PROJECT REPORT

SKILL / JOB RECOMMENDER APPLICATION

SUBMITTED BY

TEAM ID : PNT2022MID48792
TEAM LEADER : DEEPA.S -921019104009
TEAM MEMBER : ABIRAMI.V-921019104001
TEAM MEMBER : MADHUMITHA.C-921019104025
TEAM MEMBER : DEEPTHIKA.K-921019104010
TEAM MEMBER : PREAMEENA.S-9210191039

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 PROJECT OVERVIEW

Job recommendation has traditionally been treated as a filter-based match or as a recommendation based on the features of jobs and candidates as discrete entities. In this paper, we introduce a methodology where we leverage the progression of job selection by candidates using machine learning. Additionally, our recommendation is composed of several other sub-recommendations that contribute to at least one of a) making recommendations serendipitous for the end user b) overcoming cold-start for both candidates and jobs. One of the unique selling propositions of our methodology is the way we have used skills as embedded features and derived latent competencies from them, thereby attempting to expand the skills of candidates and jobs to achieve more coverage in the skill domain. We have Deployed our model in a real-world job recommender system and have achieved the best click-through rate through a blended approach of machinelearned recommendations and other subrecommendations. For recommending jobs through machine learning that forms a significant part of our recommendation, we achieve the best results through BiLSTM with attention.

1.2 PURPOSE

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

CHAPTER 2
LITERATURE SURVEY

2.1 EXISTING PROBLEM

The internet hold a considerable number of websites where job seekers can search for jobs,we do find it relevent to put some emphasis on contributions published by linkedIn employees for a numbers of reasons. furthermore, although many job seekers use well-known general-purpose search engine /social network

2.2 REFERENCES

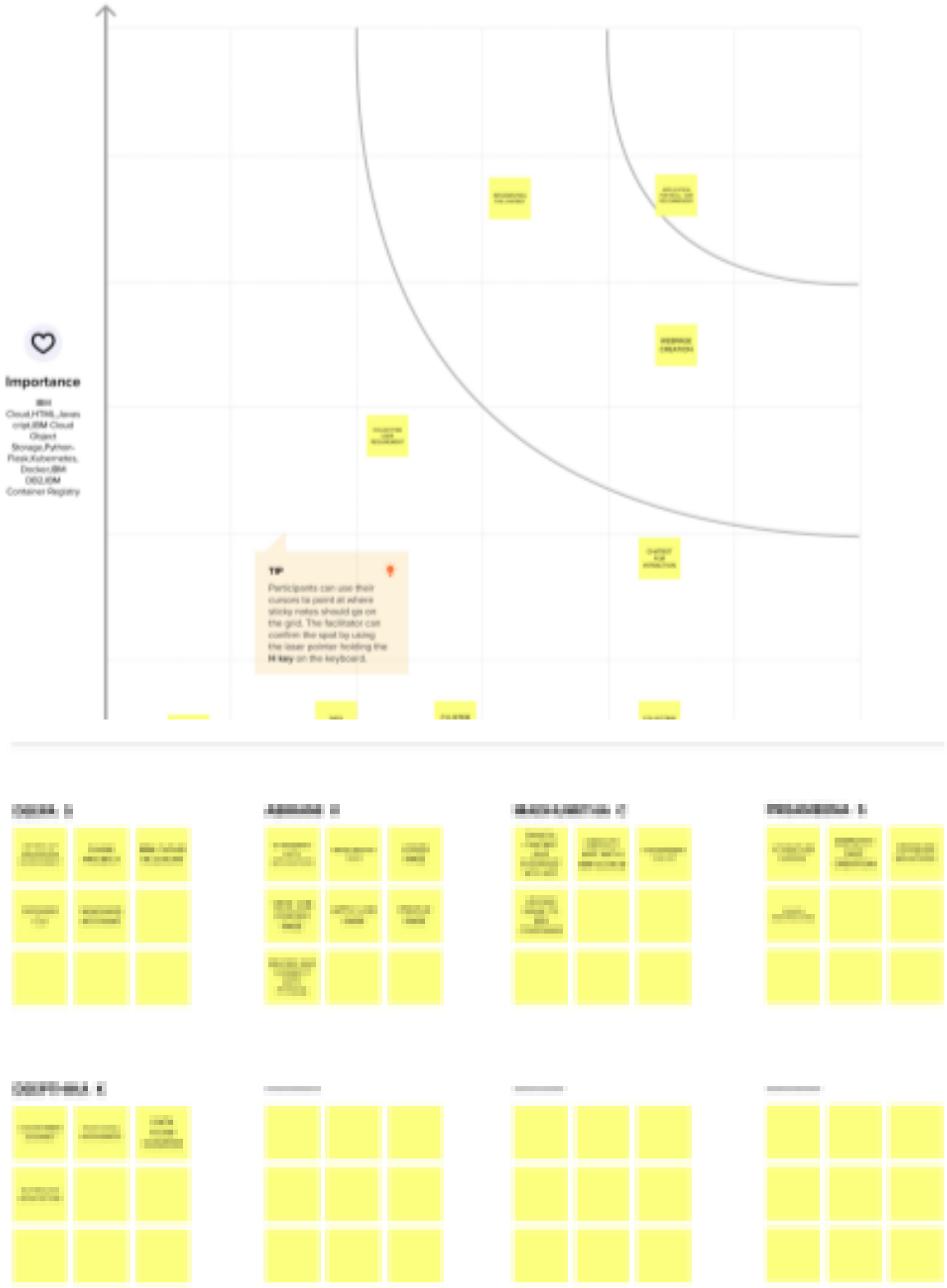
https://www.researchgate.net/publication/325697854_Job_Recommendation_based_on_Job_Seeker_Skills_An_Empirical_Study

Mauricio Noris Freire and Leandro Nunes de Castro. e-Recruitment recommender systems: a systematic review. Knowledge and Information Systems, pages 1–20, 2020

CHAPTER 3
IDEATION AND PROPOSED SOLUTION

3.1 EMPHATHY MAP CANVAS

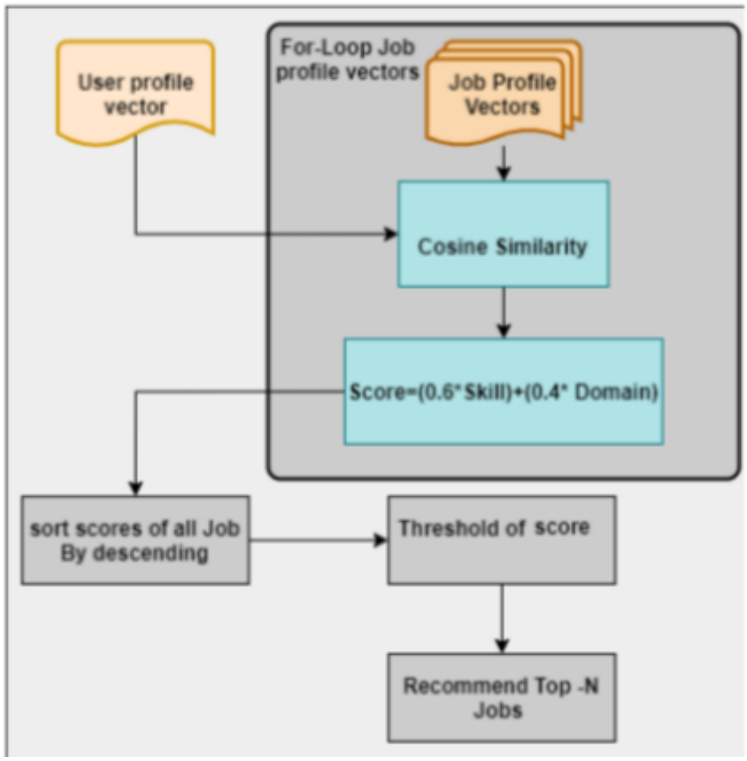
3.2 IDEATION & BRAINSTORMING



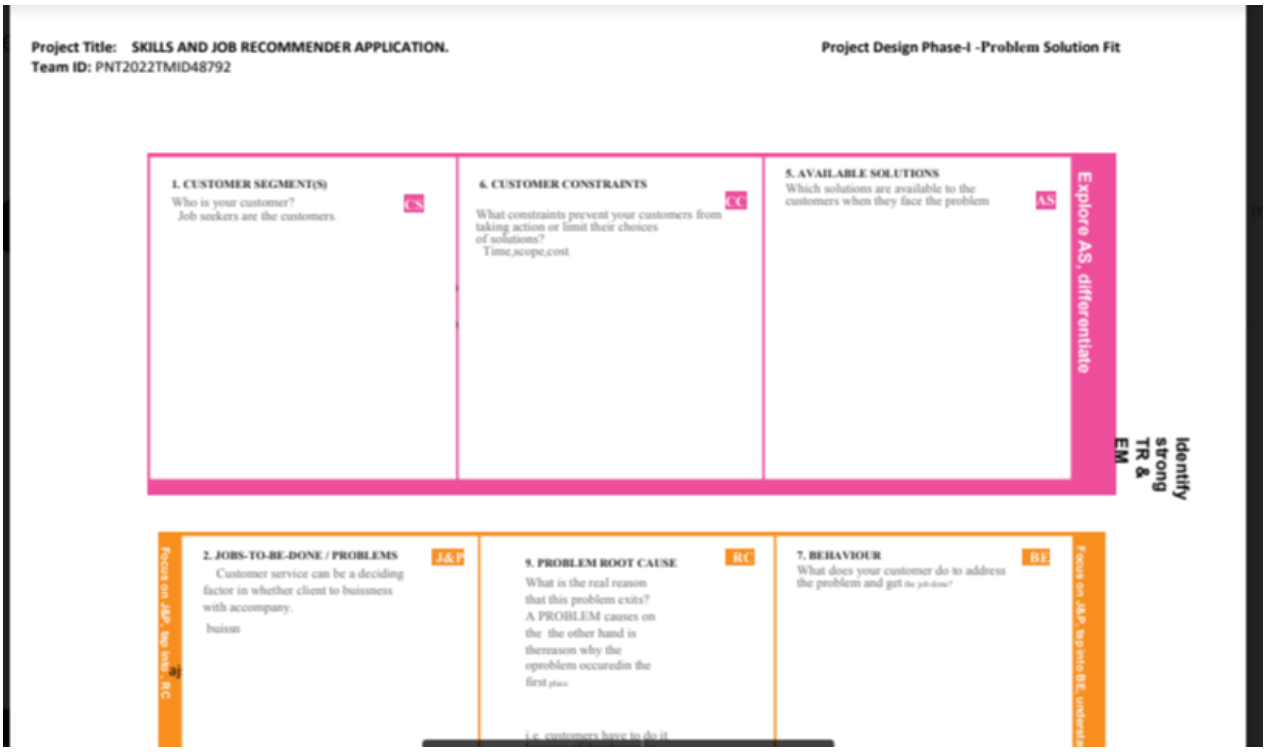
3.3 PROPOSED SOLUTION

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The dataset used for this research are sourced from Stack overflow survey data which is modelled as the user data for this research. Another dataset was created by web scrapping the Job board Using R programming language to fulfill the road map of this dissertation. The research question proposed by this research is "Can an efficient recommender system be modeled for the Job seekers which recommend Jobs with the user's skill set and job domain and also addresses the issue of cold start".
2.	Idea / Solution description	Implement a recommender system to recommend jobs from the list of job dataset for a particular user based on the user profile vector; Which includes the details such as what language user would like to work on, what frameworks he has worked on, what was his role or domain of his work. This information is utilised to check similarity between the job profile vector. This led to generation of score against each job. The score is filtered using the rating scale approach, where we set a particular threshold value and subset the recommendation list by considering jobs with score greater than threshold value. To select the threshold, we performed the evaluation by taking random user for analysis of best threshold value for the recommendation .
3.	Novelty /	As early as 1999, Baeza-Yates and Ribiero-Neto briefly

	Uniqueness	discussed the novelty in information retrieval, the novelty of a retrieval set has been defined with respect to the end-user as the proportion of known and unknown relevant items in the recommended list[17]. That is, given is the set of items in R that the user likes, L can be partitioned as into those items, is already known items to the user and Lu is unknown items to the uer. Then the novelty is NOVELTY(R)=[LU]/[L].
4.	Social Impact / Customer Satisfaction	Advantage: Use many attributes. Transition history is included. Disadvantage: _One way recommendation. – No relational aspects are included. – Scalability, ramp-up, and data sparsity problems.

5.	Business Model (Revenue Model)	
6.	Scalability of the Solution	Therefore, We conclude that job recommendation system with analysis of job description to recommend a job based on user's skills and preferences presents itself as worthy Recsys model in recommending open position to the job seekers when looking for a new positions. Thus, among the different threshold and filtering techniques, we chose to model the recommender system using content-based filtering which is achieving F1-score of 66% with the threshold of 0.3 with average coverage of 53%.

3.4 PROBLEM SOLUTION FIT



CHAPTER 4
REQUIREMENT ANALYSIS

4.1 FUNTIONAL REQUIREDMENT

Functional Requirements:

Following are the functional requirements of the proposed solution.

Fr No.	Functional requirement (epic)	Sub requirement (story / sub-task)
Fr-1	User registration	Registration through form Registration through gmail Registration through linkedin
Fr-2	User confirmation	Confirmation via email Confirmation via otp
Fr-3	Sign in	To login and see the recommended jobs
Fr-4	Signup	To create a new profile

NON FUNCTIONAL REQUIREMENT

Following are the non-functional requirements of the proposed solution.

FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	Usability study is a study in which a product can be used by specific users to achieve specific objectives with effectiveness, efficiency, and satisfaction in a specific context of use.

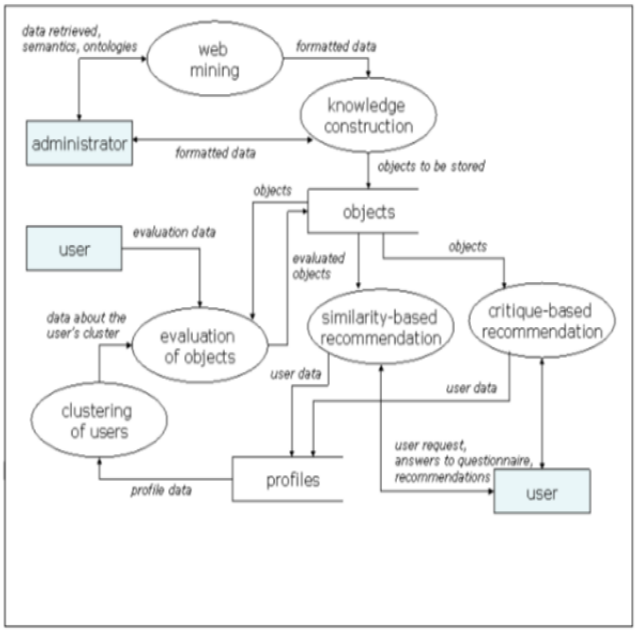
NFR-2	**Security**	An application security engineer ensures that every step of the software development lifecycle (SDLC) follows security best practices. They are also responsible for adhering to secure coding principles and aid in testing the application against security risks/parameters before release.
NFR-3	**Reliability**	According to a dictionary definition, being reliable is the quality of being trustworthy and performing consistently well. Reliability is considered to be a soft skill.
NFR-4	**Performance**	The common way to assess the performance

		recommender system would be through standard metrics such as Accuracy, Precision or Recall [1,2]. However, these metrics require ground truth knowledge about which recommendations are correct, which is hard to obtain at a large scale in our specific problem setting.
NFR-5	**Availability**	the quality or state of being available trying to improve the availability of affordable housing
NFR-6	**Scalability**	Scalability is the measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processing demands.

CHAPTER 5
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

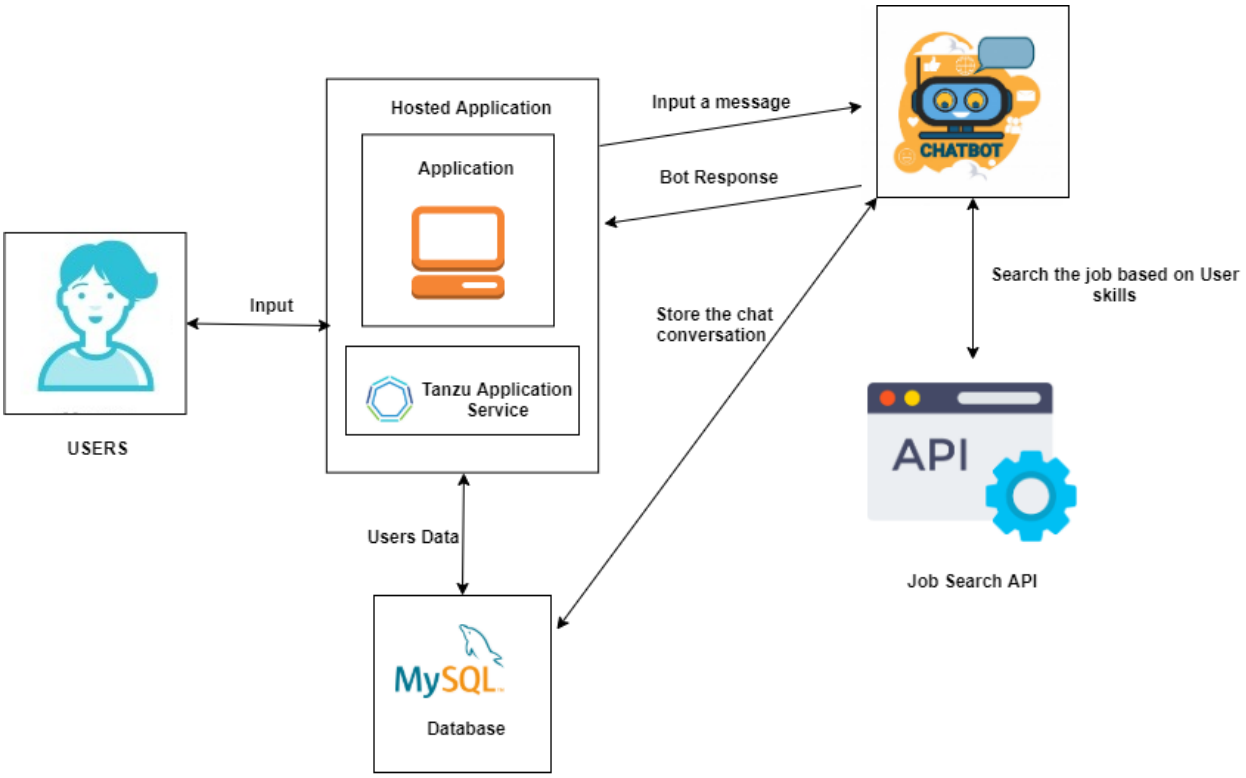
- Job skill match (Recruitment)
- Employee attrition prediction (Employee Management)
- Pay gap by gender, ethnicity, profession (Employee Compensation)
- Organizational network analysis (ONA) FUTURE WORK



5.2 SOLUTION AND TECHNICAL ARCHITECTURE

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to: Find the best tech solution to solve existing business problems. Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders. Define features, development phases, and solution requirements. Provide specifications according to which the solution is defined, managed, and delivered.

Example - Solution Architecture Diagram:



5.3 USER STORIES

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

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application	I can register & access the dashboard	Low	Sprint-2

		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1

		USN-3	As a user, I can register for the application	I can register & access the dashboard	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1

CHAPTER 6
PROJECT PLANING AND SCHEDULING

6.1 SPRINT PLANING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Priority Points	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2 High	Deepa, Abirami
Sprint-2		USN-2	As a user, I will receive confirmation email once I have registered for the application	1 High	Deepthika, Preameena
Sprint-3		USN-3	As a user, I can recommend job according to user data	2 High	Madhumitha

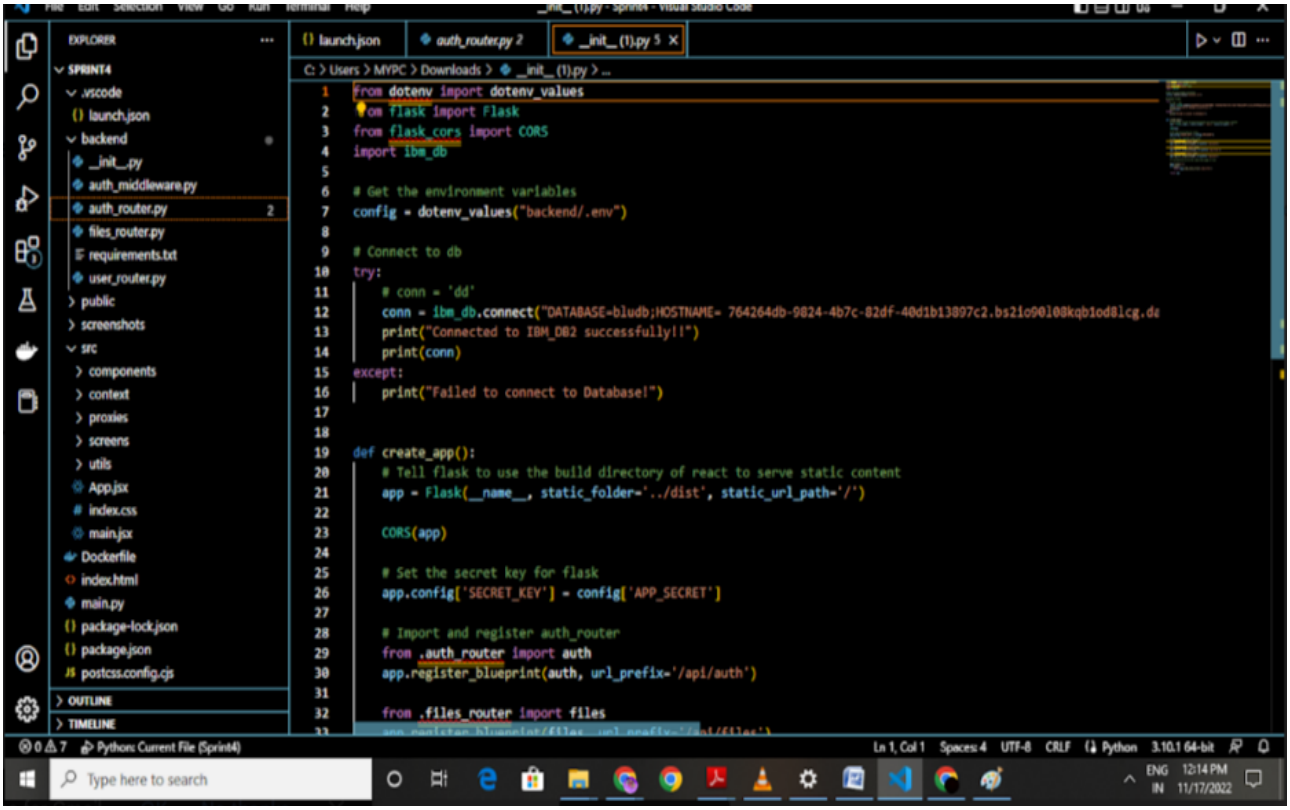
Sprint-4	Dashboard	USN-4	As a user I can show user data	1 High	Deepthika, Preameena

6.2 SPRINT DELIVERY SCHEDULE

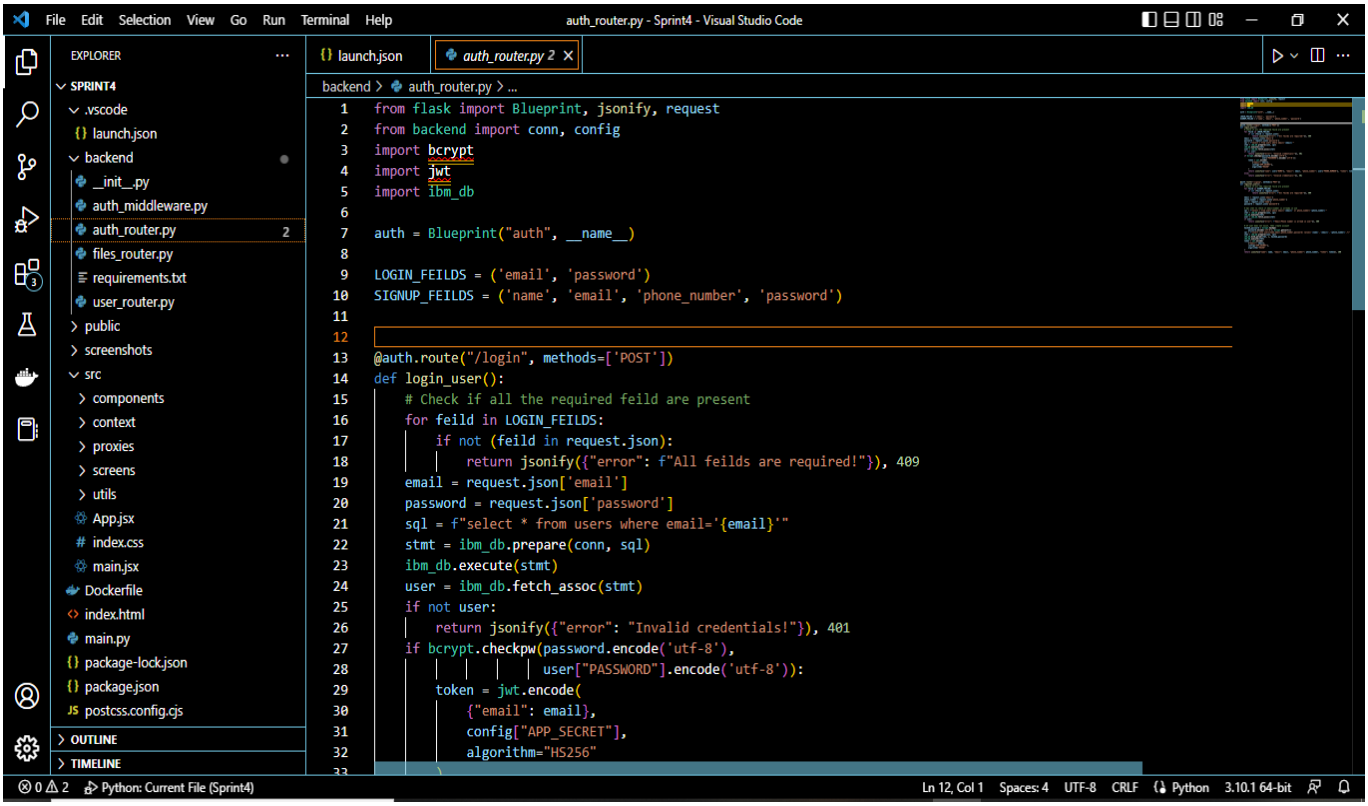
Sprint	Total Story Points	Duration	Sprint start date & Sprint End date	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022 29 Oct 2022 20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022 05 Nov 2022	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022 12 Nov 2022	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022 19 Nov 2022	19 Nov 2022

CHAPTER 7
CODING & SOLUTIONING

7.1 FEATURE1



7.2 FEATURE 2


File Edit Selection View Go Run Terminal Help
auth_router.py - Sprint4 - Visual Studio Code
EXPLORER
launch.json
auth_router.py 2
backend > auth_router.py > ...
SPRINT4
.vscode
launch.json
backend
__init__.py
auth_middleware.py
auth_router.py
files_router.py
requirements.txt
user_router.py
public
screenshots
src
components
context
proxies
screens
utils
App.jsx
index.css
main.jsx
Dockerfile
index.html
main.py
package-lock.json
package.json
postcss.config.cjs
OUTLINE
TIMELINE
from flask import Blueprint, jsonify, request
from backend import conn, config
import bcrypt
import jwt
import ibm_db
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"
)
0 2 Python: Current File (Sprint4)
Ln 12, Col 1 Spaces: 4 UTF-8 CRLF Python 3.10.1 64-bit

CHAPTER -8
TESTING

8.1 TEST CASES

B	C	D	E	F	G	H	I	J	K	L	M	N	O
			Date	03-Nov-22									
			Team ID	PNT2022TMID48792									
			Project Name	Project - Skill/Job Recommende									
			Maximum Marks	4 marks									
Feature Type	**Component**	**Test Scenario**	**Pre-Requisite**	**Steps To Execute**	**Test Data**	**Expected Result**	**Actual Result**	**Status**	**Commnets**	**TC for Automation(Y/N)**	**BUG ID**	**Executed By**	
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	https://drive.google.com/drive/folders/1OiftIud88VX7m0v-iEyDMq41Nmv1KFp5	Users should be navigated to registration page	Working as expected	Pass	Excellent	'N		DEEPA.S	
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	https://drive.google.com/drive/folders/1OkpV13FJnZ1OboSw2zPAIkBELGsY9hRok	Users should be notified of login process	Not working	Fail	Trying To Recover	N	BUG-12	ABIRAMI.V	
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	Username: abivnscet@gmail.com password: abirami01	User should be logged into website properly	Working as expected	Pass	Good	N		MADHUMITHA.C	
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 homepage		Categories of skills and jobs should be shown in homepage	Working as expected	Pass	Good	N	BUG-14	PREAMEENA.S	
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		DEEPTHIKA.K	
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 page		When clicked on job link it should be redirected to correct page	Working as expected	Pass	Excellent	N		ABIRAMI.V	

8.2 USER ACCEPTANCE TESTING

8.2.1 DEFECT ANALYSIS

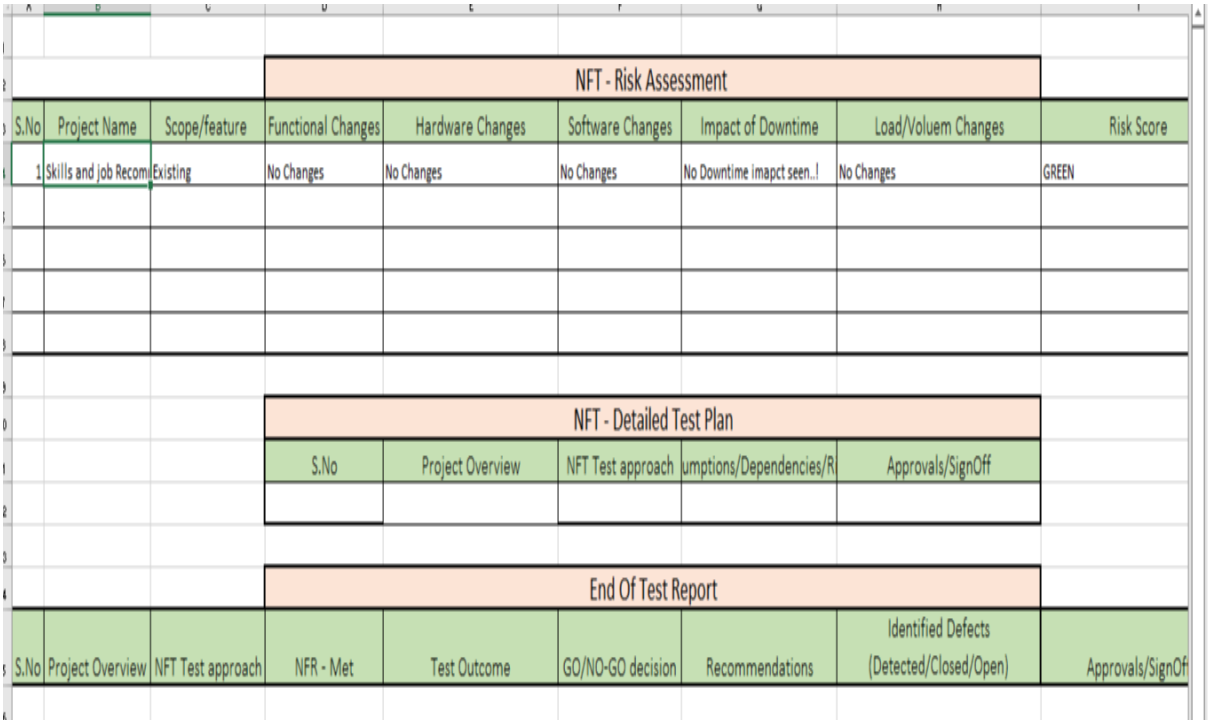
Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	1	3	1	6
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	80

8.2.2 TEST CASE ANALYSIS

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	1	7
Client Application	51	0	1	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
RESULTS

9.1 PERFORMANCE METRICES



NFT - Risk Assessment

S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Voluem Changes	Risk Score
1	Skills and job Recom	Existing	No Changes	No Changes	No Changes	No Downtime imapct seen..!	No Changes	GREEN

NFT - Detailed Test Plan

S.No	Project Overview	NFT Test approach	umptions/Dependencies/R	Approvals/SignOff

End Of Test Report

S.No	Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Recommendations	Identified Defects (Detected/Closed/Open)	Approvals/SignOff

CHAPTER 10
ADVANTAGES & DISADVANTAGES

ADVANTAGES

Recommender systems can be used across multiple verticals such as e-commerce, entertainment ,mobile apps,education and more details(discussed in detail later).In general a recommendation engine can be helpful in any situation where there is a need to give users personalized suggestions and advice.

DISADVANTAGES

Faulty recommendation engines that inaccurately estimate consumers' true preferences stand to pull down willingness to pay for some items and increase it for others, regardless of the likelihood of actual fit. This may tempt less ethical organizations to inflate recommendations artificially.

CHAPTER 11

CONCLUSION

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.

CHAPTER 12
FUTURE SCOPE

This project is far from complete and there is a lot of room for improvement. some of the improvements that can be made to this project are as follows:

- It will store the huge data in the cloud so we can easily access and retrieve the data.
- AND,IT will match the job based on the skill. There will be a huge employeement opportunity.
- enormous storage of datas
- no need to maintain the sever

APPENDIX

SOURCE CODE

package.json :

```
{
  "name": "react-flask-app",
  "private": true,
  "version": "0.0.0",
  "type": "module",
  "scripts": {
    "start": "vite",
    "build": "vite build",
    "preview": "vite preview",
    "server": "cd backend && flask --debug run"
  },
  "dependencies": {
    "axios": "^1.1.3",
    "daisyui": "^2.33.0",
    "react": "^18.2.0",
    "react-dom": "^18.2.0",
    "react-icons": "^4.6.0",
    "react-router-dom": "^6.4.2"
  },
  "devDependencies": {
    "@types/react": "^18.0.17",
    "@types/react-dom": "^18.0.6",
    "@vitejs/plugin-react": "^2.1.0",
    "autoprefixer": "^10.4.12",
    "postcss": "^8.4.18",
    "tailwindcss": "^3.1.8",
    "vite": "^3.1.0"
  }
```

}

index.html:

```
{
  "name": "react-flask-app",
  "private": true,
  "version": "0.0.0",
  "type": "module",
  "scripts": {
    "start": "vite",
    "build": "vite build",
    "preview": "vite preview",
    "server": "cd backend && flask --debug run"
  },
  "dependencies": {
    "axios": "^1.1.3",
    "daisyui": "^2.33.0",
    "react": "^18.2.0",
    "react-dom": "^18.2.0",
    "react-icons": "^4.6.0",
    "react-router-dom": "^6.4.2"
  },
  "devDependencies": {
    "@types/react": "^18.0.17",
    "@types/react-dom": "^18.0.6",
    "@vitejs/plugin-react": "^2.1.0",
    "autoprefixer": "^10.4.12",
    "postcss": "^8.4.18",
    "tailwindcss": "^3.1.8",
    "vite": "^3.1.0"
  }
}
```

Backend:

init.py:

```
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.connect("DATABASE=bludb;HOSTNAME= 764264db-9824-4b7c-82df-
40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=
32536;SECURITY=SSL;
SSLServerCertificate=backend/DigiCertGlobalRootCA.crt;UID=khd31033;PWD=CBYBNxi
K00QKGKJR",", ")
    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='../dist', 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_middleware:

```
from functools import wraps
import jwt
from flask import request
from backend import conn, config
import ibm_db

# Middleware function that checks for JWT token in header
# All routes that have the @token_required decorator will be protected


def token_required(f):
    @wraps(f)
    def decorated(*args, **kwargs):
        token = None
        if "Authorization" in request.headers:
            token = request.headers["Authorization"].split(" ")[1]
        if not token:
            return {
                "error": "Unauthorized"
            }, 401
        try:
            # Get the user's email from the decoded token
            data = jwt.decode(
                token, config["APP_SECRET"], algorithms=["HS256"])

            # Retreive user's info from the database
```

```
        sql = f"select * from users where email='{data['email']}'"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.execute(stmt)
        current_user = ibm_db.fetch_assoc(stmt)

        # If user does not exist throw error.
        if current_user is None:
            return {
                "error": "Unauthorized"
            }, 401
    except Exception as e:
        return {
            "error": str(e)
        }, 500

    # Pass the authorized user in function args.
    return f(current_user, *args, **kwargs)

return decorated
```

GITHUB:

https://github.com/IBM-EPBL/IBM-Project-41633-1660643523

PROJECT DEMO :

https://drive.google.com/file/d/1EJMO52gjdtEwbQU1LONtQQdvHp9Lij7v/view?usp=drivesdk