP('smtp.gmail.com', 587)

# start TLS for security
s.starttls()

# Authentication
s.login(fromaddr, "hneucvnontsuwgpj")
```

```
    # Converts the Multipart msg into a string
    text = msg.as_string()

    # sending the mail
    s.sendmail(fromaddr, toaddr, text)

    # terminating the session




if__name__ == '__main__':
    app.run(host='0.0.0.0', debug='TRUE')
```

job.py

```
importrequests
importjson
importpandasaspd
from json2html import *
url = "https://linkedin-jobs-search.p.rapidapi.com/"

payload = {
    "search_terms": "python programmer",
    "location": "india",
    "page": "1"
}
headers = {
    "content-type": "application/json",
    "X-RapidAPI-Key": "b045b9af95msha8d7c3160785729p1674cdjsnbdf4adbf9868",
    "X-RapidAPI-Host": "linkedin-jobs-search.p.rapidapi.com"
}
```

```
response = requests.request("POST", url, json=payload, headers=headers)

print(response.text)

infoFromJson = json.loads(response.text)
print(json2html.convert(json = infoFromJson))
#data = json.loads(elevations)
df = pd.json_normalize(infoFromJson)
print(df)
```