SKILL AND JOB RECOMMENDER

PROJECT BASED LEARNING (NALAIYA THIRAN)

On

PROFESIONAL READINESS FOR INNOVATION,

EMPLOYABILITY AND ENTREPRENEURSHIP

Submitted by

TEAM ID:PNT2022TMID41796

TEAM MEMBERS:

K.SURESH	**622019106026**
S.SATHISH	**622019106019**
A.MUTHU SELVAN	**622019106015**
T.VISANTH	**622019106029**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"SKILL AND JOB RECOMMENDER"** is the bonafide work of **K.SURESH(**622019106026**), S.SATHISH(**622019106019**)**, **A.MUTHU SELVAN**(622019106015), **T.VISANTH**(622019106029) who carried out the project under my supervision

SIGNATURE	**SIGNATURE**	**SIGNATURE**
MR.L.Vasanth	**MR.V.Sundhararaj**	**MR.P.Gopinath**
M.E.,Ph.D.,SPOC	M.E.,Ph.D.,	M.E.,
Associate Proffessor	Assistant Proffessor	Assistant Proffesor
Department of	Department of	Department of
Electronics And	Electronics And	Electronics And
Communication Engineering	Communication Engineering	Communication Engineering
Paavai college of Engineering	Paavai college of Engineering	Paavai college of Engineering
Nammakal-637 018	Namakkal-637 018	Nammakal-637 018

ACKNOWLEDGEMENT

A great deal of time and effort has been spent in completing this project work. Several people have contributed significantly to this work and so this becomes obligatory to record our thanks to them

We express our profound gratitude to our honourable chairman Shri.CA.N.V. NATARAJAN, B.Com., FCA, and also to our correspondent Mrs.N.MANGAI NATARAJAN, M.Sc., for giving motivation and providing all necessary facility for the successful completion of this project

We have immense pleasure in expressing our sincere gratitude to our beloved Director Administration Dr.K.K RAMASAMY, M.E., Ph.D., and also extending our gratitude to Principal Dr.A.IMMANUVEL,Ph.D., for allowing us to have the expensive use of our college facilities to do this project.

We express my heartly gratitude Mr. R.PRABU M.E, HOD, Department of Electronics and Communication Engineering for extended encourage to fulfill this project

We would like to express our sincere thanks and heartiest gratitude to our project guide Mrs. K.POONKODI, M.E., Associate Professor in the department of Electronics and Communication Engineering for her valuable guidance, suggestions, encouragement and co-operation throughout our project work

We express our sincere thanks to All the Staff members of our Department for their advice and encouragement to do the project work with full interest and enthusiasm

ABSTRACT

The Internet-based recruiting platforms become a primary recruitment channel in most companies. 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. The recommender system technology aims to help users in finding items that match their personnel interests; it has a successful usage in e-commerce applications to deal with problems related to information overload efficiently. In order to improve the e-recruiting functionality, many recommender system approaches have been proposed. This article will present a survey of e-recruiting process and existing recommendation approaches for building personalized recommender systems for candidates/job matching. In the last years, job recommender systems have become popular since they successfully reduce information overload by generating personalized 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 twofold, i) We 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 engine sites. ii) We put forward the proposal of a framework for job recommendation based on professional skills of job seekers. iii) We provide a chatbot for user convenience, regarding their doubts to clarify. 4 We thus present a general panorama of job recommendation task aiming to facilitate research and real-world application design regarding this important issue.

TABLE OF CONTENTS

CHAPTER-1

1.INTRODUCTION

1.1 PROJECT OVERVIEW

There has been a sudden boom in the technical industry and an increase in the number of good startups. Keeping track of various appropriate job openings in top industry names has become increasingly troublesome. This leads to deadlines and hence important opportunities being missed. Through this research paper, the aim is to automate this process to eliminate this problem. To achieve this, IBM cloud services like db2, Watson assistant, cluster, kubernetes have been used. A hybrid system of Content-Based Filtering and Collaborative Filtering is implemented to recommend these jobs. The intention is to aggregate and recommend appropriate jobs to job seekers, especially in the engineering domain. The entire process of accessing numerous company websites hoping to find a relevant job opening listed on their career portals is simplified. The proposed recommendation system is tested on an array of test cases with a fully functioning user interface in the form of a web application. It has shown satisfactory results, outperforming the existing systems. It thus testifies to the agenda of quality over quantity.

1.2 PURPOSE

To develop an end-to-end web application capable of displaying the current job openings based on the skillset of the users.

The users and their information are stored in the Database.

An alert is sent when there is an opening based on the users skill set.

CHAPTER-2

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

Building and managing recommender systems today requires specialized expertise in analytics, applied machine learning, software engineering, this makes it challenging regardless of your background or skillset.

Intelligent_ Chatbot Description A Chatbot is a software application that replaces a live human agent to conduct a conversation via text or text to speech. In this system, we demonstrate a chatbot that uses Artificial Intelligence to produce dynamic responses to online client enquiries. This web-based platform provides a vast intelligent base that can help humans to solve problems. The Chatbot recognizes the user's context, which prompts an intended response. Its objective is to reduce human dependency in every organization and reduce the need for different systems for different processes.

2.2 REFERENCES

1. Sha ha T Al-Ota and Mourad Ykhlef. "A survey of job recommender systems". In: International Journal of the Physical Sciences 7.29 (2012), pp. 5127—5142.

2. N Deniz, A Noyan, and O G Ertosun. "Linking Person-job Fit to Job Stress: The Mediating Effect of Perceived Person-organization Fit". In: Procedia - Social and Behavioural Sciences 207 (2015), pp. 369— 376.

3. M Diaby, E Viennet, and T Launay. "Toward the next generation of recruitment tools: An online social network-based job recommender system". In: Proc. of the 2013 IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining, ASONAM 2013 (2013), pp. 821—828. 8

4. M Diaby and E Viennet. "Taxonomy-based job recommender systems on Facebook

and Linkedln profiles". In: Proc. of Int. Conf. on Research Challenges in Information Science (2014), pp. 1—6. issn: 21511357. doi: 10.1109/RClS.2014.6861048.

5. M Kusner et al. "From word embeddings to document distances". In: Proc. of the 32nd Int. Conf. on Machine Learning, ICML'15. 2015, pp. 957— 966.

6. T Mikolov et al. "Distributed Representations of Words and Phrases and Their Compositionality". In: Proc. of the 26th Int. Conf. on Neural Information Processing Systems - Volume 2. NIPS' 13. Lake Tahoe, Nevada, 2013, pp. 3111— 3119.

2.3 PROBLEM STATEMENT DEFINITION

When we start to do a project, the first thing that we should always do is define the problem. It's not only about dividing the big project into small parts, but also representing how to think about the problem, which may have varying performance in our final solution.

To build a job recommendation system providing recommendations to millions of users with millions of items, the first thing is to define the problem. Dealing with the enormous amount of recruiting information on the Internet, a job seeker always spends hours to find useful ones. Many times, people who lack industry knowledge are unclear about what exactly they need to learn in order to get a suitable job for them. We address the problem of recommending suitable jobs to people who are seeking a new job. Job recommender technology aims to help job seekers in finding jobs that match their skills. The Internet caused a substantial impact on the recruitment process through the creation of e-recruiting platforms that become a primary recruitment channel in most companies. While companies established job positions on these portals, job-seeker uses them to publish their profiles. E-recruitment platforms accomplished clear advantages for 9 both recruiters and job-seekers by reducing the recruitment time and advertisement cost. Recommender system technology aims to help users in finding items that match their preferences; it has a successful usage in a widerange of applications to deal with problems related to information overload efficiently. In order to improve the e-recruiting functionality, many system approaches have been proposed. This paper will e-recruiting process and related issues for building personalized recommender systems of candidates/job matching.

CHAPTER-3

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 helps teams 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.

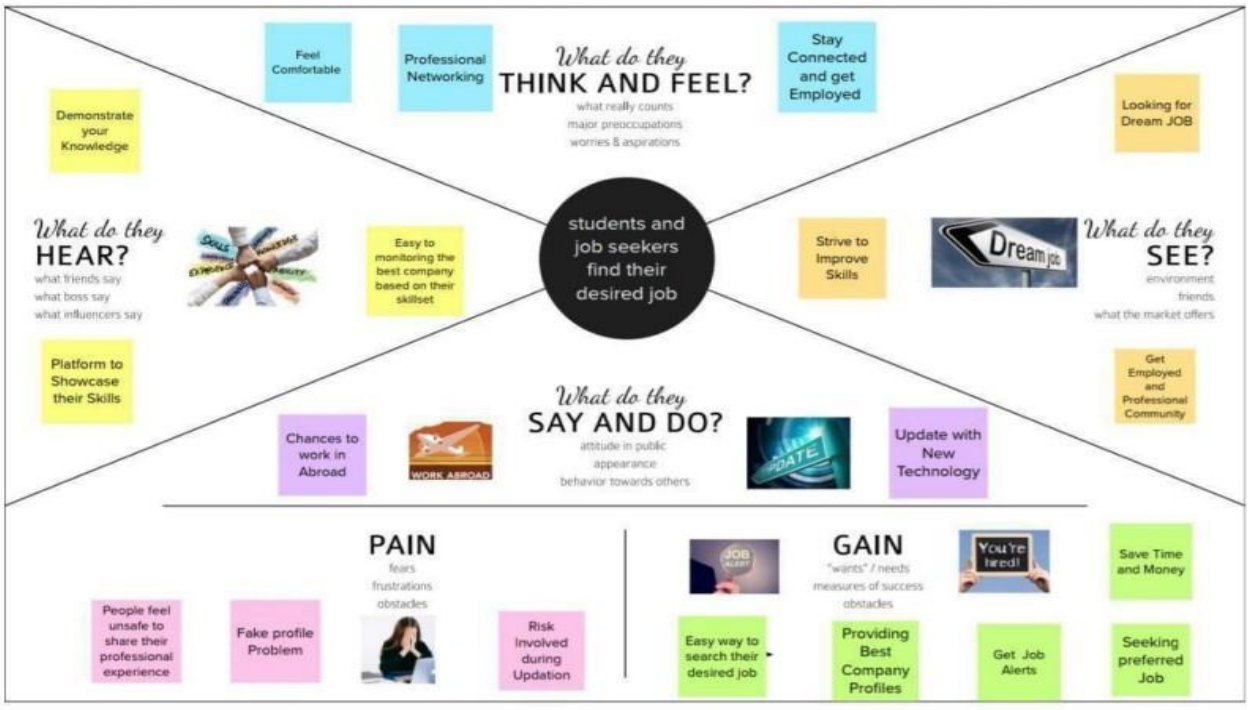


Fig 3.1 Empathy Map

3.2 IDEATION AND BRAINSTORMING

➢E-recruitment 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. The recommender system technology aims to help users in finding items that match their personnel interests; it has a successful usage in e-commerce applications to deal with problems related to information overload efficiently

➢Job recommendation application with intelligence of chatbot. In this system, we demonstrate a chatbot that uses Artificial Intelligence to produce dynamic responses to online client enquiries. This web-based platform provides a vast intelligent base that can help humans to solve problems. The chatbot recognizes the user's context, which prompts an intended response. Because this is a dynamic response, the user's desired response will be generated. This also uses a machine-learning algorithm to learn the chatbot by experiencing various requests and responses. Chatbots come to use in numerous fields of our daily life. Because AI enhances the human touch in every communication, chatbots are becoming increasingly robust. It triggers accurate responses after understanding a user's query. Its objective is to reduce human dependency in every organization and reduce the need for different systems for different processes.

➢Job seekers struggling to get the desired job for skills they have. we are proposing an application which will help the students to give Suggestions on the jobs based the skills. In this application freshers or skilled person can sign up and find the jobs by using either the search option or they can directly interact with the chatbot and get their dream job. In this application freshers or skilled person can sign up and find the jobs by using either the search option or they can directly interact with the chatbot and get their dream job. An alert is sent when there is an opening based on the user skillset. Users will interact with the chatbot and can get the recommendations based on their skills. We can use a job search API to get the current job openings in the market which will fetch the data directly from websites.

Fig 3.2 Ideation and Brainstorming

3.3PROPOSED SOLUTION

Job recommender systems are desired to attain a high level of accuracy while making the predictions which are relevant to the customer, as it becomes a very tedious task to explore thousands of jobs, posted on the web, periodically. The web recommender System suffers from many challenges.

S.NO	PARAMETERS	DESCRIPTION
1	Problem Statement (Problem to be solved)	To develop an end-to-end application capable of displaying the current job openings based on the user skillset

2	Idea / Solution description	In this paper, we proposed a framework for job recommendation task. This framework facilitates the understanding of job recommendation process as well as it allows the use of a variety of text processing and recommendation methods according to the preferences of the job recommender
		system designer. Moreover, we also contribute making publicly available a new dataset containing job seekers profiles and job vacancies. Future directions of our work will focus on performing a more exhaustive evaluation considering a greater amount of methods and data as well as a comprehensive evaluation of the impact of each professional skill of a job seeker on the received job recommendation.

3	Novelty/ Uniqueness	With the development of information technology and application of the Internet, People gradually entered the time of information overload from information scarcity. User satisfaction with recommender systems is related not only to how accurately the system recommends but also to how much it supports the user's decision making. Novelty is one of the important metrics of customer satisfaction. There is an increasing realization in the

		Recommender Systems (RS) field that novelty is fundamental qualities of recommendation effectiveness and addedvalue. This paper combed research results about definition and algorithm of novel recommendation, and starting from the meaning of "novel", defined novelty of item in recommendation system. Experiment proved using the definition of novelty to recommend can effectively recognize the item that the user is familiar with and ensure certain accuracy

4	Social Impact/ Customer Satisfaction	we develop several recommender systems and measure their ability to deliver accurate and diverse recommendations and their ability to generate customer satisfaction with diverse data sets. The results show that accuracy and diversity positively affect customer satisfaction when applying a deep learning-based recommender system. By contrast, only accuracy positively affects customer satisfaction when applying traditional recommender systems. These results imply that developers or managers of recommender systems need to identify factors that further improve customer satisfaction with the recommender system and promote the sustainable development of e-commerce.

5	Business Model (Revenue Model)	Recommendation systems allow brands to personalize the consumer experience and make suggestions for the information that make the most sense to them. A recommendation engine also lets businesses analyse the customer's current usage and past browsing history to deliver relevant service and product recommendations
6	Scalability of the Solution	Recommendation system is a which provides techniques with information, which he/she may be interested in or accessed in past. Traditional recommender techniques such as content and collaborative filtering used in various applications such as education, socia.

3.4 PROBLEM SOLUTION FIT

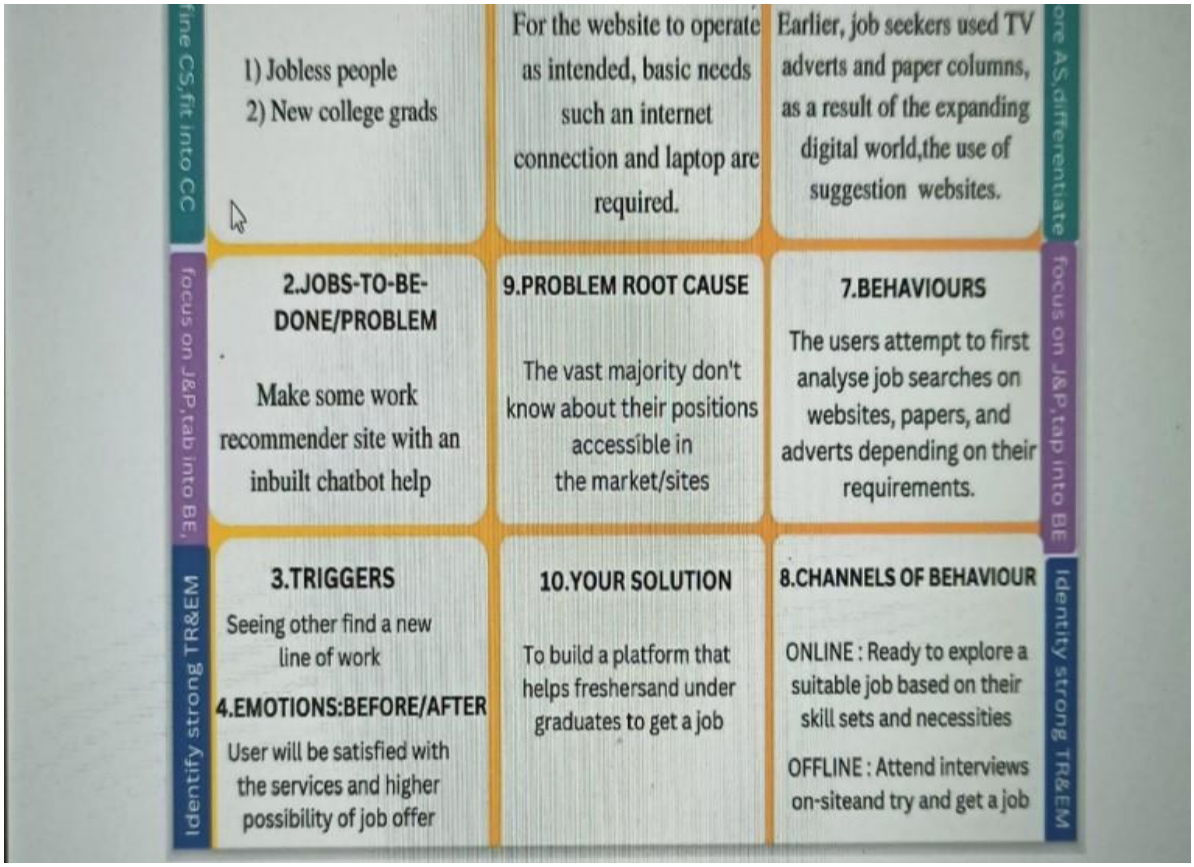
The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem. It helps entrepreneurs, marketers and corporate innovators identify behavioural patterns and recognize what would work and why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

- ❑ Solve complex problems in a way that fits the state of your customers.
- ❑ Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behaviour.

❑ Sharpen your communication and marketing strategy with the right triggers and messaging. ❑ Increase touch-points with your company by finding the right problembehaviour fit and building trust by solving frequent annoyances, or urgent or costly problems.

❑ Understand the existing situation in order to improve it for your target group.

Table 3.4 Problem Solution Fit:



CHAPTER-4

4.REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

Following are the functional requirements of the proposed solution

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 Skills set.
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 takes lesser 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

5. PROJECT DESIGN

5DATA FLOW DIAGRAMS

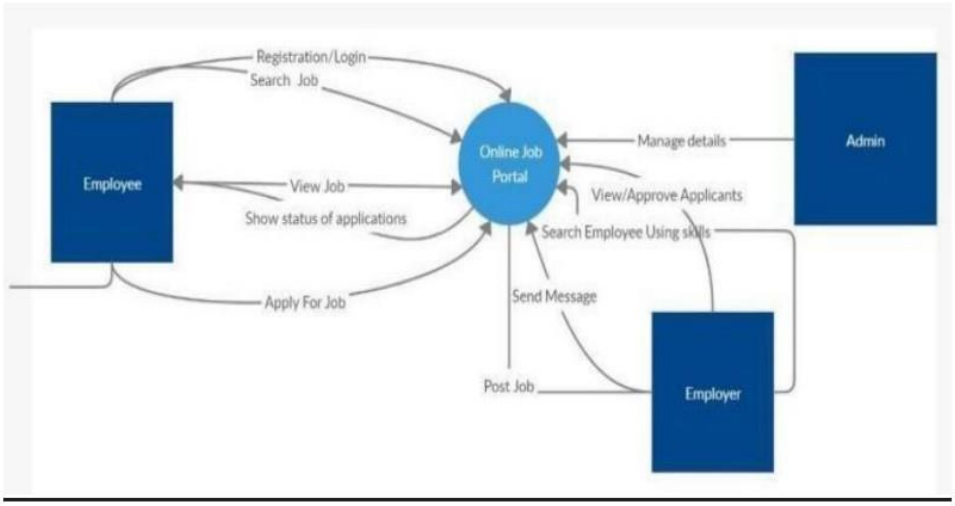


Fig 5.1 Data Flow Diagram

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 AND TECHNICAL ARCHITECTURE

JOB RECOMMENDED APPLICATION: (SOLUTION ARCHITECTURE)

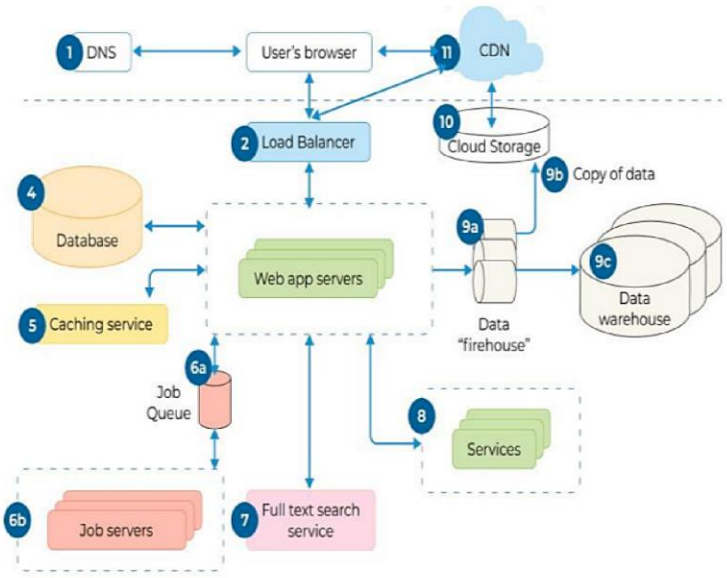


Fig 5.2: SOLUTION AND TECHNICAL ARCHITECTURE

5.3 USER STORIES

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

Table 5.3 User Stories

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 job 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 job application through Gmail	I can receive confirmation Email and apply for the job	Medium	Sprint-2
		USN-4	As a user, I	I can	Low	Sprint-1
			can register for the job application through Facebook	register & access the dashboard with Facebook Login		
	Login	Login	As a user, I can log into the application by entering email & password	I can apply for a job	High	Sprint-1
	Dashboard	USN-6	As a user, I can Search & Apply jobs posted by top companies & consultants as per your skills	Update your resume for latest jobs	High	Sprint-1

Customer (Web user)		USN-7	As a user, I can limit who can see her resume	I can receive a information from company can post new job openings	High	Sprint-1

Customer Care Executive		USN-8	As a user, I want to select a desired jobs	I can select a job based on my skills	Medium	Sprint-2
		USN-9	As a user, I can Update my resume for latest jobs	I can receive confirmati on form job portal	High	Sprint-1
		USN-10	As a use, I want to read a privacy and rules	I can access and see the privacy statement and read it in the job portal	High	Sprint-2

		USN-11	As a user, I want to quickly and easily apply for a job	I can start searching in the job portable so Its quickly as possible	Medium	Sprint-1

CHAPTER-6

6.PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6 .2 Sprint Delivery Schedule

Table 6.1: Sprint Planning &Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	7	High	1

Sprint1		USN-2	As a user, I will receive confirmation email once I have registered for the application	7	High	1
Sprint2		USN-4	As a user, I can register for the	5	Low	1
			application through Facebook			
Sprint2		USN-5	As a user, I can register for the application through Gmail	5	Medium	1
Sprint2	Login	USN-6	As a user, I can log into the application by entering email & password	10	High	1

Sprint3	Profile and details	USN-7	Update user skills in their account to use it for job search.	7		1

Sprint3		USN-8	Make user able to edit their skill set	7	Low	1
Sprint1	Communication	USN-3	A customer care executive is a professional responsible for communicating the how's and why's regarding service expectations within a company	6		1
Sprint3		USN15	Create a chat assistant for the users.	6	Low	1

Sprint4	Backend processes	USN10	Backend to search job based on user skill set.	20	High	1
Sprint5	Deployment	USN13	Containerize the application.	10	High	1
Sprint5		USN14	Deploy the application for public access.	10	High	1

Project Tracker, Velocity & Burndown Chart:

Table 6.2: Project Tracker

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint1	20	6 Days	24 Oct 2022	29 Oct 2022	20	02 Nov 2022
Sprint2	20	6 Days	31 Oct 2022	05 Nov 2022	20	06 Nov 2022
Sprint3	20	6 Days	07 Nov 2022	12 Nov 2022	20	
Sprint4	20	6 Days	14 Nov 2022	19 Nov 2022	20	

Velocity:

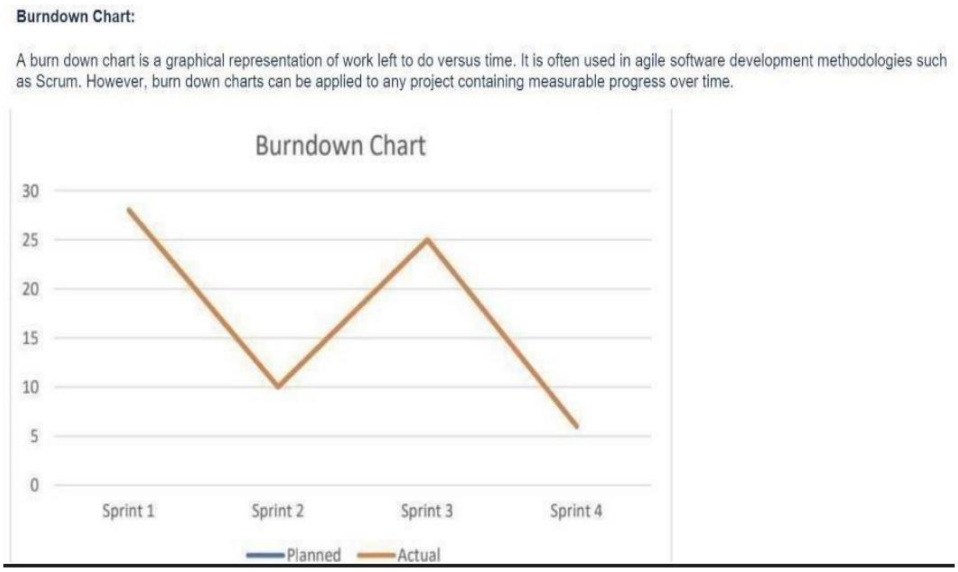
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{\text{sprint duration}}{\text{velocity}}$$

$$= \frac{20}{10}$$

$$= 2$$

6.3 Reports From JIRA



Fig;6.3 Burndown chart

CHAPTER-7

CODING & SOLUTIONING:

FEATURE 1: SIGN UP

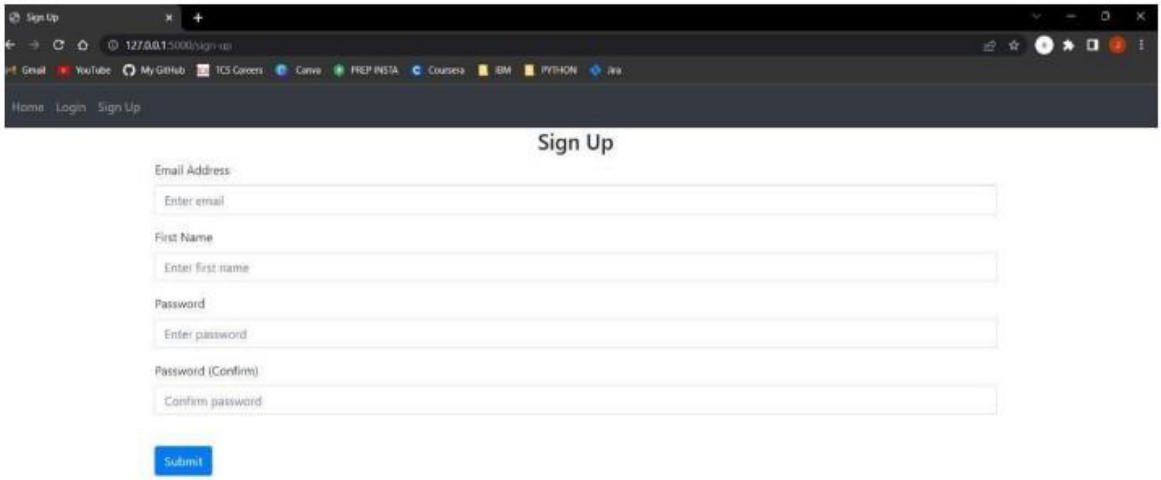
FEATURE 2: LOGIN

FEATURE 3: ADD SKILLS

FEATURE 4: APP WILL RECOMMEND THE JOBS ACCORDING TO THE SKILLS

FEATURE 5: USER CAN APPLY FOR JOBS.

SIGNUP PAGE



CODE

sign_up.html

website > templates > sign_up.html > ...

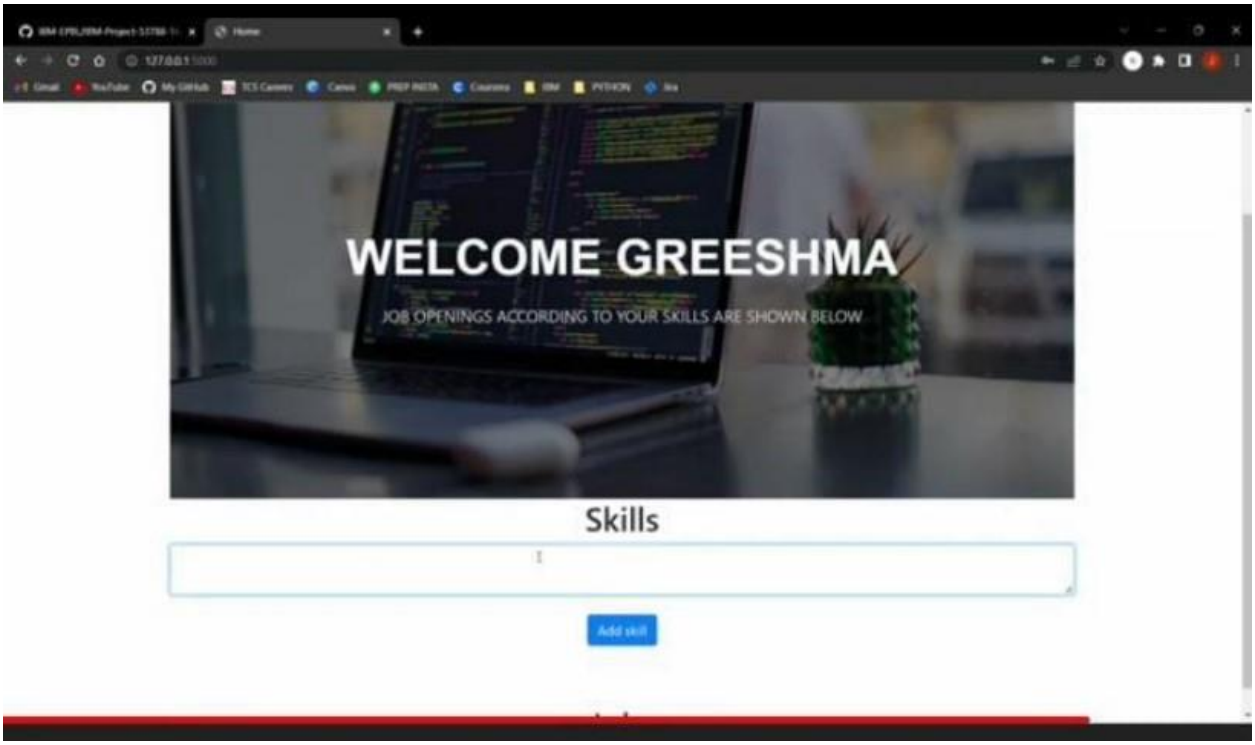
```
{% extends "base.html" %} {% block title %}Sign Up{% endblock %} {% block
content %}
<form method="POST">
  <h3 align="center">Sign Up</h3>
  <div class="form-group">
    <label for="email">Email Address</label>
    <input
      type="email"
      class="form-control"
      id="email"
      name="email"
      placeholder="Enter email"
    />
  </div>
  <div class="form-group">
    <label for="firstName">First Name</label>
    <input
      type="text"
      class="form-control"
      id="firstName"
      name="firstName"
      placeholder="Enter first name"
    />
  </div>
  <div class="form-group">
    <label for="password1">Password</label>
    <input
      type="password"
      class="form-control"
      id="password1"
      name="password1"
      placeholder="Enter password"
    />
  </div>
  <div class="form-group">
    <label for="password2">Password (Confirm)</label>
    <input
      type="password"
      class="form-control"
```

auth.py

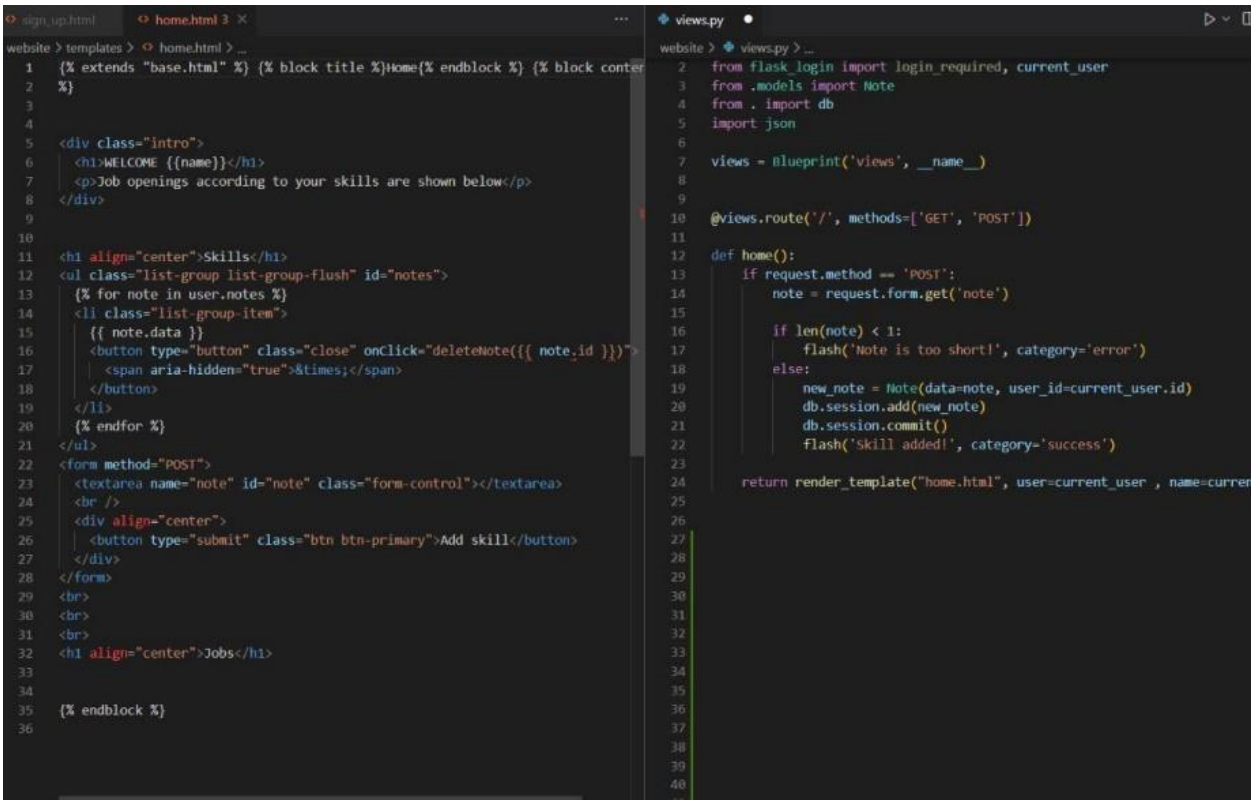
website > auth.py > ...

```
@auth.route('/sign-up', methods=['GET', 'POST'])
def sign_up():
    if request.method == 'POST':
        email = request.form.get('email')
        first_name = request.form.get('firstName')
        password1 = request.form.get('password1')
        password2 = request.form.get('password2')

        user = User.query.filter_by(email=email).first()
        if user:
            flash('Email already exists.', category='error')
        elif len(email) < 4:
            flash('Email must be greater than 3 characters.', category='error
        elif len(first_name) < 2:
            flash('First name must be greater than 1 character.', category='
        elif password1 != password2:
            flash('Passwords don\'t match.', category='error')
        elif len(password1) < 7:
            flash('Password must be at least 7 characters.', category='error
        else:
            new_user = User(email=email, first_name=first_name, password=gene
                password1, method='sha256'))
            db.session.add(new_user)
            db.session.commit()
            login_user(new_user, remember=True)
            flash('Account created!', category='success')
            return redirect(url_for('views.home'))

    return render_template("sign_up.html", user=current_user)
```

HOME PAGE


WELCOME GREESHMA
JOB OPENINGS ACCORDING TO YOUR SKILLS ARE SHOWN BELOW
Skills
Add skill

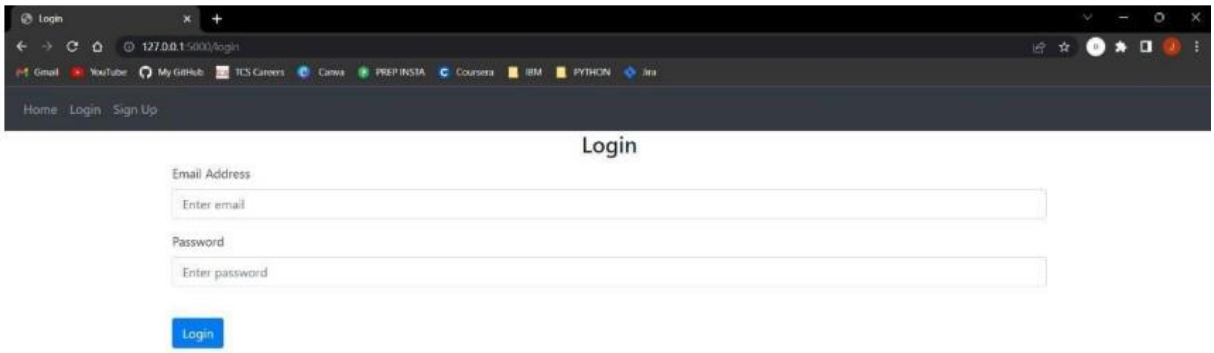
CODE



```
{% extends "base.html" %} {% block title %}Home{% endblock %} {% block conter
%}


<div class="intro">
  <h1>WELCOME {{name}}</h1>
  <p>Job openings according to your skills are shown below</p>
</div>


<h1 align="center">Skills</h1>
<ul class="list-group list-group-flush" id="notes">
  {% for note in user.notes %}
  <li class="list-group-item">
    {{ note.data }}
    <button type="button" class="close" onClick="deleteNote({{ note.id }})">
      <span aria-hidden="true">&times;</span>
    </button>
  </li>
  {% endfor %}
</ul>
<form method="POST">
  <textarea name="note" id="note" class="form-control"></textarea>
  <br />
  <div align="center">
    <button type="submit" class="btn btn-primary">Add skill</button>
  </div>
</form>
<br>
<br>
<br>
<h1 align="center">Jobs</h1>


{% endblock %}
```

```
from flask_login import login_required, current_user
from .models import Note
from . import db
import json

views = Blueprint('views', __name__)


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

def home():
    if request.method == 'POST':
        note = request.form.get('note')

        if len(note) < 1:
            flash('Note is too short!', category='error')
        else:
            new_note = Note(data=note, user_id=current_user.id)
            db.session.add(new_note)
            db.session.commit()
            flash('Skill added!', category='success')

    return render_template("home.html", user=current_user , name=curren
```

LOGIN



CODE

login.html

website > templates > login.html > ...

```
{% extends "base.html" %} {% block title %}Login{% endblock %} {% block conte
%}
<form method="POST">
  <h3 align="center">Login</h3>
  <div class="form-group">
    <label for="email">Email Address</label>
    <input
      type="email"
      class="form-control"
      id="email"
      name="email"
      placeholder="Enter email"
    />
  </div>
  <div class="form-group">
    <label for="password">Password</label>
    <input
      type="password"
      class="form-control"
      id="password"
      name="password"
      placeholder="Enter password"
    />
  </div>
  <br />
  <button type="submit" class="btn btn-primary">Login</button>
</form>
{% endblock %}
```

auth.py

website > auth.py > ...

```
from . import db
from flask_login import login_user, login_required, logout_user, current_user


auth = Blueprint('auth', __name__)


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

        user = User.query.filter_by(email=email).first()
        if user:
            if check_password_hash(user.password, password):
                flash('Logged in successfully!', category='success')
                login_user(user, remember=True)
                return redirect(url_for('views.home'))
            else:
                flash('Incorrect password, try again.', category='error')
        else:
            flash('Email does not exist.', category='error')

    return render_template("login.html", user=current_user)


@auth.route('/logout')
@login_required
def logout():
    logout_user()
    return redirect(url_for('auth.login'))
```

DATABASE SCHEME:

models.py

website > models.py > ...

```
from . import db
from flask_login import UserMixin
from sqlalchemy.sql import func


class Note(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    data = db.Column(db.String(10000))
    date = db.Column(db.DateTime(timezone=True), default=func.now())
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))


class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(150), unique=True)
    password = db.Column(db.String(150))
    first_name = db.Column(db.String(150))
    notes = db.relationship('Note')
```

CHAPTER-8

8.TESTING

8.1 TEST CASES:

Table 8.1: Test Cases

				Date	03-Nov-22						
				Team ID	PNT2022TMID25046						
				Project Name	Project - Skill/Job Recommender						
				Maximum Marks	4 marks						
	Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	Executed By
	LoginPage_TC_OO1	Functional	Login page	Verify that after registration users are navigated to login page	Mail id, Username, Password,Phone number, Pin	1. Open the website and go to register page. 2.Enter details and press register 3.Verify that users are navigated to registration page	Users should be navigated to registration page	Working as expected	Pass	Excellent	N	JEFRIN J
	LoginPage_TC_OO2	UI	Home Page	Verify the UI elements in Login/Signup popup	Username & Password	1. Open the website 2.Enter details and press login 3.Verify that users are notified of login process	Users should be notified of login process	Working as expected	Pass	Good	N	ALBERT RAVIDOSS
	LoginPage_TC_OO3	Functional	Home page	Verify user is able to log into application with Valid credentials		1. Open the website 2.Enter details and press login 3.Verify that users are logged into website properly	User should be logged into website properly	Working as expected	Pass	Good	N	ARUN K
	HomePage_TC_001	Functional	Home Page	Verify that categories of skills and jobs are shown in homepage		1. Open the website 2.Enter details and press login 3.Verify that categories of are showing Jobs shown in	Categories of skills and jobs should be shown in homepage	Working as expected	Pass	Good	N	BALAMURUGAN K
	HomePage_TC_OO2	Functional	Home page	Verify that jobs are displayed in homepage		1. Open the website 2.Enter details and press login 3.Verify that jobs are displayed in homepage	jobs should be displayed in homepage	Working as expected	Pass	Good	N	JEFRIN J
	HomePage_TC_OO3	Functional	Home page	Verify that when clicked on jobs it is redirected to correct page		1. Open the website 2.Enter details and press login 3.Verify that when clicked on jobs it is redirected to correct	When clicked on job link it should be redirected to correct page	Working as expected	Pass	Excellent	N	JAYAKRISHNAN J

8.2 USER ACCEPTANCE TESTING:

Acceptance Testing UAT Execution & Report Submiss

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

2. Defect Analysis This report shows the number of resolved or closed bugs at each severity level, and how they were

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	**3**	**3**	**20**
Duplicate	**1**	**1**	**2**	**2**	**6**

External	**2**	**3**	**0**	**1**	**6**
Fixed	**11**	**2**	**4**	**20**	**17**
Not Reproduced	**0**	**0**	**1**	**0**	**1**
Skipped	**0**	**0**	**1**	**1**	**2**
Won't Fix	**0**	**5**	**2**	**1**	**8**
Total	**24**	**14**	**13**	**26**	**80**

Table 8.2: User Acceptance Testing

8.3. Test Case Analysis

This report shows the number of test cases

Table 8.3: Test Case Analysis

Section	Total case	Not Tested	Fail	Pass
Print Engine	**7**	**0**	**1**	**7**
Client Application	**51**	**1**	**0**	**51**
Security	**2**	**0**	**2**	**2**
Outsource shipping	**3**	**0**	**1**	**3**
Exception Reporting	**9**	**0**	**1**	**9**
Final Report Output	**4**	**0**	**1**	**4**
Version Control	**2**	**0**	**0**	**2**

CHAPTER-9

9.RESULTS

9.1 PERFORMANCE METRICS

Efficiency should be a priority for employees. This requires them to have a good sense of time management and resource utilization. They should be able to monitor missed deadlines and how well a certain task was executed. But what is efficiency?
In simple terms, it is the output that you get after putting in a certain amount of input that contributes to the overall success of a business.

Here is how you can measure an employee's efficiency. For instance:

- Choose the number of tasks completed
- Measure the number of tasks completed during a period of one month.
- Measure the output against the average figure of the workplace.
  The average of the workplace is the benchmark to measure.
- Evaluate an employee's input which is the number of hours an employee puts in.
- Divide the output by the input to get the efficiency figure Remember, efficiency is a key indicator that reveals whether an employee is meeting expectations or not.

When measuring efficiency, remember to evaluate the following as well:

- The job description
- The nature of work
- Amount of work assigned
- Deadline for completing tasks
- Quality of work done

CHAPTER-10

10.ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

It can speak volumes for a candidate-in-question when they are referred by an existing employee. Not only will the current employee, the referrer, likely want to add to—and not detract from—company culture, but they'll also vouch for required skillsets and competencies. Here are the top advantages of employee referrals:

1. Your company will save time and money.

Sourcing candidates requires a lot of effort, which means it can cost a company both time and money. It was found in one study that referred candidates are 55% faster to hire, compared with employees sourced through career sites. An advantage of employee referrals is that your current team member makes the connection and saves the recruiter that initial time of sourcing the candidate. Further, the candidate could be a better match compared to other candidates who apply externally. This will also help expedite the process and cut back on the need to find alternative options.

2. Your company will receive qualified, quality candidates.

Employees will want to work with someone who will improve their own output and day-to-day workload. So, in most cases, you can have more confidence in the candidate's ability to perform the necessary tasks. Further, according to research done by Zao, nearly three in ten employers have caught a fake reference on an application. So, a personal recommendation that is already within the company can instill confidence that the reference is in fact valid and reputable.

3. Retention rate is typically better.

After two years, retention of referred employees is 45% compared to 20% from job boards. Employee referrals tend to stay around longer, perhaps because 38 they are personally connected to their peers. That's not to mention that the referrer themselves may feel more respected and valued too after their company takes their recommendation. And when an employee feels respected and valued, they can become more dedicated in turn. You may also want to give an employee referrer a bonus to show your appreciation.

DISADVANTAGES:

To properly answer "What are the advantages and disadvantages of employee referral?" we must now also look at the disadvantages. The disadvantages of employee referrals do not outweigh the benefits, but there are still some to consider. Here are three employee referrals disadvantages to keep in mind when making a hiring decision:

1. You may get a recommendation based on bias.

While in most cases an employee's motives should be "pure," there may be circumstances where a person wants to just work with their friend or receive the referral bonus. This can result in the candidate not being as qualified as either the referrer or referee said they were. The referrer may think that they can make up for the candidate's shortcomings or give them a crash course to level-set their skills. This can impact their own production in a negative way. And now your company may have two underperforming employees—and you may have to look to fill both of these positions in the not-so-far-off future.

2.Employee referrals can invite opportunity for negative company politics.

While an advantage of employee referrals is that they can positively impact peer morale, they can also cause unnecessary tension. The twosome can be negatively received by their peers especially if the external hire was chosen over an internal promotion. Further, the referrer may be afraid to offer critique to the person they referred. This kind of dynamic can negatively impact their work.

3. Your company could end up losing both the referrer and the referee.

When one goes, the other may follow. Whether one decides to leave because of company politics, personal reasons, or a better opportunity, there is a risk that their counterpart will follow suit. This chance may heighten if problems with team dynamics aren't addressed and resolved. So, it's important to stay involved with a new hire, beyond any initial onboarding and ensure they are connected to the company and not just the employee who referred them.

CHAPTER-11

11. CONCLUSION

Conclusions and directions for further research in this paper, we have considered the job recommender system (JRS) literature from several perspectives. These include the influence of data science competitions, the effect of data availability on the choice of method and validation, and ethical considerations in job recommender systems. Furthermore, we branched the large class of hybrid recommender systems to obtain a better view on how these hybrid recommender systems differ. Both this multi-perspective view, and the new taxonomy of hybrid job recommender systems has not been discussed by previous reviews on job recommender systems. Application-oriented challenges in JRS were already highlighted in early JRS contributions, though, still most literature does not take these into account. Contributions that do take different views on the JRS problem, however, do show that such views can have considerable benefits. These benefits may include improved model performance (temporal perspective), improved distribution of candidates over a set of homogeneous vacancies (reciprocal perspective), or ensuring algorithm fairness (ethical perspective). Currently, most attention goes out to how to represent the substantial amount of textual data from both candidate profiles and vacancies to create job recommendations, for which recently especially deep representations have shown promising results. However, this focus may also create the illusion that this is the only perspective that is relevant. Especially in terms of fairness, such a single perspective can be considerably harmful. Although we are not aware of algorithm audits on job recommender systems, an audit on the candidate search engines of Indeed, Career builder, and Monster, did show significant results for both individual and group unfairness in terms of gender. The increased scientific attention towards algorithm fairness, however, does provide algorithms and metrics that can be applied to measure and ensure algorithm fairness. Hence, there is a research opportunity to study how these can be transferred to the job 41 recommender system domain. Many authors state in the introduction of their contribution that there is a vast amount of data available in the form of vacancies and job seeker profiles. However, there is a clear split in the literature with regards to contributions having also access to interaction data between these two, in particular in the form of clicks/skips on the recommendation list. Interaction data can resolve the language inconsistency between job seekers and recruiters, which is especially troublesome in content-based and some knowledgebased JRS. In case interaction data is missing, one common resort is to use one of the available datasets originating from JRS competitions, in particular the CareerBuilder 2012, Rec Sys 2016, and Rec Sys 2017 competitions, which therefore have had a considerable influence on the JRS literature. An interesting aspect with respect to the usage of these competition datasets, beyond the contributions to the competitions themselves, is that these datasets are mostly used for training, but rarely for validation. This is unfortunate, as the (to our knowledge) only contribution that compares JRS on different competition datasets shows that error metrics may differ substantially across different datasets.

I.e., this raises questions with respect to the generalizability of JRS trained on one dataset. Another interesting question why (online)interaction data is sometimes not taken into account, or along the same line, why researchers often resort to the competition datasets, beyond the motives of contributing the competition or for validation. Although there may be many valid reasons, we would like to hypothesize from anecdotal experience that it can be difficult to obtain such interaction datasets, as recruitment organizations are not always part of research communities, or given that these recruitment organizations have not always considered the implications of sharing data for research, either from a technical or legal point of view, making it difficult to use such datasets on a short term.

CHAPTER-12

12.FUTURE SCOPE

Future directions of our work will focus on performing a more exhaustive evaluation considering a greater amount of methods and data as well as a comprehensive evaluation of the impact of each professional skill of a job seeker on the received job recommendation. We can use machine learning techniques to recommend data in a efficient way.

CHAPTER-13

13.APPENDIX

SOURCE CODE

Front end

```
import { useToast } from "@chakra-ui/react";
import React, { useContext, useEffect, useState } from
"react";
import { useNavigate } from "react-router-dom"; import
{ AppContext } from "../context/AppContext"; import {
loginUser } from "../proxies/backend_api"; import {
emailRegex } from "../utils/helper";

const Login = () => {
const toast = useToast();
    const { setUser } = useContext(AppContext);
    const navigate = useNavigate();
const [inputs, setInputs] = useState({
email: "", password: "",
});
const [error, setErrors] = useState({
email: "", password: "",
});
const handleChange = ({ target: { name, value } }) => {
setErrors((prev) => { return { ...prev, [name]: ""
};
}); setInputs((prev) => ({ ...prev, [name]:
value }));
};
 const checkInputErrors = () => {
let status = true;
            if (inputs.email.trim() === "" ||
!emailRegex.test(inputs.email.trim())) {
setErrors((prev) => {
        return { ...prev, email: "Enter a valid email" };
        });
```

```
status = false;
          }
          if (inputs.password.trim() === "") {
setErrors((prev) => {  return { ...prev, password: "Enter
          a valid password"
};       });
status = false;
     }
    if (inputs.password.trim().length < 6) {
          setErrors((prev) => {
               return { ...prev, password: "Minimum 6 characters" }; }); status = false; } return
          status;
};
          });
status = false;
    }    return
status;
};
const handleLogin = async () => {
if (checkInputErrors()) {  const data =
          await loginUser(inputs);
if (data.error) {           toast({
title: data.error           , status:
"error"           , duration: 3000,
isClosable:  true,         variant: "left
accent",          position: "top",
          });
return;            }
setUser(data);
toast({
          title: `Welcome back ${data.name}`,
          status: "success",
duration: 3000, isClosable:
true, variant: "left-accent",
position: "top",
          });
```

```
        localStorage.setItem("user", JSON.stringify(data));
        navigate("/dashboard");
      }
 };
 return (
<>
        <div>
           <button className="big-base-300 rounded-box flex flex-row flex
flex -row justify-evenly items -center gap-10 px -10 py-5 wfit  mx-auto">
<span>Sign in with Github</span>        <img src={'github-dark.png}
alt="github"
width="14%"/>
       </button>
       <div classname ="divider  max -w-xs">or</div>
      <form
          onSumbit={(e)                              =>
e.preventfault()}
 className="card bg-base-300 rounded-box flex flexcol

 justify-center items-center gap-5 px-10 py-5 w-fit mxauto"

       >

          <div>

              <input

value={inputs.email}

type="text"                  name="email"

          placeholder="email"

          className="input input -bordered input-primary  w-

full"

           onChange={handleChange}

           />
```

```
          {error.email  !== " "&&  (
             <p  className="text-sm text -red-500 mt-1
fontmedium">
              {error.email}
            </p>
          )}
</div>
  <div>
<input
  value ={inputs .password } type="password
name ="password" placeholder="password"
                 className="input  input -boardered input -
primary w-full"
                onChange={handlechange}
                />
                {error.password  !== " "&& (
                      <p className="text-sm text - red -500 mt -1
fontmedium">
                  {error. password}
                 </p>
                )}
               </div>
```

```
                <div  className="text -center">
                    <button
type ="submit"
onClick={handleLogin}
             className="btn btn-sm  btn-primary mb-4 "
            >
               Login
                </button>
            </div>
          </form>
        </div>
    </>
);
};
export default Navbar;
 styling
 @import
 url("https://fonts.googleapis.com/css2?family=Ubuntu&displ
 ay=swap");
@tailwind base;
 @tailwind components;
@tailwind utilities;
```

```
:root
{
font-family:
Inter, Avenir, Helvetica, Arial, sansserif; font-size: 16px;
line-height: 24px; font-weight:
400;  color-scheme:
light;
/* color: rgba(255, 255, 255, 0.87); backgroundcolor: #242424; */
font-synthesis: none; text- font-synthesis:
none;  text-rendering: optimizeLegibility;
-webkit-font-smoothing: antialiased
; -moz-osx-font-smoothing: grayscale;
-webkit-text-size-adjust: 100%;
 *{
 margin: 0;
padding: 0;
font-family: "Ubuntu", sans-serif;
}
body::-webkit-scrollbar {
width: 5px; background-color:
none; border-radius: 20px;
}
```

```
body::-webkit-scrollbar-thumb { background-
color: #adada;  border-radius: 20px;
} body {   max-
height: 100vh;
}
```

Backend from backend import

```
create_app  app = create_app() if
 __name__ == '__main__':
from waitress import serve
serve(app, port=5000)

from dotenv import dotenv_values
from flask import Flask from
 flask_cors import CORS import
 ibm_db
 # Get the environment variables config =
dotenv_values("backend/.env")
 # Connect to db try:
 # conn = 'dd  ' conn
 = ibm_db.pconnect(

f"DATABASE= { config[ 'DB2_DATABASE ' ] } ;
HOSTNAME={conf ig[ 'DB2_HOSTNAME' ] } ;
```

```
PORT={ config['DB2_PORT' ] } ; SECURITY=SSL ;
SSLServerCertificate=backend/DigiCertGlobalRootCA .crt;UID=

{config ['DB2_USERNAME']};PWD={config['DB2_PASSWORD']}", ", ")

 print("Connected to IBM_DB2 successfully!!")

 print(conn) except:

 print("Failed to connect to Database!")

 def create_app():

 # Tell flask to use the build directory of react to serve

static content

 app = Flask(__name__, static_folder='../build',

 static_url_path='/')

 CORS(app)

 # Set the secret key for flask

 app.config['SECRET_KEY'] = config['APP_SECRET']

 # Import and register auth_router

 from .auth_router import auth

 app.register_blueprint(auth, url_prefix='/api/auth')

 from .files_router import files

app.register_blueprint(files, url_prefix='/api/files')

from .user_router import user

app.register_blueprint(user, url_prefix='/api/user')

 # In production serve the index.html page at root

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

```
def home():
return app.send_static_file('index.html')
return app
auth = Blueprint("auth", __name__)
LOGIN_FEILDS = ('email', 'password')
SIGNUP_FEILDS = ('name', 'email', 'phone_number', 'password')
@auth.route("/login", methods=['POST']) def
login_user():
        # Check if all the required feild are present
for feild in LOGIN_FEILDS:
            if not (feild in request.json):
                return jsonify({"error": f"All feilds are required!"}), 409
        email =request.json['email']
password = request.json['password']
        sql = f"select * from users where
email='{email}'"  stmt = ibm_db.prepare(conn, sql)
ibm_db.execute(stmt)          user =
ibm_db.fetch_assoc(stmt)      if not user:
        return jsonify({"error": "Invalid credentials!"}),
401         if bcrypt.checkpw(password.encode('utf-8'),
user["PASSWORD"].encode('utf-8')):
        token = jwt.encode(
```

```
{"email": email},
 config["APP_SECRET"],
 algorithm="HS256"
) return jsonify({"name": user["NAME"],
 "email": email, "phone_number" :
 user["PHONE_NUMBER"],
 "token":
token}), 200
 else:
 return jsonify({"error": "Invalid credentials!"}),
401
 @auth.route("/signup", methods=['POST']) def
 register_user():
 # Check if all the required feild are present
for feild in SIGNUP_FEILDS:

if not (feild in request.json):
 return jsonify({"error": f"All feilds are required!"}),409
 email = request.json['email']
 phone_number = request.json['phone_number']
name = request.json['name'] password =
request.json['password']
```

```
# Sql stmt to check if email/number is already in use
sql = f"select * from users where email='{email}'  or
phone_number='{phone_number}' "              stmt =
ibm_db.prepare(conn, sql)
 ibm_db.execute(stmt)          user
= ibm_db.fetch_assoc(stmt) if user:
            return jsonify({"error": f"Email/Phone number is alread in use!"}), 409
                # If user does not exist, then create account
 hashed_password = bcrypt.hashpw(
                        password.encode('utf-8'), bcrypt.gensalt())
                                sql = f"insert into
 users(name,email,phone_number,password)
 values('{name}','{email}','{phone_number}',?)" stmt =
ibm_db.prepare(conn, sql) ibm_db.bind_param(stmt, 1, hashed_password)
              ibm_db.execute(stmt)
                token = jwt.encode(
{"email": email},
 config["APP_SECRET"],
 algorithm="HS256"
)
       return jsonify({"name": name, "email": email,
"phone_number": phone_number, "token": token}), 200
```

GITHUB & PROJECT DEMO LINK:

GITHUB link - https://github.com/IBM-EPBL/IBM-Project-53788-1661495800

PROJECT DEMO VIDEO link-