DOCUMENTATION OF FLASK PROJECT – A JOB PORTAL

WORKING OF A FLASK APP

from flask import Flask

app = Flask(__name__)

@app.route('/')

def hello_world():

    return 'Hello, World!'

if __name__=='__main__':

    app.run(debug=True)   #any changes made will immediately show

It runs on http://127.0.0.1:5000/   where '127.0.0.1' is the IP address of my computer and '5000' is the port

In Pycharm terminal----------

```
pip install flask
env\scripts\activate
```

```
from flask import *
import pymysql
```

---- STEP 2----------

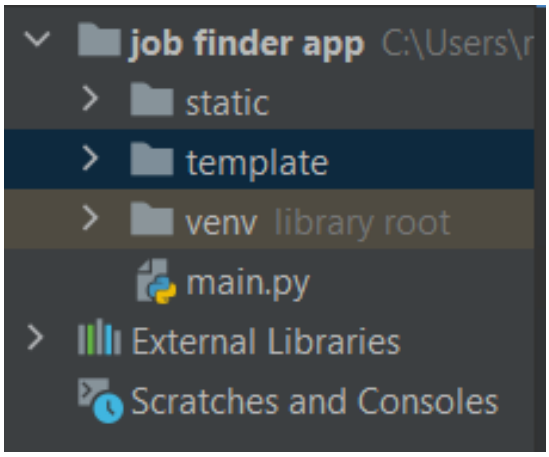
Base.html, all other html files extend from this base file. Base file defines a simple HTML skeleton document that you might use for a simple two-column page. It's the job of "child" templates to fill the empty blocks with content:

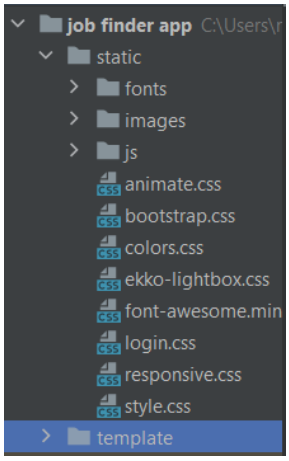
base.html

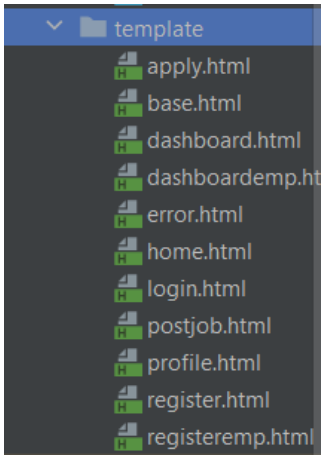
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="static/main.css">
    <link href="https://fonts.googleapis.com/css?family=Ubuntu" rel="stylesheet">
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" inte
    <!-- url_for is a jynga mehtod-->
    {% block head %} {% endblock %}
</head>
<body>
    {% block body %} {% endblock %}
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js" integrity="sha384-9/reFT
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js" integrity="sha384-B4gt
</body>
</html>
```

---- STEP 3----------

ALL CSS ,HTML and JS FILES GO IN STATIC FOLDER



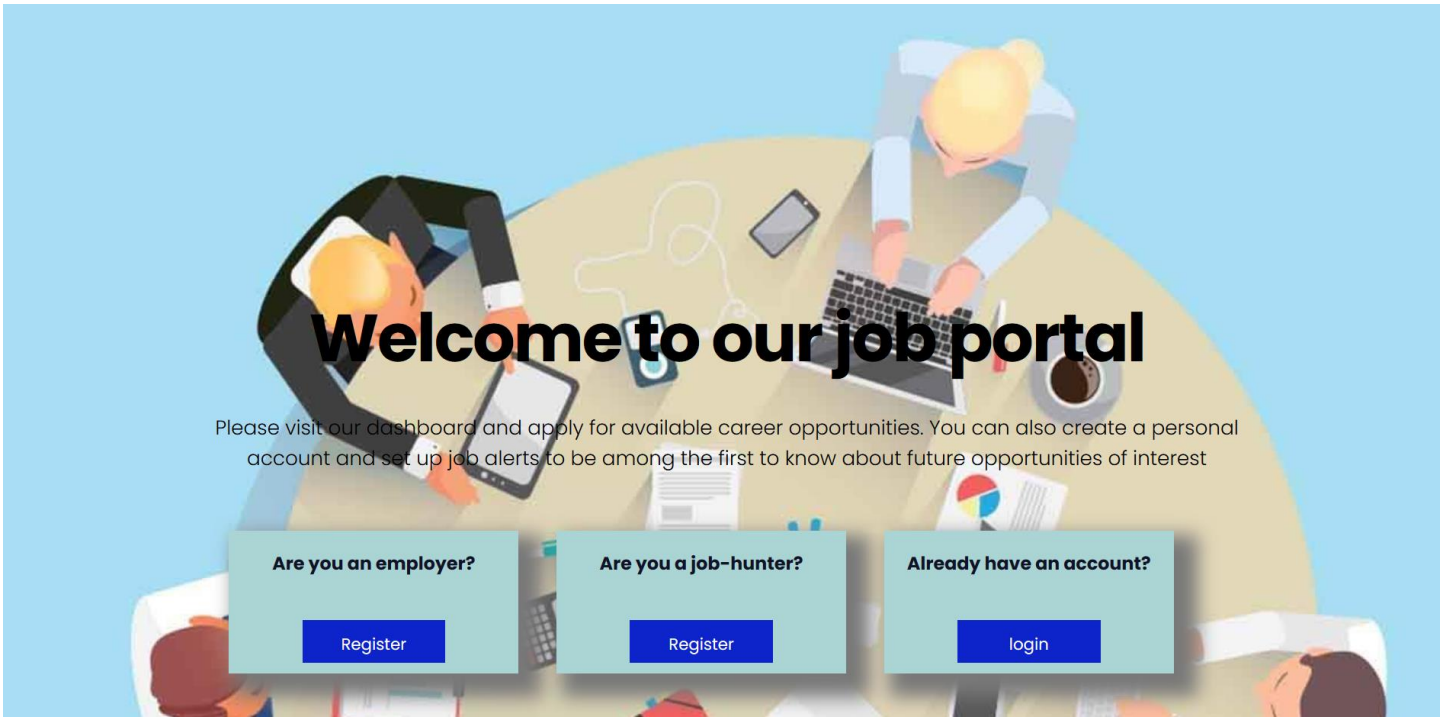




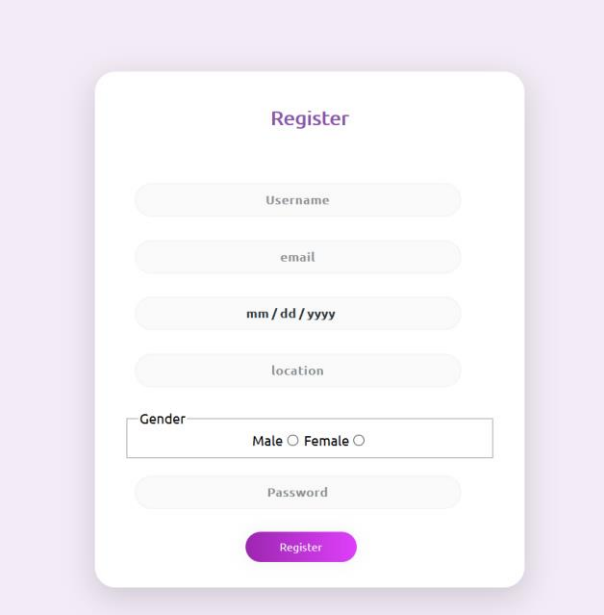
---- STEP 4----------

LANDING PAGE

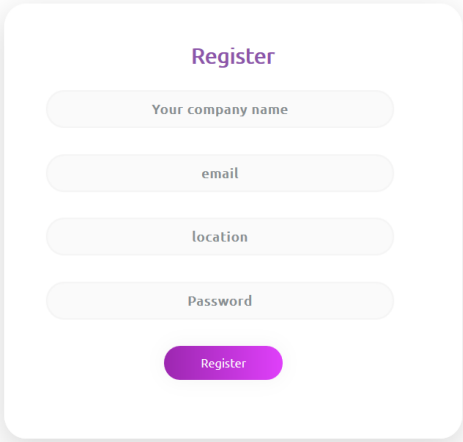
```python
app = Flask(__name__, template_folder='template')

@app.route('/', methods=['GET'])
def home():
    return render_template('home.html')
```



---- STEP 5----------



REGISTRATION FOR JOB HUNTER



REGISTRATION FOR EMPLOYER

Submitting registration information to sql database:

```
@app.route("/signup", methods=['POST'])
def signup():

    username = request.form['username']
    email = request.form['email']
    dob = request.form['dob']
    location = request.form['location']
    gender = request.form['gender']
    password = request.form['password']
    msg=''
    try:
        myconn = pymysql.connect(host='localhost', user = 'root', password = '',
database='flask_app')
        cursor = myconn.cursor()
        sql="insert into profiles_app(username,email,dob,location,gender,password)
values('{}','{}','{}','{}','{}','{}')".format(username,email,dob,location,gender,password
)
        cursor.execute(sql)
        if cursor.rowcount > 0:
            myconn.commit()
            msg = "registration done"
        else:
            myconn.rollback()
            msg = "try again"
    except Exception as e:
        msg = '{}, try again'.format(e)
        myconn.rollback()
    finally:
```

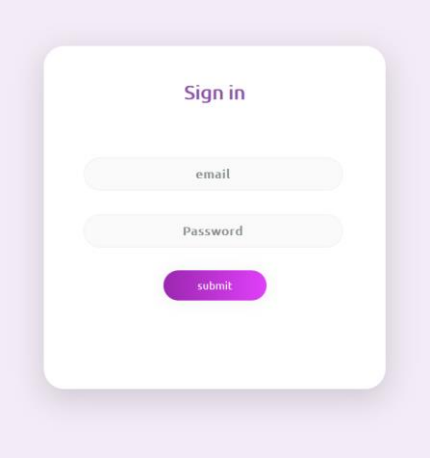
```
        myconn.close()
        return render_template("home.html", msg=msg)

@app.route("/signup/emp", methods=['POST'])
def emp():

    company_name = request.form['company_name']
    email = request.form['email']
    location = request.form['location']
    password = request.form['password']

    msg=''
    try:
        myconn = pymysql.connect(host='localhost', user = 'root', password = '',
database='flask_app')
        cursor = myconn.cursor()
        sql="insert into profiles_app(email,location,password,company_name)
values('{}','{}','{}','{}')".format(email,location,password,company_name)
        cursor.execute(sql)
        if cursor.rowcount > 0:
            myconn.commit()
            msg = "registration done"
        else:
            myconn.rollback()
            msg = "try again"
    except Exception as e:
        msg = '{}, try again'.format(e)
        myconn.rollback()
    finally:
        myconn.close()
        return render_template("home.html", msg=msg)
```

Sql database of profiles

username	email	dob	location	gender	password	company_name
si	sfd@gmail.com	2020-05-04	bangalore	female	gucchi	*NULL*
sid	sid@gmail.com	2020-10-08	sid@gmail.com	Female	123	*NULL*
nid	nid@gmail.com	2020-10-09	Mumbai	Female	90	*NULL*
lata	latabansal@gmail.com	1951-01-08	Bangalore	Female	lata	*NULL*
siddhi	siddhi@yahoo.com	2020-10-09	Ahmedabad	Female	123	*NULL*
ravi	ravi@gmail.com	2020-10-09	Bangalore	Male	123	*NULL*
lata	latavazecollege@gmail.com	2020-09-28	Mumabi	Female	12	*NULL*
NULL	careers@infosys.in	*NULL*	Bangalore	*NULL*	12	infosys

---- STEP 6----------

LOGIN AND LOGOUT WITH HELP OF COOKIES




```
@app.route('/login', methods=['GET','POST'])
def login():
    return render_template('login.html')

@app.route('/logininfo', methods=['POST'])
def logininfo():
    email = request.form['email']
    password = request.form['password']
    error = None
    try:
        myconn = pymysql.connect(host='localhost', user='root', password='',
database='flask_app')
        cursor = myconn.cursor()
        sql = "select * from profiles_app where email='{}' and
password='{}'".format(email, password)
        cursor.execute(sql)
        data = cursor.fetchall()
        if len(data) > 0:
            if data[0][6] == None:
                resp = make_response(render_template('dashboard.html'))
                resp.set_cookie('email', email)
            else:
                resp = make_response(render_template('dashboardemp.html'))
                resp.set_cookie('email', email)
            return resp
        else:
            return "invalid id and password"
    except Exception as e:
        msg = '{}, try again '.format(e)

    finally:
        myconn.close()
```

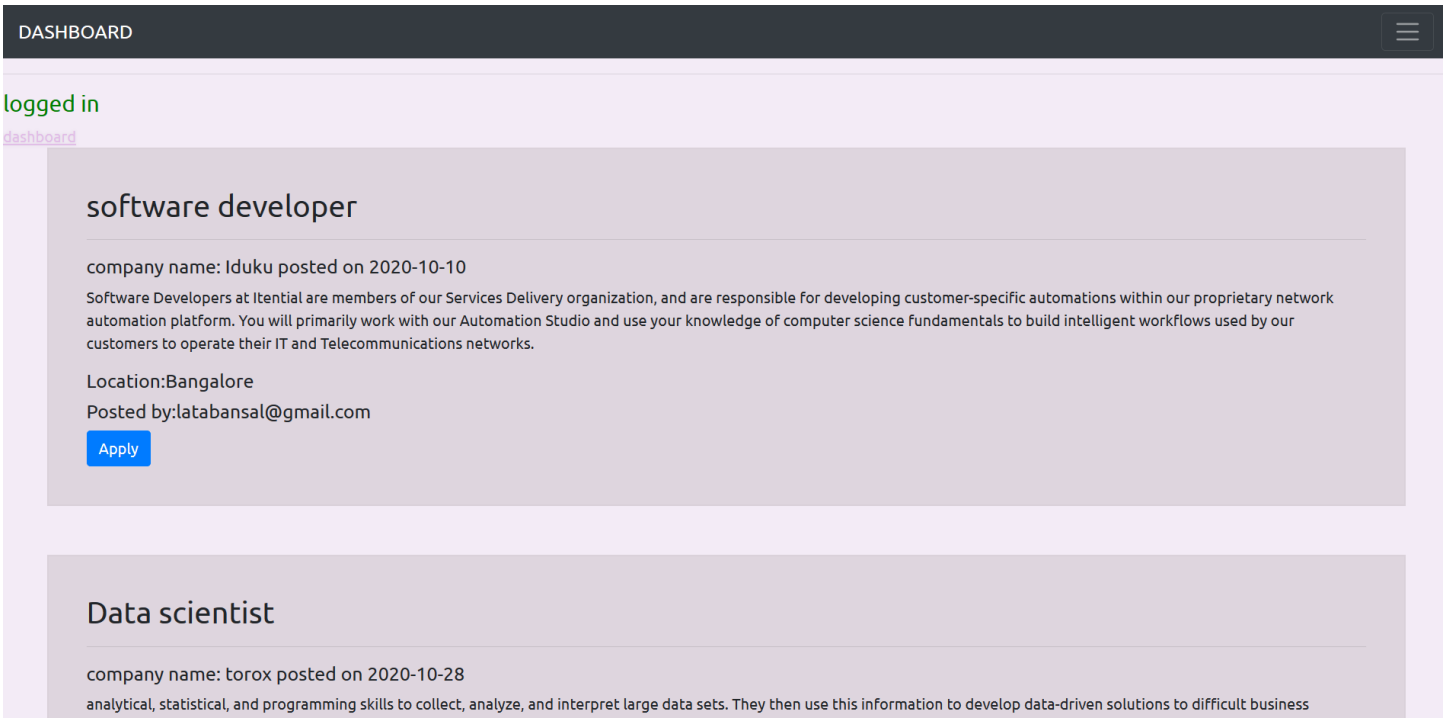
```
app.route('/logout')
def logout():
    email = request.cookies.get('email')
    if email:
        resp = make_response(render_template('home.html', name=email))
        resp.set_cookie('email','',expires=0)
        resp.delete_cookie('email')
        return resp
    else:
        resp = make_response(render_template('home.html', name=email))
        return resp
```

---- STEP 7----------

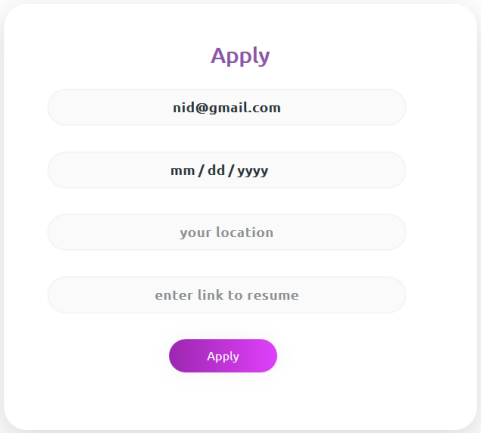
DASHBOARD FOR JOB-HUNTER

The dashboard shows the available job that have been posted by employers

The job hunter can apply for a job




```
def dashboard():


    myconn = pymysql.connect(host='localhost', user='root', password='',
database='flask_app')
    cursor = myconn.cursor()
    sql = "select * from dashboard_jobs"
    cursor.execute(sql)
    data = cursor.fetchall()
    return render_template('dashboard.html', data=data)

@app.route('/dashboardemp')
def dashboardemp():


    myconn = pymysql.connect(host='localhost', user='root', password='',
database='flask_app')
    cursor = myconn.cursor()
    sql = "select * from dashboard_jobs"
    cursor.execute(sql)
    data = cursor.fetchall()
    return render_template('dashboardemp.html', data=data)
```



email	dob	location	resume	job_id	posted_by
latabansal@gmail.com	2020-10-12	Andaman	http/resumemth//	5	latabansal@gmail.com
latavazecollege@gmail.com	2020-10-05	Mumbai	http/lataresume//	6	nid@gmail.com
nid@gmail.com	2020-10-08	Nainital	http/dcrfgv/rgverv	1	latabansal@gmail.com
ravi@gmail.com	2019-10-14	Chennai	http/myresume//	6	nid@gmail.com
siddhi@yahoo.com	2020-10-14	Mumabi	http/krviasucks/pdf	5	latabansal@gmail.com

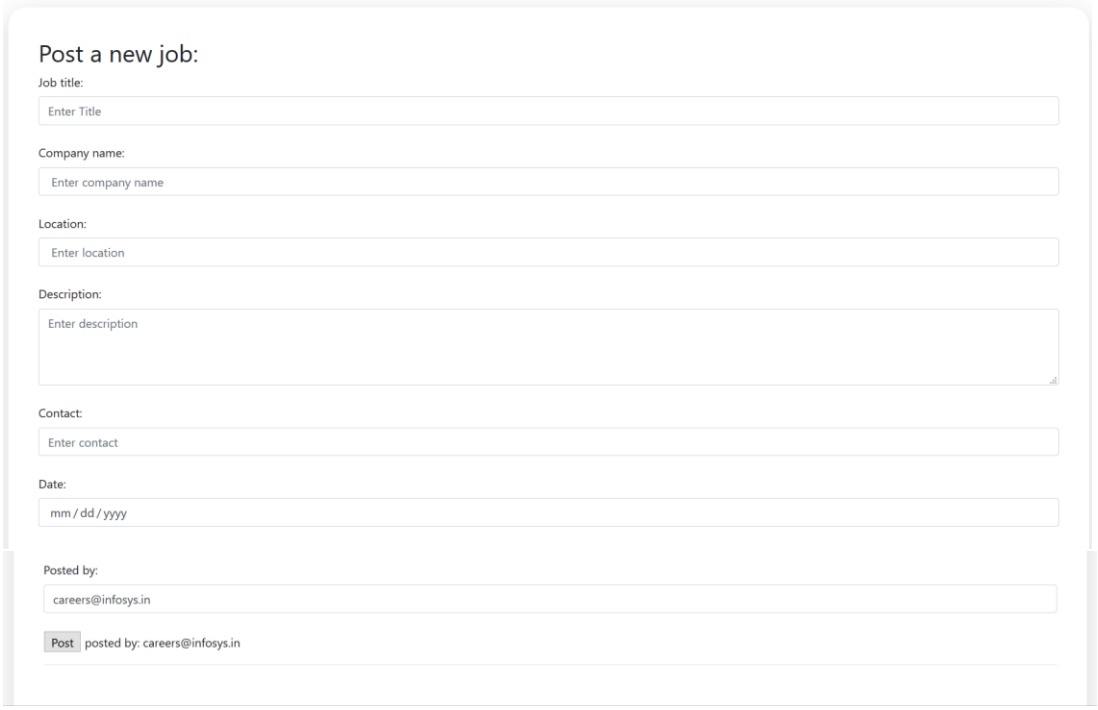
APPLICATIONS GET POSTED TO ANOTHER DATABASE CALLED APPLICATIONS ALONG WITH JOB ID

```
app.route('/apply/<int:job_id>', methods=['POST','GET'])
def apply(job_id):
    email = request.cookies.get('email')
    try:
        myconn = pymysql.connect(host='localhost', user='root', password='',
database='flask_app')
        cursor = myconn.cursor()
        cursor.execute("SELECT * FROM dashboard_jobs WHERE id={}".format(job_id))
        row = cursor.fetchone()
        if row:
            msg='successfull'
            if email:
                resp = make_response(render_template('apply.html', name=email, row=row))
            else:
                resp = make_response(render_template('dashboard.html'))
            return resp

    except Exception as e:
        msg = '{}, try again'.format(e)
        return render_template("error.html", msg=msg)
    finally:

        myconn.close()




@app.route('/application', methods=['POST','GET'])
def application():
    email = request.form['email']
    dob = request.form['dob']
    location = request.form['location']
    resume = request.form['resume']
    job_id = request.form['id']
    posted_by = request.form['posted_by']

    try:
        myconn = pymysql.connect(host='localhost', user='root', password='',
database='flask_app')
        cursor = myconn.cursor()
```

```
        sql = "insert into applications(email,dob,location,resume,job_id,posted_by)
values('{}','{}','{}','{}',{},'{}')".format(email, dob, location,resume,job_id,posted_by)
        cursor.execute(sql)
        if cursor.rowcount > 0:
            myconn.commit()
            msg = "job applied"
        else:
            myconn.rollback()
            msg = "try again"
    except Exception as e:
        msg = '{}, try again'.format(e)
        myconn.rollback()
    finally:
        myconn.close()
        return render_template("dashboard.html")
```

DASHBOARD FOR EMPLOYER

EMPLOYER CAN POST A JOB



SUBMITTING FORM AND UPLOADING TO DATABASE

```
def posted():

    title = request.form['title']
    company_name = request.form['company_name']
    location = request.form['location']
    description = request.form['description']
    contact = request.form['contact']
```

```
    date_posted = request.form['date_posted']
    posted_by = request.form['posted_by']
    msg=''
    try:
        myconn = pymysql.connect(host='localhost', user = 'root', password = '',
database='flask_app')
        cursor = myconn.cursor()
        sql="insert into
dashboard_jobs(title,company_name,location,description,contact,date_posted,posted_by)
values('{}','{}','{}','{}','{}','{}','{}')".format(title,company_name,location,descriptio
n,contact,date_posted,posted_by )
        cursor.execute(sql)
        if cursor.rowcount > 0:
            myconn.commit()
            msg = "job posted"
        else:
            myconn.rollback()
            msg = "try again"
    except Exception as e:
        msg = '{}, try again'.format(e)
        myconn.rollback()
    finally:
        myconn.close()
        return render_template("dashboard.html")
```

JOBS POSTED GET ADDED TO A DIFFERENT DATABASE 'DASHBOARD JOBS' WITH AN ID

title	company_name	location	description	contact	date_posted	id	posted_by
software developer	Iduku	Bangalore	Software Developers at Itential are members of our...	careers_info@Iduku.com	2020-10-10	1	latabansal@gmail.com
Data scientist	torox	Chennai	analytical, statistical, and programming skills to...	34567744	2020-10-28	4	*NULL*
Node.js Developer	lihib	Nainital	writing server-side web application logic in JavaS...	44356787898	2020-10-29	5	latabansal@gmail.com
Front end developer	SAas	Mumbai	website code with programming languages such as HT...	2458345	2020-10-21	6	nid@gmail.com
yhyrh	retghty	tryhty	tyhty	tyhty	2020-10-21	9	nid@gmail.com

EMPLOYER CAN CHECK PROFILE OF APPLICATIONS FOR THE JOB

For job id 6

Email: ravi@gmail.com
DOB: 2019-10-14
Location:Chennai
RESUME LINK: http/myresume//

For job id 6

Email: latavazecollege@gmail.com
DOB: 2020-10-05
Location:Mumbai
RESUME LINK: http/lataresume//

```
@app.route('/viewprofile')
def profile():
    email = request.cookies.get('email')
    myconn = pymysql.connect(host='localhost', user='root', password='',
database='flask_app')
    cursor = myconn.cursor()

    sql = "select * from applications where posted_by='{}'".format(email)
    cursor.execute(sql)
    data = cursor.fetchall()

    if email:
        resp = make_response(render_template('profile.html', name=email,data=data))
    else:
        resp = make_response(render_template('dashboard.html'))
    return resp
```