SKILL AND JOB RECOMMENDER APPLICATION

A PROJECT REPORT

Submitted by

Vishnu Priyan M
Thamarai Kannan T
Vinodhyan R
Vijay Kumar A

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

MAHENDRA ENGINEERING COLLEGE

NAMAKKAL

ANNA UNIVERSITY:CHENNAI-600 025
NOV 2022

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled " ***Skill And Job Recommender Application*** "

is the bonafide work of " **VISHNU PRIYAN M (6113192071116), THAMARAI KANNAN T(6113192071104), VINODHAYAN R (6113192071112), VIJAY KUMAR A (6113192071114)" who carried out the project work under mysupervision.**

SIGNATURE	**SIGNATURE**
Dr. S.RADHA.,M.E.,Ph.D.	**Prof.Dr.S.RAJU.,M.sc.,M.C.A.,M.Phil.,M.E.,Ph.D.**
MENTOR	**HEAD OF THE DEPARTMENT**
BACHELOR OF THECHNOLOGY	**BACHELOR OF TECHNOLOGY**
MAHENDRA ENGINEERING COLLEGE	**MAHENDRA ENGINEERING COLLEGE**
NAMAKKAL	**NAMAKKAL**

CHAPTER

TITLENO

ABSTRACT

In the last years, job recommender systems have become popular since they successfully reduce information overload by generating personal-ized job suggestions.Although in the literature exists a variety of techniques and strategies used as part of job recommender systems, most of them fail to recommending job vacancies that fit properly to the job seekers profiles. Thus, the contributions of this work are threefold, we: i) made publicly available a new dataset formed by a set of job seekers profiles anda set of job vacancies collected from different job search engine sites; ii) put forward the proposal of a framework for job recommendation based on professional skills of jobseekers; and iii) carried out an evaluation to quantify empirically the recommendation abilities of two state-of-the-art methods, considering different configurations, within the proposed framework. We thus present a general panorama of job recommendation taskaiming to facilitate research and real-world application design regarding this importantissue.

CHAPTER 1

INTRODUCTION 1.1

1.2 PROJECT OVERVIEW:

With population growth, there are tons of flats and apartments which have been built in the rapid urbanization areas like in Nairobi, Kenya. This is due to rural to urban migrationin a quest to make ends meet for most inhabitants. There are several issues faced by the inhabitants of the flats. One of them is the issue of the domestic solid waste disposal, which cause pollutions. Unlike landed houses, the flats' waste disposal bins are shared among residents which live in the same building, and thus, the bins tend to be filled very quickly. Thus, an unsystematic and inefficient disposal waste managementmay cause the bins to be always full with of garbage, and further littering from the residents will cause the garbage piles to be scattered outside the bins.

Besides, there are also problems regarding the attitudes of each inhabitant of the flats. There are cases where some irresponsible residents, who normally live at the higher levels of the building, littered or simply threw their domestic waste directly from the floor which they live into the bins.

Implementation of environmental conservation and management system is of no doubt the solution to the major problems that are currently faced when it comes to proper disposal of waste and management.

Indiscriminate disposal of solid waste is a major issue in urban centers of most developing countries and it poses a serious threat to healthy living of the citizens. Access to reliable data on the state of solid waste at different locations within the city will help both the local authorities and the citizens to effectively manage the menace. Inthis paper, an intelligent solid waste monitoring system is developed using Internet of Things (IoT) and cloud computing technologies. The fill level of solid waste in each of the containers, which are strategically situated across the communities, is detected using ultrasonic sensors.

A Wireless Fidelity (Wi-Fi) communication link is used to transmit the sensor data to an IoT cloud platform known as ThingSpeak. Depending on the fill level, the system sends appropriate notification message (in form of tweet) to alert relevant authorities and concerned citizen(s) for necessary action. Also, the fill level is monitored on ThingSpeak in real-time. The system performance shows that the proposed solution may be found useful for efficient waste management in smart and connected communities.

1.2 PURPOSE

Enormous amounts of jobs are posted on the job websites on daily basis and large numbers of new resumes are also added to job websites daily. In such scenario it‟s a very tough job to suggest matching jobs to the job applicants. A recommendation system can solve this problem to the great extent. A recommendation system has already been proved to be very effective in the area of Online shopping websites and Movie recommendation. Given a user, the goal of an employment recommendation system is to predict those job positions that are likely to be relevant to the user. An Employment recommendation system would suggest matching jobs to the users using matching, collaborative filtering and content based recommendation based on ranking.

CHAPTER 2

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

A lot of research has been carried out in the field of job recommender systems. A large variety of job recommendation systems already exist that try to provide one or the other aspect of the information by applying different methods.

mThe key problem is that most of job hunting websites just provides recruitment information to website viewers. Students have to retrieve information among those displayed by websites to find jobs they want to apply. The whole procedure is lengthy and inefficient. In addition, many ecommerce websites, uses collaborative filteringalgorithm without considering user's resume and item's properties .

An online job recommendation system that classifies users into groups by using historical behaviors of users and individual information and then uses the appropriate recommendation approach for each group of users. This approach is suitable for the cases in which different users may have different attributes and a single recommendation approach may not be appropriate for all users.

This system considers input as a CV to create the user profile. These user profiles are then compared with the available jobs. Moreover, the RS has been enhanced with implicit relevance feedback, which allows the system to find out user preferences. Mamadou et al. presented an online social network-based recommender system that extracts users' interests for jobs and then make recommendations according to user's interest.

2.2 REFERENCES

1. Toon De Pessemier, Kris Vanhecke, and Luc Martens. 2016. A scalable, high-performance Algorithm for hybrid job recommendations. In Proceedings of the Recommender Systems Challenge(RecSys Challenge '16). ACM, New York, NY, USA, Article 5, 4 pages. DOI:https://doi.org/10.1145/2987538.2987539

2. Fedor Borisyuk, Liang Zhang, and Krishnaram Kenthapadi. 2017. LiJAR: A System for Job Application Redistribution towards Efficient Career Marketplace. In Proceedings of KDD "17, Halifax, NS, Canada, August 13-17, 2017, 10 pages. https://doi.org/10.1145/3097983.3098028

3. Ravi Kumar, Silvio Lattanzi, Sergei Vassilvitskii, and Andrea Vattani. 2011. Hiring a Secretary from a Poset. In EC. https://doi.org/10.1145/1993574.19935822

4. Esteban Arcaute and Sergei Vassilvitskii. 2009. Social Networks and Stable Matchings in the Job Market. In WINE. https://doi.org/10.1007/978-3-642-10841- 9_21

5. Hafizovic, NedzadLinnaeus University, Faculty of Technology, Department of computer science and media technology (CM), Department of Computer Science. Linneuniversitetet.

6. Job Recommendation Systems for Enhancing E-recruitment Process Shaha T.Al-Otaibi, M.Ykhetef, Published 2012 Business, Computer Science

7. Job Recommendation System Using Maching Learning And Natural Language Prcessing-JEEVANKRISHNA Dublin Business SchoolDissertation submitted in partial fulfilment of the requirements for the degree of MSc in Data Analytics,May 2020.

8. Skill Scanner: Connecting and Supporting Employers, Job Seekers and Educational Institutions with an AI-based Recommendation SystemJune 2022 Conference: The Learning Ideas Conference 2022 (15th annual conference) At: New York, New York, USA

9. INTERNATIONAL JOURNAL OF ADVANCE SCIENTIFIC RESEARCHAND ENGINEERING TRENDS WWW.IJASRET.COM 29-Job Recommendation System Using Profile Matching And Web-Crawling,Deepali V Musale 1, Mamta K Nagpure2, Kaumudini S Patil.

2.3 PROBLEM STATEMENT DEFINITION

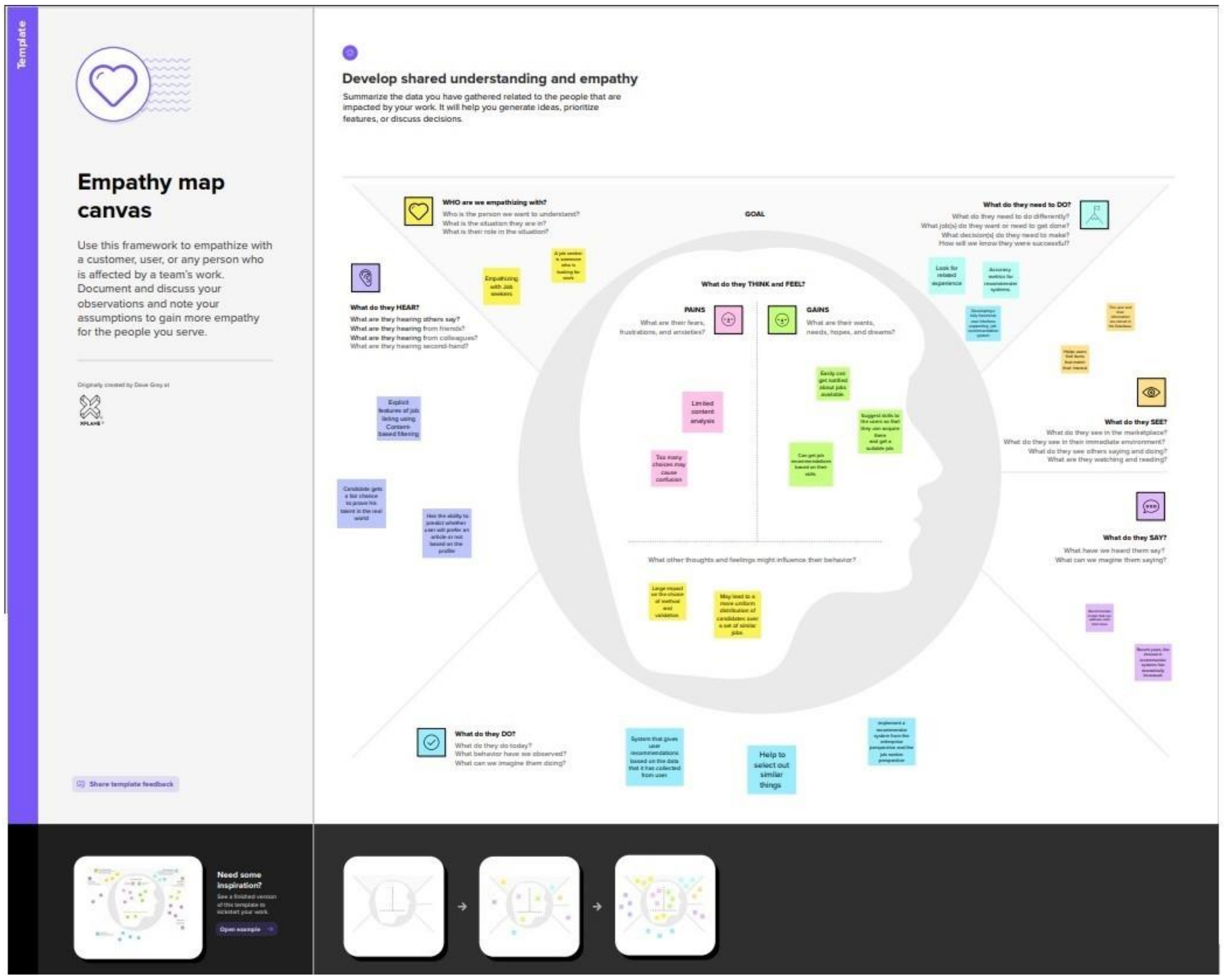
The key problem is that most of job-hunting websites just display recruitment information to website viewers. Websites just display recruitment information to website viewers. Students have to retrieve among all the information to find jobs they want to apply. The whole procedure is tedious and inefficient. By creating an easy job recommendation system where everyone will have a fair and square chance. This savesa lot of potential time and money both on the industrial as well as the job seeker's side.

Moreover, as the candidate gets a fair chance to prove his talent in the real world it is a lot more efficient system. The basic agenda of every algorithm used in today's world beit a traditional algorithm or a hybrid algorithm is to provide a suitable job that the user actually seeks and wishes for.

CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



Template

Empathy map canvas

Use this framework to empathize with a customer, user, or any person who is affected by a team's work. Document and discuss your observations and note your assumptions to gain more empathy for the people you serve.

Originally created by Dave Gray at XPLANE

Share template feedback

Need some inspiration?

See a finished version of this template to kickstart your work.

Open example

Develop shared understanding and empathy

Summarize the data you have gathered related to the people that are impacted by your work. It will help you generate ideas, prioritize features, or discuss decisions.

WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?

Empathizing with Job seekers

A job seeker is someone who is looking for work

GOAL

What do they need to DO?
What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?

Look for related experience

Accuracy metrics for recommender systems

What do they HEAR?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?

Explicit features of job listing using Content-based filtering

Candidate gets a fair chance to prove his talent in the real world

Has the ability to predict whether user will prefer an article or not based on the profile

What do they THINK and FEEL?

PAINS
What are their fears, frustrations, and anxieties?

Limited content analysis

Too many choices may cause confusion

GAINS
What are their wants, needs, hopes, and dreams?

Easily can get notified about jobs available

Suggest skills to the users so that they can acquire them and get a suitable job

Can get job recommendations based on their skills

What other thoughts and feelings might influence their behavior?

Large impact on the choice of method and validation

May lead to a more uniform distribution of candidates over a set of similar jobs

What do they SEE?
What do they see in the marketplace?
What do they see in their immediate environment?
What do they see others saying and doing?
What are they watching and reading?

What do they SAY?
What have we heard them say?
What can we imagine them saying?

What do they DO?
What do they do today?
What behavior have we observed?
What can we imagine them doing?

System that gives user recommendations based on the data that it has collected from users

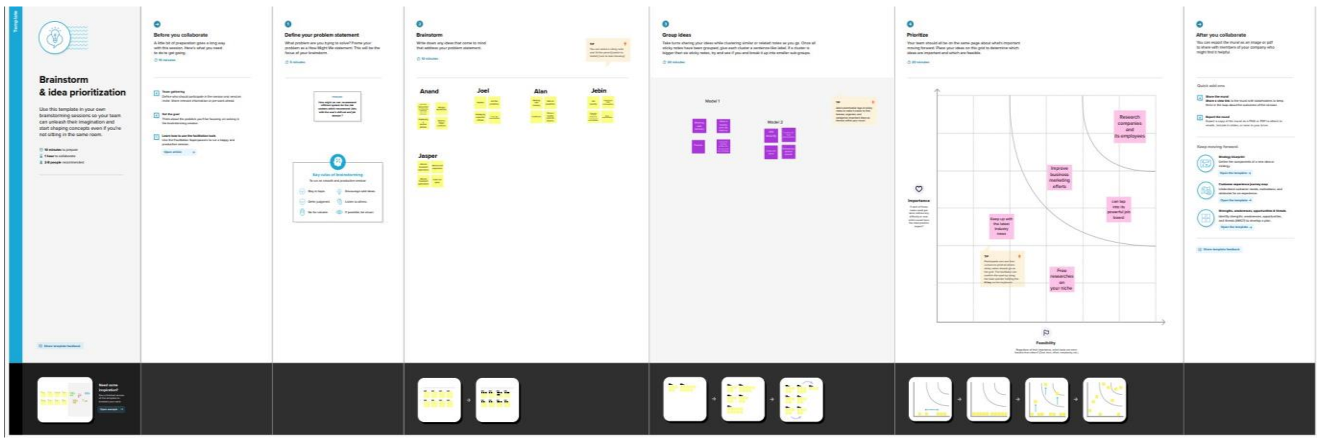
Help to select out similar things

Implement a recommender system from the enterprise perspective and the job seeker perspective

3.2 IDEADTION AND BRAINSTORMING

Brainstorming is a method design teams use to generate ideas to solve clearly defined design problems. Brainstorming is a method of generating ideas and sharing knowledge to solve a particular commercial or technical problem, in which participants are encouraged to think without interruption. Brainstorming is a group activity where each participant shares theirideas as soon as they come to mind. At the conclusion of the session, ideas are categorised and ranked for follow-on action.

When planning a brainstorming session it is important to define clearly the topic to be addressed. A topic which is too specific can constrict thinking, while an ill-defined topic will not generate enough directly applicable ideas. The composition of the brainstorming group is important too. It should include people linked directly with the subject as well as those who can contribute novel and unexpected ideas. It can comprise staff from inside or outside the organisation.

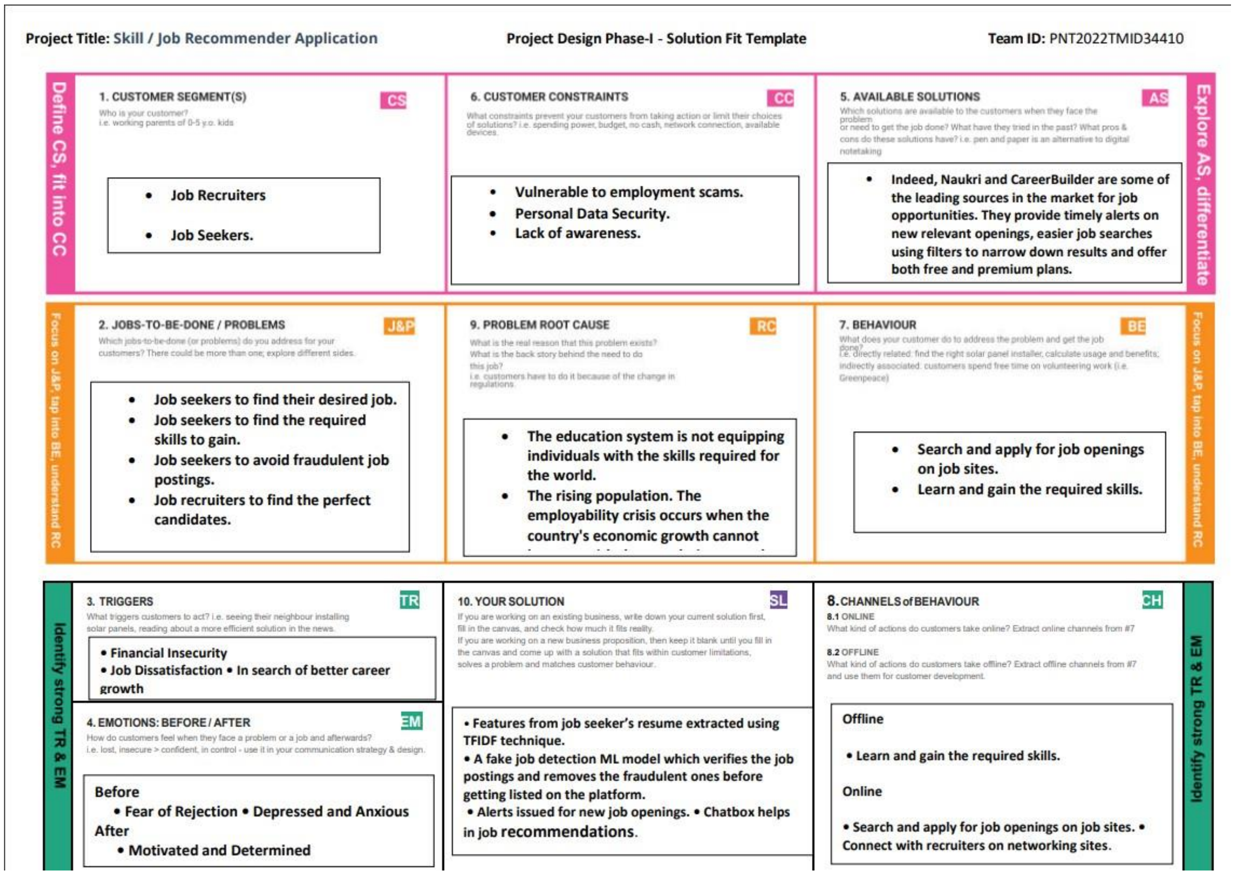


3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To develop an end-to-end web application capable of displaying the current job openings based on the user skillset.So, to eradicate the unemployment crisis, for the job seekers to find a job they desire, match their qualifications and Skills.
2.	Idea / Solution description	● The skills (basic features) are extracted from thejob seeker's resume using the TF-IDF technique. ● The job seeker's profile may get outdated sometimes as they fail to update the resumeregularly. ● The dynamic behaviour of the job seeker is noted by observing the jobs he applied for.
3.	Novelty / Uniqueness	A fake job detection ML model which verifies the job postings and removes the fraudulent ones before getting listed on the platform is integrated with the recommendation engine tobring down the employment scams.
4.	Social Impact / Customer Satisfaction	The customer satisfaction can be measured by customer loyalty and customer reviews after deployment of the project.
5.	Business Model (Revenue Model)	A subscription model can be provided for both employees and employers with additional costs.

6.	Scalability of the Solution	The cloud is capable of increasing or decreasing IT resources as needed to meet the changing demand

3.4 PROBLEM SOLUTION FIT



Project Title: Skill / Job Recommender Application

Project Design Phase-I - Solution Fit Template

Team ID: PNT2022TMID34410

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

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

- Job Recruiters
- Job Seekers.

6. CUSTOMER CONSTRAINTS CC

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

- Vulnerable to employment scams.
- Personal Data Security.
- Lack of awareness.

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

- Indeed, Naukri and CareerBuilder are some of the leading sources in the market for job opportunities. They provide timely alerts on new relevant openings, easier job searches using filters to narrow down results and offer both free and premium plans.

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.

- Job seekers to find their desired job.
- Job seekers to find the required skills to gain.
- Job seekers to avoid fraudulent job postings.
- Job recruiters to find the perfect candidates.

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.

- The education system is not equipping individuals with the skills required for the world.
- The rising population. The employability crisis occurs when the country's economic growth cannot

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)

- Search and apply for job openings on job sites.
- Learn and gain the required skills.

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.

• Financial Insecurity
• Job Dissatisfaction • In search of better career growth

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.

Before
• Fear of Rejection • Depressed and Anxious
After
• Motivated and Determined

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.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

• Features from job seeker's resume extracted using TFIDF technique.
• A fake job detection ML model which verifies the job postings and removes the fraudulent ones before getting listed on the platform.
• Alerts issued for new job openings. • Chatbox helps in job recommendations.

8. CHANNELS of BEHAVIOUR CH

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

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

Offline

• Learn and gain the required skills.

Online

• Search and apply for job openings on job sites. • Connect with recruiters on networking sites.

Identify strong TR & EM

CHAPTER-4

REQUIREMENT

ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	User Registration	Registration via Form And via Gmail
FR-2	User Confirmation	Confirmation through Email Thatis through OTP
FR-3	Chat Bot	A Chat Bot will be there in website to solve user queries and problemsrelated to applying a job,search for a job and muchmore.
FR-4	User Login	Login through Form Login through Gmail
FR-5	User Search	Explorationof Jobs based on job filters and skill recommendations.
FR-6	User Profile	Updation of the user profile through the login credentials

4.2 NON-FUNCTIONAL REQUIREMENTS

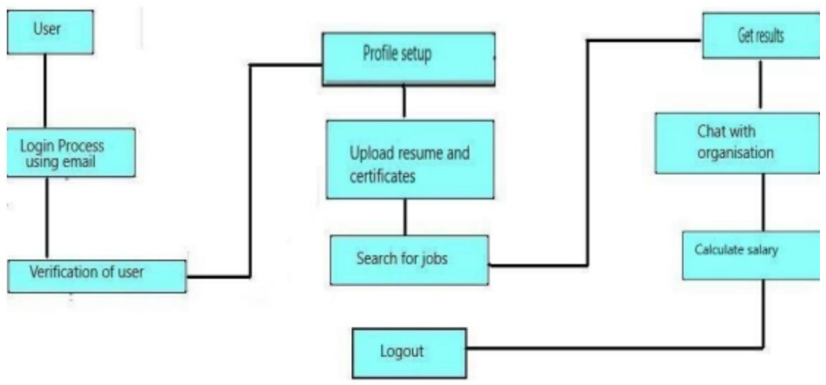
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This application can be used by the job seekers to login and search for the job based on her Skillsset
NFR-2	**Security**	This application is secure with separate login for Job Seekers as well as Job Recruiters.
NFR-3	**Reliability**	This application is open-source and feel free to use, without need to pay anything. The enormous job openings will be provided to all the job seekers without any limitation.
NFR-4	**Performance**	The performance of this application is quicker response and takeslesser time 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 time of the application is quite faster compared to any other application

CHAPTER -5

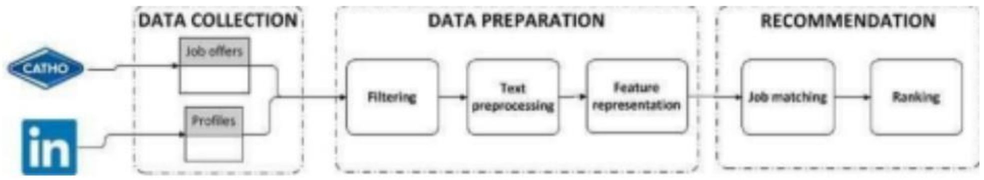
PROJECT

DESIGN

5.1 DATAFLOW DAGRAM



5.2 SOLUTION AND TECHNICAL ARCHITECTURE



5.3 USER STORIES

UserType	Functional Requirement(Epic)	User Story Number	User Story/Task	Acceptance Criteria	Priority	Release
Customer(Mobile user)	Registration	USN-1	Asauser,Ican registerforthe application by enteringmy email,password, and confirming my password.	Icanaccessmy account/dashboard	High	Sprint-1
		USN-2	Asauser,Iwill receive confirmation email onceI have registeredfor the application	Icanrecieve confirmation email &clickconfirm	High	Sprint1-
		USN-3	Asauser,Ican registerforthe application through Facebook	Ican register& accessthe dashboard with Facebook Login	Low	Sprint-1
		USN-4	Asauser,Ican registerforthe 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-6	Asauser,I can access allthe features from dashboard			

Customer(Web user)	Registration	USN-7	Asauser,Ican registerforthe application with username and password andthen confirmingit.	Icanaccessmy account/dashb oard	High	Sprint-2
		USN-8	Asauser,Iwill receive confirmation email after registration	Icanreceive confirmation email &clickconfirm	High	Sprint-2
		USN-9	Asauser, Ican registerthe application through Facebook	Ican register& accessthe dashboa rdwith Facebook Login	Low	Sprint-2
		USN-10	Asauser,Ican registerforthe application through Gmail		Medium	Sprint-2
	Login	USN-11	Asauser,Icanlog into the application by entering email& password		High	Sprint-2
	Dashboard	USN-12	Asauser,I can access allthe features from dashboard			
Customer Care Executive	Support customer with theirqueries	USN-13	Asacustomer care executive,I should helpand support the customer problems.I should solve theirqueries.		High	Sprint-3

	Guide customers	USN-13	Asacustomer care executive,I should guidethe customer begining from registrationtill applyingforjobs.		High	Sprint-3
	Encourage customer	USN-14	Asacustomer care executive,I should encourage customertouse this application.		Low	Sprint-3
Administrator	Login	USN-15	Asa administrator, I canloginwithmy username and password	Icanaccess the application from administrativeside.	High	Sprint-4
	Monitor the application	USN-16	Asaadministrator ,Ishould monitor the application whetheritis working properly withoutany error		High	Sprint-4
	Monitor the chatbot	USN-17	Asa administrator, I shouldmonitor thechatbotdaily,		Low	Sprint-4

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Milestones	Activity	Priority	Team Members
Registration	1.Design the UI for Registration page	Medium	Anand Dev.V
	2.Complete the registration page by placing required fields.	High	Joel .R.K Anand Dev.V
	3.Write the functionality for the buttons using Python Flask	High	Jebin Dyline.J Alan Pramil.J.S
	4.Send verification mail to users to register their account	High	Jasper David.G Jebin Dyline.J

Login	1.Design the UI for Login Page	Medium	Joel.R.K
	2.Complete login page and its functionality.	Medium	Anand Dev.V Jasper David.G
Search	1.Collect dataset	Low	Alan Pramil.J.S
	2.Implement recommendation using Hybrid Filtering	High	Anand Dev.V Jebin Dyline.J Alan Pramil.J.S
	3.Create Profile section for entering user details	Medium	Alan Pramil.J.S Joel .R.K
Review feature	1.Create Review feature for users	Low	Jasper David.G

Deployment	1.Upload the web app to Docker	Medium	Joel.R.,K Anand Dev.V
	2.Deploy the Docker image into a Kubernetes Cluster	Medium	Alan Pramil.J.S, Jebin Dyline.J

6.2 Sprint Delivery Schedule

Sprint	**Total Story Points**	**Durati on**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date (Actual)**
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	27 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	29 Nov 2022

CHAPTER 7

CODING AND SOLUTIONING

7.1 FEATURE 1:

- ◎ Registration page
- ◎ Login page
- ◎ Profile page
- ◎ Job Recommendation page
- ◎ Log out user

7.2 FEATURE 2:

In this application theuser can create an acount and upload their CV and they can also select their interested fields. We can update the user about their progress and information from the company they have applied for. User can also ask for the support of Chatbot which is really user friendly.

Code:
app.py:

```
import hashlib
import re
import sqlite3
import uuid
 from datetime import datetime, timedelta

import jwt
from flask import (Flask, jsonify, make_response, redirect, render_template,   request,
url_for)
request,url_for)

  fromflask_loginimport(login_required,login_user,logout_user)

  app=Flask(  name  )
  bcrypt=Bcrypt(app) salt
  ="5gz"
```

```
app.config["KEY"]="Hello"


def verify(token):
   data = jwt.decode(token, "Hello", algorithms='HS256')
   return data["email"]

@app.route('/')
defhome():
   return render_template('./sign/hrsignin.html')


@app.route("/hr/signin", methods=['GET', 'POST'])
def hrSignIn():
   ifrequest.method=="GET":
      return
   render_template("./sign/hrsignin.html")   else:
      email = request.form["email"]
      password =
      request.form["password"]
      withsqlite3.connect('hr.db')asconnection:
         cursor = connection.cursor()
         cursor.execute(
            "SELECTemailFROMRECRUITERWHEREemail=?",(email,))user
         = cursor.fetchone()
         ifuser==None:
            print("No user")
            return  redirect("/hr/profile")
         else:
            db_password = password+salt
            pw_hash=hashlib.md5(db_password.encode())


            cursor.execute(                                             "SELECTemail,password
               FROMRECRUITERWHEREemail=?",(email,))
            details=cursor.fetchone()
            print(details)
            ifpw_hash.hexdigest()==details[1]:
               token=jwt.encode({"email":email,'exp':  datetime.utcnow(
)+timedelta(minutes=30)}, "Hello", algorithm='HS256')   print(token
```

```
    response = make_response(
        render_template("./feed/feed.html"))
    response.set_cookie('token',
    token) returnresponse

else:
    return "wrong password"
```

Theothercodefeaturresareinthebelowgithublink

https://github.com/IBM-EPBL/IBM-Project-48925-1660814454

CHAPTER 8

TESTING

8.1 TESTING:

i. Loginpage
ii. Registeration page
iii. Profile page
iv. Jobrecommender page

8.2 USER ACCEPTANCE TESTING

1. Purpose of Document

The purpose of this document is to briefly explainthe test coverage and open issues of the [ProductName]projectatthetimeofthereleasetoUserAcceptanceTesting (UAT).

2. Defect Analysis

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
ClientApplication	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
ByDesign	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won'tFix	0	5	2	1	8
Totals	24	14	13	26	77

3. Test Case Analysis

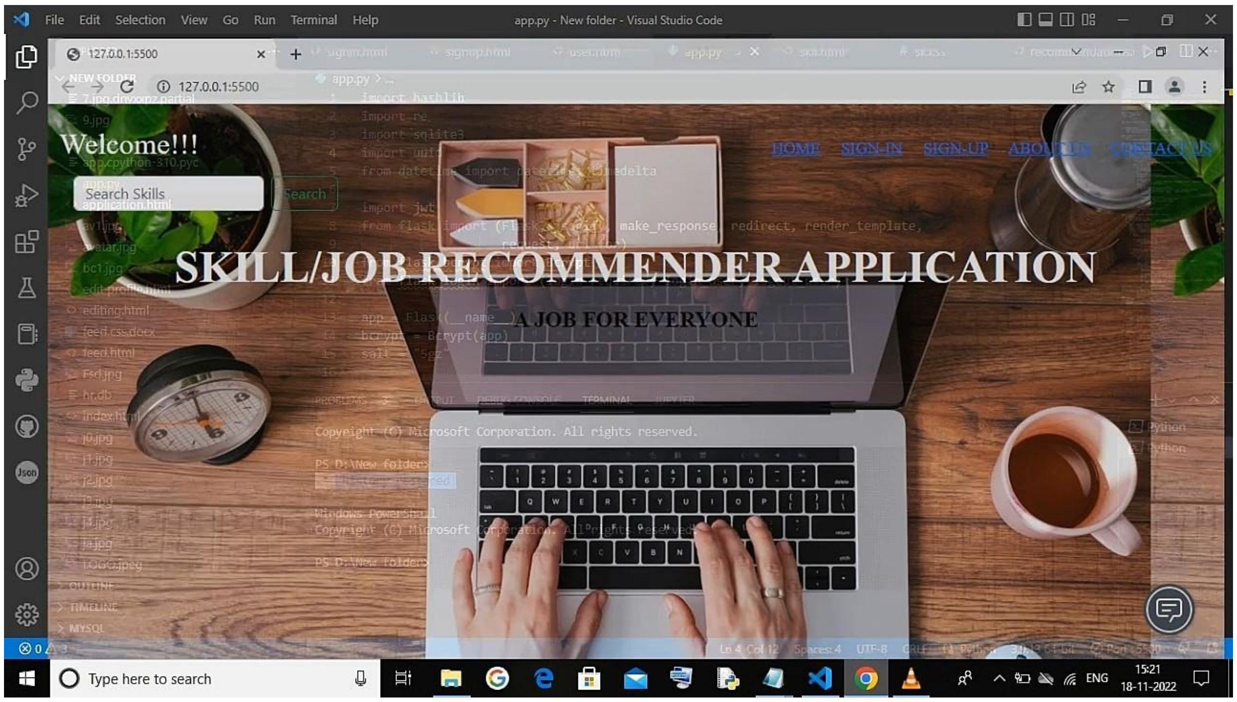
Thisreport showsthenumberoftestcasesthathavepassed,failed,anduntested

ExceptionReporting	9	0	0	9
FinalReportOutput	4	0	0	4
VersionControl	2	0	0	2

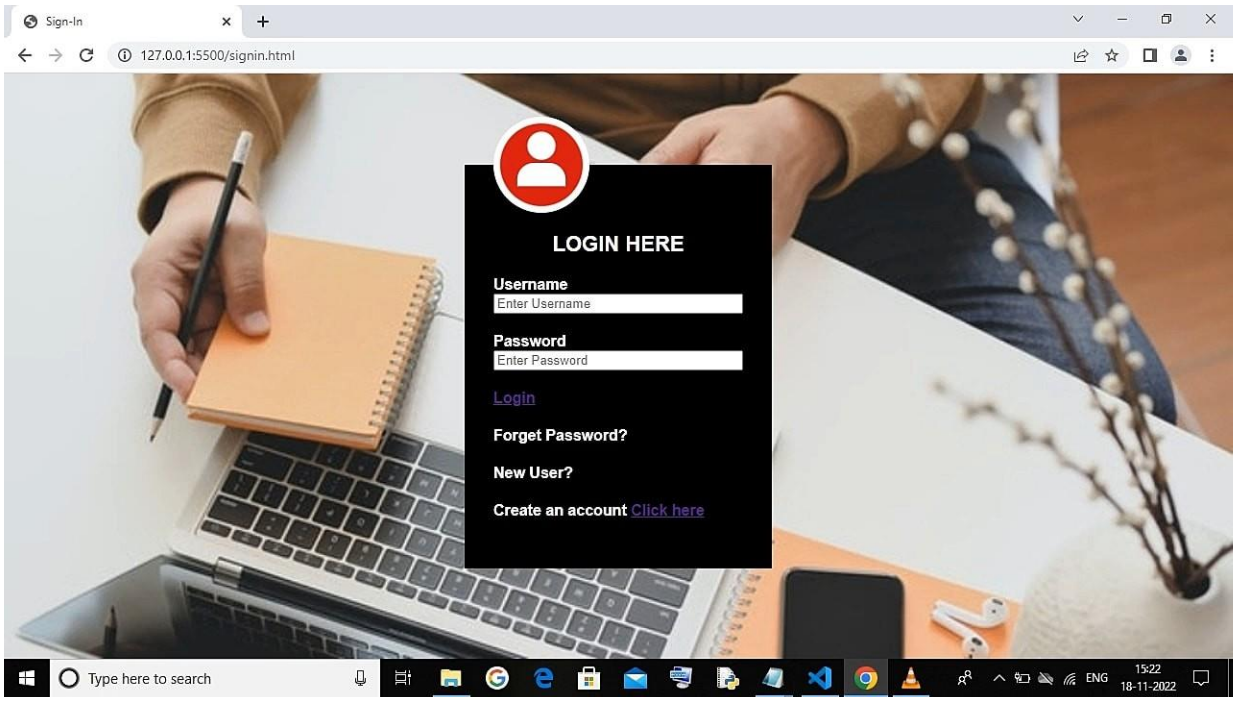
CHAPTER 9

RESULTS

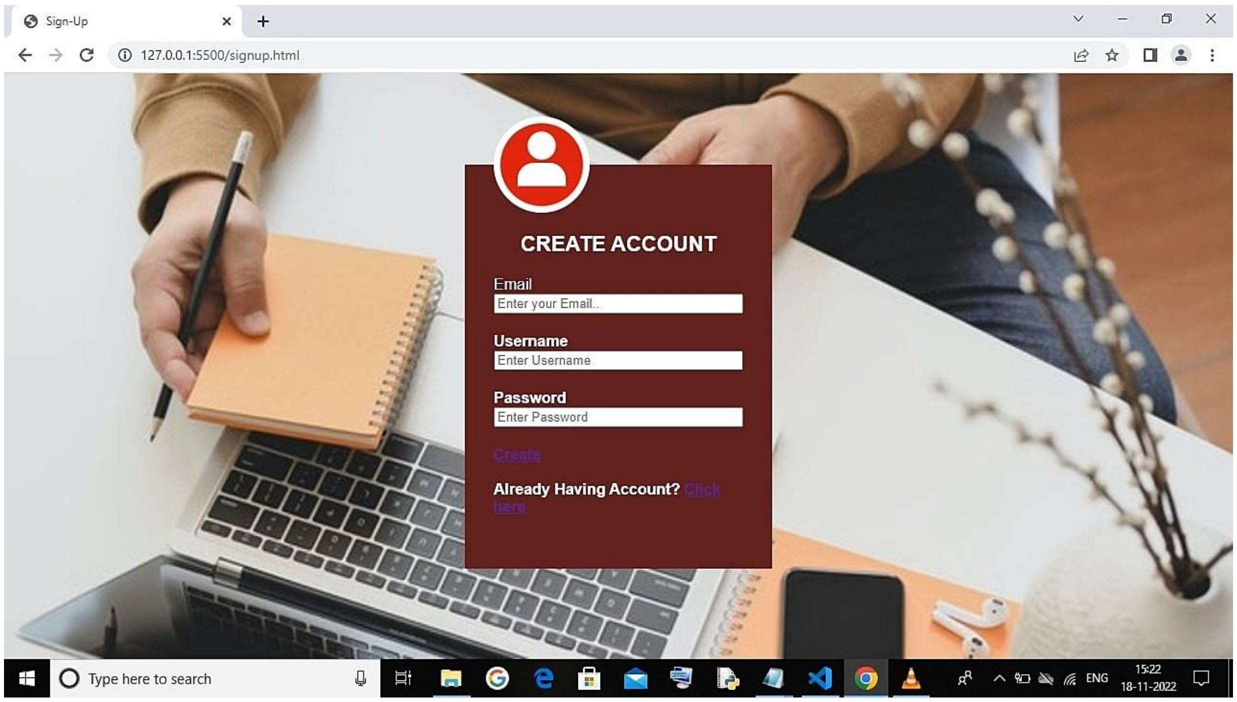
9.1 Home page:



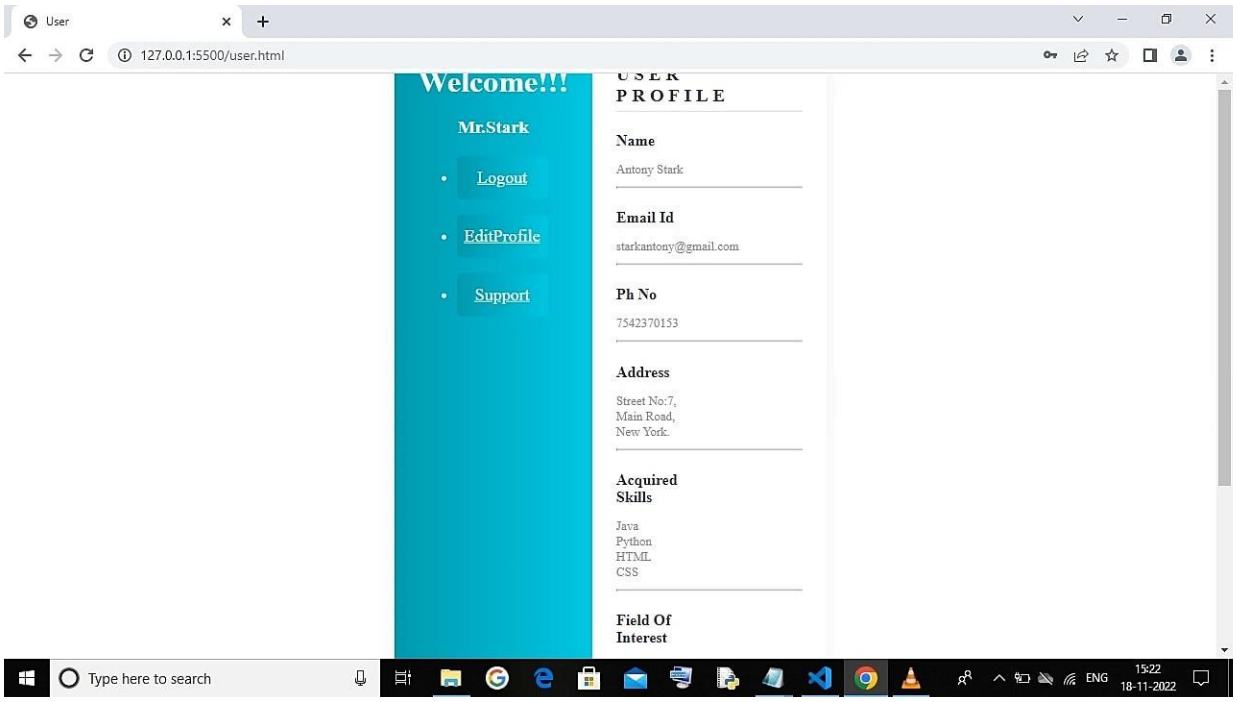
9.2 Login Page:



9.3 Sign up page:



9.4 User profile:


User

127.0.0.1:5500/user.html

Welcome!!!

Mr.Stark

- Logout
- EditProfile
- Support

USER PROFILE

Name

Antony Stark

Email Id

starkantony@gmail.com

Ph No

7542370153

Address

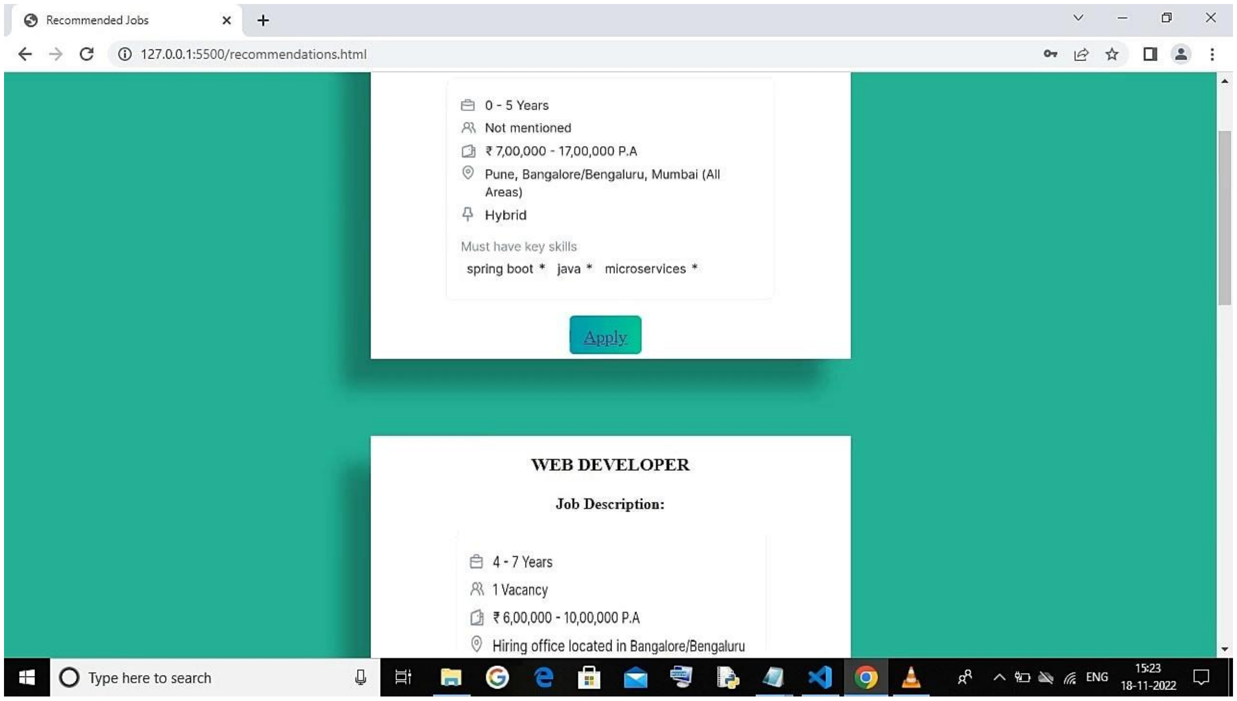
Street No:7,
Main Road,
New York.

Acquired Skills

Java
Python
HTML
CSS

Field Of Interest

9.5 Jobrecommender page:



0 - 5 Years
Not mentioned
₹ 7,00,000 - 17,00,000 P.A
Pune, Bangalore/Bengaluru, Mumbai (All Areas)
Hybrid
Must have key skills
spring boot * java * microservices *
Apply

WEB DEVELOPER
Job Description:
4 - 7 Years
1 Vacancy
₹ 6,00,000 - 10,00,000 P.A
Hiring office located in Bangalore/Bengaluru

CHAPTER 10

ADVANTAGES AND DISADVANTAGES

10.1 ADVANTAGE

- The System would benefit those users who have to use search engines to locate relevant content. They have to scroll through pages of results to find relevant content. Rather than searching for quality web pages, the users of this system would be directly taken to quality web pages matching their personal interests and preferences.

- The system would deliver quality web pages as it is not just dependent on the rating given by other users which could be deceiving at times.

- It provides personalized recommendations of Skill and jobs. User can easily search jobs of their field of interest.

- The user can easily search the skill that they need to improve for the jobs they are interested. There is no paperwork required.

- No manual work is needed.

- It is user-friendly and doesn’t require any prior knowledge to use.

10.2 DISADVANTAGE

- To access the application we need stable internet connection. The correctness of the user profile is question mark. The dataset need to be regulated and updated on the regular basis and it is a tiring process.

- The recommendation system only recommends the jobs or skill and that doesn't mean that we are hired ,hence the users need to undergo next steps to to be hired which frustrate the user.

CHAPTER 11

CONCLUSION

A literature analysis of many journals and proceedings related to the recruiting process and the job recommendation researches has been used. From our literature review and from the challenges that faced the holistic e-recruiting platforms, an increased need for enhancing the quality of candidates/job matching. The recommender system technologies accomplished significant success in a broad range of applications and potentially a powerful searching and recommending techniques.

Consequently, there is a great opportunity for applying these technologies in recruitment environment to improve the matching quality. This survey shows that several approaches for job recommendation have been proposed, and many techniques combined in order to produce the best fit between jobs and candidates. Thus presented state of the art of job recommendation as well as, a comparative study for its approaches that proposed by literatures. Additionally, we reviewed typical recommender system techniques and the recruiting process related issues. Therefore, the field of job recommendations is still unripe and require further improvements.

CHAPTER 12

FUTURE SCOPE

The proposed recommendation system provides, only the recommendations and in future users who needs moral support can be provided by the support group. The validation of the user profile and the job notification will be ensured. The dataset can be improved so that the data will be feed to the system and no need of updating the dataset.

The support group which provides motivation and support the users who are frustrated, panicked, depressed due to unemployment. The volunteer organization can be asked to collaborate to provide workshops , seminar for the key skills that required to be hired. Can collaborate with the companies to conduct a offcampus drive , so that all the userscan attend them.

CHAPTER 13

APPENDIX

Source code:

As we successfully developed and programmed our python code, lets this be the final code of execution.

App.py

```
import hashlib
import re
import sqlite3
import uuid
from datetime import datetime, timedelta
import jwt
from flask import (Flask, jsonify, make_response, redirect, render_template,
                   request, url_for)
from flask_bcrypt import Bcrypt
from flask_login import (login_required, login_user, logout_user)
app = Flask(_name_)
bcrypt = Bcrypt(app)
salt = "5gz"
app.config["KEY"] = "Hello"
def verify(token):
   data = jwt.decode(token, "Hello", algorithms='HS256')
   return data["email"]
@app.route('/')
def home():
   return render_template('./sign/hrsignin.html')
@app.route("/hr/signin", methods=['GET', 'POST'])def
hrSignIn():
    if request.method == "GET":
        return render_template("./sign/hrsignin.html")
    else:
        email = request.form["email"]
        password  =  request.form["password"]
        with  sqlite3.connect('hr.db')  as  connection:
            cursor = connection.cursor()
            cursor.execute(
                "SELECT email FROM RECRUITER WHERE email=?", (email,))user
            = cursor.fetchone()
            if user == None:
                print("No  user")
                return  redirect("/hr/profile")
            else:
                db_password = password+salt
                pw_hash = hashlib.md5(db_password.encode())
                cursor.execute(
                    "SELECT email,password FROM RECRUITER WHERE email=?",
(email,)
)               details = cursor.fetchone()
                print(details)
                if pw_hash.hexdigest() == details[1]:
                    token = jwt.encode({"email": email, 'exp': datetime.utcnow(
                    )+timedelta(minutes=30)}, "Hello", algorithm='HS256') print(token)
                    response = make_response(
                        render_template("./feed/feed.html"))
                    response.set_cookie('token', token)return
                    response
                else:
                    return  "wrong  password"
```

```
@app.route("/hr/signup", methods=["GET", "POST"])
def hrSignUp():
    if request.method == "GET":
        return render_template("./sign/hrsignup.html")
    else:
        name = request.form["name"]
        email = request.form["email"]
        phone = request.form["phone"]
        password  =  request.form["password"]
        confirm = request.form["re-password"] if
        email:
            regex  =  r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b'  def
            check(email):
                if (re.fullmatch(regex, email)):
                    print("valid email")
                else:
                    return "not an valid email"
        if password != confirm:
            return  "Password  mismatch"
        else:
            with  sqlite3.connect('hr.db')  as  connection:
                cursor = connection.cursor()
                cursor.execute(
                    """ SELECT email FROM RECRUITER WHERE email=? """, (email,))
                user = cursor.fetchone()
                print(user)
                if user == None:
                    key  =  uuid.uuid1().hex
                    print(key)

                    db_password = password+salt
                pw_hash = hashlib.md5(db_password.encode())
                cursor.execute("INSERT INTO RECRUITER
(name,email,phone,password,id) VALUES (?,?,?,?,?)", (
                    name, email, phone, pw_hash.hexdigest(), key))
                connection.commit()
                return redirect("/hr/signin")
            else:
                print("exists")
                return redirect("/hr/signin")
@app.route("/hr/logout")
def logout():
    response = make_response(render_template("./sign/hrsignin.html"))
    response.set_cookie('token', '')
    return response
@app.route('/hr/feed')
def hrFeed():
    try:
        token = request.cookies.get('token')
        data = jwt.decode(token, "Hello", algorithms='HS256')
        return render_template("./feed/feed.html")
    except:
        return render_template("./feed/feed.html")
@app.route("/hr/feed/<id>")
def hrOneFeed(id):
    try:
        token = request.cookies.get('token')
        data = jwt.decode(token, "Hello", algorithms='HS256')
        print(id)
        return render_template("./feed/oneFeed.html")
    except:
        return redirect("/hr/signin")
```

```
@app.route("/hr/application")
def hrApplication():
    try:
        token = request.cookies.get('token')
        email = verify(token)
        print(email)
        return render_template("./application/applications.html")
    except:
        return render_template("./application/applications.html")
@app.route("/hr/application/<id>")
def hrOneApplication(id):
    try:
        token = request.cookies.get('token')
        email = verify(token)
        print(email)
        return render_template("./application/oneApplication.html")
    except:
        return redirect("/hr/signin")
@app.route("/hr/profile")
def hrProfile():
    try:
        token = request.cookies.get('token')
        email = verify(token)
        print(email)
        with sqlite3.connect('hr.db') as connection:
            cursor=connection.cursor()
            cursor.execute("""
            SELECT name,
            email,
            about_me,
            designation,
            experience ,
            url ,
            company_name ,
            company_description ,
            location ,
            website ,
            in_url
             FROM RECRUITER WHERE email=?""", (email,))
            data=cursor.fetchone()
            print(data)
            if not data:
               return redirect("/hr/logout")
            else:
               return render_template("./profile/viewProfile.html",data=data)
    except Exception as e:
        print(e)
        return redirect("/hr/signin")
@app.route("/hr/profile/edit")
def hrProfileEdit():
    try:
        token = request.cookies.get('token')
        email = verify(token)
        print(email)
        with sqlite3.connect('hr.db') as connection:
            cursor=connection.cursor()
            cursor.execute("""
            SELECT name,
            email,
            about_me,
            designation,
            experience ,
            url ,
            company_name  ,
            company_description ,
            location ,
            website  ,
            in_url ,
            id
             FROM RECRUITER WHERE email=?""", (email,))
            data=cursor.fetchone()
            print(data[11])
            if not data:
                return redirect("/hr/logout")
            else:
                return render_template("./profile/editProfile.html",data=data)
        return render_template("./profile/editProfile.html")
    except:
        return redirect("/hr/signin")
@app.route("/hr/profile/edit/<id>",methods=("POST","GET"))
def profileEditIID(id):
    if request.method=="POST":
        token = request.cookies.get('token')print("post")
        try:
            print(email)
            email = verify(token)
            name=request.form["name"],
            about_me=request.form["about_me"],
            designation=request.form['designation'],
            experience=request.form['experience'],
            url=request.form['url'],
            company_name=request.form['company_name'],
        company_description=request.form['company_description'],
        location =request.form['location'],
        website=request.form['website'],
        in_url=request.form['in_url'] ,
        if not id:
           return  redirect("/hr/profile")
        with sqlite3.connect('hr.db') as connection:
           cursor=connection.cursor()
           cursor.execute("""SELECT id FROM RECRUITER WHERE
email=?""",(email,))
           if data[11]==id:
              cursor.execute("""
           SELECT name,
           email,
           about_me,
           designation,
           experience ,
           url ,
           company_name ,
           company_description ,
           location ,
           website ,
           in_url ,
           id
           FROM RECRUITER WHERE email=?""", (email,))
           data=cursor.fetchone()
           cursor.execute("""
              UPDATE RECRUITER SET name=?,
              about_me=?,
              designation=?,
              experience=?,
              url=?,
              company_name=? ,
              company_description=?  ,
              location =?,
              website=?,
              in_url=? ,
               FROM RECRUITER WHERE email=?""", (name,
              about_me,
              designation,
              experience,
              url,
              company_name,
              company_description ,
              location,
              website,
              in_url ,email))
           connection.commit()
           return "Success"
       except Exception as e:
        print(e)
        return "failed"
@app.route("/hr/profile/pwd", methods=("GET", "POST")) def
hrProfileEditPWD():
if request.method == "GET":
       try:
        token = request.cookies.get('token')
        email = verify(token)
        print(email)
        return render_template("./profile/passwordReset.html")

       except:
        return redirect("/hr/signin")
else:
       try:
        token = request.cookies.get('token')
        email = verify(token)
        print(email)
        password = request.form["password"]
        newPWD = request.form['newPassword']
        confirmPWD = request.form['confirmPassword']
        print(password, newPWD, confirmPWD)
        return redirect("/hr/profile/pwd")
       except:
        return redirect("/hr/signin")
#VIEWING OPENING
@app.route("/hr/openings")
def hrOpenings():
try:
       token = request.cookies.get('token')
       email = verify(token)
       with sqlite3.connect('hr.db') as connection:
        cursor=connection.cursor()
        cursor.execute(""" SELECT
id,title,company_name,designation,salary_range,skills_required,roles_respo
nsibilities,company_description,location,website,author FROM OPENINGS
WHERE author=?""", (email,))
        data=cursor.fetchall()
        data.reverse()
        connection.commit()
        return render_template("./openings/viewOpening.html",data=data)
except Exception as e:
   # return redirect("/hr/signin")
```

```
      return   render_template("./openings/viewOpening.html")
# CREATION NEW OPENING
@app.route("/hr/openings/new",  methods=('GET',  'POST'))
def hrOpeningsCreate():
    if request.method == 'GET':
        try:
            token = request.cookies.get('token')
            email = verify(token)
            return render_template("./openings/oneOpening.html")
        except:
            return redirect("/hr/signin")
    else:
        try:
            token = request.cookies.get('token')
            email = verify(token)
            title = request.form["title"]
            company_name = request.form["company_name"]
            designation = request.form["designation"]
            salary_range = request.form["salary_range"]
            skills_required = request.form["skills_required"]
            roles_responsibilities    =    request.form["roles_responsibilities"]
            company_description    =    request.form["company_description"]
            location = request.form["location"]
            website   =   request.form["website"]
            author = email
            with  sqlite3.connect('hr.db')  as  connection:
                key = uuid.uuid1().hex
                cursor = connection.cursor()
                cursor.execute("INSERT INTO OPENINGS
(id,title,company_name,designation,salary_range,skills_required,roles_respo
nsibilities,company_description,location,website,author) VALUES
(?,?,?,?,?,?,?,?,?,?,?)", (key, title, company_name, designation, salary_range,
skills_required, roles_responsibilities, company_description, location,
website, author))
                connection.commit()
                print("created successfully")
                return  redirect('/hr/openings')
        except Exception as e:
            print(e)
            return redirect('/hr/openings')#
DELETEING THE  OPENINGS
@app.route("/hr/opening/<id>")
def deleteOpening(id):
    try:
        token = request.cookies.get('token')
        email = verify(token)
        with sqlite3.connect('hr.db') as connection:
            cursor=connection.cursor()
            cursor.execute(""" SELECT id FROM OPENINGS WHERE id=?""",(id,))
            data=cursor.fetchone()
            if not data:
                return redirect("/hr/openings")
            else:
                print(data[0])
                cursor.execute(""" DELETE FROM OPENINGS WHERE id=?
""",(data[0],))
                connection.commit()
                return  redirect("/hr/openings")
    except Exception as e:
        connection.commit()
        print(e)
        return "null"
```

```
@app.route("/hr/openings/edit/<id>",methods=('GET','POST')) def
hrOpeningsOne(id):
    if  request.method=="GET":
        try:
            token = request.cookies.get('token')
            email = verify(token)
            print(email)
            if not id:
                return render_template("./openings/oneOpening.html") with
            sqlite3.connect('hr.db') as connection:
                cursor=connection.cursor()
                cursor.execute("""SELECT id,author FROM OPENINGS WHERE id=?
""",(id,))
                data=cursor.fetchone()
                if not data :
                    return redirect("/hr/openings")
                elif email== data[1]:
                    cursor.execute(""" SELECTid,
                    title,company_name,
                    designation,
                    salary_range,
                    skills_required,
                    roles_responsibilities,
                    company_description,
                    location,website,
                    author
                    FROM OPENINGS WHERE id=?""", (id,))
                    data=cursor.fetchone()
                    connection.commit()
                    print(data)
                    return render_template("./openings/editing.html",data=data)
                else:
                    return  redirect("/hr/openings")
            return render_template("./openings/oneOpening.html")
        except Exception as e:
            print(e)
            return redirect("/hr/signin")
    else:
        token = request.cookies.get('token')
        email = verify(token)
        title = request.form["title"]
        company_name = request.form["company_name"]
        designation = request.form["designation"]
        salary_range = request.form["salary_range"]
        skills_required = request.form["skills_required"]
        roles_responsibilities    =    request.form["roles_responsibilities"]
        company_description    =    request.form["company_description"]
        location = request.form["location"]
        website   =   request.form["website"]
        author = email
        with  sqlite3.connect('hr.db')  as  connection:
                cursor=connection.cursor()
                cursor.execute("""SELECT id,author FROM OPENINGS WHERE id=?
""",(id,))
                data=cursor.fetchone()
                if not data :
                    return redirect("/hr/openings")
                elif email== data[1]:
                    cursor.execute("""
                    UPDATE OPENINGS SET
                    title=?,
                    company_name=?,designation=?,
                    salary_range=?,
                    skills_required=? ,
                    roles_responsibilities=?   ,
                    company_description=?,
                    location=?,
                    website=?
                    WHERE id=? """,(title,
                    company_name,designation,
                    salary_range,
                    skills_required ,
                    roles_responsibilities ,
                    company_description,
                    location,
                    website,
                    id
            ))
                    data=cursor.fetchone()
                    connection.commit()
                    print(data)
                    return redirect("/hr/openings")
                else:
                    return  redirect("/hr/openings")
if__name__== "_main_":
    app.run(host="0.0.0.0",  port=8081,  debug=True)
```

PROJECT DEMONSTARTION VIDEO UPLOADED HERE

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-19938-1659709494

PROJECT DEMO LINK:

https:/ drive.google.com/file/d/1-0Wzmjr8JOQrsmskY7Te4cNmX7DS0Fb1/view?usp=drivesdk