SKILL / JOB RECOMMENDER APPLICATION

**TEAM ID : ** PNT2022TMID27099

1. INTRODUCTION

1.1 Project Overview

Having lots of skills but wondering which job will best suit you? Don't need to worry! We have come up with a skill recommender solution through which the fresher or the skilled person can log in and find the jobs by using the search option or they can directly interact with the chatbot and get their dream job.

To develop an end-to-end web application capable of displaying the current job openings based on the user skillset. The user and their information are stored in the Database. An alert is sent when there is an opening based on the user skillset. Users will interact with the chatbot and can get the recommendations based on their skills. We can use a job search API to get the current job openings in the market which will fetch the data directly from the webpage.

1.2 Purpose

- To recommend job to user based on their skill set.

2. LITERATURE SURVEY

2.1 Existing problem

- In previous system did not have any communication with help centre.
- Applicants cannot find their jobs by their specified skills.

2.2 References

1. siva, murugan,abi, "Prediction of recommendations for employment utilizing machine learning procedures and geoarea based recommender framework ", Sustainable Operations and Computers , volume: 3, issue: 9, pp: 83-92, 2021.
2. Manal Alghieth, Amal A. Shargabi, "A Map-based Job Recommender Model", International Journal of Advanced Computer Science and Applications, volume: 10, issue: 9, pp: 345-351, 2019.
3. Shivraj Hulbatte, Amit Wabale, Suraj Patil, Nikhilkumar Sathe, "Enhanced Job Recommendation System ", International Journal of Research in Engineering , volume: 1, issue: 10, pp: 212-215, 2018.

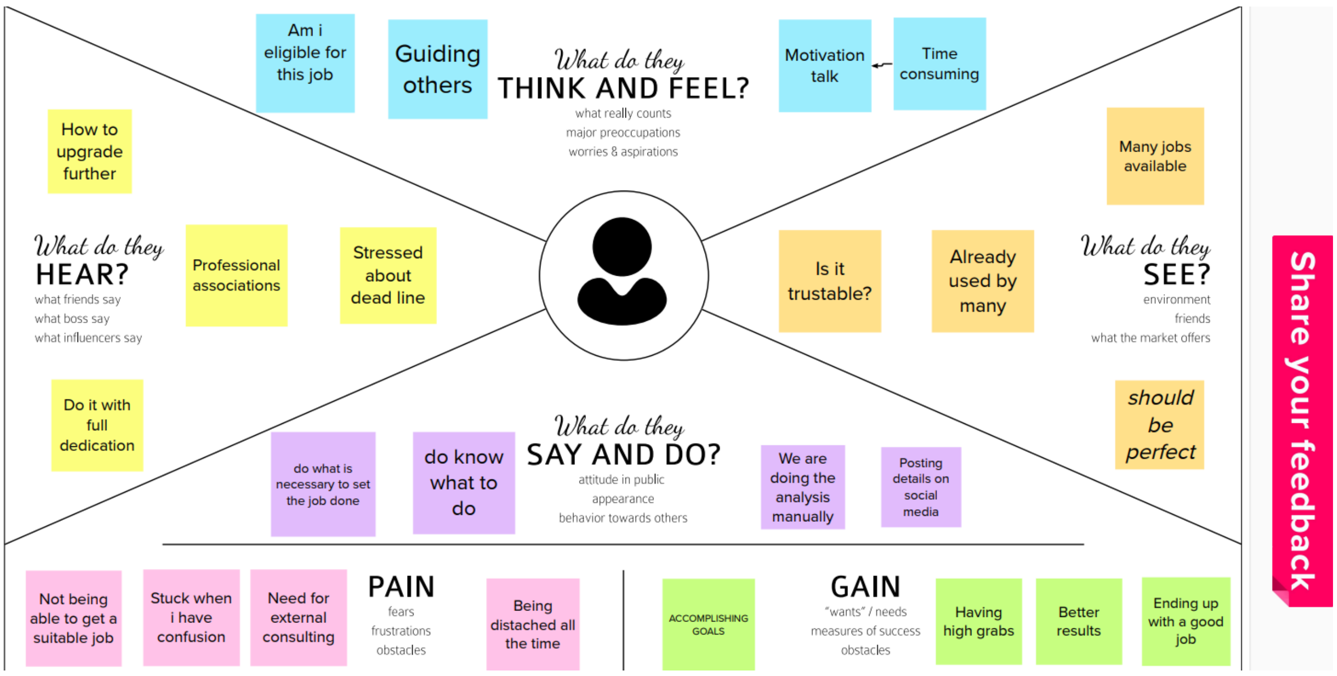
2.3 Problem Statement Definition

- The system requires applicants to search through print and visual media for job opportunities.
- This approach is hard and requires much effort and resources.

- There is need of an online job portal where applicants easily find the jobs and employer can find suitable candidates for the job.
- Most of the applicants are cheated by untrusted agency due to finding job.

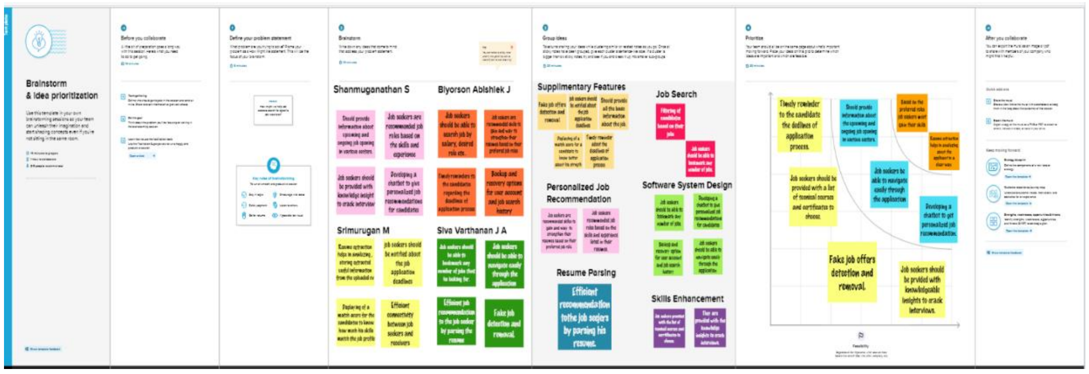
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming

- The system requires applicants to search through print and visual media for job opportunities.
- This approach is hard and requires much effort and resources.
- There is need of an online job portal where applicants easily find the jobs and employer can find suitable candidates for the job.



Brainstorm & idea prioritization
Shanmuganathan S
Biyorson Abishiek J
Srimurugan M
Siva Varthanan J A
Supplimentary Features
Job Search
Personalized Job Recommendation
Software System Design
Resume Parsing
Skills Enhancement
Efficient recommendation to the job seekers by parsing his resume.
Fake job offers detection and removal.

3.3 Proposed Solution

S.No.	Parameter	Description
1.	**Problem Statement (Problem to be solved)**	To help job seekers to find a best job related to their skill set. We plan this suggestion issue as a administered AI issue.
2.	**Idea / Solution description**	Our application will solve this problem and make things simpler and more effective at necessary moment • Alert using notification when a new relevant job is posted. • Filter jobs based on skills, location, salary, experience, etc. • API to enable users to search for various job openings. • Chatbots for recommendations. • Useful for all job seekers. .
3.	**Novelty / Uniqueness**	• Alert users about deadlines for the jobs that match their skill sets. Provide resources to stay up-to-date on their skills..
4.	**Social Impact/ Customer Satisfaction**	Social capability is especially striking for understudies who are arranged into one of the great frequency handicap gatherings like explicit learning inabilities, mental impediment, close to home unsettling influence or considaration shortfall/hyperactivity jumble. Among the most famous of the educational methodologies for these understudies has been interactive abilities preparing (SST). Different meta-examinations of the writing recommend that SST has not delivered enormous, socially significant, long haul, or summed up changes in friendly capability of

is

		understudies with high-rate handicaps. Likely clarifications for the powerless impacts in some meta-examinations are talked about and explicit proposals are presented for planning and creating more successful SST mediations.

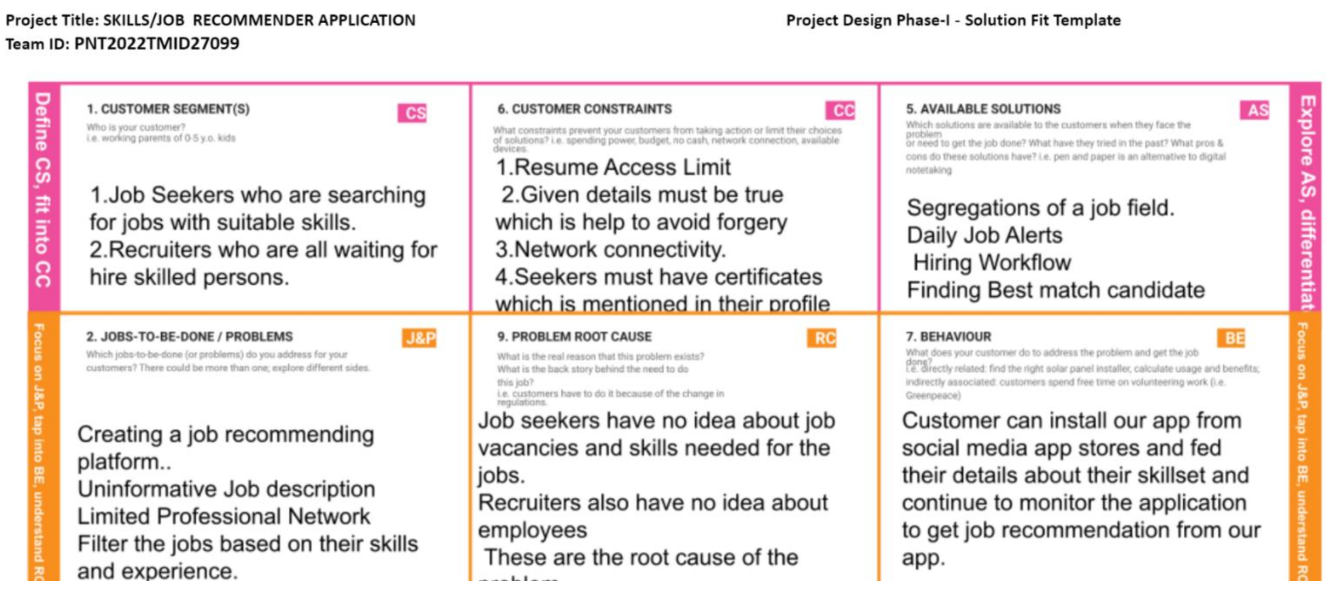
be

on

a

5.	**Business Model (Revenue Model)**	We can provide the application for job seekersin a subscription based.We can share the profiles with companies and generate the revenue by providing them bestprofiles.
6.	**Scalability of the Solution**	As the system grows (in data volume, traffic volume, or complexity), there should be reasonableways of dealingwith that growth. See “Scalability”.

3.4 Problem Solution fit



Project Title: SKILLS/JOB RECOMMENDER APPLICATION
Team ID: PNT2022TMID27099

Project Design Phase-I - Solution Fit Template

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

1.Job Seekers who are searching for jobs with suitable skills.
2.Recruiters who are all waiting for hire skilled persons.

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

1.Resume Access Limit
2.Given details must be true which is help to avoid forgery
3.Network connectivity.
4.Seekers must have certificates which is mentioned in their profile

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem
or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

Segregations of a job field.
Daily Job Alerts
Hiring Workflow
Finding Best match candidate

Explore AS, differentiat

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

Creating a job recommending platform..
Uninformative Job description
Limited Professional Network
Filter the jobs based on their skills and experience.

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

Job seekers have no idea about job vacancies and skills needed for the jobs.
Recruiters also have no idea about employees
These are the root cause of the

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

Customer can install our app from social media app stores and fed their details about their skillset and continue to monitor the application to get job recommendation from our app.

Focus on J&P, tap into BE, understand RC

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story/ Sub-Task)
FR-1	User registra on	Registra on through form
FR-2	Admin module	Administra on can get details about user and job recommender
FR-3	Job provider	Post jobs withdetailed job descrip on

4.2 Non-Functional requirements

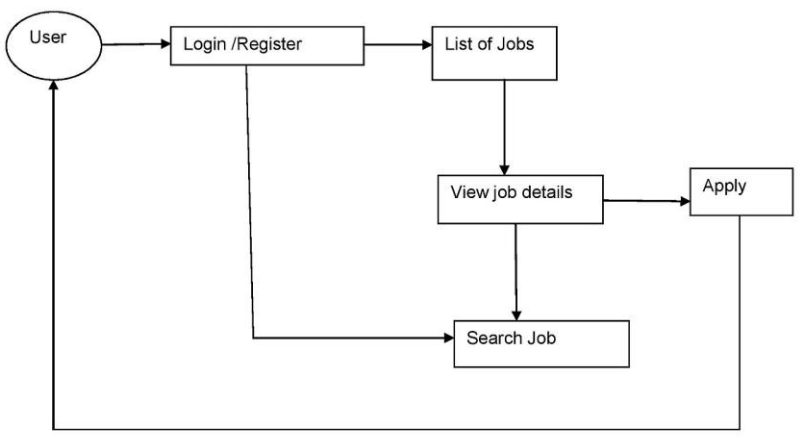
FR No.	Non-Functional Requirement	Descrip on
NFR-1	**Usability**	Ease of use and knows the details immediately.
NFR-2	**Security**	The applica on whichprotects the dataefficiently over the web.
NFR-3	**Reliability**	The applica on can be usedin a confiden al manner.
NFR-4	**Performance**	The performance of the applica on is very effec ve.
NFR-5	**Availability**	The applica on which Can be easyto access.
NFR-6	**Scalability**	User access meis less, so that applica on is scalable.

5. PROJECT DESIGN

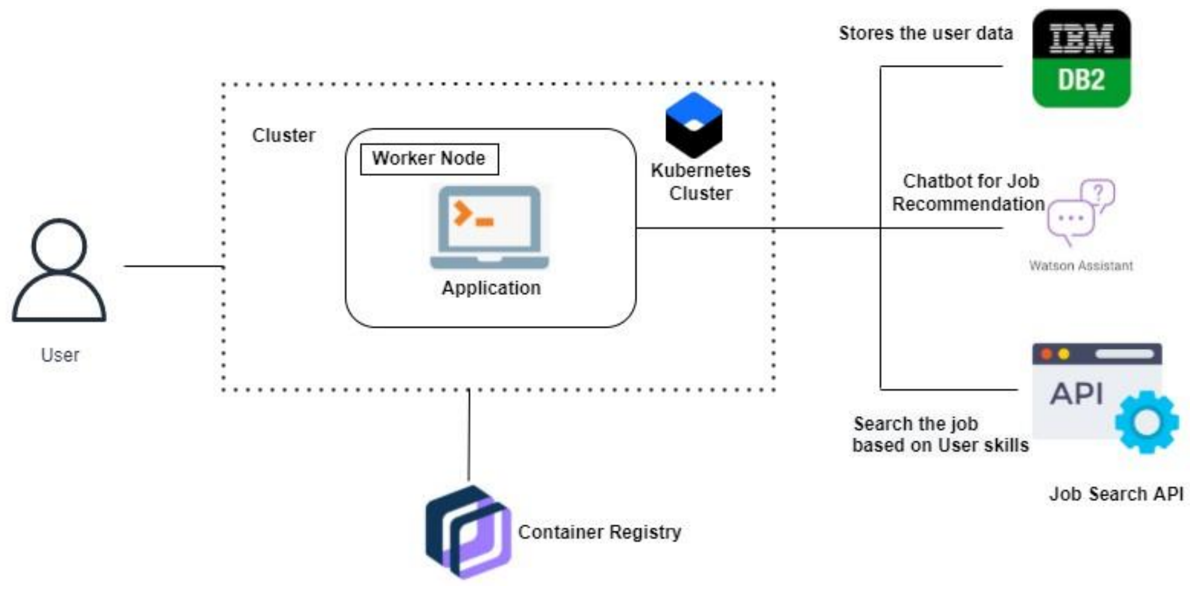
5.1 Data Flow Diagrams

Data Flow Diagrams:

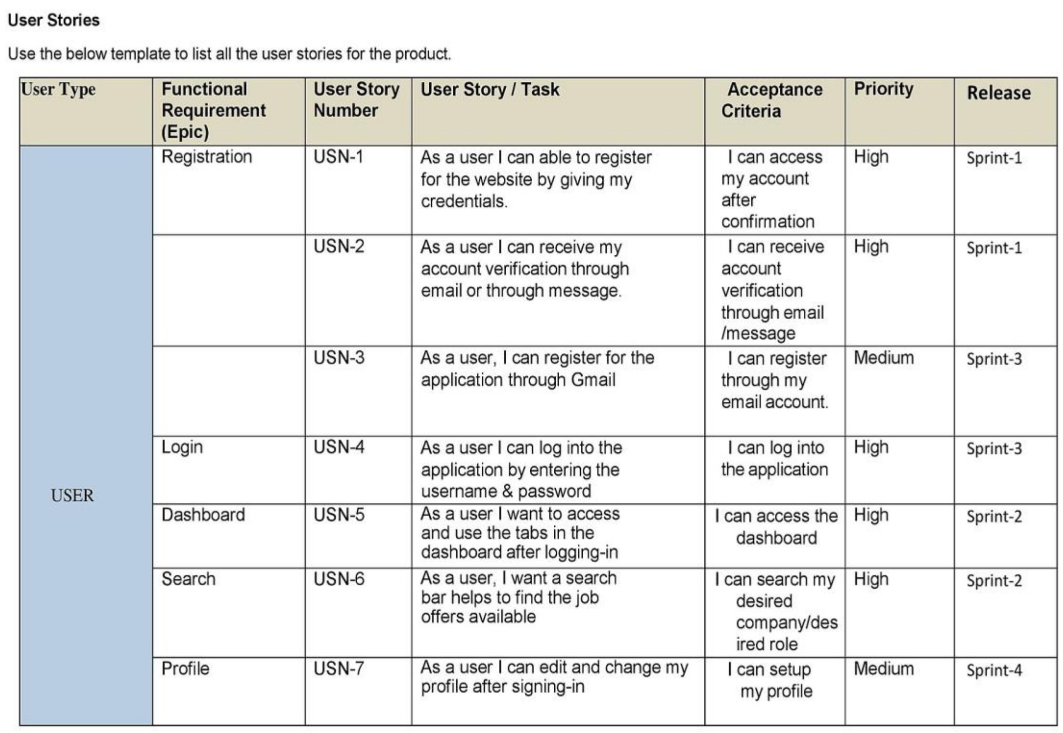
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 Solution & Technical Architecture

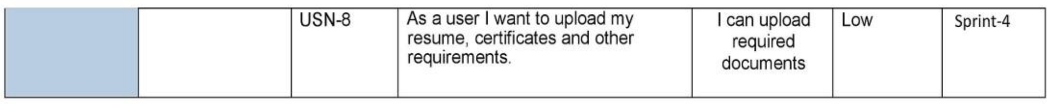


5.3 User Stories



User Stories

Use the below template to list all the user stories for the product.



User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
USER	Registration	USN-1	As a user I can able to register for the website by giving my credentials.	I can access my account after confirmation	High	Sprint-1
		USN-2	As a user I can receive my account verification through email or through message.	I can receive account verification through email /message	High	Sprint-1
		USN-3	As a user, I can register for the application through Gmail	I can register through my email account.	Medium	Sprint-3
	Login	USN-4	As a user I can log into the application by entering the username & password	I can log into the application	High	Sprint-3
	Dashboard	USN-5	As a user I want to access and use the tabs in the dashboard after logging-in	I can access the dashboard	High	Sprint-2
	Search	USN-6	As a user, I want a search bar helps to find the job offers available	I can search my desired company/desired role	High	Sprint-2
	Profile	USN-7	As a user I can edit and change my profile after signing-in	I can setup my profile	Medium	Sprint-4
		USN-8	As a user I want to upload my resume, certificates and other requirements.	I can upload required documents	Low	Sprint-4

6.

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Title	**Descrip on**
Informa on Gathering Literature Survey	Referring to the research publica ons & technical papers, etc.
Create EmpathyMap	Preparing the List of Problem Statements and to capture user pain and gains.
Idea on	Priori es a top ideasbased on feasibility and Importance.
Proposed Solu on	Solutions including feasibility, novelty, social impact, business model and scalability of solu ons.
Problem Solu on Fit	Solu on fit document.
Solu on Architecture	Solu on Architecture.
Customer Journey	To Understand User Interac ons andexperiences with applica on.
Func onal Requirement	Prepare func onal Requirement.
Data flow Diagrams	Data flow diagram.
Technology Architecture	Technology Architecture diagram.
Milestone & sprintdelivery plan	Ac vi es are done& further plans.
Project Development Delivery of sprint1,2,3 & 4	Develop and submit thedeveloped code by tes ng it.

Sprint	Func onal Requireme nt (Epic)	User Story Number	User Story / Task	Priority	Acceptance criteria	Team Members
Spri nt-1	UI Design	USN-1	As a user, I can see and experience an awesome userinterface in the website	Medium	Be er Impress ion about a website	Shanmuganathan s Biyorson abishiek j Sri murugan m Siva varthanan j a

Sprint-1	Registra on	USN-2	As a user,I can register for the applica on by entering my email, password, and confirming my password.	High	Ican access myaccount / dashboard	Shanmuganathan s Biyorson abishiek j Sri murugan m Siva varthanan j a
Sprint-1		USN-3	As a user, I will receiveconfirma on email onceI have registered for the applica on	High	I can receive confirmation email &click confirm	Shanmuganathan s Biyorson abishiek j Sri murugan m Siva varthanan j a
Sprint-1		USN-4	As a user, I can register for the applica on through Facebook	Low	I can register &access the dashboard with Facebook Login	Shanmuganathan s Biyorson abishiek j Sri murugan m Siva varthanan j a
Sprint-1		USN-5	As a user, I can register for the applica on through Gmail	Medium	I can receive confirmation email & click confirm	Shanmuganathan s Biyorson abishiek j Sri murugan m Siva varthanan j a

6.2 Sprint Delivery Schedule

7. SOURCE CODE

App.py from flask
import * import os
import ibm_db
import bcrypt
from functools import partial,wraps

```
conn = ibm_db.connect('DATABASE=bludb;HOSTNAME=fbd88901-ebdb-4a4f-
a32e9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32731;SECURITY=SSL;SSL
Serve
rCertificate=DigiCertGlobalRootCA.crt;PROTOCOL=TCPIP;UID=rwb71462;PWD=XGg6dmPJELe3XGE
K',",")
app = Flask(__name__) app.secret_key
=
'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'
PEOPLE_FOLDER = os.path.join('static', 'people_photo')
#IN THIS FILE ONLY SPRINT-2 ACTIONS ARE AVAILABLE
@app.route("/")
def login():

 return render_template('login.html')
@app.route('/login_api',methods=['GET','POST']) def
login_api():
 query1_e = 'SELECT * FROM users WHERE email = ?'
 query2_e = 'SELECT * FROM orgusers WHERE email = ?'
 email = request.form['email']
 password = request.form['password']
 stm_e1 = ibm_db.prepare(conn,query1_e)
 ibm_db.bind_param(stm_e1,1,email)
 res = ibm_db.execute(stm_e1)
 d = ibm_db.fetch_assoc(stm_e1)
 stm_e2 = ibm_db.prepare(conn,query2_e)
 ibm_db.bind_param(stm_e2,1,email)
 res = ibm_db.execute(stm_e2)
 d1 = ibm_db.fetch_assoc(stm_e2)
 print(d)
 print(d1)
 print(email)
 print(password)
if d!=False:
 if password == d['PASSWORD'] :
 return redirect(url_for('home'))
if d1!=False:
 if password == d1['PASSWORD']:
 return redirect(url_for('jobpost'))
 return "<div style='margin: 300px;'><center><div><h2>Invalid Data</h2></div></center></div>"
@app.route("/register") def
register():
 return render_template('register.html')

@app.route("/register_api",methods=['GET','POST']) def
register_api():
 fname = request.form['firstname']
 lname = request.form['lastname']
 dob = request.form['dob']
 qlf = request.form['qlf']
 skills = request.form['skills']
 email = request.form['email']
 password = request.form['password']
 query = 'insert into users (fname,lname,dob,qlf,skills,email,password) VALUES(?,?,?,?,?,?,?)'
 stm = ibm_db.prepare(conn,query)
 ibm_db.bind_param(stm,1,fname)
 ibm_db.bind_param(stm,2,lname)
 ibm_db.bind_param(stm,3,dob)
 ibm_db.bind_param(stm,4,qlf)
 ibm_db.bind_param(stm,5,skills)
 ibm_db.bind_param(stm,6,email)
 ibm_db.bind_param(stm,7,password)
 ibm_db.execute(stm)
 return 'success'
@app.route("/orgregister")
def orgregister():
 return render_template('orgregister.html')
@app.route("/orgregister_api",methods=['GET','POST']
) def orgregister_api():  email = request.form['email']
 password = request.form['password']
 query = 'INSERT INTO orgusers(email,password) VALUES(?,?)'
 stm = ibm_db.prepare(conn,query)
 ibm_db.bind_param(stm,1,email)
 ibm_db.bind_param(stm,2,password)
 ibm_db.execute(stm)
 return 'success'
@app.route("/jobpost") def
jobpost():

 return render_template('jobpost.html')
#----------SPRINT-1-----------------
# @app.route('/logout') def
logout():
 session.pop('loggedin', None)
 session.pop('id', None)
 return redirect(url_for('home'))
@app.route('/success') def
suc():
 return render_template('suc.html')
@app.route('/home') def home():
 query = 'SELECT * FROM job'
 stm = ibm_db.prepare(conn,query)
 res = ibm_db.execute(stm)
 d = ibm_db.fetch_assoc(stm)
 title = d['TITLE']
 skills = d['SKILLS']
 loc = d['LOC']
 sal = d['SALARY']
 return render_template('home.html',title=title,skills=skills,loc=loc,sal=sal)
@app.route('/postmsg',methods=['GET','POST']) def
postmsg():
 tit = request.form['jobtitle']
 skills = request.form['keyskills']
 location = request.form['location']
 salary = request.form['salary']
 query = 'insert into job(title,skills,loc,salary) VALUES(?,?,?,?)'
 stm = ibm_db.prepare(conn,query)
 ibm_db.bind_param(stm,1,tit)
 ibm_db.bind_param(stm,2,skills)
 ibm_db.bind_param(stm,3,location)
 ibm_db.bind_param(stm,4,salary)
 ibm_db.execute(stm)
 return render_template('postmsg.html')
@app.route("/orgregister",methods=['GET','POST'])
#----------SPRINT-1-----------------#
@app.route("/user_dashboard")
#----------SPRINT-1-----------------#
@app.route("/login",methods=['GET','POST'])
#----------SPRINT-1-----------------
# @app.route('/browse') def
addMarker():
 if 'loggedin' in session:
 query = "SELECT * FROM jobpost;"
 stmt = ibm_db.prepare(conn, query)
 ibm_db.execute(stmt)
 a=[]
 isUser = ibm_db.fetch_assoc(stmt)

 while(isUser!=False):
 a.append(isUser)
 isUser = ibm_db.fetch_assoc(stmt)
else:
 return redirect(url_for('login'))
 return render_template("browse.html",result=a)
```

```
@app.route('/companies') def
companies():
 if 'loggedin' in session:
 query = "SELECT * FROM RECRUITER"
 stmt = ibm_db.prepare(conn, query)
 ibm_db.execute(stmt)
 a=[]
 isUser = ibm_db.fetch_assoc(stmt)

 while(isUser!=False):
 a.append(isUser)
 isUser = ibm_db.fetch_assoc(stmt)
else:
 return redirect(url_for('login'))
 return render_template("companies.html",result=a)
@app.route("/jobpost1",methods=['GET','POST'])
def jobpost1():
 if 'loggedin' in session:
 if request.method == 'POST':
 recruiterid=request.form['recruiter_id']
 jobtitle = request.form['jobtitle']
 jobtype = request.form['jobtype']
 jobexp=request.form['jobexperience']
 keyskill=request.form['keyskills']
 location=request.form['location']
 salary=request.form['salary']
 discription=request.form['discription']
 insert_sql = "INSERT INTO JOBPOST (RECRUITER_ID,JOBTITLE, JOBTYPE, EXPERIENCE,
KEYSKILL,
LOCATION, SALARY, DISCRIPTION) VALUES (?,?,?,?,?,?,?,?)"
 prep_stmt = ibm_db.prepare(conn, insert_sql)
 ibm_db.bind_param(prep_stmt, 1, recruiterid)
 ibm_db.bind_param(prep_stmt, 2, jobtitle)
 ibm_db.bind_param(prep_stmt, 3, jobtype)
 ibm_db.bind_param(prep_stmt, 4, jobexp)
 ibm_db.bind_param(prep_stmt, 5, keyskill)
 ibm_db.bind_param(prep_stmt, 6, location)
 ibm_db.bind_param(prep_stmt, 7, salary)
 ibm_db.bind_param(prep_stmt, 8, discription)
 ibm_db.execute(prep_stmt)
 else:
 return redirect(url_for('login'))
 return render_template("jobpost.html")
```

```
@app.route("/browse/searchjob",methods=['GET','POST']) def
searchjob():
 if request.method=='POST':
 searchopt=request.form['searchopt']
 srctitle=request.form['srctitle']
 query = "SELECT * FROM JOBPOST WHERE "+searchopt+"="+chr(39)+srctitle+chr(39)
 stmt = ibm_db.prepare(conn, query)
 ibm_db.execute(stmt)
 a=[]
 isUser = ibm_db.fetch_assoc(stmt)

 while(isUser!=False):
 a.append(isUser)
 isUser = ibm_db.fetch_assoc(stmt)
 return render_template('browse.html',result=a) if __name__ == "__main__": #checking if
__name__'s value is '__main__'. __name__ is an python environment variable who's value will
always be '__main__' till this is the first instatnce of app.py running
 app.run(debug=True,port=8080) #running flask (Initalised on line 4)
orgregister.html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<link rel="stylesheet" href="../static/style/orgregister.css">
 <link rel="stylesheet" type="text/css"
href="https://fonts.googleapis.com/css?family=Rancho&effect=shadow-multiple">
 <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-
9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk"
crossorigin="anonymous">
 <link rel="stylesheet" type="text/css"
href="https://stackpath.bootstrapcdn.com/fontawesome/4.7.0/css/font-
awesome.min.css">  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js">  </script>
 <link href="https://cdn.jsdelivr.net/npm/select2@4.1.0-rc.0/dist/css/select2.min.css" rel="stylesheet"
/>
 <script src="https://cdn.jsdelivr.net/npm/select2@4.1.0-rc.0/dist/js/select2.min.js"></script>
<title></title>
</head>
<body>
<img src="{{url_for('static',filename = 'images/registerpg.jpg')}}"><br><br><br>
<center>
<h3>Registration on Jobby</h3><Br>
</center>
<form name="form_org" action="/orgregister_api" method="POST">
```

```
 <div id="Accountinfo">
 <div class="heading" id="head1">
 <div class="verbor"></div>
 <h4>Account Info</h4>
 </div>
 <div class="row Accountinfo">
 <div class="col-md-4 regr1col1">
 <label>Organization E-mail Id <a id="requi">*</a></label><br>
 <input type="email" name="email" placeholder="Enter the mail address">
 </div>
 <div class="col-md-4 regr1col2">
 <label>Password <a id="requi">*</a></label><br>
 <input type="password" name="password" placeholder="Enter the password">
</div>
 <div class="col-md-4 regr1col2">
 <label>Confirm Password <a id="requi">*</a></label><br>
 <input type="password" name="password1" placeholder="Confirm password">
</div>
</div><Br><Br>
 <center>
 <button type="submit" id="button">Save & Continue</a>
 </center>

 </div>
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<link rel="stylesheet" href="../static/style/register.css">
 <link rel="stylesheet" type="text/css"
href="https://fonts.googleapis.com/css?family=Rancho&effect=shadow-multiple">
 <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-
9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk"
crossorigin="anonymous">
 <link rel="stylesheet" type="text/css"
href="https://stackpath.bootstrapcdn.com/fontawesome/4.7.0/css/font-awesome.min.css">
 <link rel="shortcut icon" href="../static/images/j.png" type="image/x-icon">
 <title>Jobby - Register</title>
 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
 <link href="https://cdn.jsdelivr.net/npm/select2@4.1.0-rc.0/dist/css/select2.min.css" rel="stylesheet"
/>
 <script src="../static/js/register.js"></script>
 <script src="https://cdn.jsdelivr.net/npm/select2@4.1.0-rc.0/dist/js/select2.min.js"></script>
```

```
<title></title>
</head>
<body>
<img src="{{url_for('static',filename = 'images/registerpg.jpg')}}"><br><br><br>
<center>
<h3>Registration on Jobby</h3><Br>
</center>
<form name="form" action="/register_api" method="POST" enctype="multipart/form-data">
 <div id="persondet">
 <div class="heading" id="head2">
 <div class="verbor" id="verbor"></div>
 <h4>Personal Details</h4>
 </div>
 <br>
 <div class="row persondet">
 <div class="col-md-4 regr1col1">
 <label>First Name <a id="requi">*</a></label><br>
 <input type="text" name="firstname" placeholder="Enter your first name">
 </div>
 <div class="col-md-4 regr1col2">
 <label>Last Name <a id="requi">*</a></label><br>
 <input type="text" name="lastname" placeholder="Enter your last name">
 </div>
 <div class="col-md-4 regr1col2">
 <label>Date Of Birth: <a id="requi">*</a></label><br>
 <input type="text" name="dob" placeholder="Enter your DOB">
 </div>
 <div class="col-md-4 regr1col2">
 <label>Qualification:<a id="requi">*</a></label><br>
 <input type="text" name="qlf" placeholder="Enter your qualification">
 </div>
 <div class="col-md-4 regr1col2">
 <label>Skills:<a id="requi">*</a></label><br>
 <input type="text" name="skills" placeholder="Enter your Skills">
 </div>
 </div><Br>
 <div id="Accountinfo">
 <div class="heading" >
 <div class="verbor" id="verbor"></div>
 <h4>Account Info</h4>
 </div>
 <div class="row Accountinfo">
 <div class="col-md-4 regr1col1">
 <label>E-mail Id <a id="requi">*</a></label><br>
 <input type="email" name="email" placeholder="Enter the mail address">
 </div>
 <div class="col-md-4 regr1col2">
 <label>Password <a id="requi">*</a></label><br>
 <input type="text" name="password" placeholder="Enter the mail address">
</div>

</div><Br><Br>
 <center>
 <button type="submit" >Save & Continue</button>
 </center>

 </div>
</form>
<center>
 <p id="warning"></p>
</center><br><br>
```

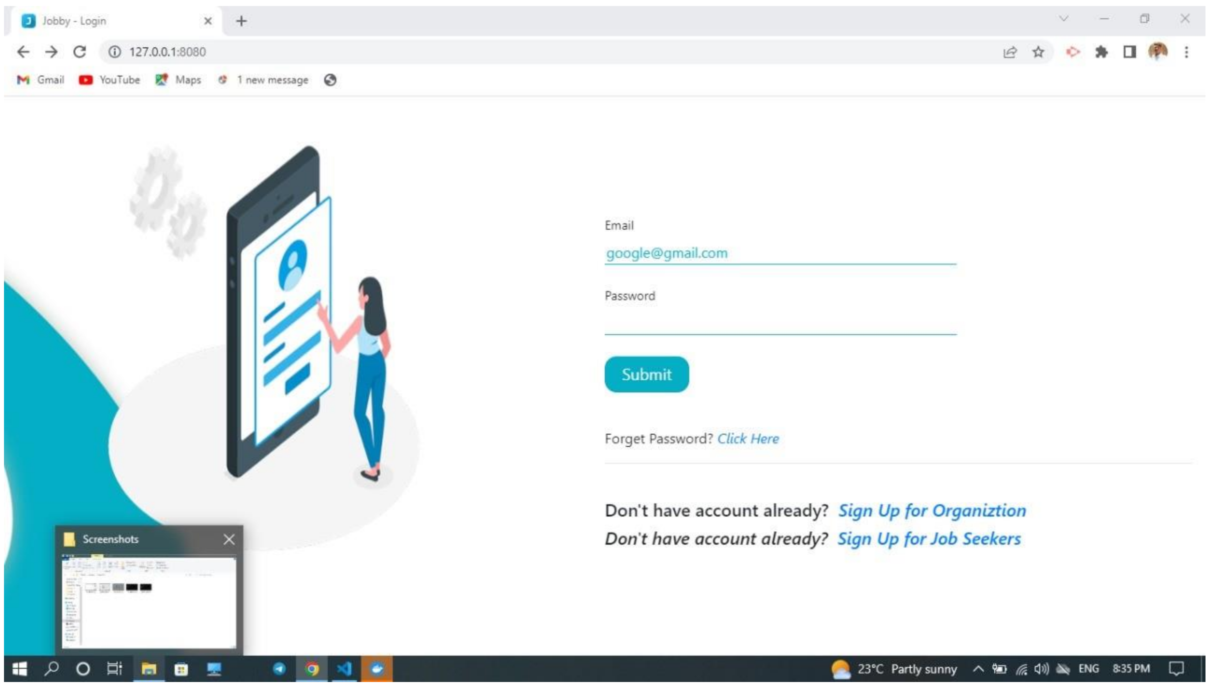
Job post.html

```
<head>
<meta charset="utf-8">
 <link rel="stylesheet" type="text/css"
href="https://fonts.googleapis.com/css?family=Rancho&effect=shadow-multiple">
 <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-
9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk"
crossorigin="anonymous">
 <link rel="stylesheet" type="text/css"
href="https://stackpath.bootstrapcdn.com/fontawesome/4.7.0/css/font-awesome.min.css">
 <link rel="shortcut icon" href="../static/images/j.png" type="image/x-icon">

 <link href="https://cdn.jsdelivr.net/npm/select2@4.1.0-rc.0/dist/css/select2.min.css" rel="stylesheet"
/>

<title></title>
</head>
<br><br><br><br><br><br><br><br>
<h4>Post Job</h4>
<br><br><bR><br>
 <div id="jobpost">
 <center>
 <form method="POST" onsubmit="required()" action="/postmsg" name="form">
 <div class="row">
 <div class="col-md-4">
 <label>Job Title</label><br>
 <input type="text" name="jobtitle"><br>
 </div>
 </div><br><br>
 <div class="row">
 <div class="col-md-4">
 <label>Key Skills</label><br>
 <input type="text" name="keyskills"><br>
 </div>
 <div class="col-md-4">
 <label>Location</label><br>
 <input type="text" name="location"><br>
 </div>
 </div><br><br>
 <div class="row" style="margin-left: -0px;">
 <div class="col-md-4">
 <label>Salary</label><br>
 <input type="text" name="salary"><br>
 </div>
 </div><br><br>
<!-- <table>
 <tr>
 <td>
 <label>Recruiter ID</label>
 <input type="text" name="recruiter_id" value="{{session['id']}}"
readonly>  </td>
 <td>
 <label>Job Title</label>
 <input type="text" name="jobtitle">
 </td>
 <td>
 <label>Job Type</label>
 <input type="text" name="jobtype">
 </td>
 <td>
 <label>Experience</label>
 <input type="text" name="jobexperience">
 </td>
 </tr>
<tr>
<td>
 <label>Key Skills</label>
 <input type="text" name="keyskills">
 </td>
 <td>
 <label>Location</label>
 <input type="text" name="location">
 </td>
 <td>
 <label>Salary</label>
 <input type="text" name="salary">
 </td>
</tr>
<tr>
<td><label>Discription</label><br>
<textarea type="text" name="discription"></textarea></td>
</tr>
 </table> -->
 <button type="submit" value="submit" id="button" style="margin-left:100px; margin-
bottom: 100px;">Post Job</button>
 </form>
 </center>
 </div>
<style type="text/css"> h4{
margin-top: -100px;
padding: 10px; color:white;
text-align: center;
background-color: #04aec4;
} textarea{
height: 190px;
width: 800px;
} input{ padding-
top: 10px; border:
none; outline:
none; color: black;
width: 300px;
box-shadow: 1px 0px 4px rgb(99, 99, 99);
 border-radius: 6px 6px 0 0;
background-color:white; border-bottom-
style: solid; border-width: 1px;
}
 form{
 margin-top: -100px;
 background-color: whitesmoke;
 }
 #button{
 border: none;
 color: white;
 padding: 7px 20px;
 border-radius: 13.5px;
 text-align: center;
 text-decoration: none;
 display: inline-block;
 font-size: 18px;
 background-color:#04aec2;
 cursor: pointer;

 }
 #button:hover,#explore:hover{
 opacity: 0.5;
 }
</style>
<script type="text/javascript">
var orgdet = document.getElementById("orgdet");
var extra = document.getElementById("extra");
var
jobpost=document.getElementById("jobpost");
orgdet.style.display = "";
extra.style.display="none";
jobpost.style.display="none"; function
orgdetFun(){ orgdet.style.display = "";
extra.style.display="none";
jobpost.style.display="none";
}
function extraDetFun2(){ orgdet.style.display
= "none"; extra.style.display="";
jobpost.style.display="none";
}
function postFun(){ orgdet.style.display
= "none"; extra.style.display="none";
jobpost.style.display="";
}
</script>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/select2@4.1.0-rc.0/dist/js/select2.min.js"></script>
Home.html
<head>
 <link rel="stylesheet" type="text/css"
href="https://fonts.googleapis.com/css?family=Rancho&effect=shadow-multiple">  <link
rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-
9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk"
crossorigin="anonymous">
 <link rel="stylesheet" href="../static/style/home.css">
 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
 <script src="{{url_for('static',filename = 'js/country.js')}}"></script>
</head>
<!-- <h2>Home Page</h2>
<p>This is the home page!</p>
<form action="/account" method="POST">
 <input type="text" name="name" id="name" placeholder="Your Name" class="form-feild"><br>
 <button type="submit" class="add-btn">See Account Page</button>
</form>
<br> -->
<!--
<p>Note how this page can be accesed on both '/' and '/home'. Look into app.py for more
information</p> -->
<!------------------------------------------------------------------------------------------------------------> <center>
 <h4>Recent Jobs</h4><br><br>
 </center>
<div class="recjobs">
 <div class="setcard">
 <div class="jobcard">
 <div class="set-1">
 <p>Amazon</p>
 </div><br>
 <p>London, UK</p><hr>
 <h5>UI/UX Designer</h5>
 <h6>$30k - $50k</h6>
 <div class="skillneed">
 <p>Adobe XD</p>
 <p>Figma</p>
 </div><br>
 <a href="/success" >Apply</a>
 </div>
 </div>
 <div class="setcard">
 <div class="jobcard">
 <div class="set-1">
 <p>Whatsapp</p>
 </div><br>
 <p>Washington, America</p><hr>
 <h5>Program Analyst</h5>
 <h6>$43k - $75k</h6>
 <div class="skillneed">
 <p>C</p>
 <p>Java</p>
 </div><br>
```

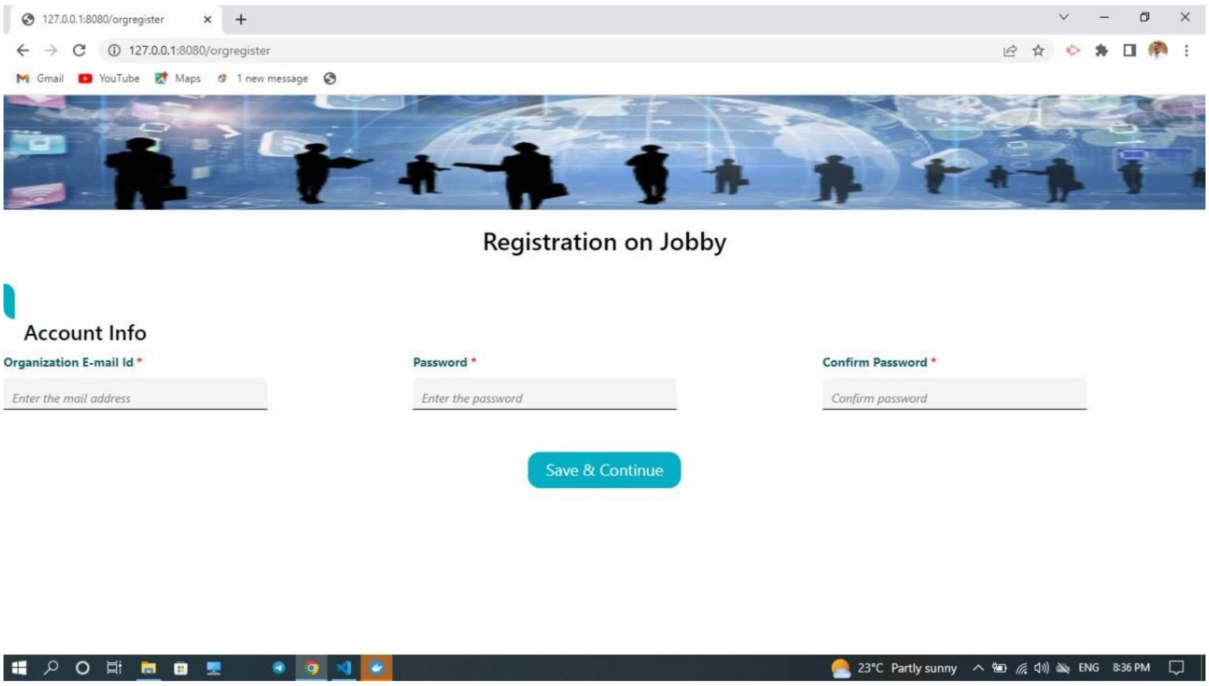
```
<a href="/success" >Apply</a>
</div>
</div>
<div class="setcard">
<div class="jobcard">
<div class="set-1">
<p>Amazon</p>
</div><br>
<p>London, UK</p><hr>
<h5>UI/UX Designer</h5>
<h6>$30k - $50k</h6>
<div class="skillneed">
<p>Adobe XD</p>
<p>Figma</p>
</div><br>
<a href="/success" >Apply</a>
</div>
</div>
<div class="setcard">
<div class="jobcard">
<div class="set-1">
<p>Amazon</p>
</div><br>
<p>London, UK</p><hr>
<h5>UI/UX Designer</h5>
<h6>$30k - $50k</h6>
<div class="skillneed">
<p>Adobe XD</p>
<p>Figma</p>
</div><br>
<a href="/success" >Apply</a>
</div>
</div>

<div class="setcard">
<div class="jobcard">
<div class="set-1">  <p>{{title}}</p>
{{sal}}
</div><br>
<p>{{loc}}</p><hr>
<div class="skillneed">
{{skills}}
</div><br>
<a href="/success" >Apply</a>
```

```
 </div>
 </div>
</div>
<!-------------------------------------SECTION - 6------------->
<script>
 window.watsonAssistantChatOptions = {
 integrationID: "f6b94708-b701-4a79-bac8-dae0cfe1403c", // The ID of this integration.
 region: "us-south", // The region your integration is hosted in.
 serviceInstanceID: "06cbeaa1-b28e-41d9-9414-e5593c7c2f6c", // The ID of your service instance.
 onLoad: function(instance) { instance.render(); }
 };
 setTimeout(function(){
 const t=document.createElement('script');
 t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";
document.head.appendChild(t);
 });
 </script>
```

SCREENSHOTS



Email
google@gmail.com
Password
Submit
Forget Password? *Click Here*
Don't have account already? *Sign Up for Organiztion*
Don't have account already? *Sign Up for Job Seekers*


127.0.0.1:8080/orgregister

Registration on Jobby

Account Info

Organization E-mail Id *

Enter the mail address

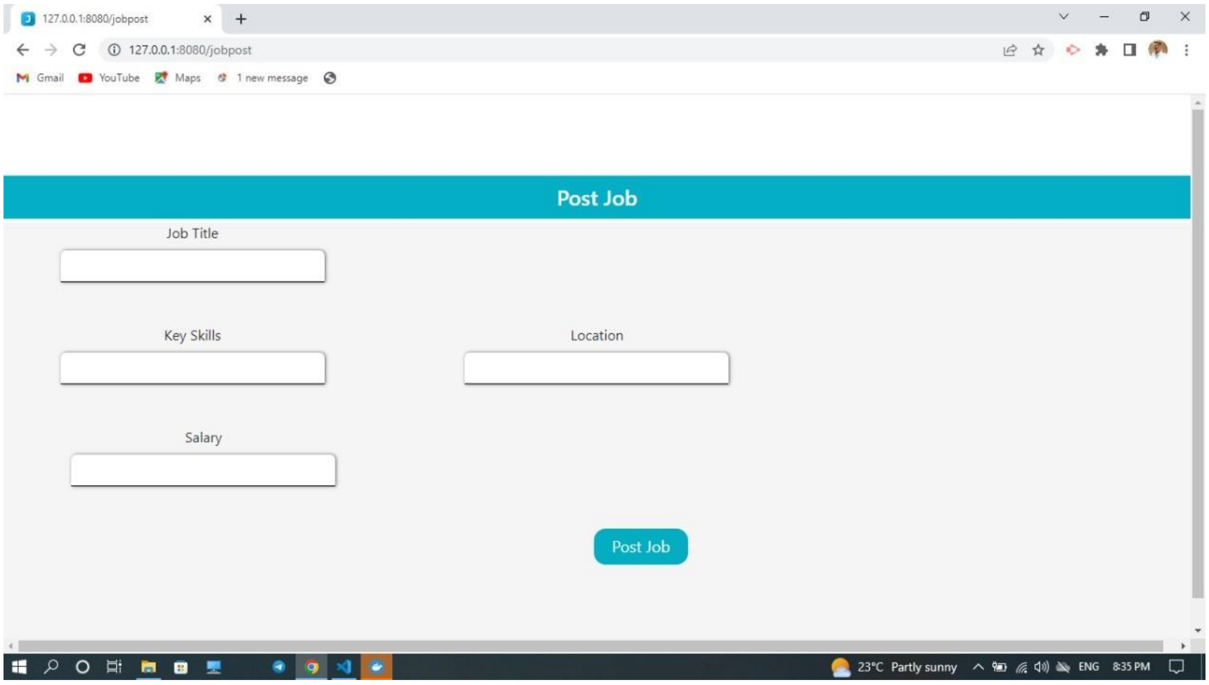
Password *

Enter the password

Confirm Password *

Confirm password

Save & Continue


127.0.0.1:8080/jobpost

Post Job

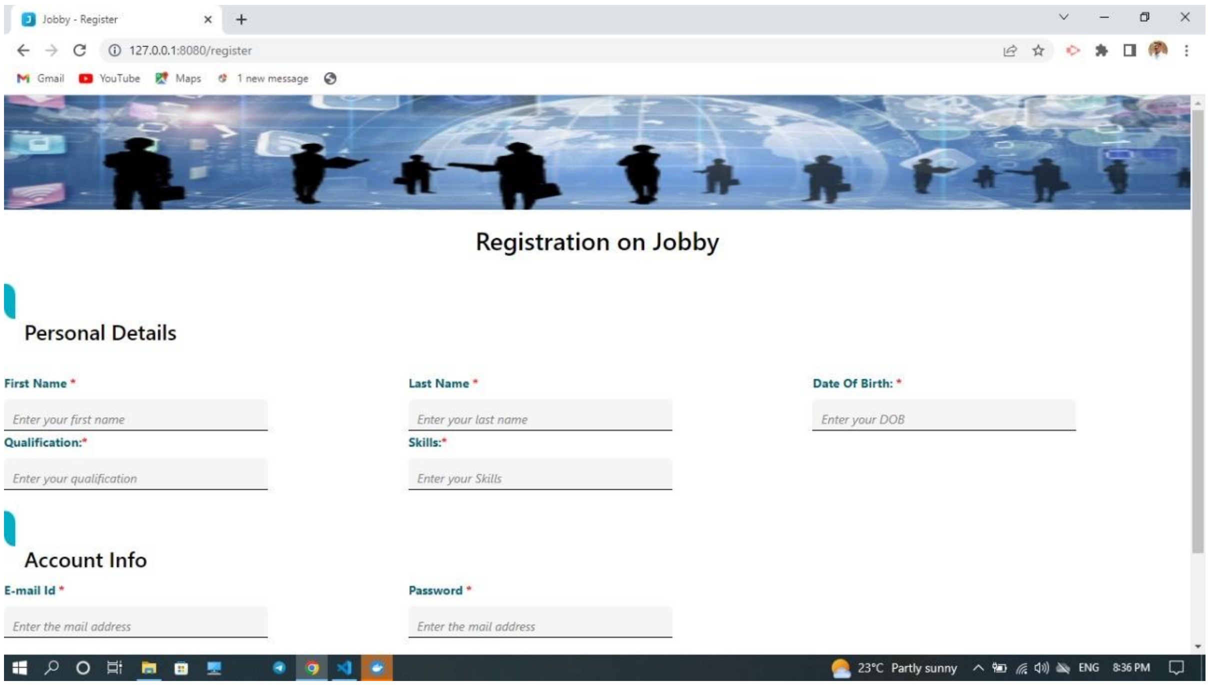
Job Title

Key Skills

Location

Salary

Post Job

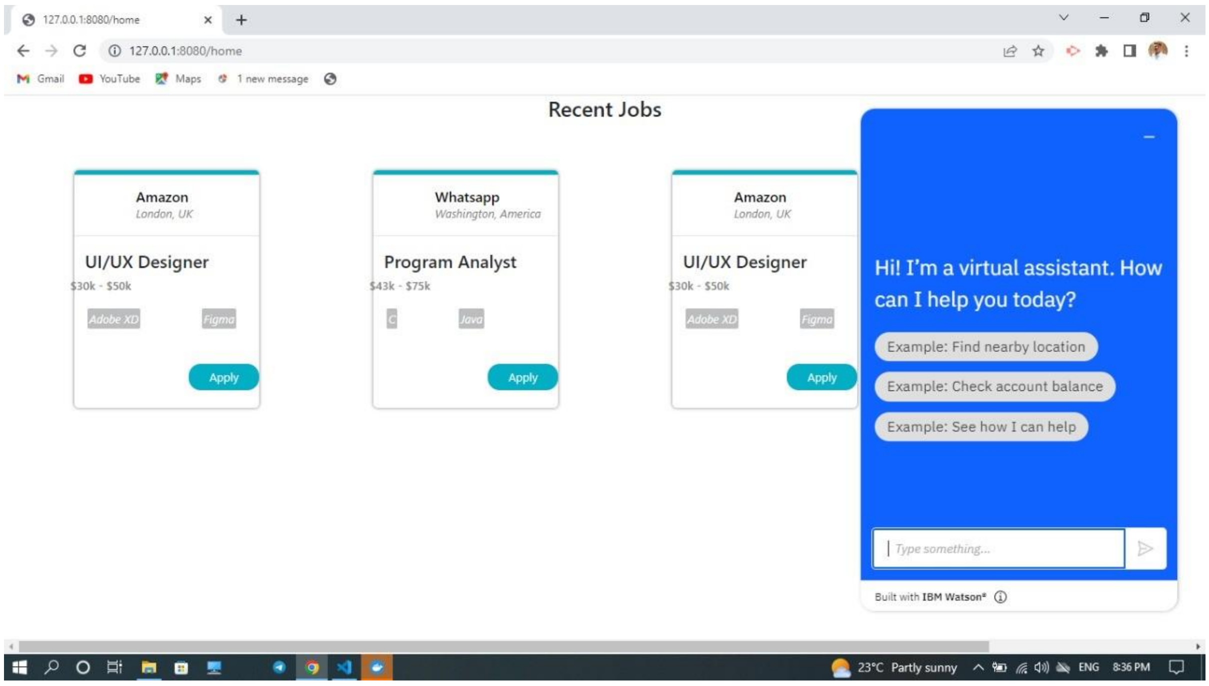

Registration on Jobby

Personal Details

First Name * Enter your first name
Last Name * Enter your last name
Date Of Birth: * Enter your DOB
Qualification:* Enter your qualification
Skills:* Enter your Skills

Account Info

E-mail Id * Enter the mail address
Password * Enter the mail address


Recent Jobs

Amazon, London, UK — UI/UX Designer — $30k - $50k — Adobe XD, Figma — Apply

Whatsapp, Washington, America — Program Analyst — $43k - $75k — C, Java — Apply

Amazon, London, UK — UI/UX Designer — $30k - $50k — Adobe XD, Figma — Apply

Hi! I'm a virtual assistant. How can I help you today?

Example: Find nearby location

Example: Check account balance

Example: See how I can help

Type something...

Built with IBM Watson®

8. CONCLUSION

we have used ibm cloud services like db2, cloud registry , kubernetes , Watson assistant to create this application , which will be very usefull for candidates who are searching for job and as well as for the company to select the right candidate for their organization.

GitHub & Project Demo Link

https://github.com/IBM-EPBL/IBM-Project-1066-1658339237

https://drive.google.com/file/d/1vCqWLqJXP0HU6-3zFrBdgnIKO8UNN4Ec/view?usp=drivesdk