SKILL/JOB RECOMMENDER APPLICATION

Domain : CLOUD

A PROJECT REPORT

Submitted by

THIRISHA. P   **(920819104047)**

THILAGAVATHY. V   **(920819104046)**

SIVAPRIYA. R   **(920819104038)**

INDHUMATHI. V   **(920819104015)**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

NPR COLLEGE OF ENGINEERING & TECHNOLOGY

NATHAM, DINDIGUL.

ANNA UNIVERSITY:: CHENNAI 600 025

TABLE OF CONTENTS

14. APPENDIX

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

This report claims that most college graduates have difficulty in choosing their domain in their job. Many engineers are trying to shift the domain from their field to IT. So, they are doing some courses in online and randomly searching for a job. Nowadays, IT fields are the targets of many students but they don't know which domain is fit for them. To avoid this situation candidates, need a Job recommendation that analyses the skills to recommend a suitable job for the candidate. The solution is to design a system that reads a resume and their skills. The resumes are going through pre-processing to make the design more efficient. For pre-processing top words and porter stemmer, PorterStemmer will make every word their root word, and stop words will remove every meaningless word. This makes the system more efficient. Useful for both resume and job description. Then compare the skills in the resume and description. For comparing, it uses the Cosine Similarity function and finds the scores of the resume for the respective jobs. Now it sorts the list in descending order with respect to their scores. Now, he got a hierarchical order of jobs from top to bottom. So, he can go with the first job or second which the skill he had already. He can be successful in that domain. The System not only shows the job but also recommends the skills to be improved for the job. Because of this, the candidate can train himself/herself for the future purpose and be a more achievable or talented person in his/her domain. The proposed work focuses on predicting the suitable jobs for the candidates.. This applicationcan be used by any candidates who need or who want to know about their suitable jobs and to improve themselves with both soft skills and hard skills. It will be helpful to them by not wasting their time searching for jobs. They can also grow their skills in their domain and grow faster in their domain.

1.2 PURPOSE

Presently recommendation frameworks are utilized to take care of the issue of the overwhelming amount of information in every domain and enable the clients to concentrate on information that is significant to their area of interest. One domain where such recommender systems can play a significant role to help college graduates to fulfill their dreams by recommending a job based on their skill set. Currently, there are plenty of websites that provide heaps of information regarding employment opportunities, but this task is extremely tedious for students as they need to go through large amounts of information to find the ideal job. And many students are not aware of which job is suitable for them. Nowadays, the IT fields are in a boom. Many engineering students are learning some technical skills by doing some courses but they don't know which skill is for which job. Simultaneously, existing job recommendation systems only take into consideration the domain in which the user is interested while ignoring their profile and skillset, which can help recommend jobs that are tailor-made for the user. This paper examines the user's resume then compares the knowledge of degree, soft skills, hard skills, and the projects he has done and then only the system recommends the jobs for that user. The system not only recommends the jobs but also shows the score of his/her resume for the respective job. Then, the system also recommends skills to improve the scores of their resume.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING PROBLEM:

The key issue is that asking to be recommended by someone can put that person's reputation at risk. If you end up pulling out, or using the job offer for leverage to get better pay elsewhere, or a promotion at your current job, you could harm their career.You might also harm your own reputation too — the person will feel used, and is unlikely to recommend you again. They might also warn others that you are unreliable. From the company's viewpoint, it will be expensive to start the hiring process again, or to replace you. People's time and efforts would have been wasted.

When you recommend someone for a role, you need to be fully confident the person can succeed in their task. Otherwise, you run the risk of creating a problem for the one who advertised the job. In addition to potentially damaging their business, you will also be harming your own reputation."In a sense, you are offering a guarantee which the person you've recommended may not fulfil, and that can cause others to doubt your claims in the future. It will reduce your credibility."There are other pitfalls, too. If you do get hired for the job, but fail to live up to expectations, both you and the person who recommended you may be looked at negatively.

2.2 REFRENCES:

➢ Naresh Kumar, Manish Gupta], and Isaac Ofori–**Technical Job RecommendationSystem Using APIs and Web Crawling**

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, Puppeteer and Representational State Transfer (REST) APIs for web crawling 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.

➢ Corno's de rujit and sandjai bhalai-**Job Recommender system: Reviews**

This paper provides a review of the job recommender system (JRS) literature published in the past decade (2011-2021). Compared to previous literature reviews, we put more emphasis on contributions that incorporate the temporal and reciprocal nature of job recommendations. Previous studies on JRS suggest that taking such views into account in the design of the JRS can lead to improved model performance. Also, it may lead to a more uniform distribution of candidates over a set of similar jobs. We also consider the literature from the perspective of algorithm fairness. Here we find that this is rarely discussed in

the literature, and if it is discussed, many authors wrongly assume that removing the discriminatory feature would be sufficient. With respect to the type of models used in JRS, authors frequently label their method as `hybrid'. Unfortunately, they thereby obscure what these methods entail. Using existing recommender taxonomies, we split this large class of hybrids into subcategories that are easier to analyse. We further find that data availability, and in particular the availability of click data, has a large impact on the choice of method and validation. Last, although the generalizability of JRS across different datasets is infrequently considered, results suggest that error scores may vary across these datasets.

> AJib susanto-**Recommendation of system for information technology job using collaborative littering method based on linkid in skills endorsement**

Students who are graduated from Informatics Engineering have wide employment opportunities in the information technology work field, such as database administrator, data scientist, UI designer, IT project manager, network engineer, system analyst, software engineer and UX designer. Each job in I nformation T echnology field has different skill requirement for the interest of work field. Therefore, IT skill classification is needed to find out the suitable career recommendation for Informatics Engineering students. Data from IT professionals which are obtained from LinkedIn account of IT professionals will be processed as reference for students. Data are ssprocessed using K-Means Clustering algorithm to find out how is feasible IT professionals data are used as a reference. Then, Collaborative Filtering method by the K-NN algorithm is used to determine classification based on the proximity between student skills and information technology job field. The output is recommendation of information technology job field which are generated from calculate of IT student skills. Result has been tested by testing one of user that has been labeled software engineer produce a recommendation output as a

5

software engineer.

- Amber rigarm ,Aakash roy ,Arpan saxena ,Hartan singh - **Job recommendation leveraging progression of job application**

    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 machine-learned recommendations and other sub-recommendations. For recommending jobs through machine learning that forms a significant part of our recommendation, we achieve the best results through Bi-LSTM with attention.

- Shaha T.,Al-Otaibi and Mourad Ykhlef-**A Survey of Job RecommenderApplication**

    The Internet-based recruiting platforms become a primary recruitmentchannel in most companies. While such platforms decrease the recruitment timeand advertisement cost, they suffer from an inappropriateness of traditionalinformation retrieval techniques like the Boolean search methods. Consequently, avast amount of candidates missed the opportunity of recruiting. The recommender system technology aims to help users in finding items that match their personnelinterests; it has a

successful usage in e-commerce applications to deal with problemsrelated to information overload efficiently. This article will present a survey of e-recruiting process and existing recommendation approaches for building personalizedrecommender systems for candidates/job matching.

S.No	TITLE	AUTHOR	TECHNOLOGY	ADVANTAGES	DISADVANTAGES
1	Technical Job Recommendation System Using APIs and Web Crawling	Naresh Kumar, Manish Gupta], and Isaac Ofori	• Hybrid Job recommender systems • Content based filtering algorithm	• Bidirectional recommendation • Adaptive system • Transition history is included • Use many attributes	➢ Binary representation only. ➢ Inefficient measures
2	Job Recommender system: Reviews	Corno's de rujit and sandjai bhalai	• Fairness algorithm • Knowledge based	• No ramp-up problem. • Intersection of machine learning and ethics.	• Need knowledge acquistion • No perfect measure
3	Recommendation of system for information technology job using collaborative littering method based on linkid in skills endorsement	AJib susanto	• Clustering : k mean Algorithm • Colatrative filtering –k-NN algorithm	• Transition history is included • Adaptive system	• No perfect measure • Binary representation only • Inefficient

				Blended approach provide significant improvement in job web portal	Using Bi-Ls7M has double LSTM cells, so it is costly
4	Job recommendation leveraging progression of job application	Amber rigarm, Aakash roy, Arpan saxena, Hartan singh	• Machine learning modules • Bilstm with attention	• Blended approach provide significant improvement in job web portal • This approach naturally solves the candidate and job cold start	• Using Bi-Ls7M has double LSTM cells, so it is costly • It is not fit for speech recognition.
5	A survey of job recommender systems		• Hybrid job recommender systems • Hybrid job recommender systems	• Bidirectional recommendation. • Relational aspects are included. • Adaptive system. • Use many attributes	• Binary representation only. • Less attributes used. • No perfect measures. • Key words search method. • One way recommendation

2.3 PROBLEM STATEMENT DEFINITION:
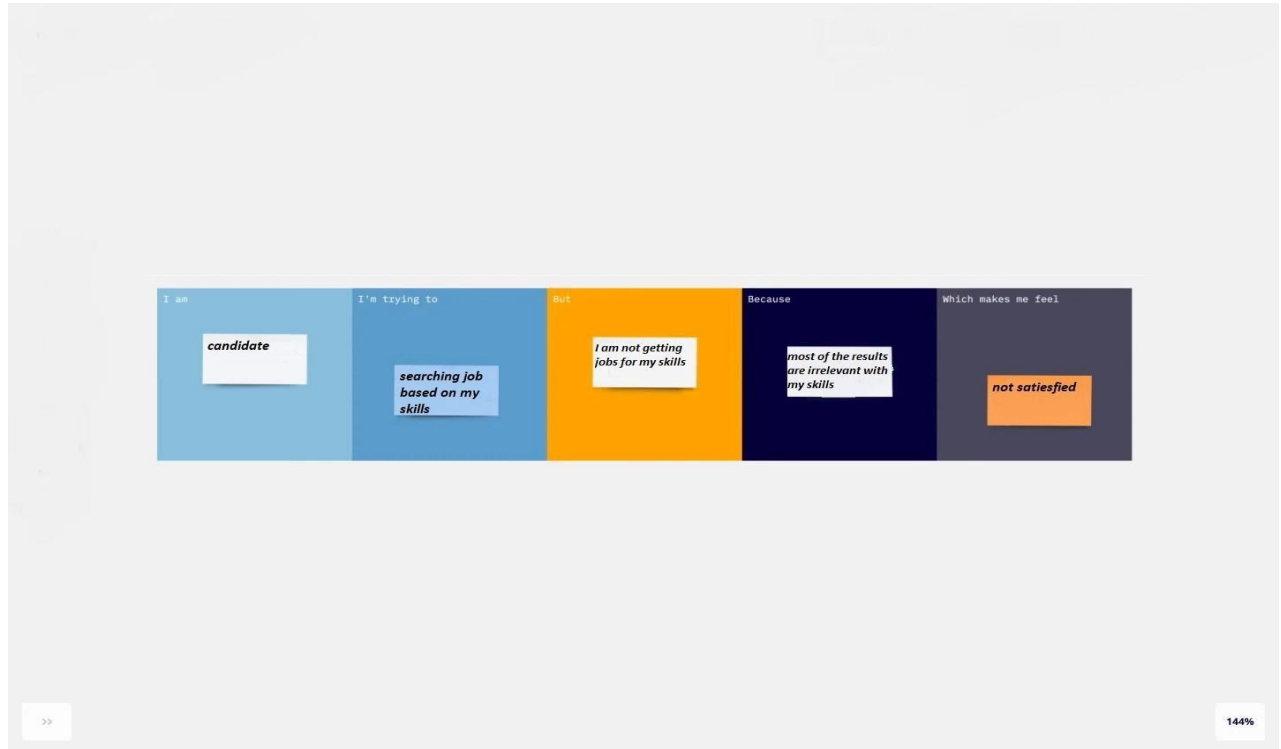
Problem Statement:

➢ Candidates who searches job for their skills doesn't get most relevant results.

➢ They will be unsatisfied with the results provided.

➢ Doesn't get a secure medium for displaying their skills.

➢ Data's provided in the profile may be misused, miscarried or can do unfair activity against the individuals.

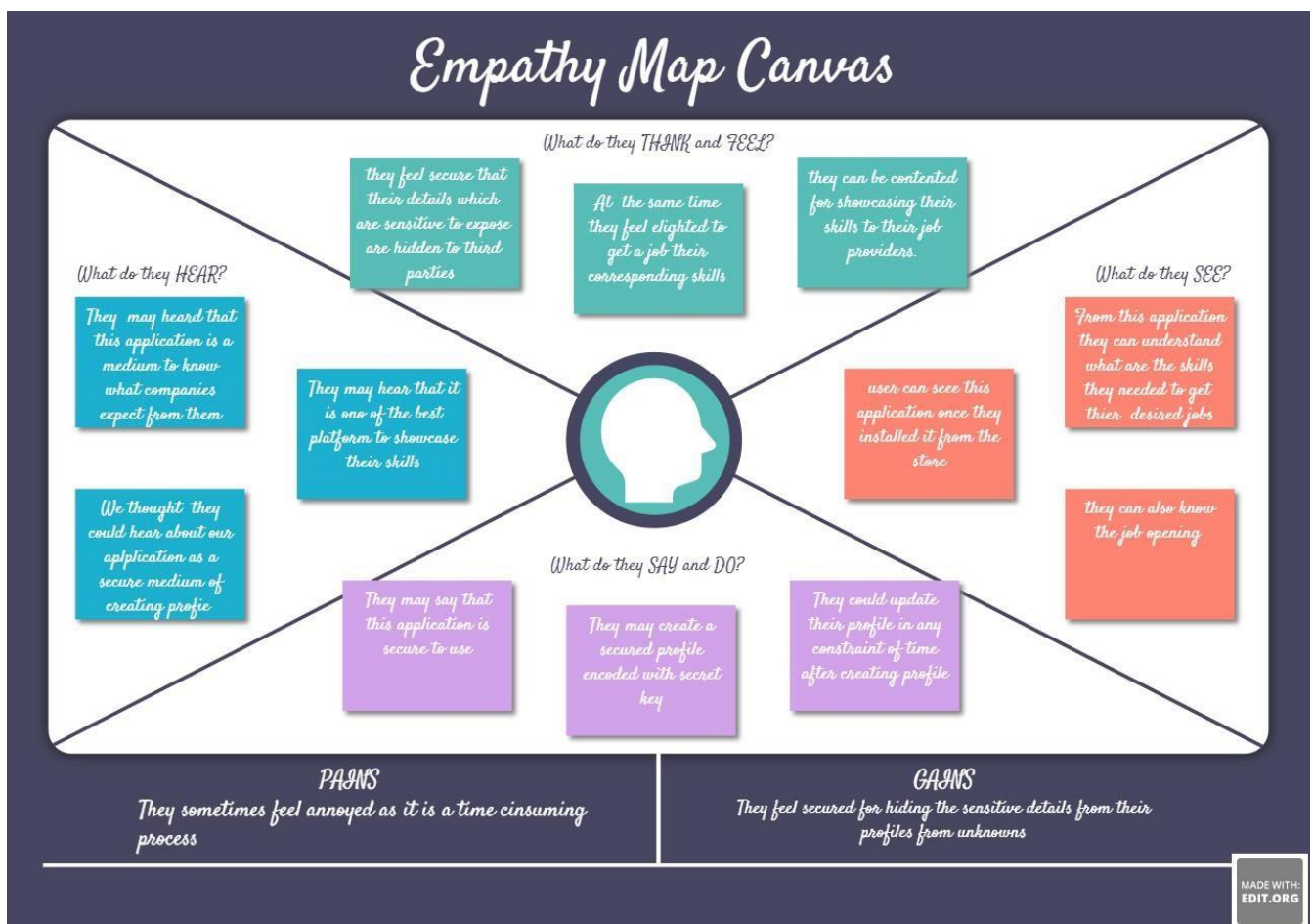➢ That details can be open to accessing of third parties.

CHAPTER 3
IDEATION PHASE

3.1 EMPATHY MAP CANVAS:

An **empathy map** is a collaborative visualization used to articulate what we know abouta particular type of user. It externalizes knowledge about users in order to

1) create a shared understanding of user needs, and

2) aid in decision making.

Explanation:

What do they think and feel?

➢ They feel secure that their detects that their details which are sensitive to expose are hidden to third parties

➢ At the same time they feel elighted to get a job for their corresponding skills.

➢ They can be contented for showcasing their skills to job providers.

What do they see?

➢ Users can see this application once they installed it from the store.

➢ From this application they can understand what are the skills they needed to get their desired jobs

➢ They can also know the job openings

What do they say and do?

➢ Application is secure to use.

➢ Secure their profile with secret key.

➢ Option for updating their profile

What do they hear?

➢ This might be medium to know what companies expect from them.

➢ They may hear that it is one of the best platform to showcase their skills

➢ They may hear about application as a secure medium of creating profile.

3.2 IDEATION AND BRAINSTROMING:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement

Step-2: Brainstorm, Idea Listing and Grouping

2 Brainstorm

Write down any ideas that come to mind that address your problem statement.

⏱ 10 minutes

Person 1

Increasing throughput of existing system	content based filtering can be used
user can provide access to one who want to access their details	collaborative filtering can be used to provide better search result

Person 2

provide related content base on the user responses	give searching prior to users skills
kNN which is a machine learning algorithm to find clusters of similar users can be used to improve the performance	we can implement two factor authentication

Person 3

collaborative filtering can be used to provide better search result	we can use cryptographic algorithm like RSA to prevent the unknown access
we can also use some of the machine learning modules to improve response	update and notify the user interested jobs along with their skills

Person 4

user should able to search for one or more parameters they wish do	use steganography image and text hiding to ensure security
content based filtering can be used to result users interested jobs results	user can able to visit their desired job providers profile

3 Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

⏱ 20 minutes

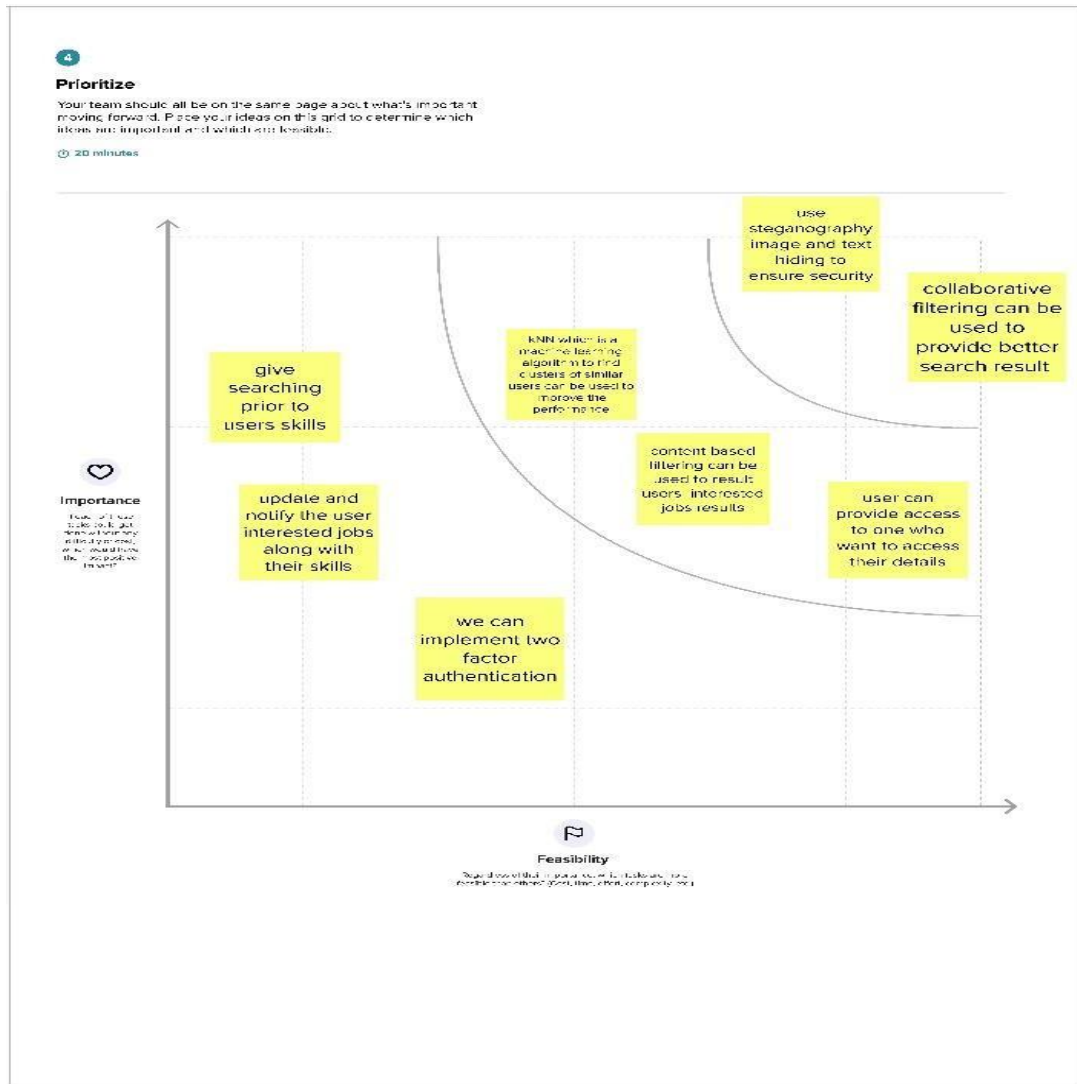|	
use steganography image and text hiding to ensure security	update and notify the user interested jobs along with their skills
collaborative filtering can be used to provide better search result	kNN which is a machine learning algorithm to find clusters of similar users can be used to improve the performance
give searching prior to users skills	user can provide access to one who want to access their details

Step-3: Idea Prioritization

3.3 PROPOSED SOLUTION:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Job portals that provide results are irrelevant with the skills acquired by the candidate and the profile they post in such application are open to access to everyone they may misuse the datas provided by the candidate
2.	Idea / Solution description	To increase the throughput using algorithms like k means and KNN and increase the security by using steganographic techniques of hiding datas.
3.	Novelty / Uniqueness	➢ Students get notified about the jobs availability. ➢ They can get a job based on their specific acquired skills. ➢ They can secure their information using their secret keys. ➢ They can able to know about one who accessing their profile.
4.	Social Impact / Customer Satisfaction	Using this application people can easily find their domain based jobs from top MNC's using AI technology that suits their skill set.
5.	Business Model (Revenue Model)	Revenue can be generated by giving ads in this application and from affiliate commission.
6.	Scalability of the Solution	Skill /job recommender system will provide job recommendations to any kind of skill set.It never fail to show recommendations.

3.4 PROBLEM SOLUTION FIT:

Problem-Solution Fit canvas

Purpose / Vision

Versions

1. CUSTOMER SEGMENT(S) — CS

students and graduates from different course and skills

6. CUSTOMER LIMITATIONS EG. BUDGET, DEVICES — CL

low budget , smart phones/desktop

5. AVAILABLE SOLUTIONS PLUSES & MINUSES — AS

PLUS : provide platform to search for job openings
MINUS: But their profiles can be accessed by third parties.

2. PROBLEMS / PAINS + ITS FREQUENCY — PR

As the profile created are open to access sometimes attackers misuse the delicate information in their corresponding profile

9. PROBLEM ROOT / CAUSE — RC

User may think this as a time consuming event as it takes more time than usual job recommendation system

7. BEHAVIOR - ITS INTENSITY — BE

User can create their profile more secure by hiding their details which do they wish that can be done by using secret key

3. TRIGGERS TO ACT — TR

misusing of their mobile numbers and other related details can get them to triggering

4. EMOTIONS BEFORE / AFTER — EM

BEFORE: Insecure
AFTER :reliable and feel free

10. YOUR SOLUTION — SL

In existing system ,the user can searches for their interested jobs and providers can searches for their needed skilled candidates but their profiles are open to everyone .they may do unethical activity with their details, so we have proposing a job recommending system with creating a secured profile option

8. CHANNELS of BEHAVIOR — CH

ONLINE: Extract channels from the behavior block

OFFLINE : Extract channels from the behavior block and use of customer.

IdeaHackers.nl

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

Functional requirements may involve calculations, technical details, data manipulation and processing, and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describe all the cases where the system uses the functional requirements, these are captured in use cases.

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 Email.
FR-2	User Confirmation	Confirmation via Email as OTP
FR-3	User Login	Login through Email Sign in through username and password
FR-4	Setting up User Profile	Complete user profile by providing personal details Upload resume and certificates Update the Skills and Experience
FR-5	Searching jobs	Search jobs based on the skillset, experience and qualification
FR-6	Job Recommendation	Recommending Job by querying datafrom DB2 Database

4.2 NON FUNCTIONAL REQUIREMENTS:

Non-Functional Requirements are the constraints or the requirements imposed on the system. They specify the quality attribute of the software. Non-Functional Requirements deal with issues like scalability, maintainability, performance, portability, security, reliability, and many more.

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

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The candidates who use the system should be students or skilled graduates
NFR-2	**security**	The users can secure their profile using steganography. The security on their databases may include firewalls to prevent unauthorized access
NFR-3	**Performance**	Applicants can access their resume 98% of the time without failure.
NFR-4	**Availability**	Employers can post jobs on the website throughout the weeks at any time during the week
NFR-5	**Scalability**	It is the ability to appropriately handle increasing Workloads without performance degradation or its ability to quickly enlarge

CHAPTER 5

PROJECT DESIGN

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.
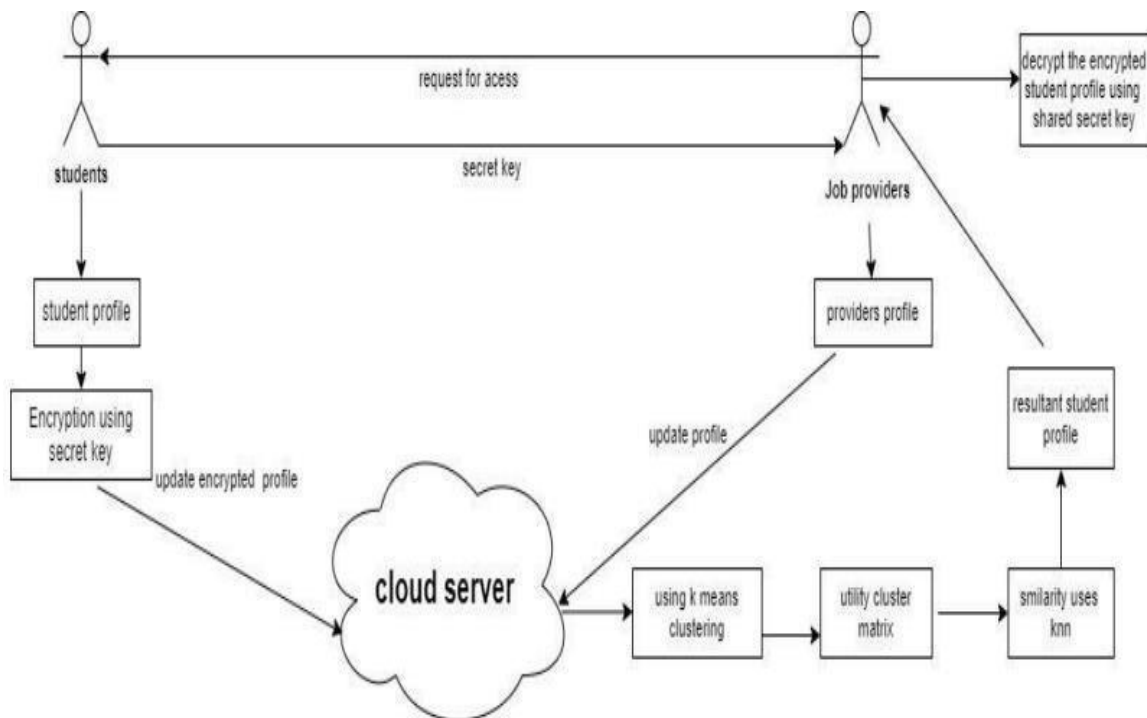
5.2 SOLUTION AND TECHNICAL ARCHITECTURE:

SOLUTION ARCHIYECTURE:

Solution architecture is a complex process – with many sub-processes – that bridges thegap 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 softwareto project stakeholders.

- Define features, development phases, and solution requirements.

- Provide specifications according to which the solution is defined, managed, anddelivered.
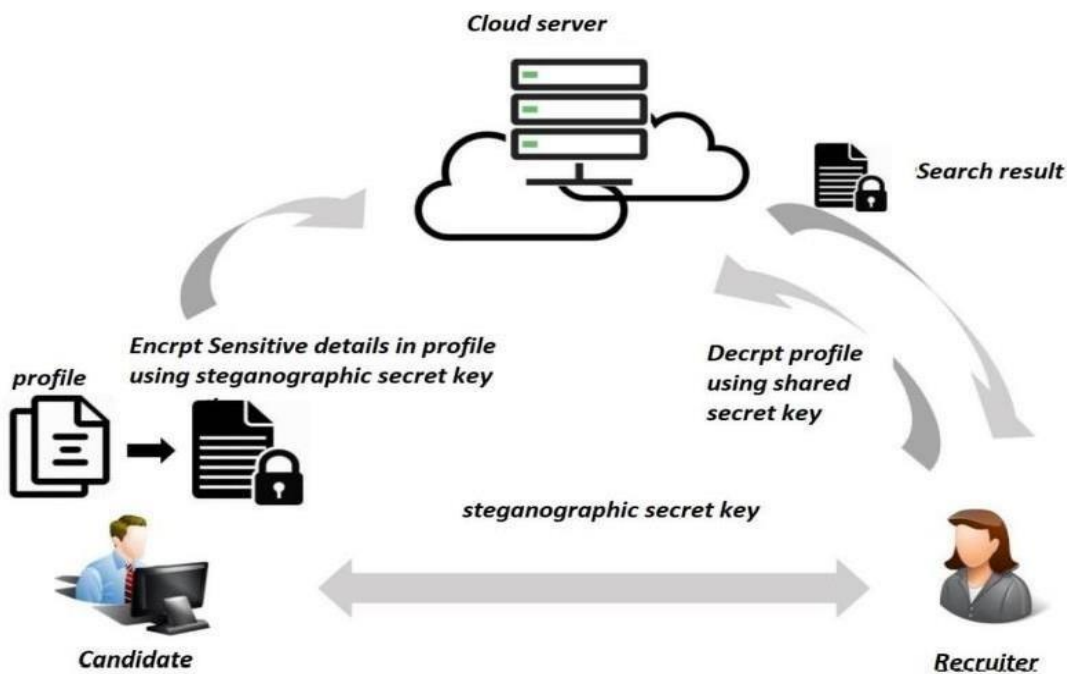
TECHNICAL ARCHITECTURE:

Technology architecture deals with the deployment of application components on technology components. A standard set of predefined technology components is provided in order to represent servers, network, workstations, and so on.
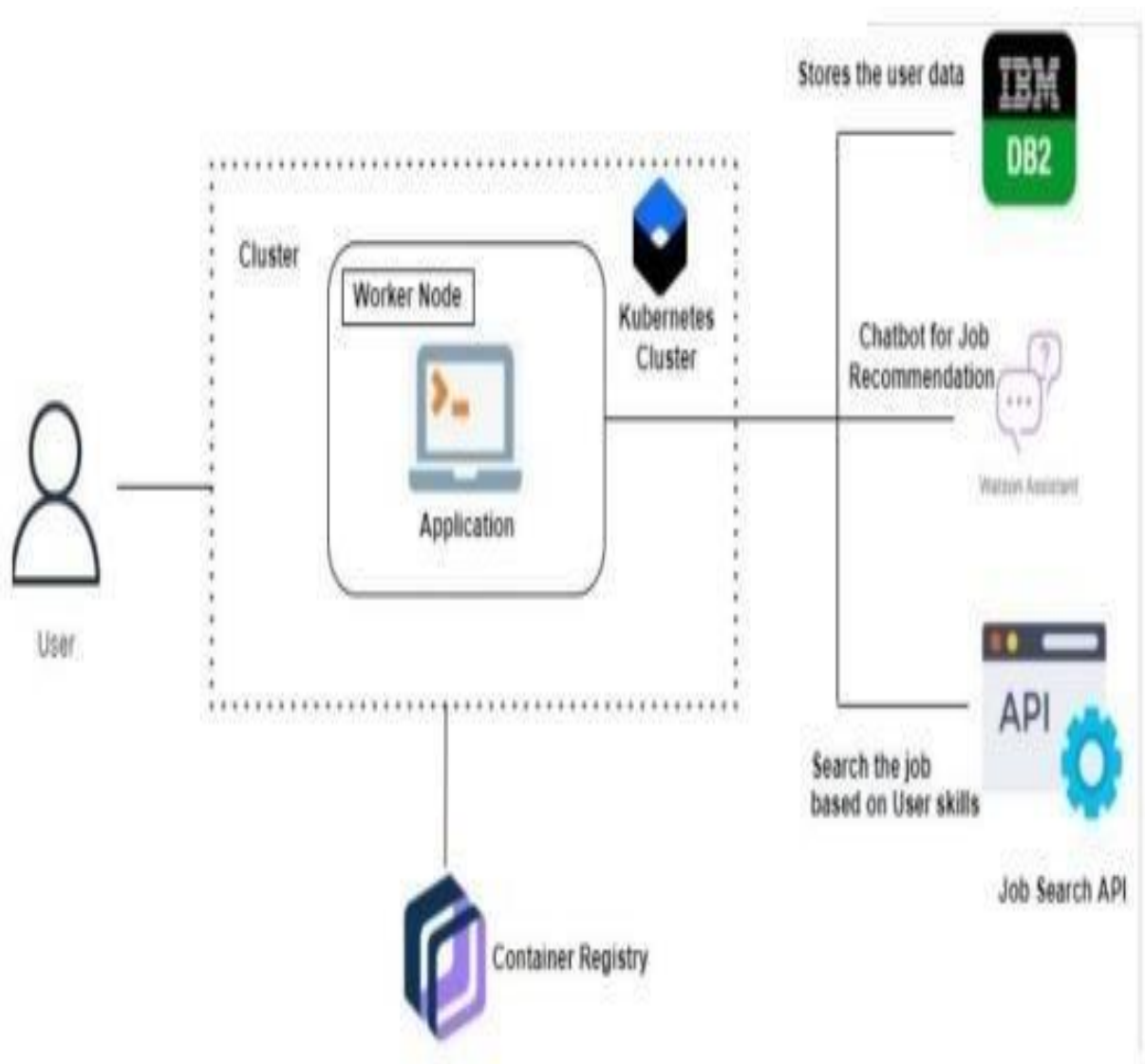
5.3 USER STORIES:

User Type	Functional Requirement (Epic)	User Story Number	User Story /Task	Acceptance criteria	Priority	Release
Customer (Student/ Graduate and job providers)	User Registration	USN-1	As a user, I need to register through registration form through Email	I should register before they access the resource.	High	Sprint-1
	User Confirmation	USN-2	As a user, I should get confirmation through email OTP	I can receive that Confirmation for registeration	High	Sprint-1
	User Login	USN-3	As a user, I want to login using username and password as my wish .	Once I registered I can login through the registered Email	High	Sprint-1
	Setting up User Profile	USN-4	As a user, I should create a profile with resumes, certificates and acquired skills and experience.	I should create a Profile that specifies my skills and experiences	Medium	Sprint-2
	Searching jobs	UN-5	As a user, I will be searching for a specified job base on my skills , experience and qualification .	I can searches for the jobs relevant to my skills and qualification.	High	Sprint-1

	Job Recommendation	USN-6	Recommending Jobs by querying data from DB2 Database	IBM cloud storage will helps for recommending jobs	High	Sprint-1
Cloud Storage	IBM Cloud	USN-7	As a user, I willbe in need to get all such information on time and in a fastest manner	IBM cloud will be helpful for this	High	Sprint-1

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Candidate	USN-1	The candidate can searches for their desired jobs based on their owned and acquired skills.	20	High	V.Thilagavathy R.Sivapriya P.Thirisha V.Indhumathi
Sprint-2	Job providers	USN-2	The role of the job providers is to searches for the deserved candidate with the knowledge and skills required for their companies.	20	High	V.Thilagavathy R.Sivapriya P.Thirisha V.Indhumathi
Sprint-3	Chatbot	USN-3	The users can directly talk with the chatbot regarding the availability of candidates and jobs. Get the recommendation based on information provided by the users.	20	High	V.Thilagavathy R.Sivapriya P.Thirisha V.Indhumathi
Sprint-4	Final delivery	USN-4	Container of the application using docker,Kubernetes and deployment of the application.create the documentation and finally submit the application.	20	High	V.Thilagavathy R.Sivapriya P.Thirisha V.Indhumathi

6.2 SPRINT DELIVERY SCHEDULE:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planne)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	26 Oct 2022	31 Oct 2022	20	31 Oct 2022
Sprint-2	20	6 Days	02 Nov 2022	07 Nov 2022	20	07 Nov 2022
Sprint-3	20	6 Days	09 Nov 2022	14 Nov 2022	20	14 Nov 2022
Sprint-4	20	6 Days	15 Nov 2022	20 Nov 2022	20	20 Nov 2022

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)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

Burndown Chart:
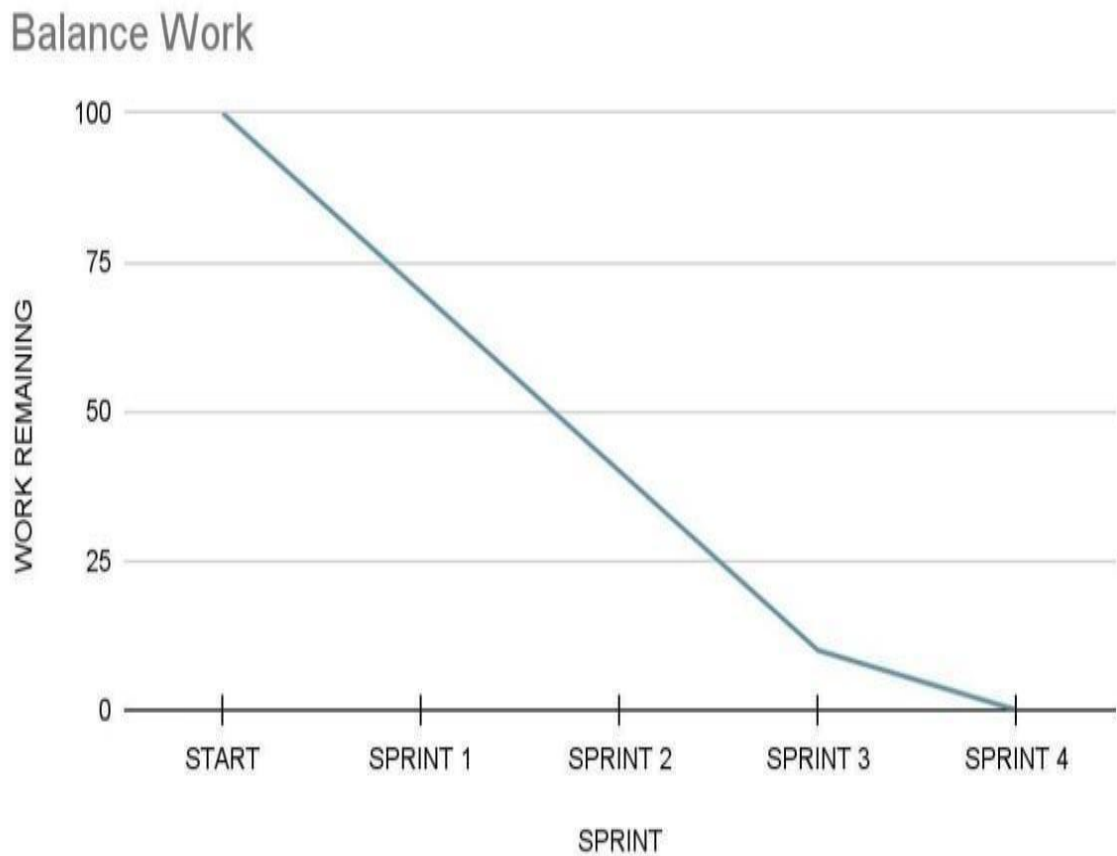
A burn down chart is a graphical representation of work left to do versus time. It is often used in agile <u>software development</u> methodologies such as <u>Scrum</u>. However, burn down charts can be applied to any project containing measurable progress over

6.3 REPORTS FROM JIRA:
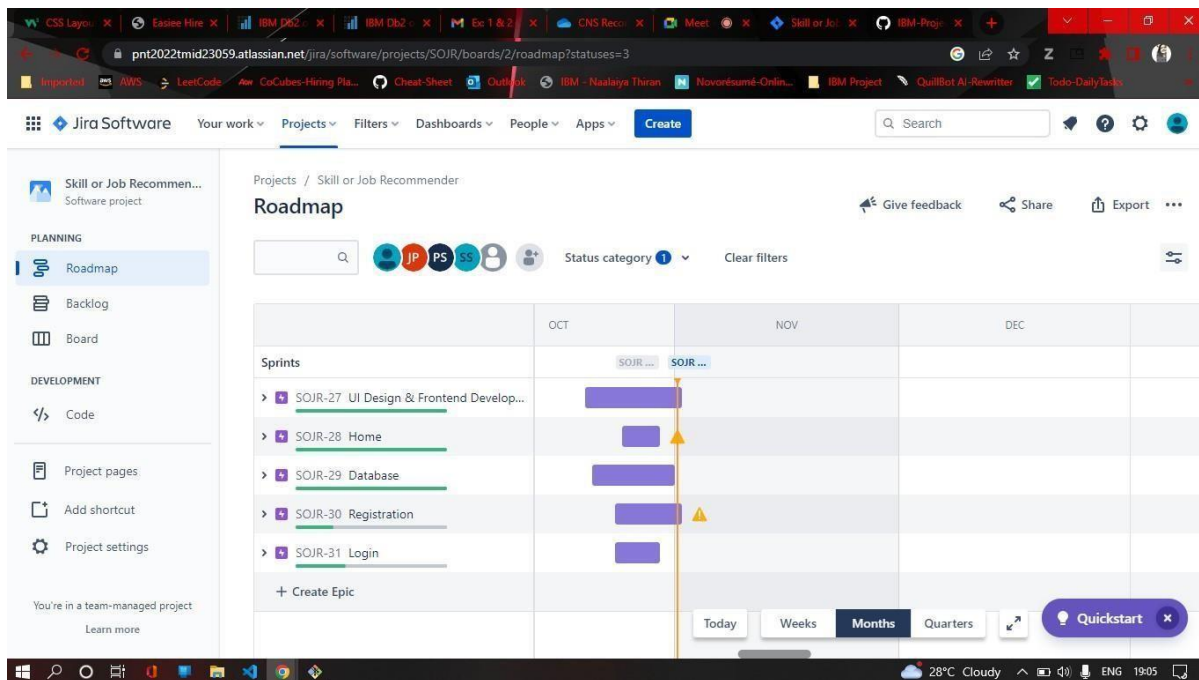
CHAPTER 7

CODING AND SOLUTIONING

7.1 FEATURE 1:

Creating IBM Watson assistant

Creating db2 connection:

Kubernetes:
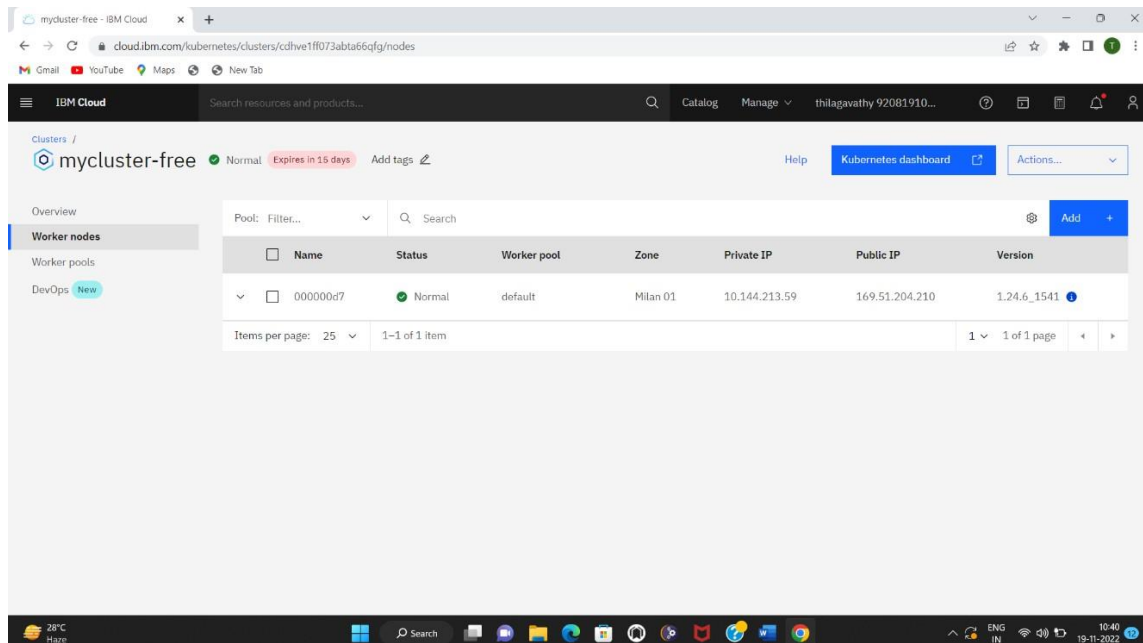
7.2 FEATURE 2:

App Backend:
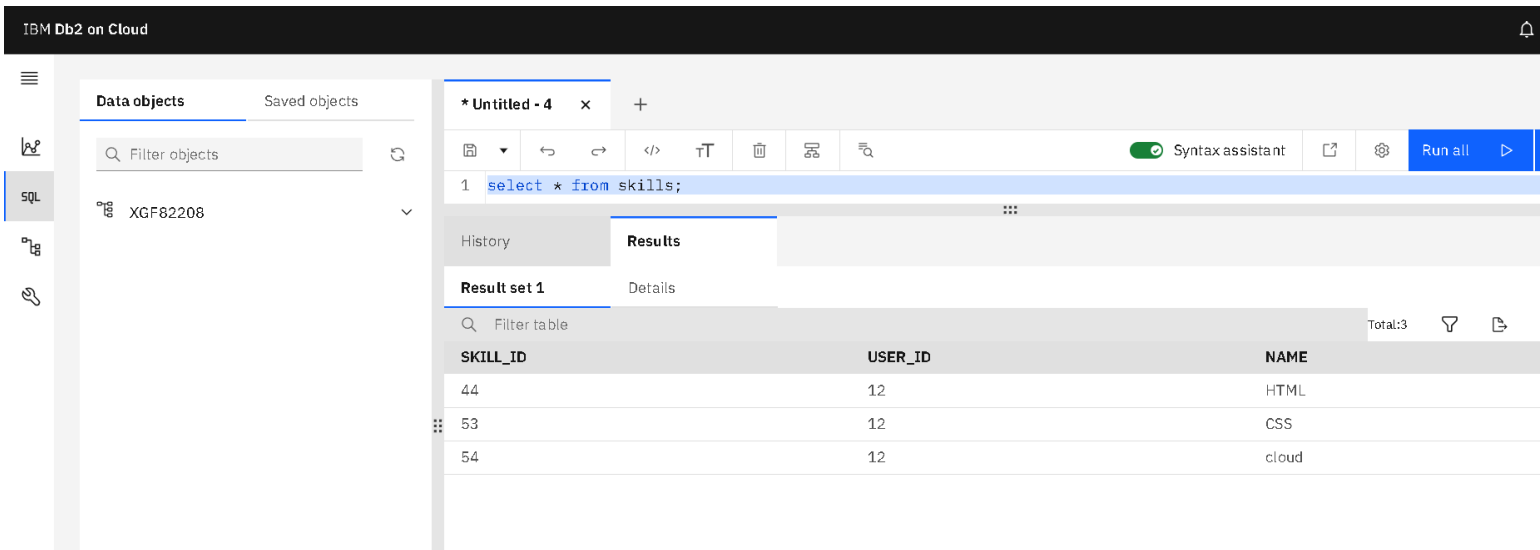
This App   is used to connect with the db2 from there all the information regarding the registration for every users are stored .
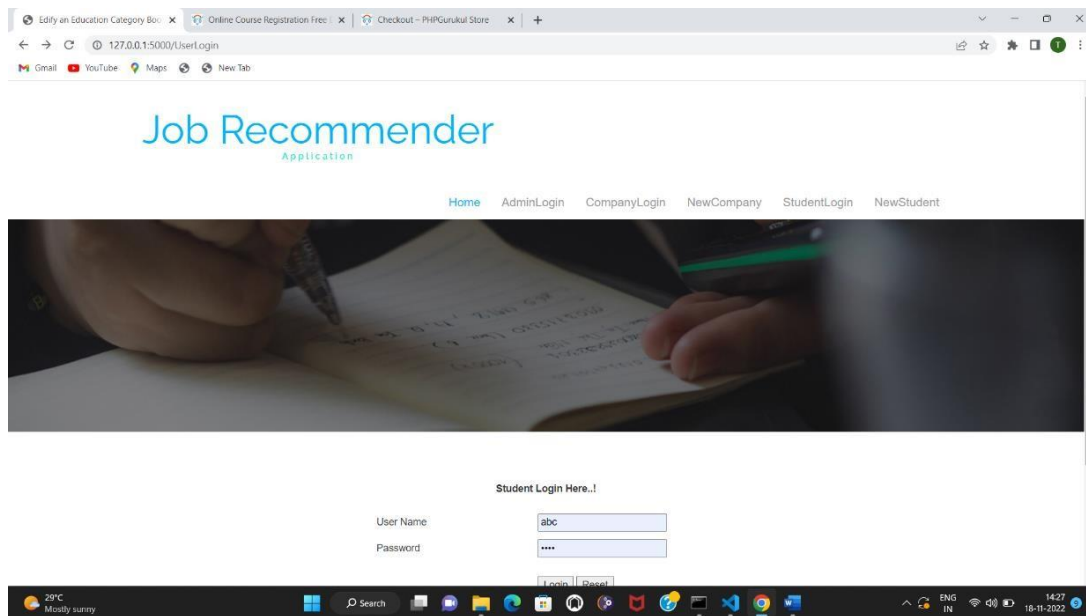
CHAPTER 8

TESTING

8.1 TEST CASES:

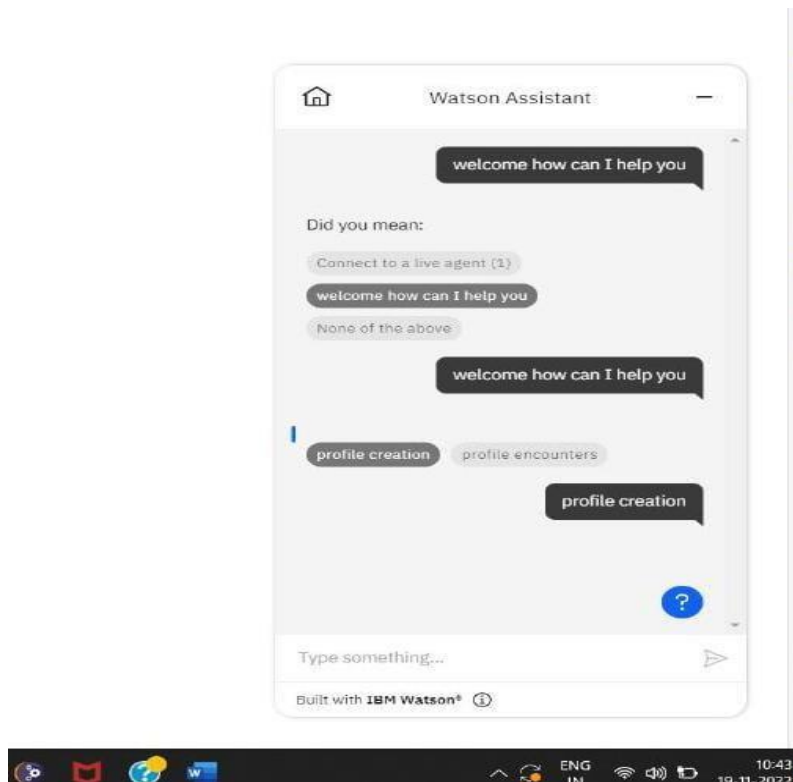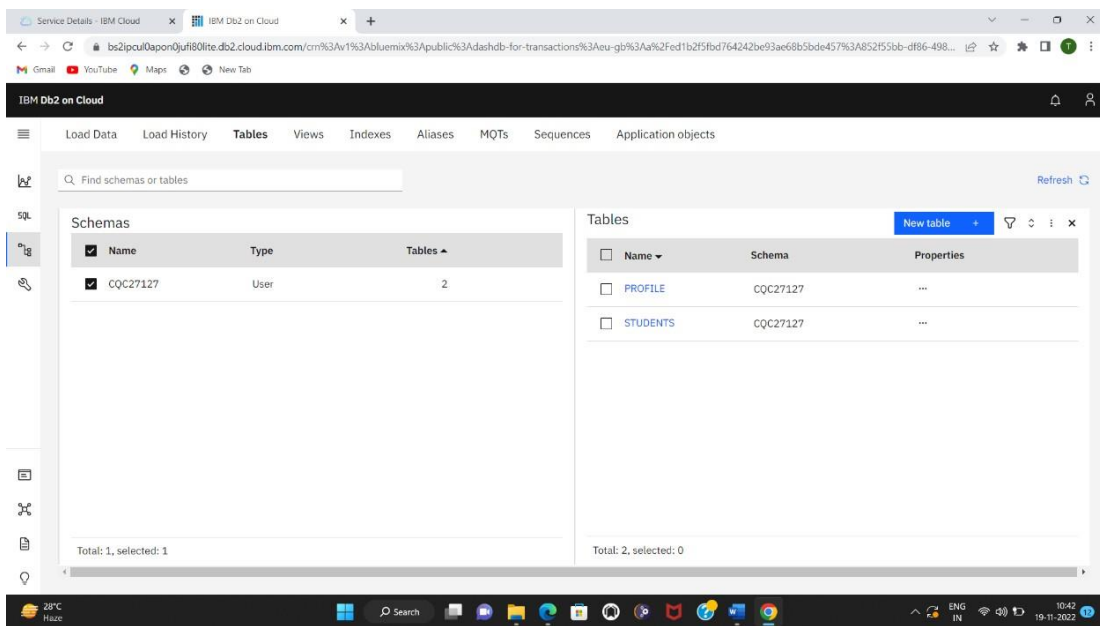| TEST MODULE	TEST CASE	EXPECTED RESULT	TEST RESULT
ADMIN	Provide valid logincredentials	User successfully logged in anddirected to the admin dashboard page	PASS
ADMIN	Enters invalid logincredentials	Displays Error message	PASS
ADMIN	Upon successful login, click on the 'List ofEmployers' tab.	Displays the details of list of active employers registered withthe application	PASS
ADMIN	Click on 'Active/Deactivate' tabunder status of the employer	The status of the employer will bechanged to active/deactivate.	PASS
EMPLOYER	Provide details forregistration	Employer successfully registeredwith the application	PASS
EMPLOYER	Upon successful login, click on 'Post New Job'tab	Employer posts jobs with therequired details	PASS
EMPLOYER	Employer trying to postjob with insufficient details	Prompts to fill in all the necessarydetails of the job	PASS
EMPLOYER	Employer clicks on the'List Posted Jobs' tab	All the jobs posted by the employer will be displayed.	PASS

EMPLOYER	Employer clicks on 'Active/deactivate' under Status	The status of the job posting willchanged to active/deactivated.	PASS
EMPLOYER	Employer clicks on the 'view' tab under candidates column	The list of the details of applicants for a particular jobposting are displayed.	PASS
JOBSEEKER	Provide details forregistration	Jobseeker successfully registeredwith the application	PASS
JOBSEEKER	Enters invalid logincredentials	Error message displayed	PASS
JOBSEEKER	Upon successful login, click on 'My Profile' tab	List details of jobseeker	PASS
JOBSEEKER	Upon successful login, click on 'Search Jobs' tab	Details of the active job postingsare displayed.	PASS
JOBSEEKER	Upon successful login, click on 'Applied Jobs' tab	Details of the jobs that are appliedby the jobseeker are displayed	PASS
JOBSEEKER	Click on 'Add Review' tab	Displays a form to fill in the review details of the company	PASS
JOBSEEKER	Logout	Redirects to the Home page of theapplication	PASS

8.2 USER ACCEPTANCE TESTING:

Purpose of Document

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

Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	8	4	2	1	15
Duplicate	1	0	1	0	2
External	3	2	0	0	5
Fixed	5	1	2	20	28
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	0	1
Won't Fix	0	0	0	1	1
Totals	17	7	7	22	53

TEST CASE ANALYSIS:

This report shows the number of test cases that have passed, failed, and untested.

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	0	5
Client Application	45	0	0	45
Security	1	0	0	1
Outsource Shipping	3	0	0	3
Exception Reporting	6	0	0	6
Final Report Output	5	0	0	5
Version Control	3	0	0	3

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS
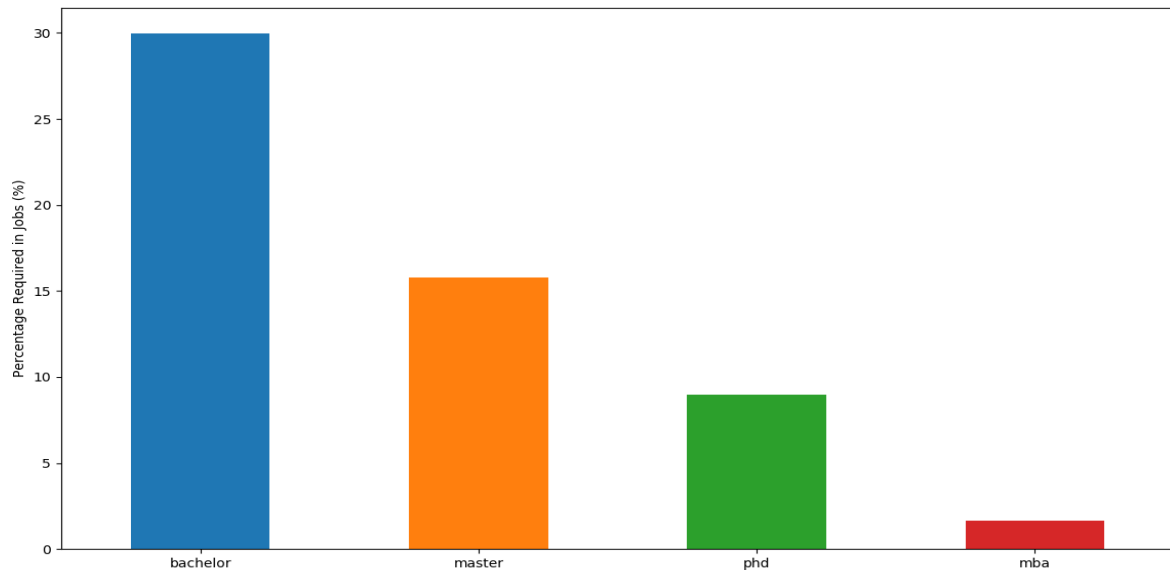
PARAMETERS	PERCENTAGE REQUIRED IN JOBS
BACHELOR	Bachelor requirement will be more.
MASTER	Master requirement will be less than Bachelor
PHD	PHD requirement will be less than Master
MBA	MBA requirement will be less than PHD

The results of job recommender system which are run in the python environment are presented. Recommender system is provided with the profile vector of user and item to generate a set of item as a recommendation to user in question. Profile vector of a user holds the information regarding the preferences of user on IT skill and domain specific information. Where as, profile vector of item holds the information regarding the skills and job domain that is required for that job. Also as defined in section 1.4, the purpose of this research was not only model a job recommender system using user's skill set and Job domain but also to address issue of cold start. To complete the research, we had set ourselves a objectives which ease our way in completion of the research.

As to conduct the research, the need for two different dataset, which will be used create a user vector and item vector was required. For the purpose of user data we had choose the data from stack overflow survey; Without the Job data set, we wouldn't be able to continue this research.

Further in this project, before starting with implementation, we had to perform the data preprocessing on the both user dataset from stack overflow and the job listing dataset , scraped from the job board.The main objective was to create matrix of user and item which describes the user and item about their implicit attribute. we were supposed to address the cold start issue of the recommender system. There could be no interaction between the user and job data set chosen for the study, With the minimal interaction with between user and job data made it hard to collaborative filtering as it requires previous interaction. When the user's and Jobs doesn't have any predefined relation it also mean they are new to the system. So we decided to perform data preprocessing on data to transform it into user and job profile vector which describes attributes of user and job. These profile vector and use of content based filtering in this Recsys allowed us to avoid the issue of cold start from both new users and new jobs perspective.

CHAPTER 10

ADVANTAGES & DISADVANTAGES

➢ ADVANTAGES:

- Bidirectional recommendation.
- Relational aspects are included.
- Adaptive system.
- Use many attributes.
- Includes many attributes.
- Relational aspects are included.
- Qualitative and quantity representation (proficiency level for skills is included)
- Key words search method.
- Search results are more relevant with the content searched by the user.
- Students get notified about the job openings of their interested companies.

➢ DISAVANTAGES:

- Knowledge acquisition and Knowledge engineering problems.
- Scalability, ramp-up, and data sparsity problems.
- No perfect measures.
- Inefficient measures.
- It have high possible rate of exposing of candidates delicate information.

CHAPTER 11

CONCLUSION

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. We 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. We conclude that the field of job recommendations is still unripe and require further improvements.

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 ibm cloud 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%.

CHAPTER 12

FUTURE SCOPE

The recommendation system works on the content-based filtering using word embedding of word2vec and similarity measure of cosine similarity. As the corpus provides general information about the word and similar words around it, It is possible to create a better recommendation by creating a corpus related to the IT skills, terminology, Job domain and jargon of the industry. By using such corpus specific to the hiring domain, the recommendation could be better when analyzing implicit text data in the job description. It can be categorized in a better way. As this Recsys is currently working on data that has no interaction, a study needs to be conducted on the data that has previous interaction in the hiring domain. This would allow us to dynamically keep recommending new jobs based on user's change in preferences. There is a recommender system in the hiring domain from LinkedIn but not in the perspective of a job seeker but from the perspective of a recruiter. Similarly, we could conduct a study based on LinkedIn data to recommend jobs using content-based filtering. As conditions change from domain to domain, it is not a good idea to recommend a job because a user liked it; instead, the recommendation has to be considered, if the profile of a user matches the requirement. So, conducting more study based on content-based filtering ensemble with other filtering technique in hiring domain in the perspective of a job seeker can be considered as a part of future work.

APPENDIX

A1 - SOURCE CODE

app.py:

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

import json
from json2html import *



import ibm_db
import pandas
import ibm_db_dbi
from sqlalchemy import create_engine

engine = create_engine('sqlite://',
                echo = False)

dsn_hostname = "3883e7e4-18f5-4afe-be8c-
fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud"
dsn_uid = "cqc27127"
dsn_pwd = "2d8IfxYcI85CayCc"

dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "BLUDB"
dsn_port = "31498"
dsn_protocol = "TCPIP"
dsn_security = "SSL"

dsn = (
    "DRIVER={0};"
    "DATABASE={1};"
    "HOSTNAME={2};"
    "PORT={3};"
    "PROTOCOL={4};"
    "UID={5};"
    "PWD={6};"
    "SECURITY={7};").format(dsn_driver, dsn_database, dsn_hostname, dsn_port,
dsn_protocol, dsn_uid, dsn_pwd,dsn_security)
```

```python
try:
    conn = ibm_db.connect(dsn, "", "")
    print ("Connected to database: ", dsn_database, "as user: ", dsn_uid, "on host: ",
dsn_hostname)

except:
    print ("Unable to connect: ", ibm_db.conn_errormsg() )

app = Flask(_name_)
app.config['DEBUG']
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'




@app.route("/")
def homepage():

    return render_template('index.html')




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

@app.route("/AdminLogin")
def AdminLogin():
    return render_template('AdminLogin.html')

@app.route("/NewUser")
def NewUser():
    return render_template('NewUser.html')

@app.route("/NewCompany")
def NewCompany():
    return render_template('NewCompany.html')

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

```python
def StudentLogin():
    return render_template('UserLogin.html')

@app.route("/CompanyLogin")
def CompanyLogin():
    return render_template('CompanyLogin.html')

@app.route("/Search")
def Search():
    return render_template('Search.html')

@app.route("/AdminHome")
def AdminHome():

    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * from regtb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()
    return render_template('AdminHome.html', data=data)

@app.route("/ACompanyInfo")
def ACompanyInfo():

    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * from companytb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()


    return render_template('ACompanyInfo.html', data=data)
```

```python
@app.route("/AjobInfo")
def AjobInfo():


    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * from jobtb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()

    return render_template('AjobInfo.html', data=data)



@app.route("/SCompanyInfo")
def SCompanyInfo():


    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * from jobtb "
    dataframe = pandas.read_sql(selectQuery, pd_conn)

    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()

    return render_template('SCompanyInfo.html', data=data)

@app.route("/CompanyHome")
def CompanyHome():
    return render_template('CompanyHome.html')



@app.route("/UserHome")
def UserHome():

    uname=  session['uname']

    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM regtb where Username='"+ uname +"' "
```

44

```python
        dataframe = pandas.read_sql(selectQuery, pd_conn)
        dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
        data = engine.execute("SELECT * FROM Employee_Data").fetchall()


        return render_template('UserHome.html', data=data)




@app.route("/CJobInfo")
def CJobInfo():


    cname=  session['cname']


    conn = ibm_db.connect(dsn, "", "")
    pd_conn = ibm_db_dbi.Connection(conn)
    selectQuery = "SELECT * FROM jobtb where Cname='"+ cname +"' "
    dataframe = pandas.read_sql(selectQuery, pd_conn)
    dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
    data = engine.execute("SELECT * FROM Employee_Data").fetchall()




    return render_template('CJobInfo.html', data=data)


@app.route("/adminlogin", methods=['GET', 'POST'])
def adminlogin():
    error = None
    if request.method == 'POST':
        if request.form['uname'] == 'admin' or request.form['password'] == 'admin':



            conn = ibm_db.connect(dsn, "", "")
            pd_conn = ibm_db_dbi.Connection(conn)
            selectQuery = "SELECT * FROM regtb  "
            dataframe = pandas.read_sql(selectQuery, pd_conn)
            dataframe.to_sql('Employee_Data', con=engine, if_exists='append')
            data = engine.execute("SELECT * FROM Employee_Data").fetchall()
```

```python
            return render_template('AdminHome.html', data=data)

    else:
        return render_template('index.html', error=error)




@app.route("/userlogin", methods=['GET', 'POST'])
def userlogin():
    error = None
    if request.method == 'POST':

        username = request.form['uname']
        password = request.form['password']


        conn = ibm_db.connect(dsn, "", "")

        pd_conn = ibm_db_dbi.Connection(conn)

        selectQuery = "SELECT * from regtb where UserName='" + username + "' and
password='" + password + "'"
        dataframe = pandas.read_sql(selectQuery, pd_conn)

        if dataframe.empty:
            data1 = 'Username or Password is wrong'
            return render_template('goback.html', data=data1)
        else:
            print("Login")
            selectQuery = "SELECT * from regtb where UserName='" + username + "' and
password='" + password + "'"
            dataframe = pandas.read_sql(selectQuery, pd_conn)

            dataframe.to_sql('Employee_Data',
                    con=engine,
                    if_exists='append')

            # run a sql query
            print(engine.execute("SELECT * FROM Employee_Data").fetchall())

            return render_template('UserHome.html', data=engine.execute("SELECT * FROM
Employee_Data").fetchall())
```

```python
@app.route("/companylogin", methods=['GET', 'POST'])
def companylogin():
    error = None
    if request.method == 'POST':

        uname = request.form['uname']
        password = request.form['password']
        session['cname'] = uname


        conn = ibm_db.connect(dsn, "", "")

        pd_conn = ibm_db_dbi.Connection(conn)

        selectQuery = "SELECT * from companytb where UserName='" + uname + "' and
password='" + password + "'"
        dataframe = pandas.read_sql(selectQuery, pd_conn)

        if dataframe.empty:
            data1 = 'Username or Password is wrong'
            return render_template('goback.html', data=data1)
        else:
            print("Login")
            selectQuery = "SELECT * from companytb where UserName='" + uname + "' and
password='" + password + "'"
            dataframe = pandas.read_sql(selectQuery, pd_conn)

            dataframe.to_sql('Employee_Data',
                     con=engine,
                     if_exists='append')

            # run a sql query
            print(engine.execute("SELECT * FROM Employee_Data").fetchall())

            return render_template('CompanyHome.html', data=engine.execute("SELECT *
FROM Employee_Data").fetchall())

@app.route("/NewStudent1", methods=['GET', 'POST'])
def NewStudent1():
    if request.method == 'POST':

        name = request.form['name']
        gender = request.form['gender']
```

```python
        Age = request.form['Age']
        email = request.form['email']
        pnumber = request.form['pnumber']
        address = request.form['address']
        Degree = request.form['Degree']
        depat =  request.form['depat']
        uname = request.form['uname']
        passw = request.form['passw']

    conn = ibm_db.connect(dsn, "", "")

    insertQuery = "insert into regtb values('" + name + "','" + gender + "','" + Age + "','" +
email + "','" + pnumber + "','" + address + "','" + Degree + "','" + depat + "','" + uname +
"','" + passw + "')"
    insert_table = ibm_db.exec_immediate(conn, insertQuery)

    sendmsg(email, "Successfully registered this website")

    data1 = 'Record Saved!'
    return render_template('goback.html', data=data1)




@app.route("/newcompany", methods=['GET', 'POST'])
def newcompany():
    if request.method == 'POST':

        cname = request.form['cname']
        regno = request.form['regno']
        mobile = request.form['mobile']

        email = request.form['email']
        Website = request.form['Website']
        address = request.form['address']
        uname = request.form['uname']
        passw = request.form['passw']

        conn = ibm_db.connect(dsn, "", "")

        insertQuery = "insert into companytb
values('"+cname+"','"+regno+"','"+mobile+"','"+email+"','"+Website+"','"+address+"','"+un
ame +"','"+passw+"')"
```

```python
        insert_table = ibm_db.exec_immediate(conn, insertQuery)

        data1 = 'Record Saved!'
        return render_template('goback.html', data=data1)




@app.route("/newjob", methods=['GET', 'POST'])
def newjob():
    if request.method == 'POST':
        cnn = session['cname']
        cname = request.form['cname']
        cno = request.form['cno']
        Address = request.form['Address']
        JobLocation = request.form['JobLocation']
        Vacancy = request.form['Vacancy']
        Job = request.form['Job']
        Department = request.form['depat']
        website = request.form['website']

        conn = ibm_db.connect(dsn, "", "")

        insertQuery = "insert into jobtb values('" + cname + "','" + cno + "','" + Address + "','"
+ JobLocation + "','" + Vacancy + "','" + Job + "','" + Department + "','" + website +
"','"+cnn+"')"
        insert_table = ibm_db.exec_immediate(conn, insertQuery)

        conn = ibm_db.connect(dsn, "", "")
        pd_conn = ibm_db_dbi.Connection(conn)
        selectQuery1 = "SELECT * FROM regtb where Department='" + Department + "'"
        dataframe = pandas.read_sql(selectQuery1, pd_conn)

        dataframe.to_sql('regtb', con=engine, if_exists='append')
        data1 = engine.execute("SELECT * FROM regtb").fetchall()

        for item1 in data1:
            Mobile = item1[5]
            Email = item1[4]
            sendmsg(Email,"Jop Title"+Job + " More Info Visit Website")
```

```python
    data = 'Record Saved!'
    return render_template("goback.html", data=data)




@app.route("/jobsearch", methods=['GET', 'POST'])
def jobsearch():
    if request.method == 'POST':
        jobname = request.form['name']


        url = "https://linkedin-jobs-search.p.rapidapi.com/"

        payload = {
            "search_terms": jobname,
            "location": "india",
            "page": "1"
        }
        headers = {
            "content-type": "application/json",
            "X-RapidAPI-Key": "dbc572baf7msh1cbca677f83bd5dp1b070djsna1bfebcbab06",
            "X-RapidAPI-Host": "linkedin-jobs-search.p.rapidapi.com"
        }

        response = requests.request("POST", url, json=payload, headers=headers)

        print(response.text)

        infoFromJson = json.loads(response.text)
        print(json2html.convert(json=infoFromJson))

        nutrients = {}

        data = json.loads(response.text)
        concepts = data['foods'][0]['foodNutrients']
        arr = ["Sugars", "Energy", "Vitamin A", "Vitamin D", "Vitamin B", "Vitamin C",
"Protein", "Fiber", "Iron",
            "Magnesium",
            "Phosphorus", "Cholestrol", "Carbohydrate", "Total lipid (fat)", "Sodium",
```

```python
"Calcium", ]
    for x in concepts:
        if x['nutrientName'].split(',')[0] in arr:
            if (x['nutrientName'].split(',')[0] == "Total lipid (fat)"):
                nutrients['Fat'] = str(x['value']) + " " + x['unitName']
            else:
                nutrients[x['nutrientName'].split(',')[0]] = str(x['value']) + " " + x['unitName']

    return render_template('display.html', x=jobname, data=nutrients,
account=session['username'])
```

```python
#send grid

def sendmsg(Mailid,message):
    import smtplib
    from email.mime.multipart import MIMEMultipart
    from email.mime.text import MIMEText
    from email.mime.base import MIMEBase
    from email import encoders

    fromaddr = "sampletest685@gmail.com"
    toaddr = Mailid

    # instance of MIMEMultipart
    msg = MIMEMultipart()

    # storing the senders email address
    msg['From'] = fromaddr

    # storing the receivers email address
    msg['To'] = toaddr

    # storing the subject
    msg['Subject'] = "Alert"

    # string to store the body of the mail
    body = message
```

```python
    # attach the body with the msg instance
    msg.attach(MIMEText(body, 'plain'))

    # creates SMTP session
    s = smtplib.SMTP('smtp.gmail.com', 587)

    # start TLS for security
    s.starttls()

    # Authentication
    s.login(fromaddr, "hneucvnontsuwgpj")

    # Converts the Multipart msg into a string
    text = msg.as_string()

    # sending the mail
    s.sendmail(fromaddr, toaddr, text)

    # terminating the session


if _name_ == '_main_':
    app.run(host='0.0.0.0', debug='TRUE')
```

GITHUB & PROJECT DEMO LINK:

Github Link:

https://github.com/IBM-EPBL/IBM-Project-46253-1660743815

Project Demo Link:

https://drive.google.com/file/d/1cXYhylTLDVOM2zyBSjC_U6kwb_jIU9G/view?usp=drivesdk