

Sprint - 4

Date	17 November 2022
Team ID	PNT2022TMID46789
Project Name	Skill/Job Recommender
Maximum Marks	20 Marks

1. Creating corporate login and post job page with python codings

1.1 Corporate login:

```
{% extends "bootstrap/base.html" %}
{% block title %}Login{% endblock %}
{% block styles %}
{{super()}}
<link rel="stylesheet" href="{{url_for('static', filename='style.css')}}">
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.
css" rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi
" crossorigin="anonymous">
{% endblock %}
{% block content %}
<div class="Login-form">
<section class="vh-100">
<div class="container-fluid h-custom">
<div class="row d-flex justify-content-center align-items-center h-100">
<div class="col-md-9 col-lg-6 col-xl-5">

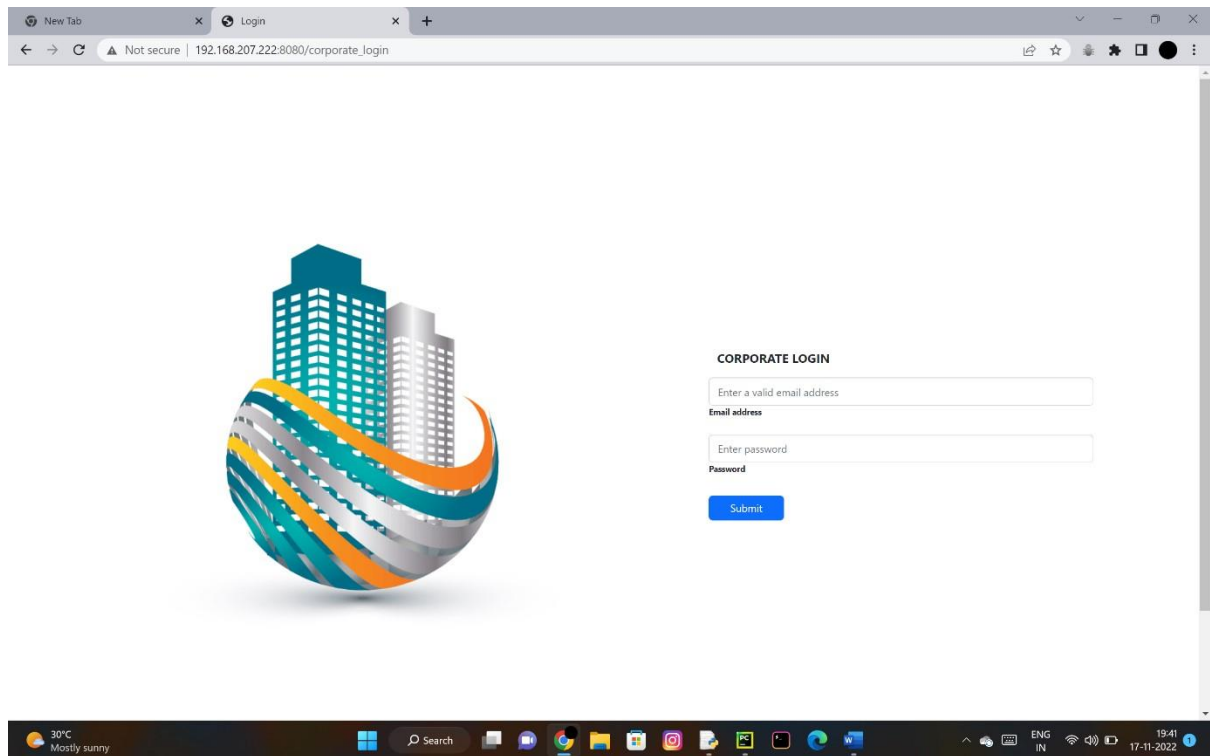
</div>
<div class="col-md-8 col-lg-6 col-xl-4 offset-xl-1">
<form class="form" method="post"
action="{{url_for('corporate_login')}}">

<div class="divider d-flex align-items-center my-4">
<h4 class="text-center fw-bold mx-3 mb-0">CORPORATE LOGIN</h4>
</div>
<div class="form-outline mb-4">
```

```

        <input type="text" id="form3Example3" name="email"
class="form-control form-control-lg"
        placeholder="Enter a valid email address" />
        <label class="form-label" for="form3Example3">Email address</label>
    </div>
    <div class="form-outline mb-3">
        <input type="password" id="form3Example4" name="password"
class="form-control form-control-lg"
        placeholder="Enter
password" />
        <label class="form-label" for="form3Example4">Password</label>
    </div>
    <div class="text-center text-lg-start mt-4 pt-2">
<input type="submit" class="btn btn-primary btn-lg"
        style="padding-left: 2.5rem; padding-right: 2.5rem;">
    </div>
    </form>
</div>
</div>
</div>
</div>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.
min.js" integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw
3" crossorigin="anonymous"></script>
    <script src="https://kit.fontawesome.com/e66a43891e.js"
crossorigin="anonymous"></script>
{% endblock %}

```



1.2 Post job

```
{% extends "bootstrap/base.html" %}
{% block title %}Register{% endblock %}
{% block styles %}
{{super()}}
<link rel="stylesheet" href="{{url_for('static', filename='style.css')}}">
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.
css" rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRI
" crossorigin="anonymous">
{% endblock %}
{% block content %}
<div class="Login-form">
<section class="vh-100">
<div class="container-fluid h-custom">
<div class="row d-flex justify-content-center align-items-center h-100">
<div class="col-md-9 col-lg-6 col-xl-5">

</div>
<div class="col-md-8 col-lg-6 col-xl-4 offset-xl-1">
<form class="form" method="post" action="{{url_for('postjob')}}">
```

```

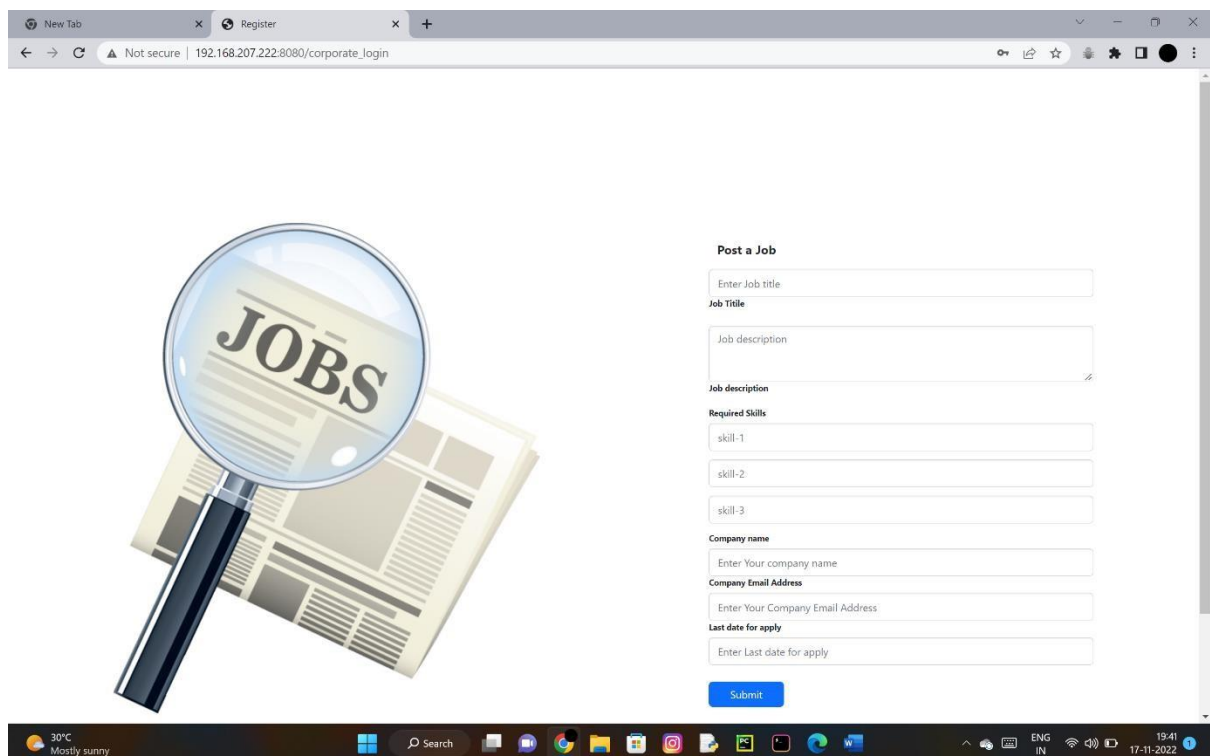
<div class="divider d-flex align-items-center my-4">
  <h4 class="text-center fw-bold mx-3 mb-0">Post a Job</h4>
</div>
<div class="form-outline mb-4">
  <input type="text" id="form3Example3" name="jt" class="formcontrol
form-control-lg"
    placeholder="Enter Job title" />
  <label class="form-label" for="form3Example3">Job Titile</label>
</div>
<div class="form-outline mb-3">
  <textarea class="form-control form-control-lg" id="Textarea1"
placeholder="Job description" name="jd" rows="3"></textarea>
  <label class="form-label" for="form3">Job description</label>
</div>
<div>
  <label class="form-label" for="form3Example3">Required
Skills</label>
  <input type="text" name="skill-1" id="skill1" class="form-control
form-control-lg"
    placeholder="skill-1" />
  <input id="skill2" type="text" name="skill-2" class="form-control
form-control-lg"
    placeholder="skill-2" />
  <input type="text" id="skill3" name="skill-3" class="form-control
formcontrol-lg"
    placeholder="skill-3" />
</div>
<div>
  <label style="margin:10px 0px 10 0px ;" class="form-label"
for="form3Example3">Company name</label>
  <input type="text" name="Company-name" class="form-control
formcontrol-lg"
    placeholder="Enter Your company name" />
</div>
<div>
  <label style="margin:10px 0px 10 0px ;" class="form-label"
for="form3Example3">Company Email Address</label>
  <input type="text" name="company-email" class="form-control
formcontrol-lg"
    placeholder="Enter Your Company Email Address" />
</div>
</div>

```

```

        <label style="margin:10px 0px 10 0px ;" class="form-label"
for="form3Example3">Last date for apply</label>
        <input type="text" name="date" class="form-control form-control-lg"
placeholder="Enter Last date for apply" />
    </div>
    <div class="text-center text-lg-start mt-4 pt-2">
        <input type="submit" name="img" class="btn btn-primary btn-lg"
style="padding-left: 2.5rem; padding-right: 2.5rem;">
    </div>
</form>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.
min.js" integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw
3" crossorigin="anonymous"></script>
    <script src="https://kit.fontawesome.com/e66a43891e.js"
crossorigin="anonymous"></script>
{% endblock %}

```



Python codings:

```
from flask import Flask,request,render_template,redirect,flash,url_for
from flask_bootstrap import Bootstrap
import sqlite3
import ibm_db
import ibm_boto3
from ibm_botocore.client import Config,ClientError
import json
from ibm_watson import AssistantV2
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
import smtplib
```

```
ibm_watson_api = 'NERNzL18grk3-kRrozsq897BF2IC-fmbhMO0FeC91XjR'
authenticator = IAMAuthenticator(ibm_watson_api)
assistant = AssistantV2(
    version = '2021-11-27',
    authenticator = authenticator
)
assistant.set_service_url('https://api.jp-
tok.assistant.watson.cloud.ibm.com/instances/403b880a-1907-4cd4-
be34ed7621ce872c')
assistant_id = '4f9960db-4d31-43a3-a58a-f747bef82cbe'
response = assistant.create_session(
    assistant_id=assistant_id
).get_result()
session_id = json.dumps(response, indent=2)
print(session_id)
```

```
COS_ENDPOINT ='https://s3.jp-tok.cloud-object-storage.appdomain.cloud'
COS_API_KEY_ID="N4Il1nx1AhrGMhoAtXgkvC6i4DZzWYE428HMPQSNByAh"
COS_INSTANCE_CRN="crn:v1:bluemix:public:iam-
identity::a/31fa813cf7134a2b87f66465bde60656::serviceid:ServiceId-8d6fd9a6-
0638-4c58-9270-
cc40705c2385" app =
Flask(__name__)
Bootstrap(app) log = 'no'
```

```

db = sqlite3.connect("Database.db", check_same_thread=False) cursor
= db.cursor()

cos =
ibm_boto3.resource("s3",ibm_api_key_id=COS_API_KEY_ID,ibm_service_instance_i
d=COS_INSTANCE_CRN,config=Config(signature_version="oauth"),endpoint_url=COS
_ENDPOINT)

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ea286ace-86c7-4d5b-
85803fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=31505;
SECU
RITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=yfp49237;PWD=KmAn
VzPQq2dDSayo;", "", "")

def get_bucket_contents(bucket_name):
    print("Retrieving bucket contents from: {}".format(bucket_name))
    try:
        files = cos.Bucket(bucket_name).objects.all()
    files_names = []    for file in files:
        files_names.append(file.key)
        print("Item: {} ({} bytes)".format(file.key,
file.size))    return files_names    except ClientError as
be:
        print("CLIENT ERROR: {}".format(be))
    except Exception as e:
        print("Unable to retrieve bucket contents: {}".format(e))

@app.route('/') def
home():
    files = get_bucket_contents('projectby13')
    return render_template("home.html",login=log,files=files)

@app.route('/login',methods=["POST","GET"])
def login():    if request.method == "POST":
        email = request.form.get('email')
password = request.form.get('password')    sql
= "SELECT * FROM USERS"    stmt =
ibm_db.exec_immediate(conn, sql)
dictionary = ibm_db.fetch_assoc(stmt)    while
dictionary != False:        if
dictionary['EMAIL']==email:            if
dictionary['PASSWORD']==password:

```

```

flash('You were successfully logged in')
log = 'Yes'
        return redirect(url_for('joblist'))
dictionary = ibm_db.fetch_assoc(stmt)
        return "wrong credantials if you are new user please signup"
return                                     render_template('login.html')
@app.route('/register',methods=["POST","GET"]) def register():  if
request.method == "POST":
        name = request.form.get("name")    email =
request.form.get("email")    password =
request.form.get("password")    confirmpassword =
request.form.get("repeatpassword")    if
password==confirmpassword:    sql1 = "SELECT * FROM
USERS WHERE NAME =?"    stmt1 =
ibm_db.prepare(conn,sql1)
ibm_db.bind_param(stmt1,1,name)
ibm_db.execute(stmt1)
        account = ibm_db.fetch_assoc(stmt1)

        if account:
print('p')    else:
        insert_sql = "INSERT INTO USERS VALUES (?,?)"
prep_stmt = ibm_db.prepare(conn, insert_sql)
ibm_db.bind_param(prepare_stmt, 1, name)
ibm_db.bind_param(prepare_stmt, 2, email)
ibm_db.bind_param(prepare_stmt, 3, password)
ibm_db.execute(prepare_stmt)
        print("inserted")    return
redirect(url_for('joblist'))    return
render_template("Register.html")
@app.route('/postjob',methods=["POST","GET"])
) def postjob():  if request.method == "POST":
jobtitle = request.form.get('jt')
jobdescription = request.form.get('jd')    skill1
= request.form.get('skill-1')    skill2 =
request.form.get('skill-2')    skill3 =
request.form.get('skill-3')
        Date = request.form.get('date')
        Companyname = request.form.get('Company-name')
CompanyEmail = request.form.get('company-email')
valve = 10
        insert_sql = "INSERT INTO joblist VALUES (?,?)"
prep_stmt = ibm_db.prepare(conn, insert_sql)

```



```

ibm_db.bind_param(prepare_stmt, 1, jobtitle)
ibm_db.bind_param(prepare_stmt, 2, jobdescription)
ibm_db.bind_param(prepare_stmt, 3, skill1)
ibm_db.bind_param(prepare_stmt, 4, skill2)
ibm_db.bind_param(prepare_stmt, 5, skill3)
ibm_db.bind_param(prepare_stmt, 6, Date)
ibm_db.bind_param(prepare_stmt, 7, Companyname)
ibm_db.bind_param(prepare_stmt, 8, CompanyEmail)
    ibm_db.execute(prepare_stmt)
return redirect(url_for('joblist')) else:
    return 'wrong credentials'

```

```

@app.route('/joblist',methods=["POST","GET"])
def joblist():    if request.method == "POST":
search_key =    request.form.get('search-bar')
sql = "SELECT * FROM joblist"          stmt =
ibm_db.exec_immediate(conn, sql)
    dictionary =
ibm_db.fetch_both(stmt)    jt_list = []
jd_list = []    companies = []    while
dictionary != False:
    if search_key == dictionary['SKILL1'] or search_key == dictionary['SKILL2'] or
search_key == dictionary['SKILL3'] :
        jt_list.append(dictionary['JOBTITLE'])
jd_list.append(dictionary['JOBDES'])
companies.append(dictionary['COMPANYNAME'])
        dictionary = ibm_db.fetch_both(stmt)
else:
        dictionary = ibm_db.fetch_both(stmt)
    lent = len(jd_list)
no = 0
    return render_template("joblist.html", jtr=jt_list,
jdr=jd_list, len=lent,cn=companies) else:
    sql = "SELECT * FROM joblist"          stmt =
ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
jt_list = []
jd_list = []
companies = []    while
dictionary != False:
    jt_list.append(dictionary['JOBTITLE'])
jd_list.append(dictionary['JOBDES'])

```

```

companies.append(dictionary['COMPANYNAME'])        dictionary =
ibm_db.fetch_both(stmt)
    lent = len(jd_list)
no = 0
    return render_template("joblist.html",jtr=jt_list,jdr=jd_list,len =
lent,cn=companies)
#-----Bucket storage-----
-#
def multi_part_upload(bucket_name, item_name,file_path):
try:
    print("Starting file transfer for {0} to bucket: {1}\n".format(item_name,
bucket_name))
    part_size = 1024 * 1024 * 5
file_threshold = 1024 * 1024 * 15
    transfer_config = ibm_boto3.s3.transfer.TransferConfig(
multipart_threshold=file_threshold,
    multipart_chunksize=part_size
    )
    print("out")    with open(file_path,
"rb") as file_data:
        cos.Object(bucket_name, item_name).upload_fileobj(
            Fileobj=file_data,
            Config=transfer_config
        )

    print("Transfer for {0} Complete!\n".format(item_name))
except ClientError as be:
    print("CLIENT ERROR: {0}\n".format(be))
except Exception as e:
    print("Unable to complete multi-part upload: {0}".format(e))

@app.route('/applyjob', methods = ['GET',
'POST']) def applyjob():  if request.method ==
'POST':    companyname =
request.form.get('cnp')
    sql = "SELECT * FROM joblist"    stmt =
ibm_db.exec_immediate(conn, sql)
dictionary = ibm_db.fetch_both(stmt)
while dictionary != False:
    print(dictionary["COMPANYNAME"])    if
companyname == dictionary["COMPANYNAME"]:
        curname=dictionary["COMPANYNAME"]
cnmail = dictionary["COMPANYEMAIL"]

```

```

dictionary = ibm_db.fetch_both(stmt)      dictionary
= ibm_db.fetch_both(stmt)      return
render_template('applyjob.html',cne=cnmail,cnn=cnn
ame)
@app.route('/mail',methods=["POST","GET"])
def mail():  if request.method == 'POST':
    my_email = "kprathap1307@gmail.com"      password =
"qsustqqoqudsxrcb"      companyname =
request.form.get('cnn')      companyemail =
request.form.get('cne')      name = request.form.get('name')
email= request.form.get('email')      experience=
request.form.get('experience')      message=
request.form.get('message')      sslc_schoolname =
request.form.get('sschool-name')      sslc_passedout =
request.form.get('sschool-year')      sslc_percentage =
request.form.get('sschool-percentage')      hslc_schoolname
= request.form.get('hschool-name')      hslc_passedout =
request.form.get('hschool-year')      hslc_percentage =
request.form.get('hschool-percentage')      college_name =
request.form.get('d-name')      college_passedout =
request.form.get('d-year')      college_cgpa =
request.form.get('d-cgpa')      skill1 = request.form.get('skill-
1')      skill2 = request.form.get('skill-2')      skill3 =
request.form.get('skill-3')

    mail = f'Hello {companyname} , Mail from Mr.perks , here are the candidate
informations for your current openings ' \
        f'Candidate name = {name} , candidate email = {email} , candidate
experience = {experience} ,candidate objectives = {message}' \
        f'candidate education details : SSLC - SSLC school name = {sslc_schoolname}
, SSLC - SSLC Passed out year = {sslc_passedout},' \
        f'Percentage = {sslc_percentage} , HSLC school name =
{hslc_schoolname},passed out year = {hslc_passedout}' \
        f'CGPA / Percentage= {college_cgpa} and candidate skills are 1.{skill1}
2.{skill2} 3.{skill3} '

    with smtplib.SMTP("smtp.gmail.com", port=587) as connection:
connection.starttls()

        connection.login(user=my_email,      password=password)
connection.sendmail(from_addr=my_email,      to_addrs=companyemail,
msg=f"Subject:Applied for job \n\n {mail}")
        return 'Sucessfully applied for the job '
@app.route('/corporate_login',methods=["POST","GET"])
def corporate_login():  if request.method == "POST":
    email = request.form.get('email')

```

```
password = request.form.get('password')
admin_email = 'admin@123'    admin_password =
'karur@123'    if email == admin_email and
password==admin_password:
    return render_template('postjob.html')
else:
    return 'username and password will be given by admin'
return render_template('corporate_login.html')
```

```
if __name__ == '__main__':
    app.secret_key = 'super secret key'
    app.config['SESSION_TYPE'] = 'filesystem'
app.run(host='0.0.0.0',port=8080,debug=True)
```