

PROJECT REPORT

SKILL/JOB RECOMMENDER APPLICATION

SUBMITTED BY

PNT2022TMID05017

Nagendhiran M	921319104128
Rishibabu RR	921319104160
Mukesh kanna	921319104121
Sarvesvara R	921319104170

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose.

2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

8. TESTING

8.1 Test Cases

8.2 User Acceptance Testing

9. RESULTS

9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub &Project Demo Link

1. INTRODUCTION

1.1 Project Overview

Finding Jobs that best suits the interest and skill set is quite a challenging task for the Job seekers. Job portal is the solution where recruiter as well as the job seeker meet aiming at fulfilling their individual requirements. They are the cheapest as well as the fastest source of communication reaching wide range of audience on just a single click irrespective of distance. The web application "Seek-Job" provides an easy and convenient search application for the job seekers to find their desired jobs and for the recruiters to find the right candidate. Job seekers can register with the application and update their details and skills. They can search for the available jobs and apply to their desired positions. The recruiters can also register and login into the web application can post the jobs with the job description and the skills needed for the candidate for applying the job. So our project makes easy for both seekers as well as recruiters. Our project are simple in structure and it is easy to understand. The aim is to come up with a Skill/Job Recommender Application System called "Seek-Job" which takes the skills from job seekers and jobs from indeed and throws the best jobs available for user according to user skills. The Skill/Job Recommender Application perform their suggestions based on the full profile of job-seekers. It aims to help job-seekers to find suitable jobs. Job offers are collected from job search websites, then they are prepared to extract meaningful attributes such as job titles and technical skills. Based on the person-job fit premise, we propose a framework for Job Recommendation based on the professional skills are job-seekers.

1.2 Purpose

In today's competitive environment, where everybody wants to be on the top, Information plays a very crucial role. As fast as information is accessed and processed , it can give good result. Normally many companies waste their money and valuable time when finding the right candidates for the job. Also Job seekers is having lots of skills but wondering which job will best suit them? For these reasons We have come up with a skill/ Job 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 chat bot and get their dream job. The main purpose of the project is to change the manual way of operation of the job application to an automatic system. This system will help to reduce the time required during the job search process. It will help the employees and the employers to reduce the human resources needed in a company. 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 chat bot 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. The main purpose of our project is to provide the best job which are suited correctly for the candidate. So that both the recruiter as well as the candidate gets benefit in finding the right person for the right position. In the job portal application user can freely enter their personal and professional details and easily apply for the job which are suited most for him.

2. LITERATURE SURVEY

2.1 Existing problem

The existing problem for job recruitment traditional methods like Employment agencies, advertising through newspapers, televisions and radios, college fairs etc., are too slow and stressful. With the advancement of internet, job seekers rely on the job portal website which makes the job search efficient. And many job seekers having lot of skills but wondering which job are suited for them. So to avoid these problems we came up with an application called Seek-Job in which job seekers can easily find their suited job for their skills.

2.2 References

Journal 1

Authors: Vachik S. Dave, MohammadAl Hasan, Baichuan Zhang, Khalifeh Aljadda and Mohammed Korayem.

Title: A combined representation learning approach for better skill Recommendation

Publisher: Association for Computing Machinery New York,NY, United States.

Yearof Publication: October 2018

DOI: Article 4 <https://doi.org/https://doi.org/10.1145/3132847.3132873>

Abstract: In this experiments, it is shown that by jointly learning the representation for the jobs and skills, Our modelprovides better recommendation for both jobs and skills.

Journal 2

Authors: Yao Lu, Sandy El Helou, Denis Gillet.

Title: A Recommender System for Job Seeking and Recruiting

Publisher: Association for Computing Machinery New York, NY, US

Year of Publication: May 2013.

DOI: 10.1145/2487788

Abstract : In this paper, a hybrid recommender system for job seeking and recruiting websites is presented. The various interaction features designed on the website help the users organize the resources they need as well as express their interest.

Journal 3

Authors: W. Shalaby, B. AlAila, M. Korayem, L. Pournajaf, K. AlJadda, S. Quinn

Title: Help Me Find a Job : A Graph-based Approach for Job Recommendation at Scale

Publisher: IEEE

Year of Publication: December 2017

DOI: 10.1109/BigData.2017.8258088

Abstract: Existing systems are mostly focused on content analysis of resumes and job descriptions, relying heavily on the accuracy and coverage of the semantic analysis and modeling of the content in which case, they end up usually suffering from rigidity and lack of implicit semantic relations that are uncovered from users behaviour .

Journal 4

Authors: S. Choudhary, S. Koul, S.Mishra, A. Thakur and R.Jain

Title: Collaborative Job Prediction based on Naïve Bayes Classifier using Python Platform

Publisher: IEEE

Year of Publication: October 2016

DOI:10.1109/CSITSS.2016.7779375

Abstract: The paper aims to implement recommendation system based on collaborative filtering technique for job portals.

Journal 5

Authors: Jorge Valverde-Rebaza, Ricardo Puma, Paul Bustios, Nathalia C. Silva

Title: Job Recommendation based on Job Seeker Skills: An Empirical Study

Publisher: A. Jorge, R. Campos, A. Jatowt, S. Nunes (eds.): Proceedings of the Text2StoryIR'18 Workshop, Grenoble, France

Year of Publication: March 2018

DOI:<https://www.researchgate.net/publication/325697854>

Abstract: The contributions of the work are, they made publically available a new dataset formed by a set of job seekers profiles put forward the proposal of a framework for job recommendation based on professional skills of job seekers.

Journal 6

Authors: G. Domeniconi, G. Moro, A. Pagliarani, K. Pasini and R. Pasolini

Title: Job Recommendation From Semantic Similarity of LinkedIn User's Skills

Publisher: SCITEPRESS - Science and Technology Publications, Lda Setubal, Portugal

Year of Publication: July 2016

DOI: 10.5220/0005702302700277

Abstract: Until recently job seeking has been a tricky, tedious and time consuming process, because people looking for a new position had to collect information from many different sources.

Journal 7

Authors: Miao Jiang, Yi Fang, Huangming Xie and Jike Chong

Title: User click prediction for personalized job recommendation

Publisher: Kluwer Academic Publishers United States

Year of Publication: April 2018

DOI: <https://doi.org/10.1007/s11280-018-0568-z>

Abstract : Major job search engines aggregate tens of millions of job postings online to enable job seekers to find valuable employment opportunities. Predicting the probability that a given user clicks on jobs is crucial to job search engines as the predictions can be used to provide personalized job recommendations for job seekers.

Journal 8

Authors: J. Malinowski, T. Keim, O. Wendt and T. Weitzel

Title: Matching People and Jobs

Publisher: IEEE

Year of Publication: January 2006

DOI: 10.1109/HICSS.2006.266

Abstract: Theory shows that a good match between persons and jobs needs to consider both, the preferences of the recruiter and the preferences of the candidate. Based on this requirement for modeling bilateral selection, decisions they present an approach applying to distinct recommendation systems to the field in order to improve the match between people and jobs.

Journal 9

Authors: Kuan Liu, Xing Shi, Anoop Kumar, Linhong Zhu, Prem Natarajan

Title: Temporal Learning and Sequence Modeling for a Job Recommender System

Publisher: Association for Computing Machinery New York, NY, United States

Year of Publication: September 2016

DOI: <https://doi.org/10.1145/2987538.2987540>

Abstract: They present their solution to the job recommendation task for RecSys Challenge 2016. The main contribution of their work is to combine temporal learning with sequence modeling to capture complex user item activity patterns to improve job recommendations.

Journal 10

Authors: I. Paparrizos, B. Cambazoglu and A. Gionis

Title: Machine Learned Job Recommendation.

Publisher: Association for Computing Machinery New York, NY, United States

Year of Publication: October 2011

DOI: <https://doi.org/10.1145/2043932.2043994>

Abstract: They address the problem of recommending suitable jobs to people who are seeking a new job . They formulate this recommendation problem as a supervised machine learning problem our technique exploits all passed job transmission as well as the data associated with employees and institutions to predict an employees next job.

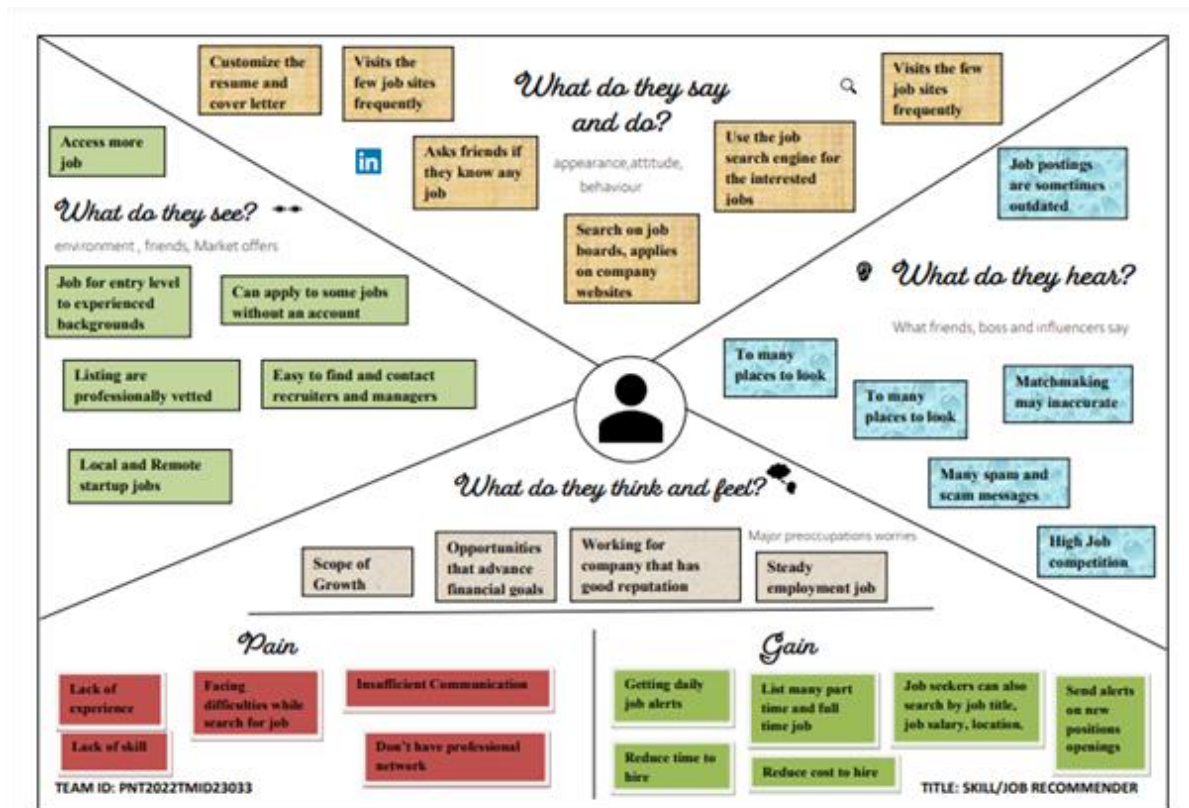
2.3 Problem Statement Definition

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 chat bot 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.

3. IDEATION & PROPOSED SOLUTION

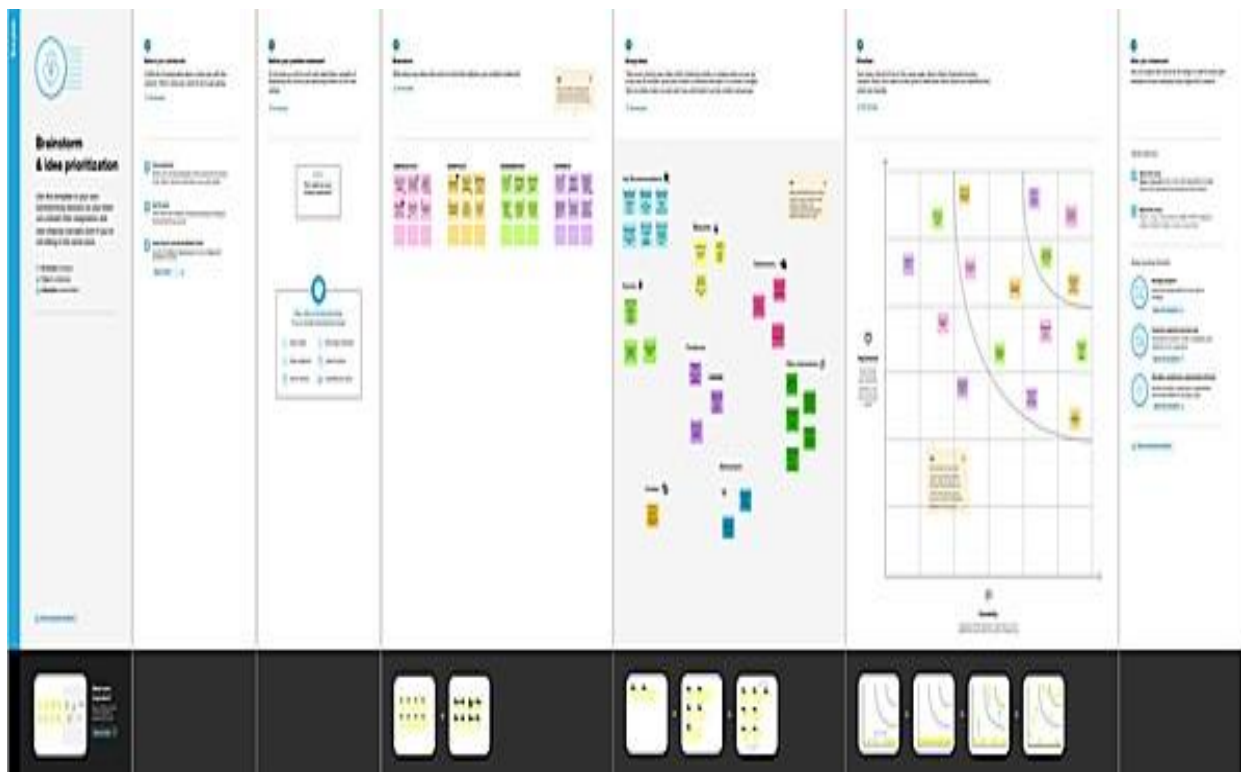
3.1 Empathy Map Canvas

An Empathy Map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to help team better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges. An Empathy Map organizes a user's behaviours and feelings to create a sense of empathy between the user and the team. The below diagram is the Empathy Map canvas for our skill/Job Recommender Application



3.2 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving.



Brainstorming ideas of our skill/Job Recommender Application

1. Search recruiters using filters like role and ongoing job postings.
2. Recommend jobs based on experience or skills.
3. Validating the resumes of the user.
4. Recommend jobs based on users search criteria.
5. Recommend jobs based on users work time need (part-time / full-time).
6. Notification of the job vacancies will reach quickly.
7. Provide company's specific resume builder.
8. Connected with recruiting companies so they can announce vacancies.
9. Suggest the firm as per the seek of user like certification, maternity leaves.
10. Individual login ID's are provided to users for security purpose.
11. Asking for users queries and user feedback.
12. Can track who viewed our profile.
13. Announcing the company recruitment process.
14. Connected with social media so, user gets update easily.
15. Checking the falsified information.

3.3 Proposed Solution

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	<p>Having lots o skills but wondering which job will best suit you? Dont need to worry! We have come up with a skill recommender solution through which the fresher or the skilled person can login and find the jobs by using the search option or they can directly interact with the chatbot and get their dream job.</p> <p>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 Dtabase.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.</p>
2.	Idea /Solution description	<p>The contributions of this work are threefold, we: i) made publicly available a new dataset formed by a set of job seekers profiles and a set of job vacancies collected from different job search engines sites ii) put forward the proposal of a framework for job recommendation based on professional skills of job seekers iii) carried out an evaluation to quantify emprically the recommendation abilities of two-state-of-the-art methods, considering the configurations within the proposal framework. We thus present a general panorama of job recommendation task aiming to facilitate research and real world application design regarding this important issue.</p>

3.	Novelty/Uniqueness	<p>The best positions are suggested to any person according to her skills. While the positions of known profiles are assumed to be correct, it should be noted that there is usually multiple advisable positions corresponding to a set of skills. A recommendation System should return a set of most likely positions and all of them can be equally valid.</p> <p>The recommendation system we use is simply based on representing both positions and profiles as comparable vectors and seeking for each profile the position with most similar vectors.</p>
4.	Social Impact/Customer Satisfaction	Students will be benefited as they will get to know which job suits them based on their skill set and therefore lack of unemployment can be reduced.
5.	Business Model (Revenue Model)	We can provide the application for job seekers in a subscription based and we can share the profiles with the companies and generate the revenue by providing them best profiles.
6.	Scalability of the solution	Data can be scaled up and scaled down according to number of current job openings available.

3.4 Problem Solution fit

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem. It helps entrepreneurs, marketers and corporate innovators identify behavioural patterns and recognize what would work and why.

Problem-Solution fit canvas 2.0

Skill/Job recommender

Team ID - PNT2022TMID05017

Define CS, fit into CC

1. CUSTOMER SEGMENT(S)
Who is your customer?
i.e. working parents of 0-5y. o. kids

CS

The main customers for our project are :

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

Focus on J&P, tap into

2. JOBS-TO-BE-DONE / PROBLEMS
Which jobs-to-be-done (or problems) do you address for your customers?
There could be more than one; explore different sides.

J&P

- 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
- Profile with safe personal data

Identify strong TR & EM

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

TR

4. EMOTIONS: BEFORE / AFTER
How do customers feel when they face a problem or a job and afterwards?

EM

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

Explore AS, differentiate

6. CUSTOMER CONSTRAINTS
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.

CC

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

5. AVAILABLE SOLUTIONS
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

AS

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

Focus on J&P, tap into C

9. PROBLEM ROOT CAUSE
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.

RC

- 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
- Users pretend to have expertise in a skillset they lack

7. BEHAVIOUR
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)

BE

- 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
- When candidates with inadequate qualifications apply for a position, employers become irritated.

Extract online & offline CH of BE

10. YOUR SOLUTION
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.

SL

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


CH

- Apply for jobs
- Review job applications
- Attend initial level assessment

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
- Finalize paperwork

 Problem-Solution fit canvas is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 license Created by Daria Niegolakhina / Amaltama.com



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Sign in / Login	Register with username, password
FR-2	Profile Registration	Register with username, password, email, qualification, skills. This data will be stored in a database.
FR-3	Job profile display	Display job profiles based on availability, location, skills.
FR-4	Chatbot	A chat on the webpage to solve user queries and issues.
FR-5	Job Registration	The company's registration/Description details will be sent to the registered email id of the user.
FR-6	Logout	Use logout option after completing job registration process.

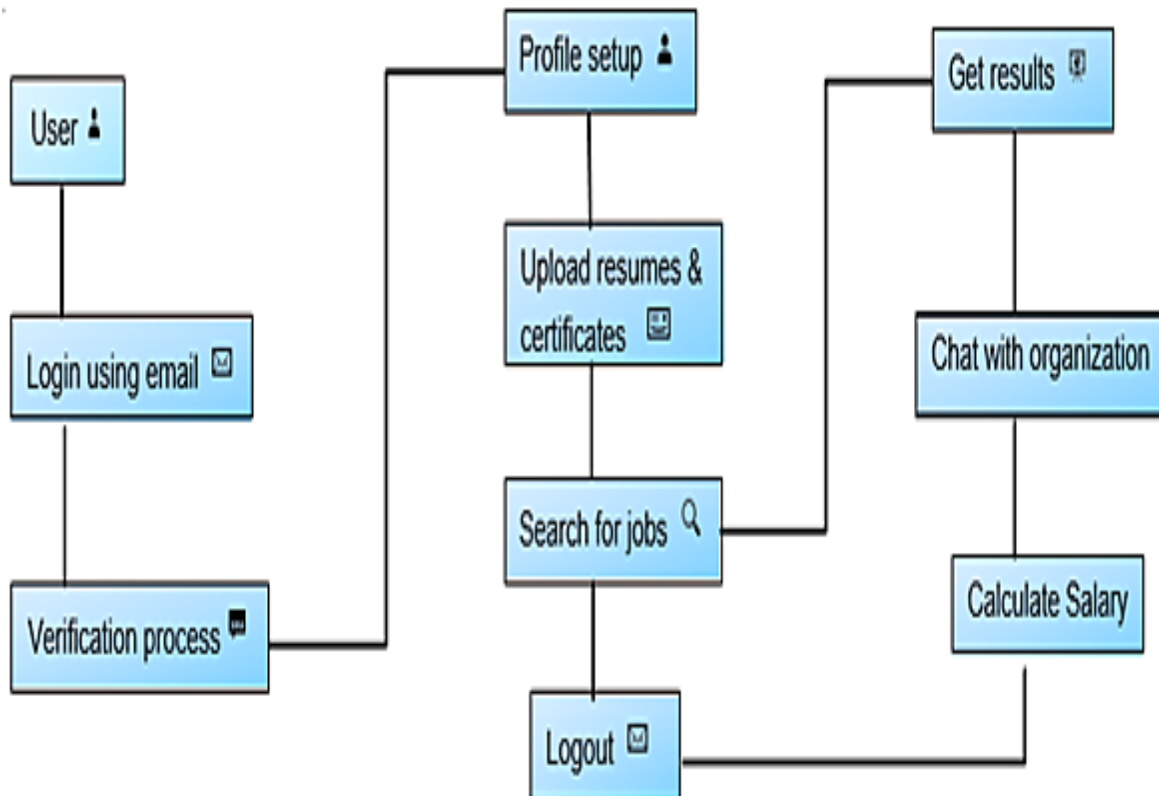
4.2 Non-Functional requirement

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The webpage will be designed in such a way that any non-technical user can easily navigate through it and complete the job registration work. (easy and simple design)
NFR-2	Security	Using of python flask to cloud connect will provide security to the project. Database will be safely stored in DB2.
NFR-3	Reliability	To make sure the webpage doesn't go down due to network traffic.
NFR-4	Performance	Focus on loading the webpage as quickly as possible irrespective of the number of user/integrator traffic.
NFR-5	Availability	The webpage will be available to all users (network connectivity is necessary) at any given point of time.
NFR-6	Scalability	Increasing the storage space of database can increase the number of users. Add some features in future to make the webpage unique and attractive.

5. PROJECT DESIGN

5.1 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

Technical Architecture

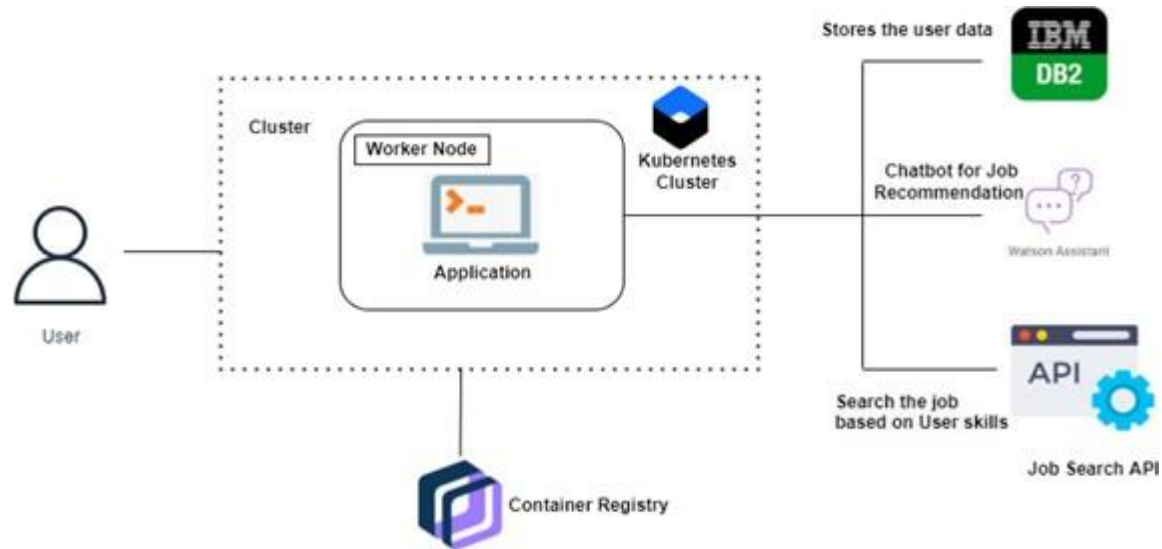


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chat bot etc.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Developing Interface	Developing application for the task	Java / Python
3.	Voice Assistance	Voice commands instead of typing.	IBM Watson STT service
4.	Chat bot Assistance	Conversational Interface	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local File System
8.	Machine Learning Model	Purpose of Machine Learning Model	Object Recognition Model, etc.
9.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Cloud Foundry, Kubernetes, etc.

Table-2: Application Characteristics:

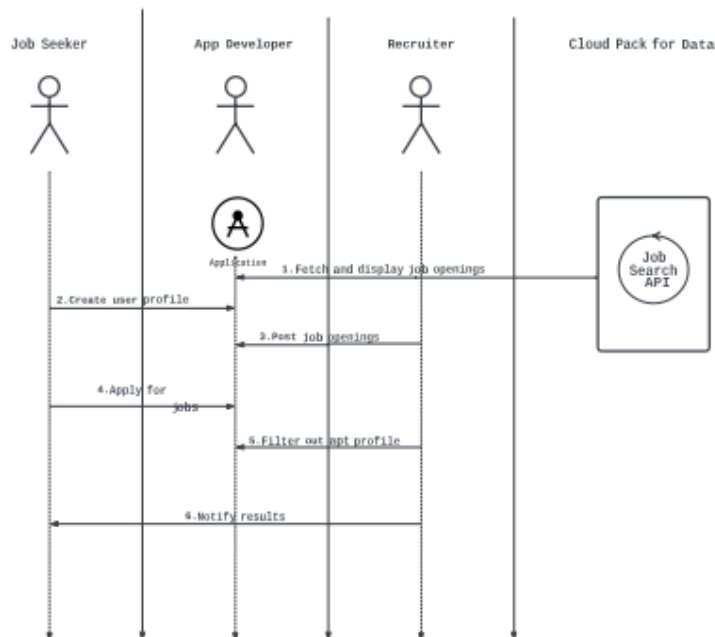
S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Technology of Open Source framework
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Artificial Intelligence (AI)
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	RAID(redundant array of independent disks)
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	DRAM or flash memory

Solution Architecture

SOLUTION ARCHITECTURE

TITLE : SKILL/JOB RECOMMENDER

TECHNOLOGY : CLOUD APPLICATION DEVELOPMENT



5.3 User Stories

User stories help articulate what value a product feature can bring and have a better understanding of why users want a certain functionality. It helps the product manager and development team shift their focus from writing about the software features to discussing the features. The user story focuses on the experience - what the person using the product wants to be able to do. A traditional Requirement focuses on functionality - what the product should do. The process of writing so-called user stories is crucial even if a development team works with already given specs and requirements.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard	USN-5	As a user, I can access my dashboard after signing in.	I can access my account / dashboard	High	Sprint-1
Customer (Web user)	Access	USN-6	As a user, I can setup a profile, and basic details by signing in.			
		USN-7	As a user, I will upload my resume, certificates, and other requirements.	I can perform several task in the application	Medium	Sprint-1
Customer Care Executive	Chatbot	USN-8	As a user, I can seek guidance from the customer care executive.		High	Sprint-1
Administrator	DBMS	USN-9	As a administrator, I can keep the applications of your organization relies on running.	I can perform various modifications in the applications.	High	Sprint-1

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning and Estimation

Title	Description	Date
Literature Survey and Information Gathering	Gathering Information by referring the technical papers, research publications etc.	2 SEPTEMBER 2022
Prepare Empathy Map	To capture user pain and gains Prepare List of Problem Statement	10 SEPTEMBER 2022
Ideation	Prioritize a top 3 ideas based on feasibility and Importance	17 SEPTEMBER 2022
Proposed Solution	Solution include novelty, feasibility, business model, social impact and scalability of solution	24 SEPTEMBER 2022
Problem Solution Fit	Solution fit document	1 OCTOBER 2022
Solution Architecture	Solution Architecture	1 OCTOBER 2022
Customer Journey	To Understand User Interactions and experiences with application	8 OCTOBER 2022
Functional Requirement	Prepare functional Requirement	15 OCTOBER 2022
Data flow Diagrams	Data flow diagram	15 OCTOBER 2022
Technology Architecture	Technology Architecture diagram	15 OCTOBER 2022
Milestone & sprint delivery plan	Activity what we done & further plans	28 OCTOBER 2022
Project Development Delivery of sprint 1,2,3 & 4	Develop and submit the developed code by testing it	28 OCTOBER 2022 – 19 NOVEMBER 2022

6.2 Sprint Delievery Schedule

A Sprint schedule is a document that outlines sprint planning from end to end . Its one of the first steps in the agile sprint planning process and it is something that requires adequate research planning and communication

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Regisration	USN-1	UI Creation Creating Registration page, Login page	10	Medium	AMRUDHA NISKRIYAA PADMASREE SAFREEN
Sprint-1	Database Connectivity	USN-2	Viewing and appling jobs Connecting UI with Database	10	High	AMRUDHA NISKRIYAA
Sprint-2	Send Grid Intezgration	USN-3	Send Grid Intezgration with Python Code	10	Low	PADMASREE SAFREEN
Sprint-2	Chat bot Development	USN-4	Building a chat bot	10	High	AMRUDHA PADMASREE
Sprint-3	Integration and Containerization	USN-5	Integrating chat bot to the HTML page and containerizing the app.	20	Medium	AMRUDHA NISKRIYAA PADMASREE SAFREEN
Sprint-4	Upload Image and deployment	USN-6	Upload the image to the IBM Registry and deploy it in the Kubernetes Cluster.	20	High	SAFREEN NISKRIYAA

7. CODING & SOLUTIONING

7.1 Feature 1

The Graphical User Interface are developed using HTML, CSS , JavaScript and BootStrap to provide and easy to understand interactive screens. Various screen and the navigation between screens are discussed below.

Web Application Screens:

Home Page:

The home page appears on the start of the application. The screen will provide various functionalities like Home, Login , Register. When clicked on the button , it navigates to the respective screens.

Source Code:

```
<html lang="en">

<head>

<link
href="https://fonts.googleapis.com/css2?family=Noto+Serif&family=Nunito&display=swap"
rel="stylesheet">

<title>Seek-Job</title>

</head>

<body>

<nav class="navbar">

<h2 class="navbar-logo"><a href="#"></a></h2>

<div class="navbar-menu">
```

Home

Register

Sign In

</div>

<header>

<h1 class="header-title">FIND YOUR
 PERFECT & DREAM JOB

EASILY
<marquee width="100%" direction="left" height="100px">We are not looking for
job seekers. We want changemakers!</marquee></h1>

</header>

<div class="filter-box">

<div class="filter-dropdown">

<option>Job level</option>

<option>Entry</option>

<option>Mid-Senior</option>

<option>Director</option>

</select>

<select class="filter-select" id="employment" name="employment">

<option>Employement Type</option>

<option>Internship</option>

<div class="job-label">

HTML

CSS

```
<a class="label-c" href="#">Javascript</a>
```

```
<div class="job-posted">Posted 2 mins ago</div></div>
```

```
<div class="job-label">
```

```
<a class="label-a" href="#">HTML</a>
```

```
<a class="label-b" href="#">CSS</a>
```

```
<a class="label-c" href="#">Javascript</a></div><div class="job-posted">Posted 7 mins ago
```

```
</div></div>
```

```
<div class="job-posted">Posted an hour ago</div></div><button class="job-more">More List  
is Loading..</button></section>
```

```

```

```
<div class="blog-detail">
```

```
<h4>Managing Time effectively!</h4>
```

```
<hr class="divider"><a href="#" class="blog-more"></a></hr></div></div></div> </section>
```

```
<script>
```

```
    window.watsonAssistantChatOptions = {
```

```
        integrationID: "7cb16381-61ae-455f-8312-8aedf4ae1f9b", // The ID of this integration.
```

```
        region: "au-syd", // The region your integration is hosted in.
```

```
        serviceInstanceID: "7ee7effa-bbb2-471b-a5c1-78adad21acf9", // 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);

});

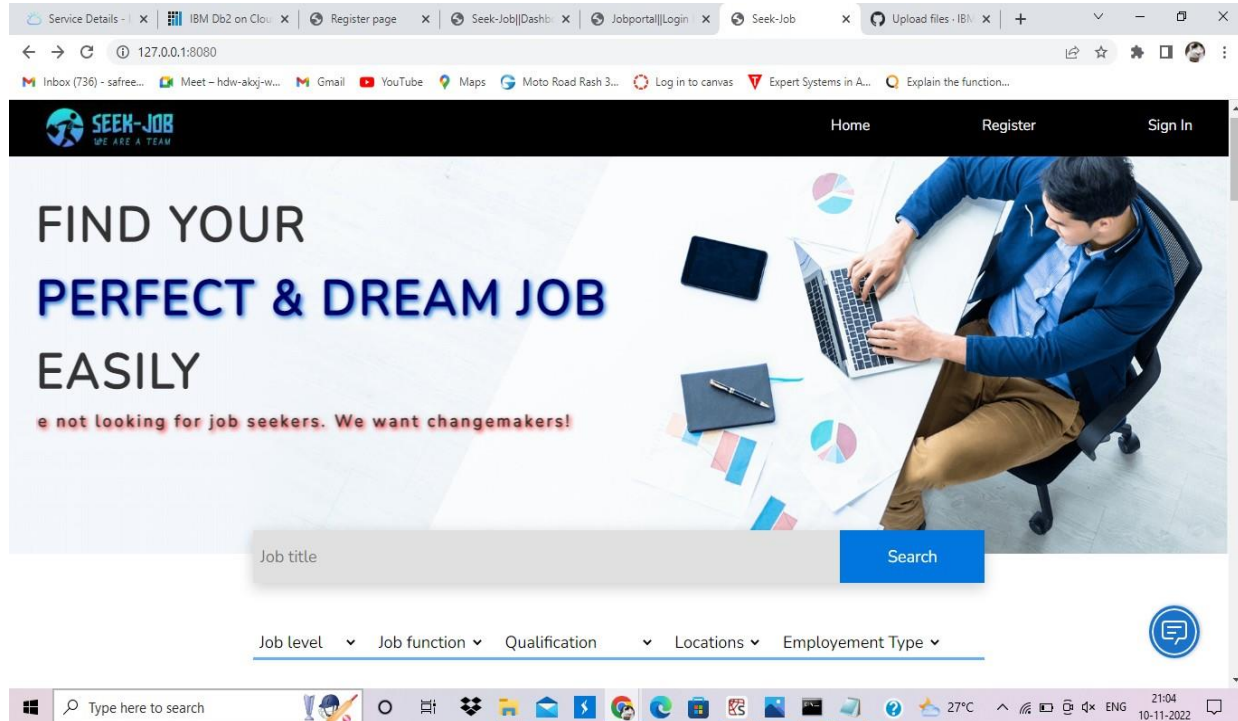
</script>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script></body>

</html>

```

Figure 7.1



Login Page

The user can login to web application by providing the valid details

Source code

```
<!DOCTYPE html>

<html lang="en" >

<head>

<title>Jobportal||Login Page</title>

<link href="https://fonts.googleapis.com/css?family=Rubik&display=swap" rel="stylesheet">

<link rel="stylesheet" href="static/logincss.css"></head>

<body>

<div class="header">

<h2>SEEK-JOB</h2><br><br><br>

<h1 class="animation a1">Welcome Back!</h1>

<h4 class="animation a2">Login for entering dashboard.</h4></div>

<form action="\login" method="post">

<div class="form">

<input type="text" class="form-field animation a3" placeholder="Username" id="username"

name="username">

<input type="password" class="form-field animation a4" placeholder="Password"

id="password" name="password"><br>

<button class="animation a5">LOGIN</button><br>

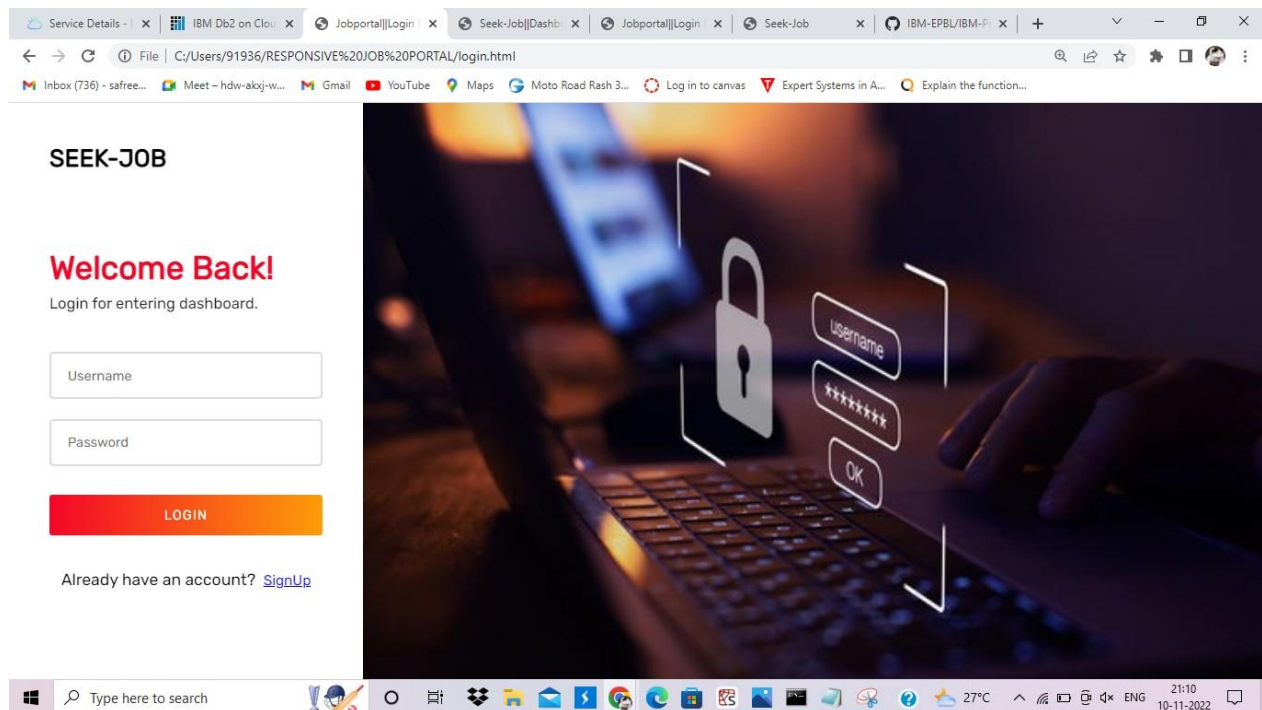
<p class="animation a6">Already have an account?&nbsp;&nbsp;&nbsp;<a href="/register"
```

```
style="color:blue">SignUp</a></p> </div>
```

```
</form></body>
```

```
</html>
```

Figure 7.2



Register page

If the user is new to this application, then user should register into the web application by providing the necessary details

Source code

```
<!DOCTYPE html>
```

```
<html lang="en" dir="ltr">
```

```
<head>
```

```
<title>Register page</title>
```

```
<link rel="stylesheet" href="static/login1css.css">

</head> <body>

<div class="container">

<div class="text">

<span class="text-1">Complete miles of journey<br> with one step</span>

<span class="text-2">Let's get started</span></div></div></div>

<div class="forms">

<form action="/login">

<div class="input-boxes">

<input type="text" placeholder="Enter Username" id="username" name="username" required>

</div>

<div class="input-box">

<input type="text" placeholder="Enter the password" id="password" name="password" required>

</div>

<div class="input-box">

<input type="password" placeholder="Enter your email" id="email" name="email" required>

</div>

<div class="button input-box"><input type="submit" value="Create Account"></div>

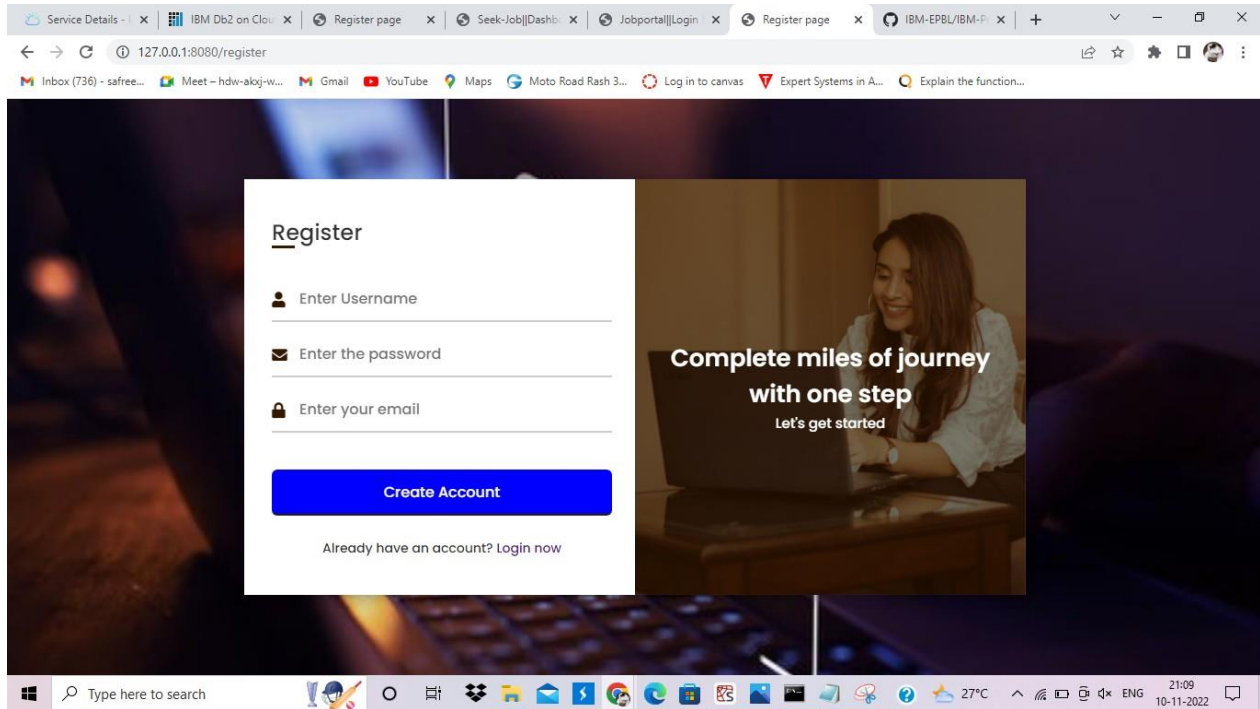
</form><div class="text sign-up-text">Already have an account? <a href="/login">Login

now</a></div></div></div>

</body>

</html>
```

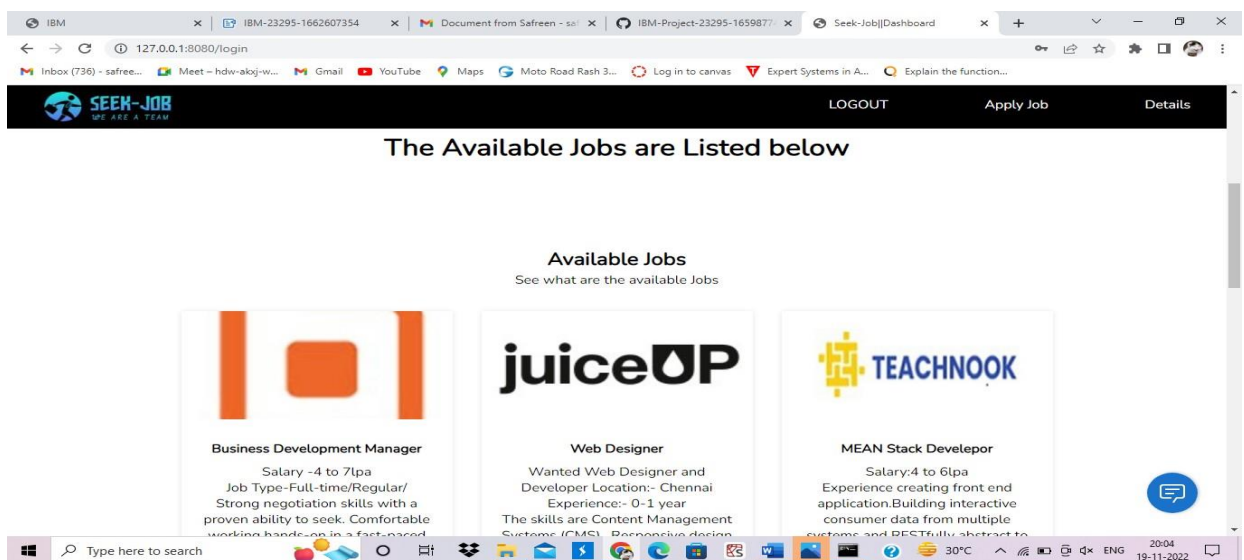
Figure 7.3



Dashboard Page

On successful registration and login, user can navigate to the candidate dashboard page where user can post a job or view the list of jobs posted

Figure 7.4



Apply Job Page

On clicking the apply job user can navigate to the page where user can apply for the job

Figure 7.5

APPLY NOW

Enter Username

Enter your email

Enter your qualification

Enter your skills

Select the role: Manager

Select the Company: Updev

Submit

click here to go to dashboard Dashboard!

Complete miles of journey with one step
Let's get started

Display page

When user navigates to display page , user finds their own information

Figure 7.6

SEEK-JOB
WE ARE A TEAM

LOGOUT Apply Job Details

Your informations

With our Job portal Module, online job application process is possible.
Job portals are managed where vacancies are filled in.
Jpb seekers can log in and apply for jobs online by simply choosing the job
All of the details that are required may be filled in by the user.

Information

User name	email	qualification	Skills	Role	Company
Safreen	safreen@gmail.com	BE	Good Communication	Web designer	7 seas
Safreen	safreen@gmail.com	BE	Good Communication	Web designer	7 seas

Team Details Authorship Contacts Account Community

7.2 Feature 2

Chatbot development

We developed chatbot for our web application , where user can interact with the application and solve any queries. We included the source code in home page and dashboard page

Source Code

```
<script>
```

```
window.watsonAssistantChatOptions = {
```

```
integrationID: "7cb16381-61ae-455f-8312-8aedef4ae1f9b", // The ID of this integration.
```

```
region: "au-syd", // The region your integration is hosted in.
```

```
serviceInstanceID: "7ee7effa-bbb2-471b-a5c1-78adad21acf9", // 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/" +
```

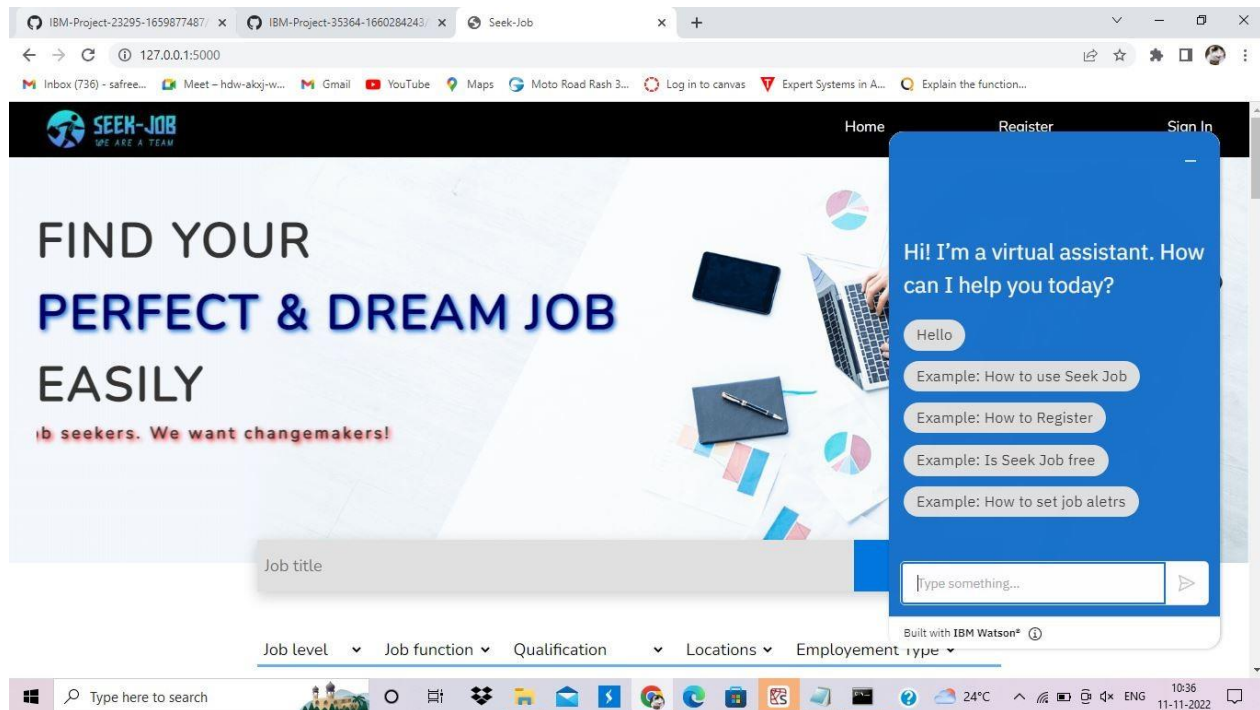
```
(window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";
```

```
document.head.appendChild(t);
```

```
});
```

```
</script>
```

Figure 7.7



7.3 Feature 3

Database Connection

We used IBM Db2 as database and two tables are used , user and Job for storing information

Source code

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

```
import ibm_db
```

```
import re
```

```
app = Flask(__name__)
```

```
app.secret_key = 'a'
```

```
conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=ba99a9e6-d59e-4883-8fc0-  
d6a8c9f7a08f.c1ogj3sd0tgutu0lqde00.databases.appdomain.cloud;
```

```
PORT=31321;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=mpj24332;PWD=Au1yolZxmaIQMdcP",",")
```

```
@app.route('/')
```

```
def homer():
```

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

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

```
def login():
```

```
    msg=" "
```

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

```
        username=request.form['username']
```

```
        password=request.form['password']
```

```
        sql="SELECT * FROM users WHERE username=? AND password=?"
```

```
        stmt=ibm_db.prepare(conn,sql)
```

```
        ibm_db.bind_param(stmt,1,username)
```

```
        ibm_db.execute(stmt)
```

```
        account=ibm_db.fetch_assoc(stmt)
```

```
        print(account)
```

```
        if account:
```

```
            session['Loggedin']=True
```

```
            session['id']=account['USERNAME']
```

```
            userid=account["USERNAME"]
```

```
            session['username']=account["USERNAME"]
```

```
            msg='Logged in successfully!'
```

```
            return render_template("dashboard.html",msg=msg)
```

```
        else:
```

```

msg="Incorrect username/password"

return render_template('login.html',msg=msg)

return render_template('login.html')

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

def register():

    msg=" "

    if request.method=="POST":

        username=request.form['username']

        password=request.form['password']

        email=request.form['email']

        sql="SELECT * FROM users WHERE username=?"

        stmt=ibm_db.prepare(conn,sql)

        ibm_db.bind_param(stmt,1,username)

        ibm_db.execute(stmt)

        account=ibm_db.fetch_assoc(stmt)

        print(account)

        if account:

            msg="Account already exists!"

            elif not re.match(r'^[@]+@[^@]+\.[^@]+' ,email):

                msg="Invalid email address"

            elif not re.match(r'[A-Za-z0-9]+' ,username):

                msg="name must contain only characters and numbers"

            else:

                insert_sql="INSERT INTO user VALUES(?,?,?)"

                prep_stmt=ibm_db.prepare(conn,insert_sql)

```

```
ibm_db.bind_param(prepare_stmt,1,email)
ibm_db.bind_param(prepare_stmt,2,username)
ibm_db.bind_param(prepare_stmt,3,password)
ibm_db.execute(prepare_stmt)

msg='You have successfully registered!'

elif request.method=='POST':
    msg='Please fill out of the form'
    return render_template('register.html')

if __name__ == '__main__':
    app.run(host='0.0.0.0',port=8080)
```

8. TESTING

8.1 Test Cases

A Test cases is a set of rules or conditions to check if system or any of the features works according to its requirement.

Test case ID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
TC_001	UI	Home Page	Verify user is able to see the Login and Signup button when user is in the homepage and able to navigate to login and Register page when user click it	UI Should work properly	Working as expected	Pass
TC_002	UI	Login page	Verify the UI elements in Login page	Application should show below UI elements: a.Username text box b.password text box c.Login button with Red colour d.New user? Create Register page link	Working as expected	Pass
TC_003	Required Fields	Login page	Verify the required field by not entering the data	Application should display the required field message if Username or password are not entered .	Working as expected	Pass
TC_004	Functional	Login page	Verify user is able to log into application with Valid credentials	User should navigate to user account Dashboard page	Working as expected	Pass
TC_005	Functional	Login page	Verify user is not able to log into application with Invalid credentials	Application should show 'Incorrect Username or password ' validation message.	Working as expected	Pass
TC_006	Functional	Login page	Verify user is able to log into application with Invalid credentials	Application should show 'Incorrect Username or password ' validation message.	Working as expected	Pass
TC_007	Functional	Login page	Verify user is able to log into application with Invalid credentials	Application should show 'Incorrect Username or password ' validation message.	Working as expected	Pass
TC_008	UI	Register Page	Verify the UI elements in Register page	Application should show below UI elements: a.Username text box b.password text box c.Email text box c.Create account button with blue colour d.Already a user? Create Login page link	Working as expected	Pass
TC_009	Required Fields	Register page	Verify the required field by not entering the data	Application should display the required field message if Username or password or email are not entered .	Working as expected	Pass
TC_010	Functional	Register Page	Verify user is able to create the account by providing the username, password, email.	User should register successfully and can login using the details	Working as expected	Pass

TC_011	Functional	Register Page	Verify user is able to create the account providing the invalid email address format	Application should display the pop up message showing that the entered email address is invalid	Working as expected	Pass
TC_012	Functional	Register Page	Verify user is able to create an account providing the existing email address	Application should display the pop up message showing that the entered email address is existing email account	Working as expected	Pass
TC_013	Database	Login page	Verify user is able to connect to the database and login to the dashboard page successfully	Application should display the message logged in successful and navigate to the dashboard page	Working as expected	Pass
TC_014	Database	Login Page	Verify user is able to connect to the database and cannot login with false information	Application should display the message incorrect username or password	Working as expected	Pass
TC_015	Database	Register Page	Verify user is connected to the database and can register successfully by storing data into database	Application should display the message Registered successfully	Working as expected	Pass
TC_016	Database	Register Page	Verify user is connected to the database and cannot register with incorrect format	Application should display the message the Unable to Register	Working as expected	Pass
TC_017	Functional	Chatbot	Verify user is able to interact with the application chatbot	Application should display the chatbot and user can easily interact with the chatbot	Working as expected	Pass
TC_018	UI	Dashboard Page	Verify user is able to see the Apply job link in the dashboard page and able to navigate to apply job page when click it	UI Should work properly	Working as expected	Pass
TC_019	Functional	Apply Job	Verify user is able to enter the details in the apply job page	User can able to register by filling all the required fields and the message You have applied for the job should be displayed	Working as expected	Pass

8.2 User Acceptance Testing

User Acceptance Testing (UAT) is a type of testing performed by the end user or client to verify / accept the software system before moving the software system to the production environment.

UAT is done in the final phase of testing after functional, integration, and the system testing.

User acceptance testing is to validate end to end business flow . It does not focus on cosmetic errors , spelling mistakes or system testing. UAT is carried out in a seperate testing environment with production like data setup.It is kind of black box testing where two or more end users will be involved. UAT is performed by-

Client

End Users

9. RESULT

9.1 Performance Metrics

Performance testing is performed to determine how well the system can perform in terms of responsiveness under all kinds of load. The web application is tested to see if it can sustain huge amount of requests providing higher throughput under different loads. I have simulated multiple hits on various pages of the application to evaluate the overall performance. We have used Apache JMeter to test functional and performance both on static and dynamic resources. It can be used to simulate a heavy load on a server, network or object to test its strength or to analyze overall performance under different load types. JMeter is configured to an increased user count while having the same Ramp-Up time and Loop Count. The analysis is done with initial 10 records for jobs posted. Extending the testing to increased records of 30 for jobs posted, we have observed a decrease in throughput under the same Test Plans. In both testing analysis, there is an increase in average throughput time with an increase in user's count. The first test case is executed for 500 samples i.e., simulating 500 requests from 50 users with a loop count of 10. Similarly, the second and third test cases are executed with 100 and 200 users simulating 1000 and 2000 samples respectively. From the above results, we can see that with the increase in the number of users the throughput increases thus ensuring good performance of the application.

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

- Cost efficient
- Time efficient
- Portable
- Easy Access
- Can find suitable job
- It helps user to update their skills

DISADVANTAGES

- While such platforms decrease the recruitment time and advertisement cost
- They suffer from an inappropriateness of traditional information retrieval techniques like the Boolean search methods.
- Consequently, a vast amount of candidates missed the opportunity of recruiting

11. CONCLUSION

Job Search Portals stands as a revolutionizing element in the sphere of recruitment. They act as a communication bridge between applicants and recruiters facilitating their requirements. This application helps organizations to have a greater exposure to the candidate pool and also job seekers facilitating wide search of jobs matching their interests. The android application provides flexibility to the jobseekers to view the openings and applied jobs without the need to carry a laptop. This application provides an enhanced user experience for both employer and jobseeker. It provides user friendly interface which facilitates in reaching wide range of audience. The application has achieved all the requirements that were initially set in the requirements gathering phase.

12. FUTURE WORK

This project fulfills the primary requirements of the job seekers and employers. It can be extended in several ways – We can provide recommendations and email updates for new job postings based on the job seeker's search history. Since, the job seekers might be interested in building a strong Resume, we can provide tips and information for the same. We can also provide templates for building the Resumes which might interest most applicants. The mobile application is developed fulfilling the functionalities of job seeker, it can be extended to support functionalities of Employer as well.

13. APPENDIX

SOURCE CODE

```
<html lang="en">

<head>

<link

href="https://fonts.googleapis.com/css2?family=Noto+Serif&family=Nunito&display=swap"

rel="stylesheet">

<title>Seek-Job||Dashboard</title>

</head>

<body>

<!--navbar-->

<nav class="navbar">

<h2 class="navbar-logo"><a href="#"></a></h2>

<div class="navbar-menu">

<a type="button" id="donate" href="/logout">LOGOUT</a>

<a type="button" id="sponsor" href="apply">Apply Job</a>

<a type="button" id="sponsor" href="display">Details</a>

</div>

<header>

<h1 class="header-title">

<br> WELCOME TO <br> <span>DASHBOARD</span>

<br><center><h1>The Available Jobs are Listed below</h1></center>
```

```
<section class="blog" id="blog" >

<h1 class="section-title">Available Jobs</h1>

<p>See what are the available Jobs</p>

<div class="blog-wrapper">

<div class="blog-card">



<div class="blog-detail">

<h4>Business Development Manager</h4>

<p>Salary -4 to 7lpa<br>Job Type-Full-time/Regular/<br>Strong negotiation skills with a
proven ability to seek.Comfortable working hands-on in a fast-paced start-up environment</p>

<hr class="divider">

<a href="apply" class="btn btn-primary">Apply Now</a>

</hr></div></div>

</section>

<p>Team ID:PNT2022TMID23033</p>

<div class="footer-wrapper">

<h4>Authorship</h4>

<a href="#">Amrudha N G R</a>

<a href="#">Niskriyaa B</a>

<a href="#">Padmasree B M</a>

<a href="#">Safreen B </a></div>
```

```
<div class="footer-wrapper">

<h4>Contacts</h4>

<a href="#">amrudha@gmail.com</a>

<a href="#">niskriyaa@gmail.com</a>

<a href="#">padmasree@gmail.com</a>

<a href="#">safreen@gmail.com</a>

</div>
```

```
<div class="footer-wrapper">

<h4>Account</h4>

<a href="Register.html">Signup/Register</a>

<a href="login.html">Signin/Login</a>

</div>
```

```
<div class="footer-wrapper">

<h4>Community</h4>

<a href="#">Community</a>

<a href="#">Invite a friend</a>

</div>
```

```
</footer>
```

```
<script>

window.watsonAssistantChatOptions = {

    integrationID: "7cb16381-61ae-455f-8312-8aedef4ae1f9b", // The ID of this integration.

    region: "au-syd", // The region your integration is hosted in.
```

serviceInstanceID: "7ee7effa-bbb2-471b-a5c1-78adad21acf9", // 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>
```

app.py

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

import ibm_db

import re

app = Flask(__name_)

app.secret_key = 'a'

conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=ba99a9e6-d59e-4883-8fc0-

d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;

PORT=31321;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=mpj243

32;PWD=Au1yolZxmaIQMdcP",",")
```



```

@app.route('/')
def homer():
    return render_template('index.html')

@app.route('/login',methods =['GET', 'POST'])
def login():
    msg=" "
    if request.method=='post':
        username=request.form['username']
        password=request.form['password']
        sql="SELECT * FROM users WHERE username=? AND password=?"
        stmt=ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account=ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            session['Loggedin']=True
            session['id']=account['USERNAME']
            userid=account["USERNAME"]
            session['username']=account["USERNAME"]
            msg='Logged in successfully!'
            return render_template("dashboard.html",msg=msg)
        else:
            msg="Incorrect username/password"
            return render_template('login.html',msg=msg)

```

```

return render_template('login.html')

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

def register():

msg=" "

if request.method=="POST":

username=request.form['username']

password=request.form['password']

email=request.form['email']

sql="SELECT * FROM users WHERE username=?"

stmt=ibm_db.prepare(conn,sql)

ibm_db.bind_param(stmt,1,username)

ibm_db.execute(stmt)

account=ibm_db.fetch_assoc(stmt)

print(account)

if account:

msg="Account already exists!"

    elif not re.match(r'^@]+@^[^@]+\.[^@]+' ,email):

        msg="Invalid email address"

    elif not re.match(r'[A-Za-z0-9]+' ,username):

        msg="name must contain only characters and numbers"

    else:

        insert_sql="INSERT INTO user VALUES(?,?,?)"

        prep_stmt=ibm_db.prepare(conn,insert_sql)

ibm_db.bind_param(prepare_stmt,1,email)

ibm_db.bind_param(prepare_stmt,2,username)

```

```
ibm_db.bind_param(prepare_stmt,3,password)
ibm_db.execute(prepare_stmt)
msg='You have successfully registered!'
elif request.method=='POST':
msg='Please fill out of the form'
return render_template('register.html')
if __name__ == '__main__':
app.run(host='0.0.0.0',port=8080)
```

Github link and Demo video link

GithubLink

[https://github.com/IBM/EPBL/IBM](https://github.com/IBM/EPBL/IBM-Project-9066-1658950906)

[M-Project-9066-1658950906](https://github.com/IBM/EPBL/IBM-Project-9066-1658950906)

Demo Video Link

<https://drive.google.com/drive/folders/1-ELoDJ7sQPpLwpN98y3HzrYrhItdmFFc>