Skill and Job Recommender

PROJECT REPORT

Submitted by

TARIQ JAMAL A	**721219205057**
Mohammed Arshad Z	**721218205038**
Gokaldas A	**721219205018**
Rithisha R	**721219205045**

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

KARPAGAM INSTITUTE OF TECHNOLOGY

ANNA UNIVERSITY : CHENNAI 600 025

NOV 2021

\

TABLE OF CONTENT

1. INTRODUCTION

1.1 Project Overview

Nowadays, job search is a task commonly done on the Internet using job search engine sites like LinkedIn,Indeed, and others. Commonly, a job seeker has two ways to search a job using these sites: 1) doing a query based on keywords related to the job vacancy that he/she is looking for, or 2) creating and/or updating a professional profile containing data related to his/her education, professional experience, professional skills and other, and receive personalized job recommendations based on this data. Sites providing support to the former case are more popular and have a simpler structure; however, their recommendations are less accurate than those of the sites using profile data. Personalized job recommendation sites implemented a variety of types of recommender systems, such as content-based filtering, collaborative filtering, knowledge-based and hybrid approaches.

1.2 Purpose

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

2. LITERATURE SURVEY

2.1 Existing Problem

1. Students/Job seekers find their desired job based on their skillset.
2. Integrating Intelligent CHATBOT for job recommendation application.
3. A study of LinkedIn as an Employment Tool for Job seeker & Recruiter.
4. Cloud storage and sharing services.

2.2 References

1. A combined representation learning approach for better job and skill recommendation-Vachik S.Dave Baichuan Zhang

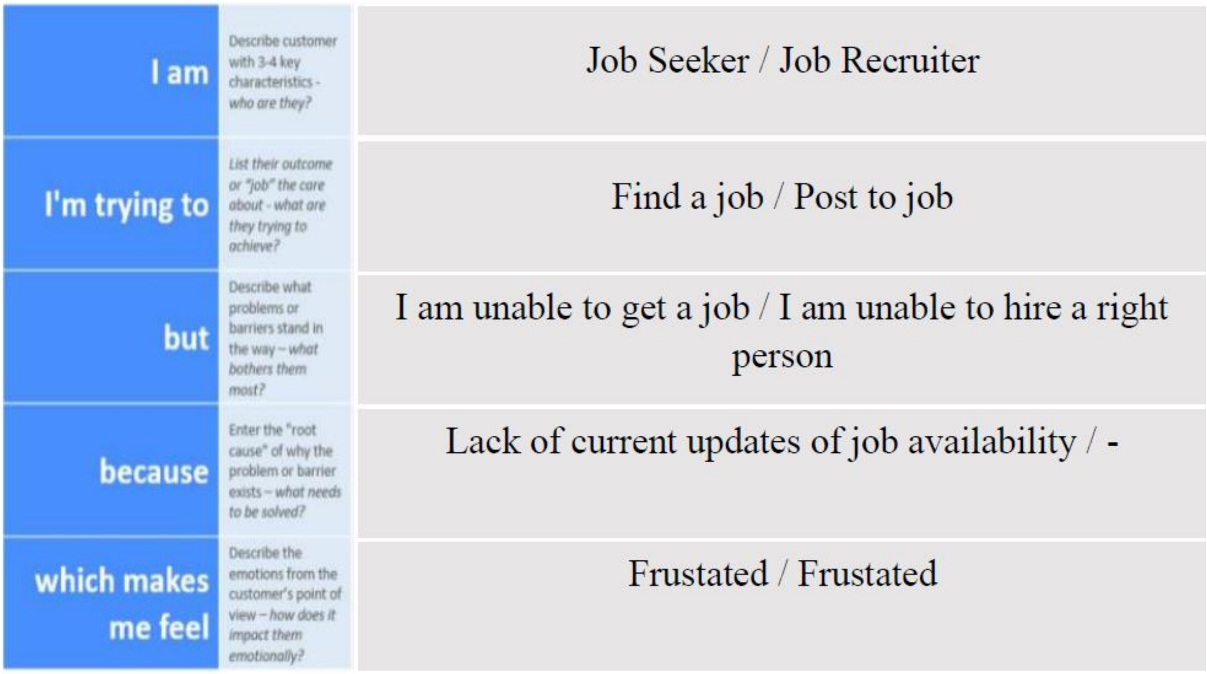
2. A Recommender System for Job Seeking and Recruiting - Yao Lu, Sandy El Helou, Denis Gillet.

3. Job Recommendation based on Job Seeker Skills an Empirical Study - Jorge ValverdeRebaza, Ricardo Puma, Paul

2.3 Problem Statement Definition

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.

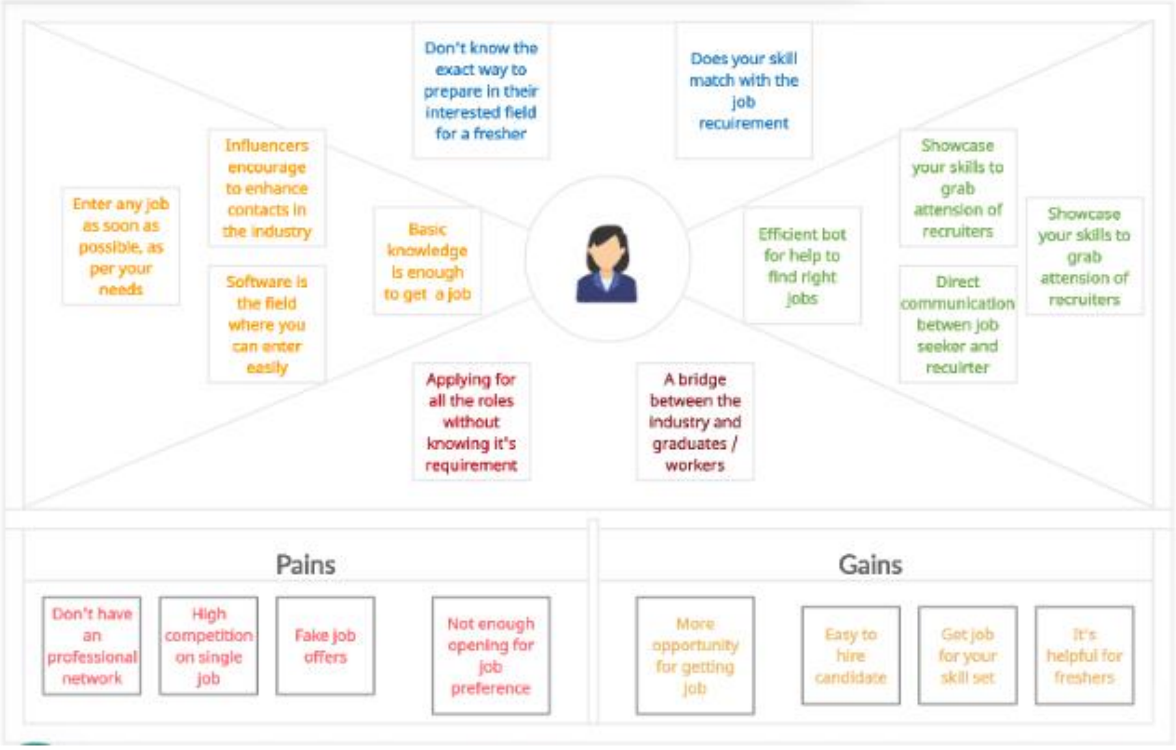
To develop an end-to-end web application capable of displaying the current job openings based on the user skillset. The user and their information are stored in the Database. An alert is sent when there is an opening based on the user skillset. Users will interact with the chatbot and can get the recommendations 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.



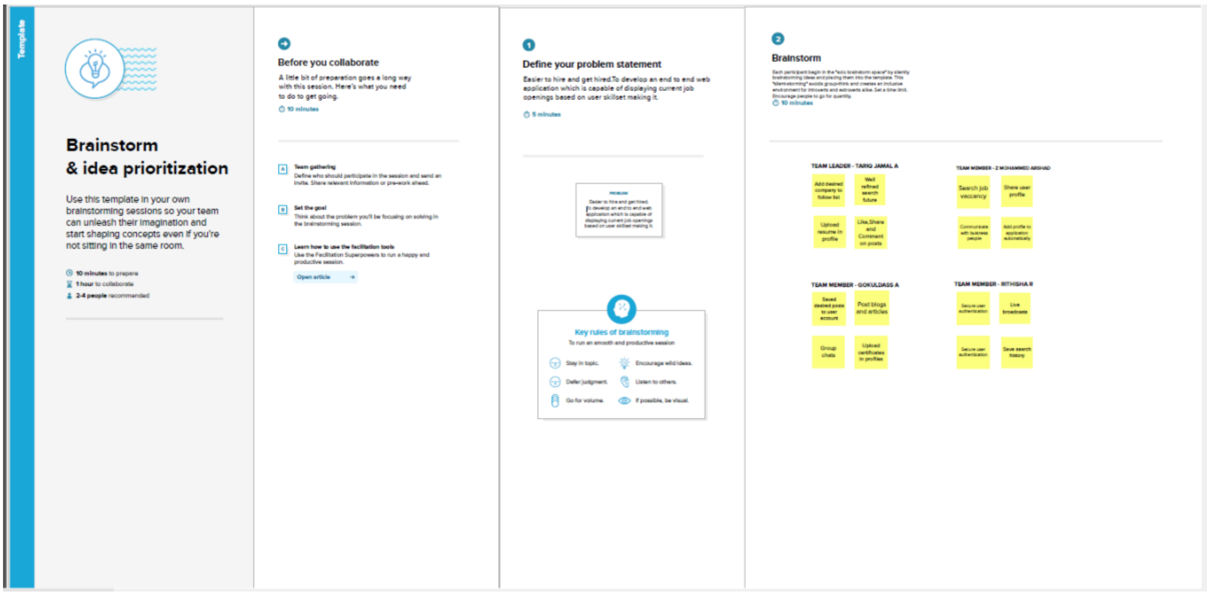
I am	Describe customer with 3-4 key characteristics - *who are they?*	Job Seeker / Job Recruiter
I'm trying to	*List their outcome or "job" the care about - what are they trying to achieve?*	Find a job / Post to job
but	Describe what problems or barriers stand in the way – *what bothers them most?*	I am unable to get a job / I am unable to hire a right person
because	Enter the "root cause" of why the problem or barrier exists – *what needs to be solved?*	Lack of current updates of job availability / -
which makes me feel	Describe the emotions from the customer's point of view – *how does it impact them emotionally?*	Frustated / Frustated

3.IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas:


Don't know the exact way to prepare in their interested field for a fresher
Does your skill match with the job recuirement
Influencers encourage to enhance contacts in the industry
Enter any job as soon as possible, as per your needs
Basic knowledge is enough to get a job
Software is the field where you can enter easily
Showcase your skills to grab attension of recruiters
Efficient bot for help to find right jobs
Showcase your skills to grab attension of recruiters
Direct communication betwen job seeker and reculrter
Applying for all the roles without knowing it's requirement
A bridge between the industry and graduates / workers
Pains
Don't have an professional network
High competition on single job
Fake job offers
Not enough opening for job preference
Gains
More opportunity for getting job
Easy to hire candidate
Got job for your skill set
It's helpful for freshers

3.2 Ideation & Brainstorming:


Template

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-4 people recommended

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

C Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 Define your problem statement

Easier to hire and get hired.To develop an end to end web application which is capable of displaying current job openings based on user skillset making it.

5 minutes

PROBLEM
Easier to hire and get hired. To develop an end to end web application which is capable of displaying current job openings based on user skillset making it.

Key rules of brainstorming
To run an smooth and productive session

Stay in topic.
Encourage wild ideas.
Defer judgment.
Listen to others.
Go for volume.
If possible, be visual.

2 Brainstorm

Each participant begin in the "solo brainstorm space" by silently brainstorming ideas and placing them into the template. This "silent-storming" avoids group-think and creates an inclusive environment for introverts and extroverts alike. Set a time limit. Encourage people to go for quantity.

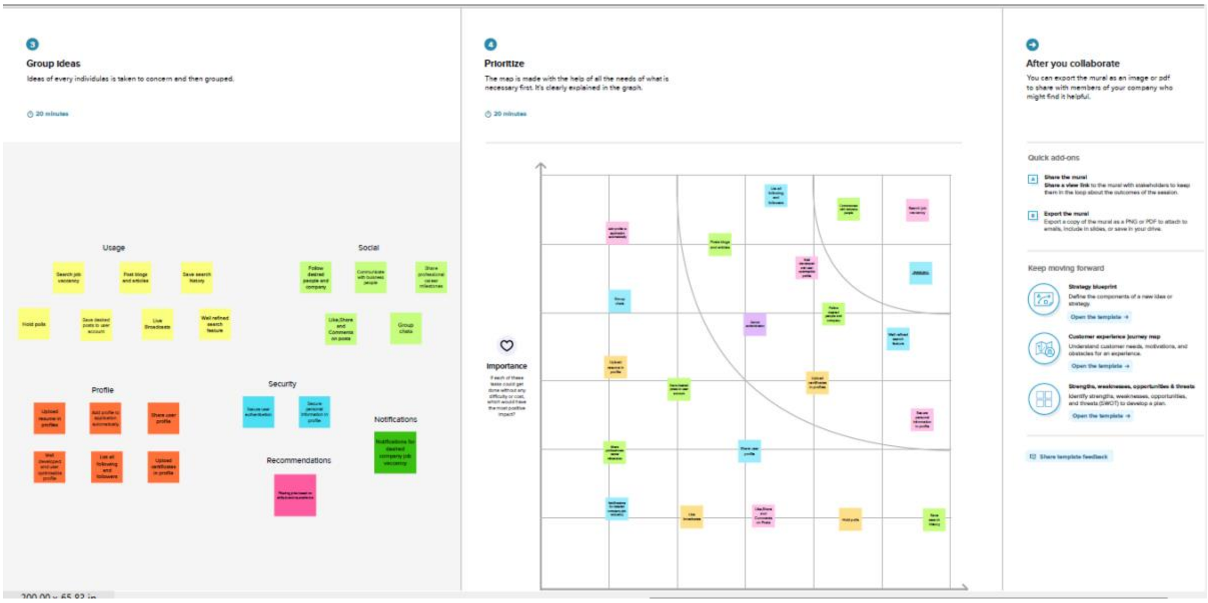
10 minutes

TEAM LEADER - TARIQ JAMAL A
Add desired company to follow list
Well refined search future
Upload resume in profile
Like,Share and Comment on posts

TEAM MEMBER - Z MOHAMMED ARSHAD
Search job vaccancy
Share user profile
Communicate with business people
Add profile to application automatically

TEAM MEMBER - GOKULDASS A
Saved desired posts to user account
Post blogs and articles
Group chats
Upload certificates in profiles

TEAM MEMBER - RITHISHA R
Secure user authentication
Live broadcasts
Secure user authentication
Save search history



3 Group Ideas

Ideas of every individules is taken to concern and then grouped.

20 minutes

4 Prioritize

The map is made with the help of all the needs of what is necessary first. It's clearly explained in the graph.

20 minutes

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

Share the mural
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

Export the mural
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

Share template feedback

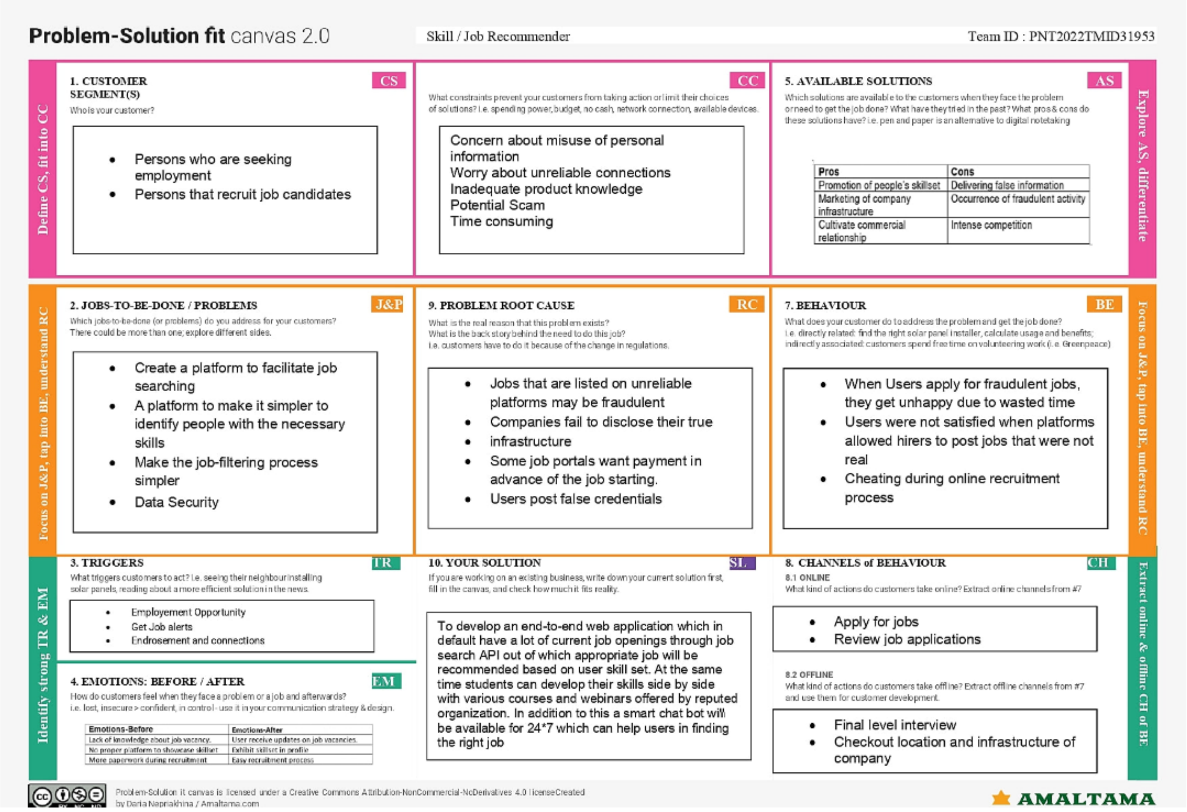
Usage
Search job vaccancy
Post blogs and articles
Save search history
Hold polls
Save desired posts to user account
Live Broadcasts
Well refined search feature
Social
Follow desired people and company
Communicate with business people
Share professional career milestones
Like,Share and Comments on posts
Group chats
Profile
Upload resume in profiles
Add profile to application automatically
Share user profile
Well developed and user centralize profile
List all following and followers
Upload certificates in profile
Security
Secure user authentication
Secure personal information in profile
Notifications
Notifications for desired company job vaccancy
Recommendations
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
200.00 × 65.83 in

3.3 Porposed solution:

Problem Statement (Problem to be solved):

To enable job seekers to find jobs relevant to their skill set.

3.4 Problem Solution fit:



Problem-Solution fit canvas 2.0

Skill / Job Recommender

Team ID : PNT2022TMID31953

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?

- Persons who are seeking employment
- Persons that recruit job candidates

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.

Concern about misuse of personal information
Worry about unreliable connections
Inadequate product knowledge
Potential Scam
Time consuming

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

Pros	Cons
Promotion of people's skillset	Delivering false information
Marketing of company infrastructure	Occurrence of fraudulent activity
Cultivate commercial relationship	Intense competition

Explore AS, differentiate

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.

- Create a platform to facilitate job searching
- A platform to make it simpler to identify people with the necessary skills
- Make the job-filtering process simpler
- Data Security

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.

- Jobs that are listed on unreliable platforms may be fraudulent
- Companies fail to disclose their true
- infrastructure
- Some job portals want payment in advance of the job starting.
- Users post false credentials

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)

- When Users apply for fraudulent jobs, they get unhappy due to wasted time
- Users were not satisfied when platforms allowed hirers to post jobs that were not real
- Cheating during online recruitment process

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

Identify strong TR & EM

3. TRIGGERS TR

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

- Employement Opportunity
- Get Job alerts
- Endrosement and connections

4. EMOTIONS: BEFORE / AFTER EM

How do customers feel when they face a problem or a job and afterwards? i.e. lost, insecure > confident, in control - use it in your communication strategy & design.

Emotions-Before	Emotions-After
Lack of knowledge about job vacancy.	User receive updates on job vacancies.
No proper platform to showcase skillset	Exhibit skillset in profile
More paperwork during recruitment	Easy recruitment process

10. YOUR SOLUTION SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.

To develop an end-to-end web application which in default have a lot of current job openings through job search API out of which appropriate job will be recommended based on user skill set. At the same time students can develop their skills side by side with various courses and webinars offered by reputed organization. In addition to this a smart chat bot will be available for 24*7 which can help users in finding the right job

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

What kind of actions do customers take online? Extract online channels from #7

- Apply for jobs
- Review job applications

8.2 OFFLINE

What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

- Final level interview
- Checkout location and infrastructure of company

Extract online & offline CH of BE

AMALTAMA

4.REQUIREMENT ANALYSIS

4.1FunctionalRequirements:

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	User Registration	Registrationthrough Gmail
FR-2	User Confirmation	Confirmation via Email
FR-3	Chat Bot	A Chat Bot will be there in website to solve userqueries and problems related to applying a job, searchfor a joband muchmore.
FR-4	User Login	Login through Gmail
FR-5	User Search	Exploration of Jobsbased on job filters and skillrecommendations.
FR-6	User Profile	Updation of theuser profile through the logincredentials
FR-7	User Acceptance	Confirmation of the Job.

4.2Non-functional Requirements:

NFR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	This application can be used by the job seekers to loginand search for the job based on her Skillsset.
NFR-2	**Security**	This application is secure with separate loginfor JobSeekers as well as Job Recruiters.
NFR-3	**Reliability**	In This application user can able to n number of jobs for free without any limitations and also the enormous job openings will be provided to all thejob seekers without any limitation.
NFR-4	**Performance**	The performance of this application is quicker response and takes lessertime to do any process.
NFR-5	**Availability**	This application provides job offers and recommends Skills for a Particular Job openings.
NFR-6	**Scalability**	The Response timeof the application is quite fastercompared to anyother application.

5.PROJECT DESIGN

5.1 Data flow diagram:

A Data Flow Diagram (DFD) is a traditional visual representation of theinformation 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 leavesthe system, what changesthe information, and where data is stored.

5.2 Solution and technical architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1& table2

5.3 User Stories:

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

User type	Functional requirements	User Story number	Task	Acceptance Criteria	Priority	Release
Customer	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access myaccount / dashboard	High	Sprint-1
		USN-2		I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through online websites	I can register & access the dashboard with online website Login	Low	Sprint-2
		USN-4	As a user, I can register for	Confirm	Medium	Sprint-1

			the application through Gmail			
	Login	USN-5	As a user, I can log into the application by entering email & password	Confirm	High	Sprint-1
Customer		USN-6	As a user, I can able to take up the skill assessment and view the appropriate test score. Based on the skill sets I canable to get personalized job recommendations.	I can receive job recommendation	Medium	Sprint-1
Customer care executive		USN-7	24/7 chatbot support.	24/7 chatbot support.	High	Sprint-1
Admin		USN-8	As an administrator, I can able to viewthe progress andmake required changes in the project	Deploy user specific and personalized job recommendation	High	Sprint-1

6 . PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation:

Sprint	Functional Requirement	User Story Number	User Story / Task	Priority	Acceptancecriteria	Team Members
Sprint-1	UIDesign	USN-1	As a user, I can see and experiencean awesome user interface in the website	Medium	Better Impression about a website	Gokuldass A, Rithisha R
Sprint-1	Registration	USN-2	As a user, I can register for the application by entering my email,password, and confirming my password.	High	Ican access my account I dashboard	Tariq Jamal A, Mohammed Arshad Z
Sprint-1		USN-3	As a user,I will receive confirmation email once I have registered for theapplication	High	I can receive confirmation email & click confirm	Gokuldass A, Mohammed Arshad Z
Sprint-1		USN-4	As a user,I can register for theapplication throughFacebook	Low	I can register & access the dashboard with Facebook Login	Mohammed Arshad Z, Rithisha R
Sprint-1		USN-5	As a user,I can register for theapplication through Gmail	Medium	I can receive confirmation email & click confirm	Tariq Jamal A, Rithisha R
Sprint-1	Login	USN-6	As a user,I can log into the applicationby entering email& password	High	Ican access my account I dashboard	Tariq Jamal A, Mohammed Arshad Z

Sprint-1	Flask	USN-7	As a user, Ican access the website in asecond	High	Ican access my account I dashboard	Mohammed Arshad Z, Gokuldass A

Sprint	**FunctionalRequirement (Epic)**	**User Story Number**	**User Story / Task**	**Priority**	**Acceptance criteria**	**Team Members**
Sprint-1	Dashboard	USN-8	As a user,If I Logged in correctly, I canview my dashboard and I can navigateto any pages which are already listed there.	High	I can accessall the pages/dashboard	Tariq Jamal A
			SubmissionOf Sprint-1			
Sprint-2	User Profile	USN-9	As a user, I can view and update mydetails	Medium	I can modify my details/data	Tariq Jamal A, Gokuldass A, Mohammed Arshad Z
Sprint-2	Database	USN-10	As a user, I can store my details anddata in the website	Medium	I can store my data	Rithisha R, Gokuldass A
Sprint-2	Cloud Storage	USN-11	As a user, I can upload my photo,resumeand much more in the website.	Medium	I can Uploadmy documents anddetails	Tariq Jamal A, Mohammed Arshad Z

Sprint-2	Chatbot	USN-12	As a user, I can ask the Chatbot aboutlatest job openings, which willhelpme and show the recentjob openings based on my profile.	High	I can knowthe recent job openings	Tariq Jamal A, Gokuldass A
Sprint-2	Identity-Aware	USN-13	As a User, I can access my account by entering by correct logincredentials. My user credentials is only displayed to me.	High	I can have my account safely	Rithisha R, Mohammed Arshad Z
			Submissionof Sprint-2			

Sprint	FunctionalRequirement (Epic)	User Story Number	User Story / Task	Priority	Acceptance criteria	Team Members
Sprint-3	Sendgrid service	USN-14	As a user, I can get a notification or mail abouta job opening with the helpof send grid service.	Medium	I can get a notification in a second.	Mohammed Arshad Z, Gokuldass A
Sprint-3	Learning Resource	USN-15	As a user, I can learn the course andI will attain the skills which will be useful for developing my technical skills.	High	I can gainthe knowledge andskills	Tariq Jamal A, Rithisha R,

Sprint-3	Docker	USN-16	As a user, I can access the website in anydevice	High	I can accessmy account in anydevice	Tariq Jamal A, Gokuldass A
Sprint-3	Kubernates	USN-17	As a user, I can access thewebsite in any device	High	I can accessmy account in anydevice	Tariq Jamal A, Gokuldass A
Sprint-3	Deployment in cloud	USN-18	As a user, I can access thewebsite in any device	High	I can accessmy account in anydevice	Rithisha R, Mohammed Arshad Z, Gokuldass A
Sprint-3	Technical support	USN-19	As a user, I can get a customercaresupport from the website which will solve my queries.	Medium	Ican tackle my problem &queries.	Tariq Jamal A, Rithisha R
			Submission of Sprint-3			
Sprint-4	Unit Testing	USN-15	As a user, I can access the website withoutany interruption	High	I can accessthe website withoutany interruption	Tariq Jamal A, Mohamme

						dArshadZ, Gokuldass A, Rithisha R
Sprint-4	Integrationtesting	USN-16	As a user, I can access the website withoutany interruption	High	I can accessthe web sitewithout any interruption	TariqJamalA, MohammedArshadZ, Gokuldass A, Rithisha R

Sprint	FunctionalRequirement (Epic)	User Story Number	User Story / Task	Priority	Acceptancecriteria	Team Members

Sprint-4	System testing	USN-17	As a user, I can access the website withoutany interruption	High	I can accessthe web sitewithout any interruption	Tariq Jamal A, Mohammed Arshad Z, Gokuldass A, Rithisha R
Sprint-4	Correction	USN-18	As a user, I can access the website withoutany interruption	High	I can accessthe web sitewithout any interruption	Tariq Jamal A, Mohammed A

						rshadZ, GokuldassA, RithishaR
Sprint-4	Acceptancetesting	USN-19	As a user, I can access the website withoutany interruption	High	I can accessthe web sitewithout any interruption	TariqJamalA, MohammedArshadZ, GokuldassA,

						Rithisha R
			Submission of Sprint-4			

6.2 Sprint Delivery Schedule:

Project Tracker, Velocity & Burndown Chart

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date(Planned)	Story PointsCompleted (as on Planned End Date)	Sprint ReleaseDate (Actual)
Sprint-1	20	6 Days	2022-10-30	2022-11-04	20	2022-11-04
Sprint-2	20	5 Days	2022-11-05	2022-11-09	20	2022-11-09
Sprint-3	20	6 Days	2022-11-10	2022-11-15	20	2022-11-15
Sprint-4	20	4 Days	2022-11-16	2022-11-19	20	2022-11-19

6.3 Reports from JIRA:

Average velocity = sprint duration / velocity
= 20 / 10 = 2

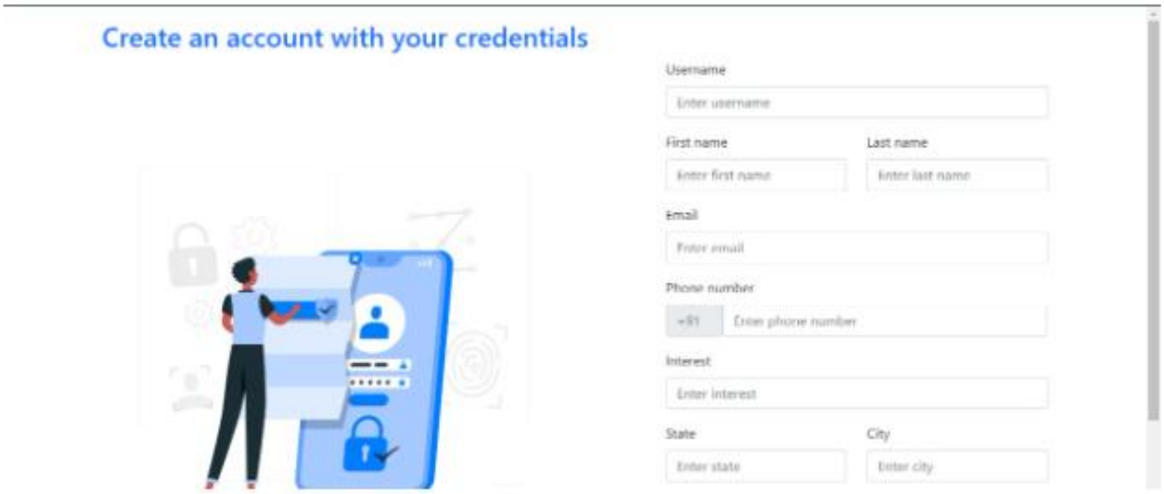
7. CODING & SOLUTIONING

7.1 Feature 1:
Create Account:

```
{% extends 'auth_layout/app.html' %}

{% block title %} Register {% endblock title %}

{% block heading %}
    Create an account with your credentials
{% endblock heading %}

{% block body %}
    <form method="post">
        <div class="mb-3">
            <label for="username" class="form-label">Username</label>
            <input type="text" name="username" class="form-control"
id="username" placeholder="Enter username" minlength="3" maxlength="15"
required>
        </div>
        <div class="row">
            <div class="col-md-6">
                <div class="mb-3">
                    <label for="first_name" class="form-label">First name</label>
                    <input type="text" name="first_name" class="form-control"
id="first_name" placeholder="Enter first name" minlength="3" maxlength="15"
required>
                </div>
            </div>
            <div class="col-md-6">
                <div class="mb-3">
                    <label for="last_name" class="form-label">Last name</label>
                    <input type="text" name="last_name" class="form-control"
id="last_name" placeholder="Enter last name" minlength="1" maxlength="15"
required>
                </div>
            </div>
        </div>
        <div class="mb-3">
            <label for="email" class="form-label">Email</label>
            <input type="email" name="email" class="form-control" id="email"
placeholder="Enter email" required>
```

```
        </div>
        <div class="mb-3">
            <label for="phone_number" class="form-label">Phone number</label>
            <div class="row">
                <div class="col-md-2 col-sm-3 col-4 phno_code">
                    <input type="number" value="+91 " readonly placeholder="+91"
class="form-control">
                </div>
                <div class="col-md-10 col-sm-9 col-8 phno">
                    <input type="number" name="phone_number" class="form-
control" id="phone_number" placeholder="Enter phone number" required>
                </div>
            </div>
        </div>
        <div class="mb-3">
            <label for="interest" class="form-label">Interest</label>
            <input type="text" name="interest" class="form-control" id="interest"
placeholder="Enter interest" required>
      </div>
        <div class="row">
            <div class="col-md-6">
                <div class="mb-3">
                    <label for="state" class="form-label">State</label>
                    <input type="text" name="state" class="form-control" id="state"
placeholder="Enter state" minlength="3" maxlength="15" required>
                </div>
            </div>
            <div class="col-md-6">
                <div class="mb-3">
                    <label for="city" class="form-label">City</label>
                    <input type="text" name="city" class="form-control" id="city"
placeholder="Enter city" minlength="3" maxlength="15" required>
                </div>
            </div>
        </div>
        <div class="mb-3">
            <label for="password" class="form-label">Password</label>
            <input type="password" name="password" class="form-control"
id="password" placeholder="Enter password" minlength="3" maxlength="15"
required>
        </div>
        <p class="mt-3">Already have an account <a href="{{ url_for('login')
}}">login</a></p>
```

```
        <button type="submit" class="btn btn-primary">Register</button>
    </form>
{% endblock body %}
```



Create an account with your credentials

Username

Enter username

First name

Enter first name

Last name

Enter last name

Email

Enter email

Phone number

+91

Enter phone number

Interest

Enter interest

State

Enter state

City

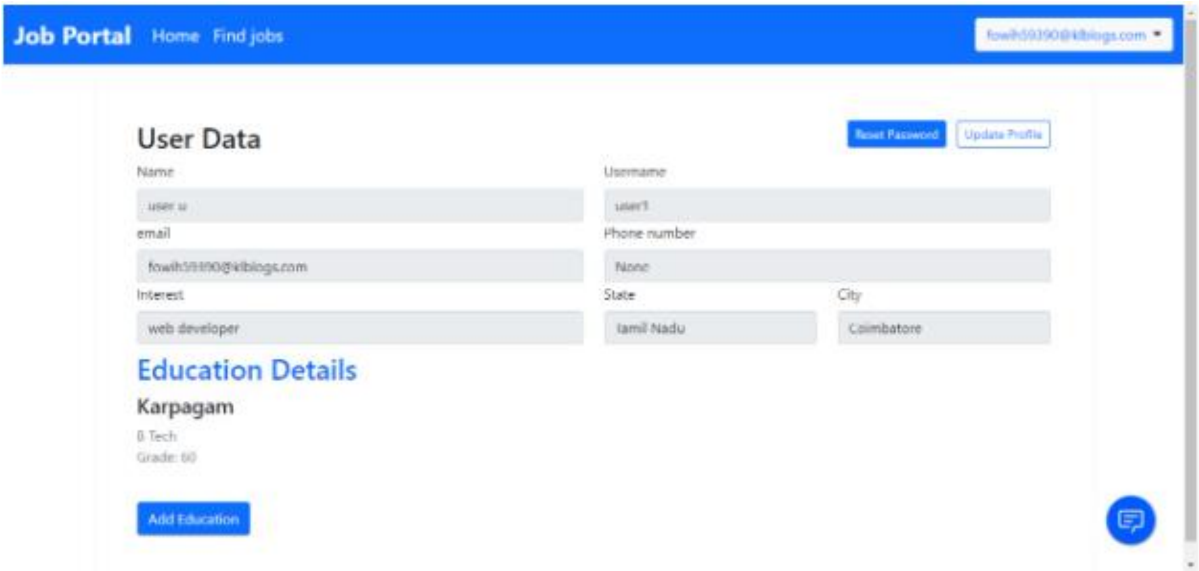
Enter city

User Data:

```
{% extends 'layout/app.html' %}

{% block title %} Profile Page {% endblock title %}
<!-- Page content  -->
{% block body %}
<div class="container my-3 p-5 shadow-sm">
    <div class="d-flex justify-content-between">
        {% if error %}
            <div class="alert alert-primary" role="alert">
                {{ error }}
            </div>
        {% endif %}
        <h2>User Data</h2>
        <div>
            <button type="button" class="btn btn-primary btn-sm me-2" data-bs-
toggle="modal" data-bs-target="#resetpassword">Reset
                Password</button>
            <!-- Modal -->
            <div class="modal fade" id="resetpassword" tabindex="-1" aria-
labelledby="resetpasswordLabel"
                aria-hidden="true">
                <div class="modal-dialog modal-sm">
                    <div class="modal-content">
```

```
                        <div class="modal-header">
                            <h5 class="modal-title" id="resetpasswordLabel">Reset
Password</h5>
                            <button type="button" class="btn-close" data-bs-
dismiss="modal" aria-label="Close"></button>
                        </div>
                        <form action="/profile" method="post">
                            <div class="modal-body">
                                <label for="password" class="form-
label">Password</label>
                                <input type="password" class="form-control"
name="password" \>
                                <label for="confirm_password" class="form-
label">Confirm Password</label>
                                <input type="password" class="form-control"
name="confirm_password" \>
                            </div>
                            <div class="modal-footer">
                                <button type="button" class="btn btn-sm btn-secondary"
data-bs-dismiss="modal">Close</button>
                                <input type="submit" class="btn btn-sm btn-primary"
name="forgot_password" value="Save changes">
                            </div>
                        </form>
                    </div>
                </div>
            </div>
            <a class="btn btn-outline-primary btn-sm" href="{{
url_for('update_profile') }}">Update Profile</a>
        </div>
    </div>
    <div class="row">
        <div class="col-md-6">
            <div>
                <label for="name" class="form-label">Name</label>
                <input type="text" class="form-control" name="name" value="{{
user['user']['NAME'] }}" readonly>
            </div>
      </div>
        <div class="col-md-6">
            <div>
                <label for="username" class="form-label">Username</label>
        <input type="text" class="form-control" name="username" value="{{
user['user']['USERNAME'] }}"
          readonly>
      </div>
    </div>
  </div>
  <div class="row">
    <div class="col-md-6">
      <div>
        <label for="email" class="form-label">email</label>
        <input type="text" class="form-control" name="email" value="{{
user['user']['EMAIL'] }}" readonly>
      </div>
    </div>
    <div class="col-md-6">
      <div>
        <label for="phone_number" class="form-label">Phone
number</label>
        <input type="text" class="form-control" name="phone_number"
value="{{ user['user']['PHNO'] }}"
          readonly>
      </div>
    </div>
  </div>
  <div class="row">
    <div class="col-md-6">
      <div>
        <label for="interest" class="form-label">Interest</label>
        <input type="text" class="form-control" name="interest"
value="{{user['user']['INTEREST']}}" readonly>
      </div>
    </div>
    <div class="col-md-3">
      <div>
        <label for="state" class="form-label">State</label>
        <input type="text" class="form-control" name="state"
value="{{user['user']['STATE']}}" readonly>
     </div>
    </div>
    <div class="col-md-3">
      <div>
        <label for="city" class="form-label">City</label>
```

```
            <input type="text" class="form-control" name="city"
value="{{user['user']['CITY']}}" readonly>
          </div>
        </div>
    </div>
    <h2 class="text-primary my-2">Education Details</h2>
    {% for education in user['educations'] %}
        <h4>{{ education['COLLEGE_NAME'] }}</h4>
        <p class="text-secondary m-0">{{ education['DEGREE'] }}</p>
      <p class="text-secondary">Grade: {{ education['GRADE'] }}</p>
    {% endfor %}
    <a class="btn btn-primary mt-4" href="{{ url_for('add_education') }}">Add
Education</a>
</div>
{% endblock body %}
```



User Data

Name: user u
Username: user1
email: fowih59390@klblogs.com
Phone number: None
Interest: web developer
State: Tamil Nadu
City: Coimbatore

Education Details

Karpagam

B Tech

Grade: 60

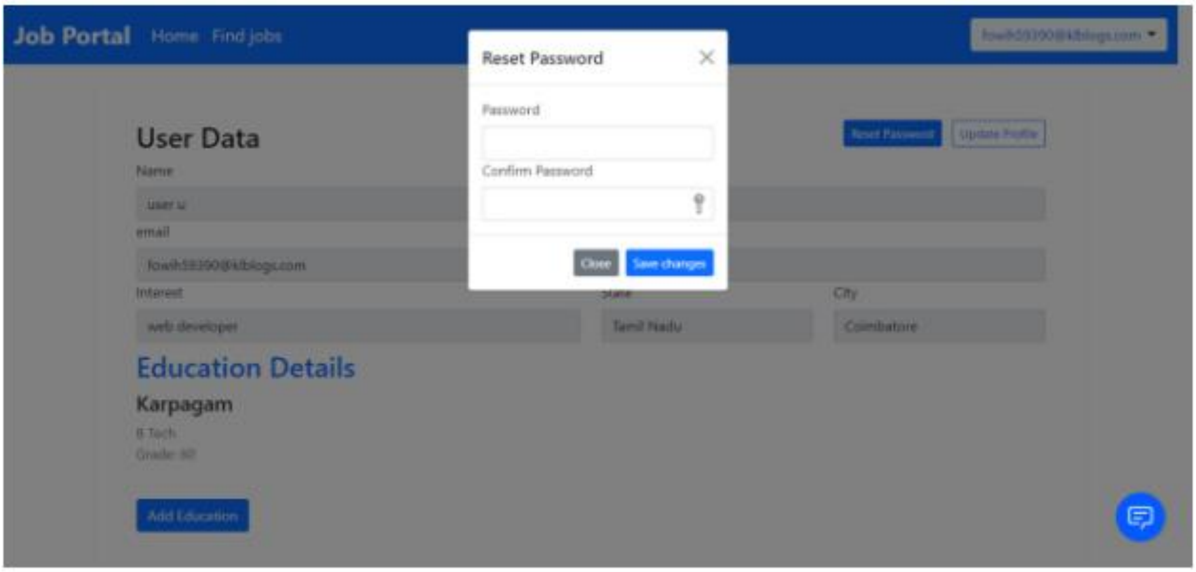
Reset Password:

```
{% extends 'auth_layout/app.html' %}

{% block title %} Reset Password {% endblock title %}

{% block heading %}
    Reset Password
{% endblock heading %}

{% block body %}
    <p>Forgotten your password? Enter your email address below, and we'll
instruction for setting a new password</p>
```

```
    <form method="post">
        <div class="mb-3">
            <label for="email" class="form-label">Email</label>
            <input type="email" class="form-control" name="email" id="email">
        </div>
        <input type="submit" class="btn btn-primary" value="Send email">
   </form>
{% endblock body %}
```

Reset Password Completed:

```
{% extends 'auth_layout/app.html' %}

{% block title %} Password Resetted {% endblock title %}


{% block body %}
    <h3 class="text-primary">Password reset complete</h3>
    <p>Your password has been set. You may go ahead and log in now.</p>

    <a href="{% url 'login' %}" class="btn btn-primary">Login</a>
{% endblock body %}
```



User Data

Name: user u

email: fowih59390@kblogs.com

Interest: web developer

State: Tamil Nadu

City: Coimbatore

Education Details

Karpagam

B Tech

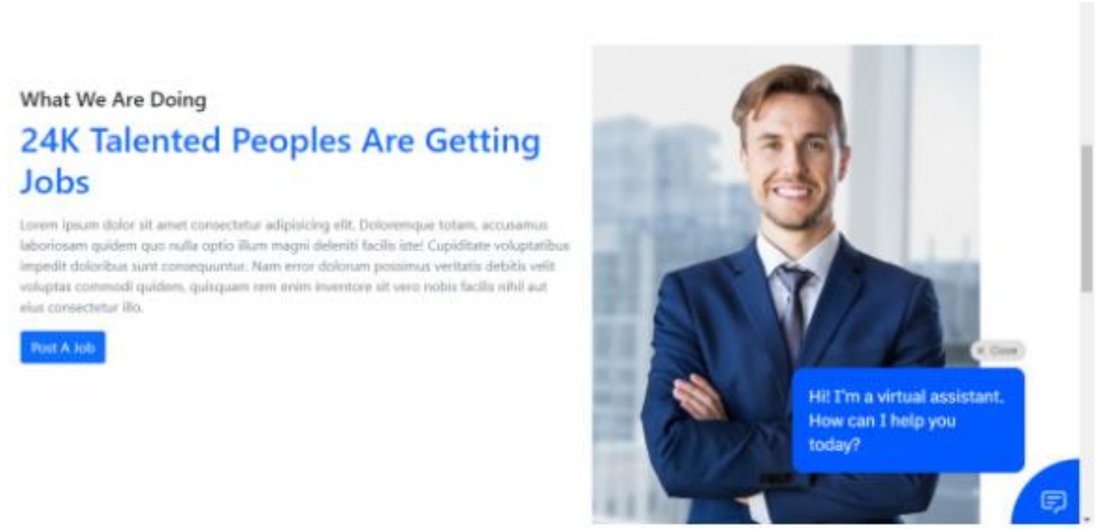
Grade: 80

7.2 Feature 2:

Post A Job:

```
{% extends 'layout/app.html' %}

{% block title %} Jobs {% endblock title %}
<!-- Page content  -->
{% block body %}

    <div class="container py-3">
        <div class="row">
            <div class="col-md-6">
                <h3 class="text-primary">Jobs</h3>
            </div>
            <div class="col-md-6">
                <form action="{{url_for('jobs')}}" method="post">
                    <div class="d-flex justify-content-center align-items-center">
                        <input type="text" class="form-control" name="search_query"
placeholder="Enter...">
                        <button class="btn btn-primary" type="submit">Search</button>
                    </div>
                </form>
            </div>
        </div>
        <div class="row my-4 py-3">
            {% for job in jobs %}
                <div class="col-md-5 shadow rounded p-3 my-3 mx-4">
                    <div class="d-flex justify-content-between align-items-center">
                        <h2 class="text-dark fw-bold">{{ job['title'] }}</h2>
                    </div>
                    <p class="text-secondary">Company: {{
job['company']['display_name'] }}</p>
                    <span class="text-secondary">Locations:
                        {% for location in job['location']['area'] %}
                            {{location}},
                        {% endfor %}
                    </span>
                    <p class="text-secondary">Salary: {{ job['salary_max'] }}</p>
                    <hr>
                    <p class="text-secondary">Job type: {{ job['contract_type'] }}</p>
                    <h4>Description</h4>
                    <p class="text-secondary">{{ job['description'] }}</p>
                    <a href="{{ job['redirect_url'] }}" class="btn btn-primary"
target="_blank">Apply</a>
                </div>
```

```
          {% endfor %}
        </div>
    </div>
{% endblock body %}
```



What We Are Doing

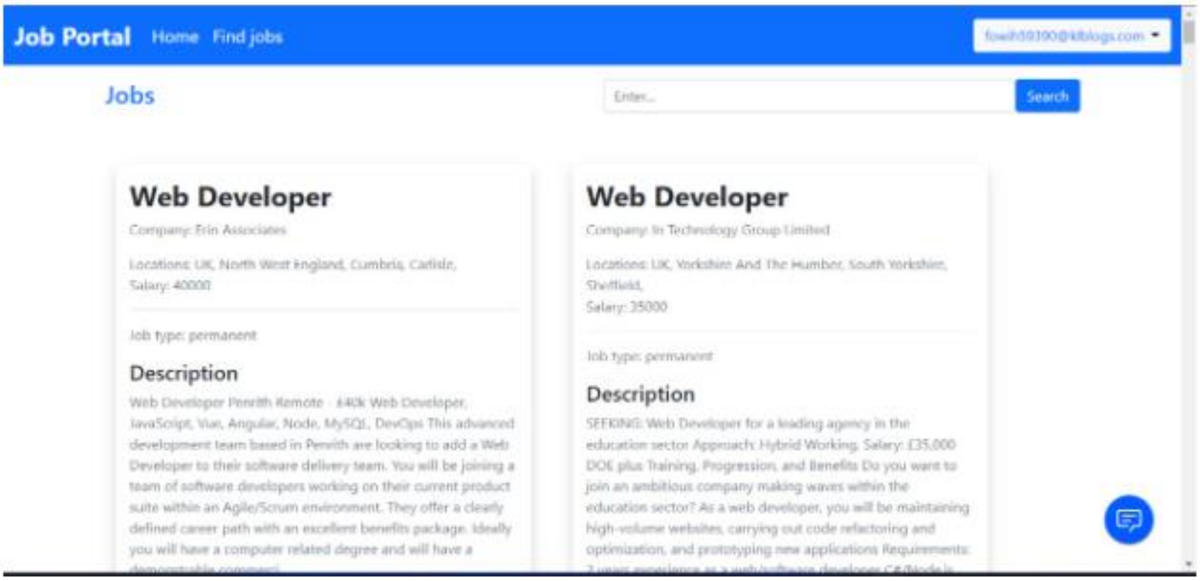
24K Talented Peoples Are Getting Jobs

Lorem ipsum dolor sit amet consectetur adipisicing elit. Doloremque totam, accusamus laboriosam quidem quo nulla optio illum magni deleniti facilis iste! Cupiditate voluptatibus impedit doloribus sunt consequuntur. Nam error dolorum possimus veritatis debitis velit voluptas commodi quidem, quisquam rem enim inventore sit vero nobis facilis nihil aut eius consectetur illo.

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

Jobs:

```
{% extends 'layout/app.html' %}

<!-- Page content  -->
{% block body %}

    <div class="container py-3">
        <div class="row">
            <div class="col">
                <a class="btn btn-primary" href="{{ url_for('jobs') }}">Back</a>
            </div>
        </div>
        <div class="row my-3 p-3 shadow-sm rounded">
            <div class="col-md-12">
                <h2 class="fw-bold">{{ job["TITLE"] }}</h2>
                <h5 class="text-secondary">{{ job["COMPANY_NAME"] }}</h5>
                <p class="text-secondary">{{ job["LOCATION"] }}</p>
                <div class="d-flex" >
                    <a  class="btn  btn-primary  me-2"  href="/apply_job/{{  job['ID']
}}">Apply job</a>
                    <a class="btn btn-outline-primary">Save job</a>
                </div>
                <hr>
                <h4 class="fw-bold">Job details</h4>
                <h5>Job type</h5>
                <P class="text-secondary">{{ job["JOB_TYPE"] }}</P>
                <h5>Experience</h5>
                <P class="text-secondary">{{ job["EXPERIENCE"] }}</P>
                <h5>Salary</h5>
                <P class="text-secondary">{{ job["SALARY"] }}</P>
                <h5>Vacancy</h5>
                <P class="text-secondary">{{ job["VACANCY"] }}</P>
                <h5>Description</h5>
                <p class="text-secondary">{{ job["DESCRIPTION"] }}</p>
                <h5>Education</h5>
                <p class="text-secondary">{{ job["EDUCATION"] }}</p>
                <h5>Responsibility</h5>
                <p class="text-secondary">{{ job["RESPONSIBILITY"] }}</p>
                <h5>Application Deadline {{ job["DEADLINE"] }}</h5>
                <p class="text-secondary">Posted on {{ job["POSTED_ON"] }}</p>
            </div>

        </div>
    </div>
{% endblock body %}
```



Web Developer

Company: Frin Associates

Locations: UK, North West England, Cumbria, Carlisle,
Salary: 40000

Job type: permanent

Description

Web Developer Penrith Remote - £40k Web Developer, JavaScript, Vue, Angular, Node, MySQL, DevOps This advanced development team based in Penrith are looking to add a Web Developer to their software delivery team. You will be joining a team of software developers working on their current product suite within an Agile/Scrum environment. They offer a clearly defined career path with an excellent benefits package. Ideally you will have a computer related degree and will have a demonstrable commerci

Web Developer

Company: In Technology Group Limited

Locations: UK, Yorkshire And The Humber, South Yorkshire, Sheffield,
Salary: 35000

Job type: permanent

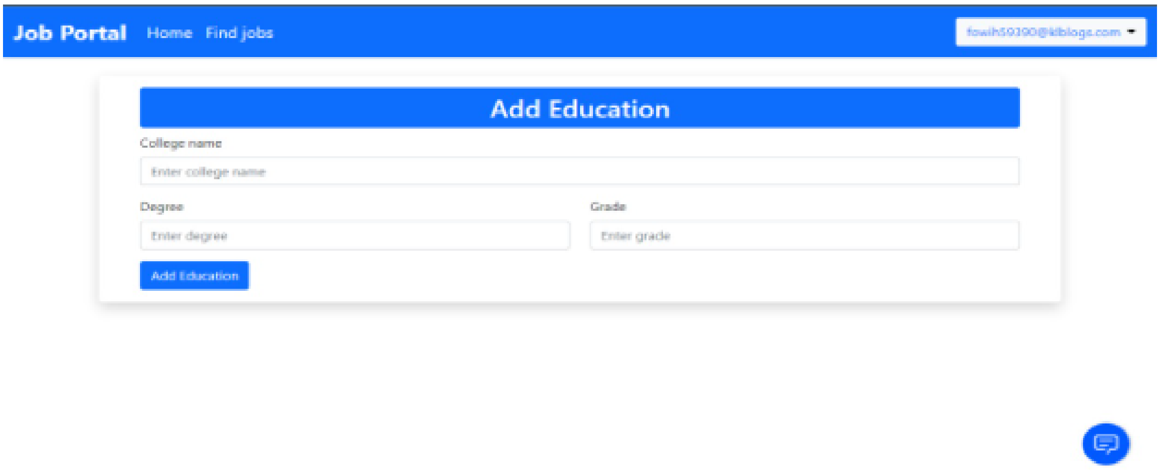
Description

SEEKING: Web Developer for a leading agency in the education sector Approach: Hybrid Working. Salary: £35,000 DOE plus Training, Progression, and Benefits Do you want to join an ambitious company making waves within the education sector? As a web developer, you will be maintaining high-volume websites, carrying out code refactoring and optimization, and prototyping new applications Requirements: 2 years experience as a web/software developer C#/Node.js

Added Education:

```
{% extends 'layout/app.html' %}

{% block title %} Add Education {% endblock title %}

<!-- Page content  -->
{% block body %}
```

```
<div class="container my-4 shadow rounded-3 px-5 py-3 border border-1 border-light">
    <h2 class="text-white text-center bg-primary py-2 rounded">Add Education</h2>
    <form action="/add_education" method="post">
        <div class="row">
            <div class="col-md-12">
                <div class="mb-3">
                    <label for="college_name" class="form-label">College name</label>
                    <input type="text" name="college_name" class="form-control" id="college_name" placeholder="Enter college name">
                </div>
            </div>
        </div>
        <div class="row">
            <div class="col-md-6">
                <div class="mb-3">
                    <label for="degree" class="form-label">Degree</label>
                    <input type="text" name="degree" class="form-control" id="degree" placeholder="Enter degree">
                </div>
            </div>
            <div class="col-md-6">
                <div class="mb-3">
                    <label for="grade" class="form-label">Grade</label>
                    <input type="text" name="grade" class="form-control" id="grade" placeholder="Enter grade">
                </div>
            </div>
        </div>
        <button type="submit" class="btn btn-primary">Add Education</button>
    </form>
</div>
{% endblock body %}
```


Job Portal
Home
Find jobs
fowih59390@kdblogs.com
Add Education
College name
Enter college name
Degree
Enter degree
Grade
Enter grade
Add Education

8. TESTING

8.1 Test Cases:

8.2 User Acceptance Testing:

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Skills and Job Recommendation project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and howthey were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	8	3	3	3	17
Duplicate	1	1	3	1	6
External	2	3	0	1	6
Fixed	12	2	4	19	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	23	14	11	25	75

3. Test Case Analysis

This report shows the numberof test cases that have passed, failed,and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	6	0	1	6
ClientApplication	49	0	1	49
Security	2	0	2	2
OutsourceShipping	3	0	1	3

Exception Reporting	7	0	1	7
Final ReportOutput	4	0	1	4
VersionControl	3	0	0	3

9. RESULTS:

9.1 PERFORMANCE METRICS:

The project has been completed as we expected. We ensured that Database was designed and well connected to our project. The Expected results were gotten.

10. ADVANTAGES & DISADVANTAGES:

ADVANTAGES:

- Person who looks for a job can easily find a suitable job based on their skill set.
- Person can check their eligibility by attending eligibility test.
- Most of the Recruiters find the suitable person based on the scores they have gotten in the eligibility.

DISADVANTAGES:

- Person Job May get technical difficulty while taking the eligibility
- Job seeker may have trouble to contact recruiters directly.

11. CONCLUSION

- The application has been developed to make job search easier .
- The application that we have developed is user friendly .
- User can find a job based on their skillset in the short period of time.
- The jobseeker certainly get benefit by using this application. In the addition,Chatbot Has been implemented with the help of IBM whatson .
- The chatbot helps jobseeker and organization when they experience the difficulties.

12. FUTURE SCOPE

The linked in the well known application to find a job and stay connected with professional and organization. The job seekers and organization use linked in to find a job. In the future , There are lots of possibilities to enhance our project similar to linked in.

13. APPENDIX

Source Code:

FRONT END:

Authentication:

login.html:

```
{% extends 'auth_layout/app.html' %}

{% block title %} Login {% endblock title %}

{% block heading %}
   Login with your credentials
{% endblock heading %}

{% block body %}
   <form method="post">
       {% with messages = get_flashed_messages() %}
           {% if messages %}
               {% for message in messages %}
                   <h4 class="text-danger">{{ message }}</h4>
               {% endfor %}
           {% endif %}
       {% endwith %}
       <div class="mb-3">
           <label for="username" class="form-label">Username</label>
           <input type="text" name="username" class="form-control"
id="username" placeholder="Enter username">
       </div>
       <div class="mb-3">
           <label for="password" class="form-label">Password</label>
           <input type="password" name="password" class="form-control"
id="password" placeholder="Enter password">
       </div>
       <p class="mt-3">Don't have an account <a href="{{ url_for('register')
}}">register</a></p>
       <button type="submit" class="btn btn-primary">Login</button>
   </form>
{% endblock body %}
```

register.html:

```
{% extends 'auth_layout/app.html' %}

{% block title %} Register {% endblock title %}

{% block heading %}
    Create an account with your credentials
{% endblock heading %}

{% block body %}
    <form method="post">
        <div class="mb-3">
            <label for="username" class="form-label">Username</label>
            <input type="text" name="username" class="form-control"
id="username" placeholder="Enter username" minlength="3" maxlength="15"
required>
        </div>
        <div class="row">
            <div class="col-md-6">
                <div class="mb-3">
                    <label for="first_name" class="form-label">First name</label>
                    <input type="text" name="first_name" class="form-control"
id="first_name" placeholder="Enter first name" minlength="3" maxlength="15"
required>
                </div>
            </div>
            <div class="col-md-6">
                <div class="mb-3">
                    <label for="last_name" class="form-label">Last name</label>
                    <input type="text" name="last_name" class="form-control"
id="last_name" placeholder="Enter last name" minlength="1" maxlength="15"
required>
                </div>
            </div>
        </div>
        <div class="mb-3">
            <label for="email" class="form-label">Email</label>
            <input type="email" name="email" class="form-control" id="email"
placeholder="Enter email" required>
        </div>
        <div class="mb-3">
            <label for="phone_number" class="form-label">Phone number</label>
            <div class="row">
        <div class="col-md-2 col-sm-3 col-4 phno_code">
          <input type="number" value="+91 " readonly placeholder="+91"
class="form-control">
        </div>
        <div class="col-md-10 col-sm-9 col-8 phno">
          <input type="number" name="phone_number" class="form-
control" id="phone_number" placeholder="Enter phone number" required>
        </div>
      </div>
    </div>
   <div class="mb-3">
      <label for="interest" class="form-label">Interest</label>
      <input type="text" name="interest" class="form-control" id="interest"
placeholder="Enter interest" required>
    </div>
    <div class="row">
      <div class="col-md-6">
        <div class="mb-3">
          <label for="state" class="form-label">State</label>
          <input type="text" name="state" class="form-control" id="state"
placeholder="Enter state" minlength="3" maxlength="15" required>
        </div>
      </div>
      <div class="col-md-6">
        <div class="mb-3">
          <label for="city" class="form-label">City</label>
          <input type="text" name="city" class="form-control" id="city"
placeholder="Enter city" minlength="3" maxlength="15" required>
        </div>
      </div>
    </div>
    <div class="mb-3">
      <label for="password" class="form-label">Password</label>
      <input type="password" name="password" class="form-control"
id="password" placeholder="Enter password" minlength="3" maxlength="15"
required>
    </div>
    <p class="mt-3">Already have an account <a href="{{ url_for('login')
}}">login</a></p>
    <button type="submit" class="btn btn-primary">Register</button>
  </form>
{% endblock body %}
```

verification_email_send.html:

```
{% extends 'auth_layout/app.html' %}

{% block title %} Account verification {% endblock title %}


{% block body %}
   <h3 class="text-primary">Verification email send</h3>
   <p>Please confirm your email address to complete the registration</p>
{% endblock body %}
```

account_activation_success.html:

```
{% extends 'auth_layout/app.html' %}

{% block title %} Account verification {% endblock title %}


{% block body %}
   {% if is_success %}
      <p><b>{{ username }}</b> your account has been activated successfully.</p>
      <p>You can login now</p>
      <a class="btn btn-primary" href="{{ url_for('login') }}">Login</a>
   {% else %}
      <h2 class="text-danger">Something went wrong token is not validated</h2>
   {% endif %}
{% endblock body %}
```

Layout:

app.html:

```
<!doctype html>
<html lang="en">

<head>
   <!-- Required meta tags -->
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">

   <!-- Bootstrap CSS -->
```

```
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCO
mLASjC" crossorigin="anonymous">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.2.0/css/all.min.css">

    <title>{% block title %} {% endblock title %}</title>

    <style>
        .main {
            background-image: linear-gradient(rgba(0, 0, 0, 0.6), rgba(83, 83, 83,
0.3)), url('https://images.pexels.com/photos/838413/pexels-photo-
838413.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1');
            background-size: cover;
            background-position: center;
            min-height: 90vh;
            width: 100%;
        }

        .apply {
            background: url('static/images/background.png')no-repeat;
            background-size: cover;
            height: 110vh;
            width: 100%;
        }

        .apply .card {
          background-color: rgba(4, 26, 100, 0.801);
            color: white;
        }

        .cir1 img {
            border-radius: 50%;
        }
    </style>

</head>

<body onload="myFunction()">
    {% include 'layout/header.html' %}
    <!-- Page content  -->
    {% block body %}

    {% endblock body %}

    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.j
s"
        integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/t
WtIaxVXM"
        crossorigin="anonymous"></script>
        <script>
            window.watsonAssistantChatOptions = {
              integrationID: "14c61a15-4802-4e67-b6b6-beeff792ff88", // The ID of
this integration.
              region: "jp-tok", // The region your integration is hosted in.
              serviceInstanceID: "a9372ca4-6fba-4a6b-b71c-8a9db5d9f79c", // 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>
</body>

</html>
```

header.html:

```
<nav class="navbar navbar-expand-lg navbar-light bg-primary shadow-sm">
    <div class="container-fluid">
        <a class="navbar-brand text-white fw-bold fs-3" href="{{ url_for('home')
}}">Job Portal</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
data-bs-target="#navbarSupportedContent"
```

```
      aria-controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav me-auto mb-2 mb-lg-0">
        <li class="nav-item">
          <a class="nav-link active text-white fs-5" aria-current="page"
href="{{ url_for('home') }}">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link active text-white fs-5" aria-current="page"
href="{{ url_for(‘jobs') }}">Find jobs</a>
        </li>
      </ul>
      <div>
        {% if session['is_loggedin'] %}
          <div class="dropdown">
            <button class="btn btn-light dropdown-toggle" type="button"
id="dropdownMenuButton1" data-bs-toggle="dropdown" aria-
expanded="false">
              <span class="text-primary">{{ session['email'] }}</sp>
            </button>
            <ul class="dropdown-menu" aria-
labelledby="dropdownMenuButton1">
              <li><a class="dropdown-item" href="{{ url_for('profile')
}}">Profile</a></li>
              <li><a class="dropdown-item" href="{{ url_for('logout')
}}">Logout</a></li>
            </ul>
          </div>
        {% else %}
          <a href="" class="btn btn-light me-2">Register</a>
          <a href="" class="btn btn-outline-light">Login</a>
        {% endif %}
      </div>
    </div>
  </div>
</nav>
```

index.html:

```
{% extends 'layout/app.html' %}

{% block title %} Home Page {% endblock title %}
<!-- Page content  -->
{% block body %}
<section class="main py-5 ">
    <div class="container py-5">
        <div class="row">
            <div class="col-lg-7 mb-5">
                <h1 class="text-primary fs-1 fs-bolder">Find The Most Exciting
Startup Jobs</h1>
            </div>
            <div class="col-lg-7 mt-3">
                <form action="{{url_for('jobs')}}" method="post">
                    <div class="row">
                            <div class="col-md-8">
                                <input type="text" class="form-control py-2"
name="search_query" id="" placeholder="find jobs...">
                            </div>
                            <div class="col-md-4">
                                <button class="btn btn-primary py-2" type="submit">Find
Job</button>
                            </div>
                    </div>
                </form>
            </div>

      </div>
</section>
<section class="job py-5">
    <div class="container py-5">
        <div class="row py-5">
            <div class="col-lg-7">
                <h4 class=" pt-5">What We Are Doing</h4>
                <h1 class="text-primary">24K Talented Peoples Are Getting
Jobs</h1>
                <p class="my-3 text-secondary">Lorem ipsum dolor sit amet
consectetur adipisicing elit. Doloremque
                    totam, accusamus laboriosam quidem quo nulla optio illum magni
deleniti facilis iste! Cupiditate
                    voluptatibus impedit doloribus sunt consequuntur. Nam error
dolorum possimus veritatis debitis velit
```

```
                    voluptas commodi quidem, quisquam rem enim inventore sit vero
nobis facilis nihil aut eius
                    consectetur illo.</p>
                <button class="btn btn-primary me-5">Post A Job</button>
            </div>
            <div class="col lg-5">
                <img src="static/images/mens.jpg" class="img-fluid" alt="">
            </div>
        </div>
    </div>
</section>

<section class="apply py-5">
    <div class="container text-white text-center py-5">
        <p class="red">Apply Process</p>
        <h1>How It Works</h1>
        <div class="row py-5">
            <div class="col-lg-4">
                <div class="card p-3">
                    <div class="card-body text-center">
                        <div class="cir1 pt-3 mb-3">
                            <img src="static/images/cate4.jpg" class="img-fluid" alt="">
                        </div>
                        <h2>1.Search a Job</h2>
                        <p class="py-3">Sorem spsum dolor sit amsectetur adipisclit,
seddo eiusmod tempor
                            incididuntutlaborea.
                        </p>
                    </div>
                </div>
            </div>
            <div class="col-lg-4">
                <div class="card p-3">
                    <div class="card-body text-center">
                        <div class="cir1 pt-3 mb-3">
                            <img src="static/images/cate3.JPG" class="img-fluid" alt="">
                        </div>
                        <h2>1.Apply for a Job</h2>
                        <p class="py-3">Sorem spsum dolor sit amsectetur adipisclit,
seddo eiusmod tempor
                            incididuntutlaborea.
                        </p>
                    </div>
                </div>
            </div>
            <div class="col-lg-4">
                <div class="card p-3">
                    <div class="card-body text-center">
                        <div class="cir1 pt-3 mb-3">
                            <img src="static/images/cate5.JPG" class="img-fluid" alt="">
                        </div>
                        <h2>1.Get hired by MNC's</h2>
                        <p class="py-3">Sorem spsum dolor sit amsectetur adipisclit,
seddo eiusmod tempor
                            incididunutlaborea.
                        </p>
                    </div>
                </div>
            </div>
        </div>
    </div>
</section>
<footer class="w-100 p-3 bg-dark">
    <p class="text-center m-0 text-white">@2022 Job Portal All rights
reserved.</p>
</footer>

{% endblock body %}
```

jobs.html:

```
{% extends 'layout/app.html' %}

{% block title %} Jobs {% endblock title %}
<!-- Page content  -->
{% block body %}

    <div class="container py-3">
        <div class="row">
            <div class="col-md-6">
                <h3 class="text-primary">Jobs</h3>
            </div>
            <div class="col-md-6">
                <form action="{{url_for('jobs')}}" method="post">
                    <div class="d-flex justify-content-center align-items-center">
            <input type="text" class="form-control" name="search_query"
placeholder="Enter...">
            <button class="btn btn-primary" type="submit">Search</button>
          </div>
        </form>
      </div>
    </div>
    <div class="row my-4 py-3">
      {% for job in jobs %}
        <div class="col-md-5 shadow rounded p-3 my-3 mx-4">
          <div class="d-flex justify-content-between align-items-center">
            <h2 class="text-dark fw-bold">{{ job['title'] }}</h2>
          </div>
          <p class="text-secondary">Company: {{
job['company']['display_name'] }}</p>
          <span class="text-secondary">Locations:
            {% for location in job['location']['area'] %}
              {{location}},
            {% endfor %}
          </span>
          <p class="text-secondary">Salary: {{ job['salary_max'] }}</p>
          <hr>
          <p class="text-secondary">Job type: {{ job['contract_type'] }}</p>
          <h4>Description</h4>
          <p class="text-secondary">{{ job['description'] }}</p>
          <a href="{{ job['redirect_url'] }}" class="btn btn-primary"
target="_blank">Apply</a>
        </div>
      {% endfor %}
    </div>
  </div>
{% endblock body %}
```

profile.html:

```
{% extends 'layout/app.html' %}

{% block title %} Profile Page {% endblock title %}
<!-- Page content -->
{% block body %}
<div class="container my-3 p-5 shadow-sm">
  <div class="d-flex justify-content-between">
    {% if error %}
```

```
      <div class="alert alert-primary" role="alert">
        {{ error }}
      </div>
    {% endif %}
    <h2>User Data</h2>
    <div>
      <button type="button" class="btn btn-primary btn-sm me-2" data-bs-
toggle="modal" data-bs-target="#resetpassword">Reset
        Password</button>
      <!-- Modal -->
      <div class="modal fade" id="resetpassword" tabindex="-1" aria-
labelledby="resetpasswordLabel"
        aria-hidden="true">
        <div class="modal-dialog modal-sm">
          <div class="modal-content">
            <div class="modal-header">
              <h5 class="modal-title" id="resetpasswordLabel">Reset
Password</h5>
              <button type="button" class="btn-close" data-bs-
dismiss="modal" aria-label="Close"></button>
            </div>
            <form action="/profile" method="post">
              <div class="modal-body">
                <label for="password" class="form-
label">Password</label>
                <input type="password" class="form-control"
name="password" \>
                <label for="confirm_password" class="form-
label">Confirm Password</label>
                <input type="password" class="form-control"
name="confirm_password" \>
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-sm btn-secondary"
data-bs-dismiss="modal">Close</button>
                <input type="submit" class="btn btn-sm btn-primary"
name="forgot_password" value="Save changes">
              </div>
            </form>
          </div>
        </div>
      </div>
```

```
            <a class="btn btn-outline-primary btn-sm" href="{{
url_for('update_profile') }}">Update Profile</a>
        </div>
    </div>
    <div class="row">
        <div class="col-md-6">
            <div>
                <label for="name" class="form-label">Name</label>
                <input type="text" class="form-control" name="name" value="{{
user['user']['NAME'] }}" readonly>
            </div>
        </div>
        <div class="col-md-6">
            <div>
                <label for="username" class="form-label">Username</label>
                <input type="text" class="form-control" name="username" value="{{
user['user']['USERNAME'] }}"
                    readonly>
            </div>
        </div>
    </div>
    <div class="row">
        <div class="col-md-6">
            <div>
                <label for="email" class="form-label">email</label>
                <input type="text" class="form-control" name="email" value="{{
user['user']['EMAIL'] }}" readonly>
            </div>
        </div>
      <div class="col-md-6">
            <div>
                <label for="phone_number" class="form-label">Phone
number</label>
                <input type="text" class="form-control" name="phone_number"
value="{{ user['user']['PHNO'] }}"
                    readonly>
            </div>
        </div>
    </div>
    <div class="row">
        <div class="col-md-6">
            <div>
                <label for="interest" class="form-label">Interest</label>
                <input type="text" class="form-control" name="interest"
value="{{user['user']['INTEREST']}}" readonly>
            </div>
        </div>
        <div class="col-md-3">
            <div>
                <label for="state" class="form-label">State</label>
                <input type="text" class="form-control" name="state"
value="{{user['user']['STATE']}}" readonly>
            </div>
        </div>
        <div class="col-md-3">
            <div>
                <label for="city" class="form-label">City</label>
                <input type="text" class="form-control" name="city"
value="{{user['user']['CITY']}}" readonly>
            </div>
        </div>
    </div>
    <h2 class="text-primary my-2">Education Details</h2>
    {% for education in user['educations'] %}
      <h4>{{ education['COLLEGE_NAME'] }}</h4>
        <p class="text-secondary m-0">{{ education['DEGREE'] }}</p>
        <p class="text-secondary">Grade: {{ education['GRADE'] }}</p>
    {% endfor %}
    <a class="btn btn-primary mt-4" href="{{ url_for('add_education') }}">Add
Education</a>
</div>
{% endblock body %}
```

update_profile.html:

```
{% extends 'layout/app.html' %}

{% block title %} Profile Page {% endblock title %}
<!-- Page content  -->
{% block body %}
<div class="container my-3 p-5 shadow-sm">
    <div class="d-flex justify-content-between">
        {% if error %}
            <div class="alert alert-primary" role="alert">
                {{ error }}
            </div>
    {% endif %}
    <h2 class="w-100 bg-primary text-center py-2 rounded text-white">User
Data</h2>
  </div>
  <form action="/update_profile" method="post">
    <div class="row">
      <div class="col-md-6">
        <div>
          <label for="name" class="form-label">Name</label>
          <input type="text" class="form-control" name="name" value="{{
user['NAME'] }}" readonly>
        </div>
      </div>
      <div class="col-md-6">
        <div>
          <label for="username" class="form-label">Username</label>
          <input type="text" class="form-control" name="username"
value="{{ user['USERNAME'] }}"
            readonly>
        </div>
      </div>
    </div>
    <div class="row">
      <div class="col-md-6">
        <div>
          <label for="email" class="form-label">email</label>
          <input type="text" class="form-control" name="email" value="{{
user['EMAIL'] }}" readonly>
        </div>
      </div>
      <div class="col-md-6">
        <div>
          <label for="phone_number" class="form-label">Phone
number</label>
          <input type="text" class="form-control" name="phno" value="{{
user['PHNO'] }}">
        </div>
      </div>
    </div>
    <div class="row">
      <div class="col-md-6">
        <div>
          <label for="interest" class="form-label">Interest</label>
```

```
                    <input type="text" class="form-control" name="interest"
value="{{user['INTEREST']}}" >
                </div>
            </div>
            <div class="col-md-3">
                <div>
                    <label for="state" class="form-label">State</label>
                    <input type="text" class="form-control" name="state"
value="{{user['STATE']}}">
                </div>
            </div>
            <div class="col-md-3">
                <div>
                <label for="city" class="form-label">City</label>
                    <input type="text" class="form-control" name="city"
value="{{user['CITY']}}">
                </div>
            </div>
        </div>
        <button type="submit" class="btn btn-primary my-3">Update</button>
    </form>
</div>
{% endblock body %}
```

BACKEND:

app.py:

```
from flask import Flask ,render_template, request, redirect, url_for, session, g,
flash
from passlib.hash import sha256_crypt
from dotenv import load_dotenv
from datetime import timedelta
from flask_mail import Mail, Message
from itsdangerous import URLSafeTimedSerializer, SignatureExpired
from datetime import datetime
import ibm_db
import os
import requests
import json


load_dotenv()
```

```
app = Flask(__name__)

DATABASE_NAME = os.getenv("DATABASE_NAME")
HOST_NAME = os.getenv("HOST_NAME")
PORT_NUMBER = os.getenv("PORT_NUMBER")
USER_ID = os.getenv("USER_ID")
PASSWORD = os.getenv("PASSWORD")

# IBM cloud connection string
conn                                                                 =
ibm_db.connect(f"DATABASE={DATABASE_NAME};HOSTNAME={HOS
T_NAME};PORT={PORT_NUMBER};SECURITY=SSL;SSLServerCertificat
e=DigiCertGlobalRootCA.crt;UID={USER_ID};PWD={PASSWORD}","","")

# secret key
app.secret_key = os.getenv('SECRET_KEY')

# configuration of mail
app.config['MAIL_SERVER']='smtp.gmail.com'
app.config['MAIL_PORT'] = 465
app.config['MAIL_USERNAME'] = os.getenv('MAIL_USERNAME')
app.config['MAIL_PASSWORD'] = os.getenv('MAIL_PASSWORD')
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True

mail = Mail(app)
s = URLSafeTimedSerializer('Thisisasecret!')

app.permanent_session_lifetime = timedelta(days=30)

@app.route("/")
def home():
    if g.email:
        return render_template('index.html')
    else:
        return redirect(url_for('login'))

@app.before_request
def before_request():
    g.email = None

    if 'email' in session:
        g.email = session['email']
```

```
# auth routes

@app.route('/login', methods=['POST', "GET"])
def login():
    if request.method == 'POST':
        username = request.form.get('username')
        password = request.form.get('password')

        sql = "select * from FVS49663.USER where username = ?"
        stmt = ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt, 1, username)
        ibm_db.execute(stmt)
        user = ibm_db.fetch_assoc(stmt)
        if user['IS_VERIFIED']:
            if user:
                print(user)
                print(user['ID'])
                user_password = user['PASSWORD']
                if(sha256_crypt.verify(password, user_password)):
                    session.permanent = True
                    session['id'] = user['ID']
                    session['email'] = user['EMAIL']
                    session['interest'] = user['INTEREST']
                    session['is_loggedin'] = True
                    return redirect(url_for('home'))
                else:
                    flash('Password is incorrect')
                    return render_template('auth/login.html')
            else:
                flash('Username is incorrect')
                return render_template('auth/login.html')
        else:
            return 'Please verify your account'
    else:
        return render_template('auth/login.html')

@app.route('/logout')
def logout():
    if 'email' in session:
        session.pop('id',None)
        session.pop('email',None)
        session.pop('is_loggedin',None)
        return redirect(url_for('login'))

@app.route('/register', methods=['POST', "GET"])
def register():
    if request.method == 'POST':
        username = request.form.get('username')
        first_name = request.form.get('first_name')
        last_name = request.form.get('last_name')
        email = request.form.get('email')
        phno = str(request.form.get('phone_number'))
        state = request.form.get('state')
        city = request.form.get('city')
        password = request.form.get('password')
        encrypt_password = sha256_crypt.encrypt(password)
        name = first_name + ' ' + last_name
        interest = request.form.get('interest')
        is_verified = False

        sql = "INSERT INTO
FVS49663.USER(username,name,email,phno,state,city,password,interest,is_ver
ified)
VALUES('{}','{}','{}','{}','{}','{}','{}','{}','{}')".format(username,name,email,ph
no,state,city,encrypt_password,interest,is_verified)
        ibm_db.exec_immediate(conn,sql)


        token = s.dumps(email, salt='email-confirm')
        msg = Message('Confirm Email', sender= os.getenv('FROM_EMAIL'),
recipients=[email])
        link = url_for('confirm_email', token=token + ',' +username,
_external=True)
        msg.body = 'Your link is {}'.format(link)
        mail.send(msg)
        return render_template('auth/verification_email_send.html')
    else:
        return render_template('auth/register.html')


@app.route('/confirm_email/<token>/')
def confirm_email(token):
    data = token.split(',')
    print(data)
    try:
        email = s.loads(data[0], salt='email-confirm', max_age=3600)
    except SignatureExpired:
        return '<h1>The token is expired!</h1>'

    sql = "UPDATE FVS49663.USER set is_verified = True where username =
'{}'".format(data[1])
    ibm_db.exec_immediate(conn,sql)
    return render_template('auth/account_activation_success.html',is_success
= True)




# Job routes
@app.route('/jobs',methods=['POST','GET'])
def jobs():
    if g.email:
        if request.method == 'POST':
            search_query = request.form.get('search_query')
            req =
requests.get("http://api.adzuna.com/v1/api/jobs/gb/search/1?app_id=b8adcac4&
app_key=68108197016f443e2c24af0587b39471&results_per_page=50&what='
{}'&content-type=application/json".format(search_query))
            jobs = json.loads(req.content)
            return render_template('jobs.html', jobs = jobs['results'])
        else:
            sql = "select * from FVS49663.USER where id = ?"
            stmt = ibm_db.prepare(conn,sql)
            ibm_db.bind_param(stmt, 1, session['id'])
            ibm_db.execute(stmt)

            user = ibm_db.fetch_assoc(stmt)

            req =
requests.get("http://api.adzuna.com/v1/api/jobs/gb/search/1?app_id=b8adcac4&
app_key=68108197016f443e2c24af0587b39471&results_per_page=50&what='
{}'&content-type=application/json".format(user['INTEREST']))
            jobs = json.loads(req.content)
            return render_template('jobs.html', jobs = jobs['results'])
    else:
        return redirect(url_for('login'))


@app.route('/profile',methods=['POST','GET'])
```

```
def profile():
    if g.email:
        if request.method == 'POST':
            password = request.form.get('password')
            confirm_password = request.form.get('confirm_password')
            if password == confirm_password:
                print("Ok")
                user = session['id']
                encrypt_password = sha256_crypt.encrypt(password)
                sql = "UPDATE FVS49663.USER set password = '{}' where id =
'{}'".format(encrypt_password,user)
                ibm_db.exec_immediate(conn,sql)
                return redirect(url_for('profile'))
        else:
            sql = "select * from FVS49663.USER where id = ?"
            stmt = ibm_db.prepare(conn,sql)
            ibm_db.bind_param(stmt, 1, session['id'])
            ibm_db.execute(stmt)

            user = ibm_db.fetch_assoc(stmt)

            sql1 = "select * from FVS49663.EDUCATION where id = ?"
            stmt1 = ibm_db.prepare(conn,sql1)
            ibm_db.bind_param(stmt1, 1, session['id'])
            ibm_db.execute(stmt1)

            educations = []
            dictionary = ibm_db.fetch_both(stmt1)
            while dictionary != False:
                educations.append(dictionary)
                dictionary = ibm_db.fetch_both(stmt1)

            print(educations)
            return
render_template('profile.html',user={"user":user,"educations":educations})

    else:
        return redirect(url_for('login'))

@app.route("/add_education",methods=['POST','GET'])
def add_education():
    if g.email:
        if request.method == 'POST':
            college_name = request.form.get('college_name')
            degree = request.form.get('degree')
            grade = request.form.get('grade')
            userid = session['id']
            sql = "insert into
FVS49663.EDUCATION(college_name,degree,grade,userid) values
('{}','{}','{}','{}')".format(college_name,degree,grade,userid)
            ibm_db.exec_immediate(conn,sql)
            return redirect(url_for('profile'))
        else:
            return render_template('add_education.html')
    else:
        return redirect(url_for('login'))


@app.route("/update_profile",methods=['POST','GET'])
def update_profile():
    if g.email:
        if request.method == 'POST':
            phno = str(request.form.get('phone_number'))
            state = request.form.get('state')
            city = request.form.get('city')
            interest = request.form.get('interest')
            userid = session['id']

            sql = "UPDATE FVS49663.USER set phno = '{}',state = '{}',city =
'{}',interest = '{}' where id = '{}'".format(phno,state,city,interest,userid)
            ibm_db.exec_immediate(conn,sql)
            return redirect(url_for('profile'))
        else:
            sql = "select * from FVS49663.USER where id = ?"
            stmt = ibm_db.prepare(conn,sql)
            ibm_db.bind_param(stmt, 1, session['id'])
            ibm_db.execute(stmt)

            user = ibm_db.fetch_assoc(stmt)
            return render_template('update_profile.html',user = user)
    else:
        return redirect(url_for('login'))
```

GITHUB & PROJECT DEMO LINK:

GITHUB LINK

https://github.com/IBM-EPBL/IBM-Project-46904-1660794589

PROJECT DEMO LINK

https://drive.google.com/file/d/1IYKr01kynEIFYlV1cLVhgsRXuEV8RI0D/view?usp=sharing