Project Name: Skill/Job Recommender Application

Team Id: PNT2022TMID43302

Team Leader: JOSHVA P

Team Member 1: SURYAMOORTHI A

Team Member 2: VASANTHAKUMAR A

Team Member 3: ASHWIN VISHAL KUMAR V M

Content

1. INTRODUCTION

Nowadays, job search is a task commonly done on the Internet using job search engine sites like Linkedlnl. lndeed2, and others. Commonly, a job seeker has two ways to search a job using these sites:

1) Doing a query based on keywords related to the job vacancy that he/she is looking for, or
2) Creating and/or updating a professional profile containing data related to his/her education, professional experience, professional skills and other, and receive personalized job recommendations based on this data. Sites providing support to the former case are more popular and have a simpler structure; however, their recommendations are less accurate than those of the sites using profile data.

Personalized job recommendation sites implemented a variety of types of recommender systems, such as content-based filtering, collaborative filtering, knowledge-based and hybrid approaches

1.1 PROJECT OVERVIEW:

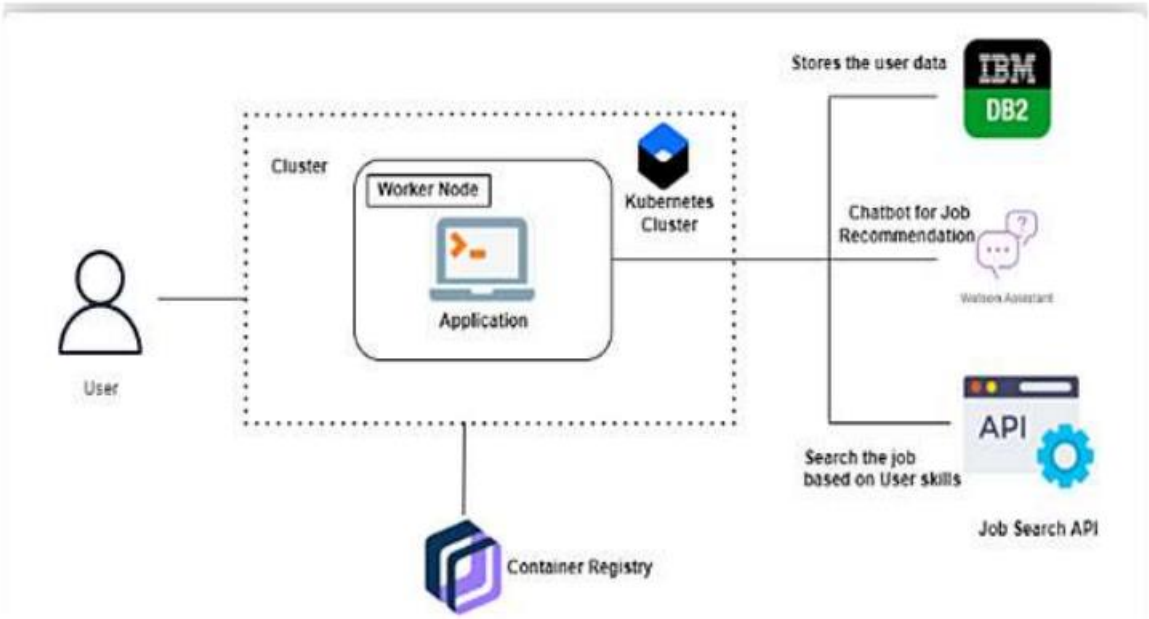
Skill/Job Recommender Application:

Now-a-days it is very difficult for people to find a job. Our application/website makes it easy to find a job. It is great platform to find a job and to improve his/her skill. We also recommend similar job as per your requirement. You can also improve your skill using this platform. Our platform is very much user-friendly and easy to navigate. It is not a piece of cake to get a job, our application provides you with an ample of opportunities.

Project Workflow:

- The user interacts with the application.
- Registers by giving the details.
- Integrate the application with job APls and store the data in the database.
- The database will have all the details and the user can search a job.

Technical architecture:



1.2 PURPOSE

- Better User Experience
- Higher Engagement
- Skill Improvement
- Revenue Opportunities
- Job Opportunities

2. LITERATURE SURVEY

ABSTRACT:

In today's world, students face an immense reper-toire of options relating to the number of courses that they may choose from. To make this seemingly massive choice relatively easy to make, many authors have created their own recommender systems to map students to the courses that are best suited for them. However, they are not widely used as they give good results only for the dataset that they consider. In our paper, we have mapped the current students to their alumni based on multiple criteria. Afterwards, unlike other papers that used k-means, we used c- means and fuzzy clustering to arrive at a better solution to predict an elective course for the student. Since all of thesis done on a broad actual dataset, the results can be applied anywhere in a real-world scenario.

INTRODUCTION:

In the highly competitive environment for jobs, students are often let down by the old curriculum and courses offered by the colleges. Many entry-level employees do not feel that the university courses prepare them for actual job roles. Even if students have knowledge of a particular domain, they are often offered a job role which they never had an interest in or they were never prepared for. Hence it is very important that students develop necessary skills required for various placement profiles by enrolling in online courses. But there is huge diversity of online courses available and there are nearly hundreds of courses for various job roles. Hence it is important for students to get enroll in courses which are suitable to their interests and understanding to save time and eventually help them in the future. This is where a recommender system can be useful for selecting a suitable track according to a student's understanding of various job domains. This paper proposes course recommendation system that suggests courses by finding students similar to target students and then searching for a pattern in their understanding of the domains and courses that they took. Finally, the system recommends courses whose association with the target student's job profile is high.

LITERATURE SURVEY:

In the Paper [1], Amer AI-Badarenah et al. have proposed a system which recommends student courses based on the courses taken by their peers, having similar interests. The similar interest students are then grouped based on their grades. Clusters are created by using the nearest neighbor algorithm. Their system also predicts grades that the student will achieve if they take up that course. The author has used Manhattan distance to find similarity between two students by calculating the distance between their grades in common subjects and to form clusters. After the clusters are formed, the similarity of the student in interests and the cluster is found from the group by using the n-nearest neighbor algorithm. Course association rules are used in finding recommendations. It is found that in some scenarios the algorithm as performance is low. The prediction depends only on the grades which may not help in knowing the students entire potential. Sh. Asadi et al.

[2] talk about using both clustering and fuzzy logic to predict courses. The author uses a clustering algorithm to group similar kinds of students and fuzzy logic to find association rules. PCA is used to decrease the number of components. The clustering is done with K means algorithm and the maximum number of clusters considered in this paper are 5. Student's scores are labeled as low, medium, high. If two regions have equal membership that is if students mark in a subject belongs to two different regions with equal membership then both will be added in a set of rules. This is a drawback. Feng-Hsu Wanh et al.

[3] talks about improving the recommendation system of websites and making the user's experience personalized. They have integrated clustering with the associative-mining method. The customers are clustered based on the time-framed navigation session assuming that their preference may change with time. For the selection of a good time frame they have used the Hierarchical Bisecting Medoids Algorithm. The nearest cluster is found and association rule is applied to that cluster. The maximum matching method preserves the sessions and finds maximum pages that match the rules and pages will be displayed whose confidence is beyond a threshold. This paper suggests that this method will be ideal in education system courses recommendation. However, the drawback of this system is that a customer won't be given recommendations if there are no pages match the association rule. Huiyi Tan et al.

[4] propose a system that can generate recommendations for E- Learning. They collect the data of the user a s previous interest and frequently visited pages are considered. Different recommendation models are stored in a separate dataset and the model is randomly selected. Once the model is selected the useful information from the user's history and recommendations are generated based on the algorithm that the model uses and sent back to the E-Learning websites.

2.1EXISTING SYSTEM:

The developed system consist of three modules: college campus recruitment system, keyword based search from online recruitment sites and Android application. In college campus recruitment system student's profiles and company's profiles are collected. Students profile generated by taking information from students through registration and login portal. Company's profile will be generated by the admin from the information and requirement provided by the company to admin. After that profile matching is perform on the students and company's profiles. In second module i.e., keyword based search module students have the provision to search for the companies from various online recruitment sites. Web crawling technique is used for searching through these sites. Students have to put the keyword e.g. C# and web crawler searches for those companies who have vacancies for C# developers through various online recruitment sites like Naukri.com .

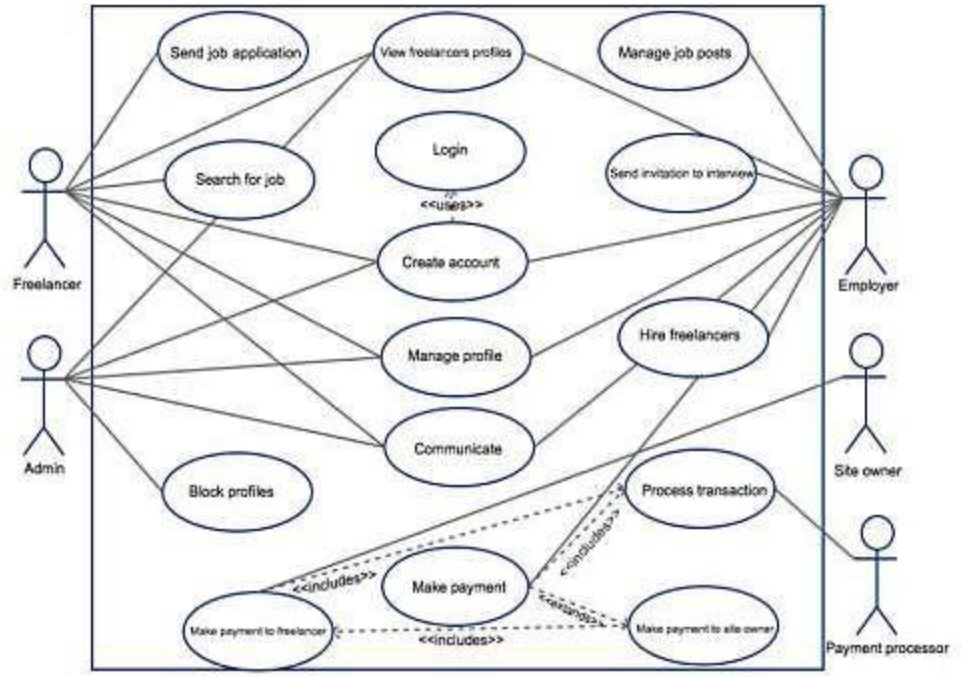
RELATED WORK:

As we have seen, the present-day job seeker is faced with an array of problems before they can find a suitable job for themselves. All existing work is so promising but lacks in some of the other aspects. The need is to eliminate these issues posed by past research and minimize the weaknesses of the systems. The proposed system is designed to go forth with developing a fully functional user interface supporting a job aggregator and recommendation system. Every aspect of the operation is made from scratch and in a customized sort of manner. Hence, the problem statement devised by us as a building starter for the research is as follows:

1. Developing a hybrid model that aggregates and recommends relevant jobs to the user based on their profile, skills, or interests.
2. Emphasizing quality over quantity and delivering only the most appropriate results to the user. The results were achieved by applying intelligent filters and filtering out great amounts of data using appropriate parameters.
3. Recommending jobs to users of any age and background in real time, based on the popularity of jobs among the other user base. Additionally, allowing users to study job popularity, skill demand, grossing market skills, etc. are discussed.
4. Finally, designing a fully useable and understandable UI for the Recommender System for practical usage.
5. The proposed system consists of the following three major modules, which are completed as part of this research as follows:
6. Data collection and preprocessing followed by the unification of the database.
7. Recommendation of suitable results using a hybrid system of content- based and collaborative filtering. Development of a fully functional user interface in the form of a web application.

USECASE DIAGRAM:

Use Case Diagrams referred as behavior diagram which describes the commutation between actors or participations and set of actions. This is set of actions or use cases will be enclosed by system boundary and can also have relation with each other. Division among tupelos will based on the information gain computed for each attribute.



PROPOSED RECOMMENDATION SYSTEM:

As discussed previously, Recommender System are the system that analyses user preference history and caters them with different options of services related to the requirement. Recommended systems emerged as an independent research area in the mid-1990s(Ricci et al., 2011). In recent years, the interest in recommender systems has dramatically increased. In the Re-commendation algorithm, it classifies into four types: Content-based filtering, Collaborative filtering, Rule-based, and Hybrid approaches (Mobasher, 2007; AI-Otaibi and Ykhlef, 2012).Collaborative Filtering (CF): Collaborative Filtering is a technique is based on the human ratings that are given to an item by a user and find similarity between different users who have given similar ratings to an items(Hu and Pu, 2011). The essential operation used here is the memory-based nearest neighbor approach to group users who have a similar interest. As the volume of data grows gradually, there will be high latency in generating recommendations Mobasher (2007); Herlocker.et.al.(1999).

Collaborative filtering has an advantage over content-based filtering techniques, but due to the nature of the hiring process, a job cannot be rated by the user and will not be possible to create a similarity matrix. Content-based filtering (CBF): These are the most subjective and descriptive based filtering. Content-based filtering can also be called as attribute-based recommender as it uses the explicitly defined property of an item. It is an approach to an information retrieval or machine learning problem. The assumption made in content-based filtering is that user prefers item with similar properties. Content-based filtering recommends items to the user whose properties are similar to the item which the user has previously shown interest. Mobasher (2007) express that drawback of this filtering technique is their tendency to over-specialize in suggesting the item to a user profile as user profiles are relayed on an attribute of the previous item opted by the user. Nevertheless, in the job domain, the job listed in the job board be available only for few days; due to the nature of the domain, the tendency to over-specialize in recommending the same item would not be any problem in the job domain recommender system. In domains like entertainment, user preference are tends to change depending on various factors, but In Job domain, the user tends to look for the job where he can use his previous skills. New recommendation of jobs can be made when there is a change in user preference, i.e. if a user thinks to change his/her job domain by updating his new skills and the job domain if he/she wishes. Another scenario of new recommendation is when new jobs are listed in the database; system would identify the properties of the job listed, such as job domain and skills required for the job and matches with the users with a high similarity score. Rule-based Filtering (RBF): These filtering techniques depend upon decision rules such as an automatic or manual decision rule that are manipulated to obtain a recommendation for the user profile. Currently, the Ecommerce industry uses a rule-based filtering technique to recommend an item based on the demographic region of a user, purchase history, and other attributes that can be used to profile an user. A drawback in rule based filtering is user feeds the information to the system. These inputs are utilized as a description of a user profile or can be considered as a preference of a user, defined by the user. Thus the data acquired is prone to bias . With the age of the user's profile, recommendation tends to hit the saturation and become static Mobasher (2007). Hybrid filtering (HF): As the title describe, its incorporation of multiple techniques to improve the performance of recommendation. The previously discussed recommendation technique has its weakness and strengths. In order to get a better recommendation and overcome the challenges posed by earlier techniques, this technique is sought after. All of the learning/model-based techniques suffer from cold-start in one or other form. It is a problem related to handling a new user or new item. These and other shortcomings of the CF,CBF, and RBF could be resolved by using hybrid filtering

techniques Burke (2007); Jain and Kakkar(2019); Dhameliya and Desai (2019). The surveys conducted by Burke (2002) and Dhameliya and Desai (2019) have identified different types of hybrid filtering techniques that could be used by integrating CF, CBF, and RBF. 1. Weighted: The similarity score obtained from different recommendation components are coupled numerically to get one better recommendation. 2. Mixed: Recommendations obtained from different recommending techniques are put together and presented as one recommendation. 3. Switching: choosing one among the recommendation components based on the scenarios where it suits best. 4. Feature Combination: Attributes derived from diverse knowledge origins are fused and supplied to a recommendation algorithm. 5. Feature Augmentation: One recommendation technique is used to compute a set of attributes of user or item, which is then part of the input to the next recommendation technique. Two or more recommendation techniques are serialized to get on recommendation. 6. Cascade: Recommending systems are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones. Here one Recsys technique refines recommendation of another. 7. There had been attempts to develop a recommendation system by several researchers. One such implementation was done by Rafter et al. (2000). They had devised a hybrid Recsys CASPER for Job finding search engine. They had implemented an automated collaborative filtering module and personalized case retrieval module in their job recommendation system. ACF module utilized user behavior information such as read time and activity on the page during his time on the system to profile the user. Similarity measure such as the Jaccard index and other clustering algorithms was used for similar grouping user against target user. Their other module PCR finds the similarity between the user's query and jobs in the system. The module computes similarity with a target user's query and jobs from the job case base using different similarity measures. This system has faced sparsity and scalability problems.

FUTURE SCOPE:

The previous recommendation system clusters students only on the basis of their grades. They also map a student to only one cluster and do not consider any other possibilities. However, this paper takes into account a student's interests and clusters them accordingly. The target student is associated with all the clusters using the c-means fuzzy algorithm. It finds the clusters in which the students are similar to the target student and group the cluster. The courses taken by the most similar students to the target, from the cluster, are found out and select the important and required the most common course is recommended

2.2 REFERENCE:

[1] AI-Badarenah, A. and Alsakran, J., 2016. An automated recommender system for course selection. International Journal of Advanced ComputerScience and Applications, 7(3), pp.166-175.

[2] Asadi, S., Jafari, S. and Shokrollahi, Z., 2019. Developing a Course Recommender by Combining Clustering and Fuzzy Association Rules. Journal of Al and Data mining, 7(2), pp.249-262.

[3] Wang, F.H. and Shao, H.M., 2004. Effective personalized recommenda-tion based on time-framed navigation clustering and association mining.Expert systems with applications, 27(3), pp.365-377.

[4] H. Tan, J. Guo and Y. Li, "E-learning Recommendation System," 2008 International Conference on Computer Science and Software Engineer- ing, Hubei, 2008, pp. 430-433.

[5] IEEE International Conference on Computing, Power and Communication Technologies (GUCON)Galgotias University, Greater Naida, UP, India. Oct-2020

[6] Students, Computer Science & Engineering, K K Wagh College of Engineering, Nashik, India

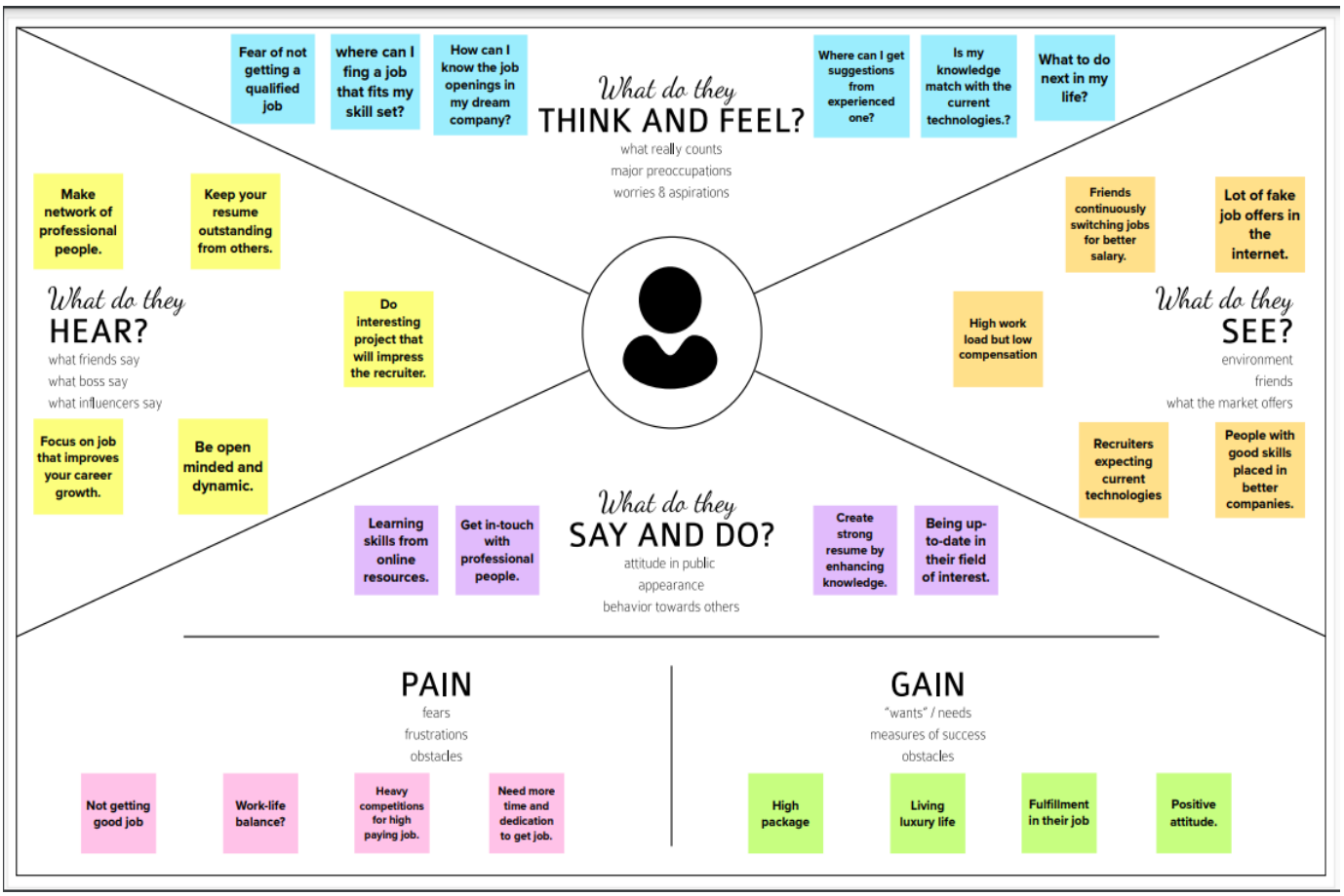
[7] The International Workshop on Artificial Intelligence and Smart City Applications {IWAISCA)August 9-12, 2020, Leuven, Belgium Job Recommendation based on Job Profile Clustering and JobSeeker Behavior

2.3 PROBLEM STATEMENT DEFINITION

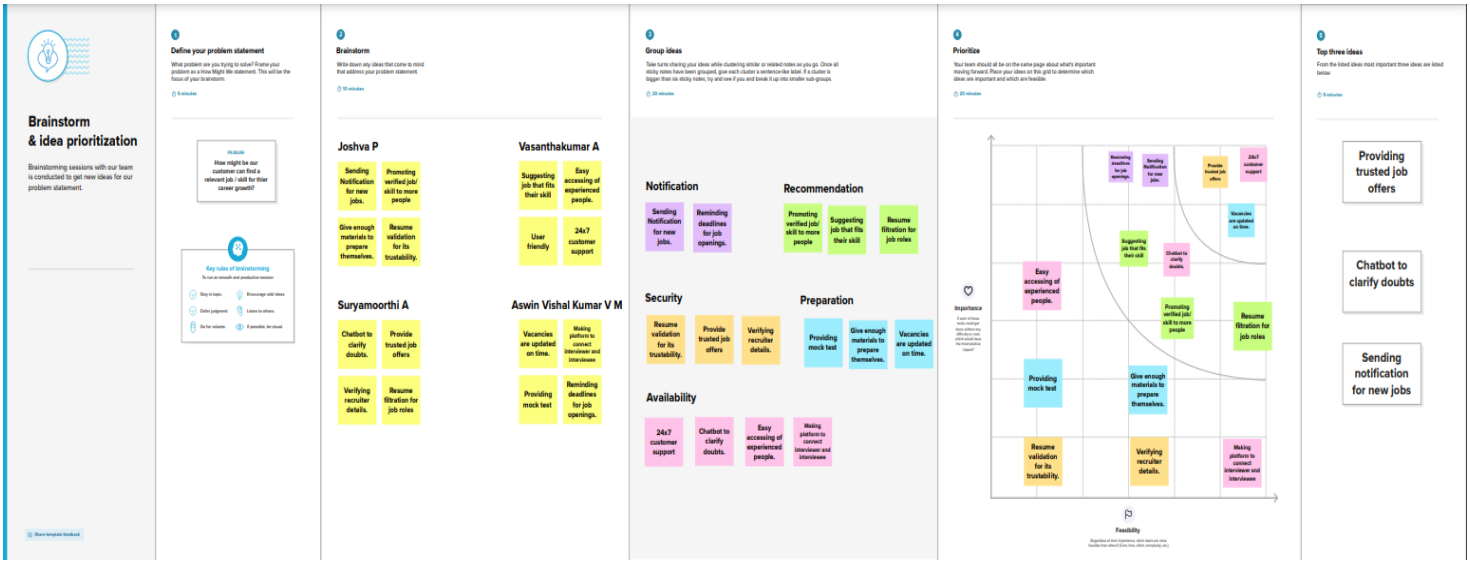
The problem statement **identifies the current state, the desired future state and any gaps between the two**. A problem statement is an important communication tool that can help ensure everyone working on a project knows what the problem they need to address is and why the project is important.

3 IDEATION AND PROPOSED SOLUTIONS

3.1 Empathy Map Canvas:



3.2 Ideation and Brainstroming

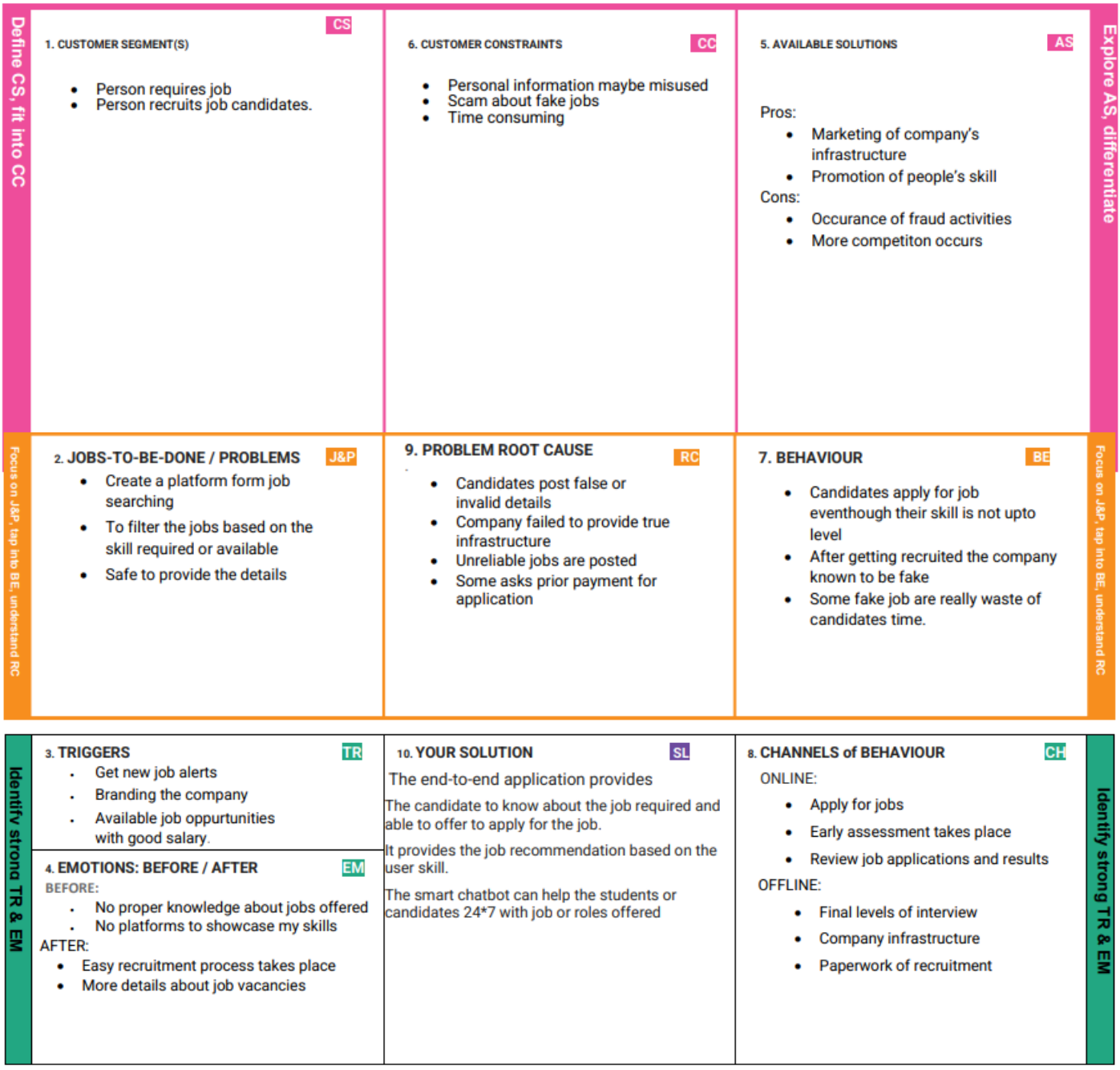


3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	• Look for field based jobs ass searching • Estimating salaries based on technical skills. • Anxiety for freshers to get a good job. • Uncertainty of fresher regarding choosing job /skill to be developed.

2.	Idea / Solution description	• We have come up with a skill recommender solution through which the fresher or the skilled person can log in and find the jobs by using the search option or they can directly interact with the chatbot and get their dreamjob. • 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 onthe user skillset. • Users will interact with the chatbot and can get the recommendations based on their skills.
3.	Novelty / Uniqueness	• The best position are suggested to any person according to her skills. • While the position of known profiles are assumed should be noted that there are usually multiple advisable positions corresponding to a set of skills. • A recommendation system should return a set of most likely positions and all of them canbe equally valid. • The recommendation method we use is simplybased on representing both positions and profiles as comparable vectors and seeking for each profile the positions with the most similar vectors.

3.4 Problem solution fit



4.REQUIREMENT ANALYSIS

4.1 Functional Requirements

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	Chatbot	Available in website to solve candidate or user queries related to jobs and for searching
FR-4	User Login	Login through Gmail Login through form
FR-5	Search	Based on job filters and candidates' recommendation searching is done
FR-6	Acceptance	Updating the jobs based on candidates' preference and providing the exact result

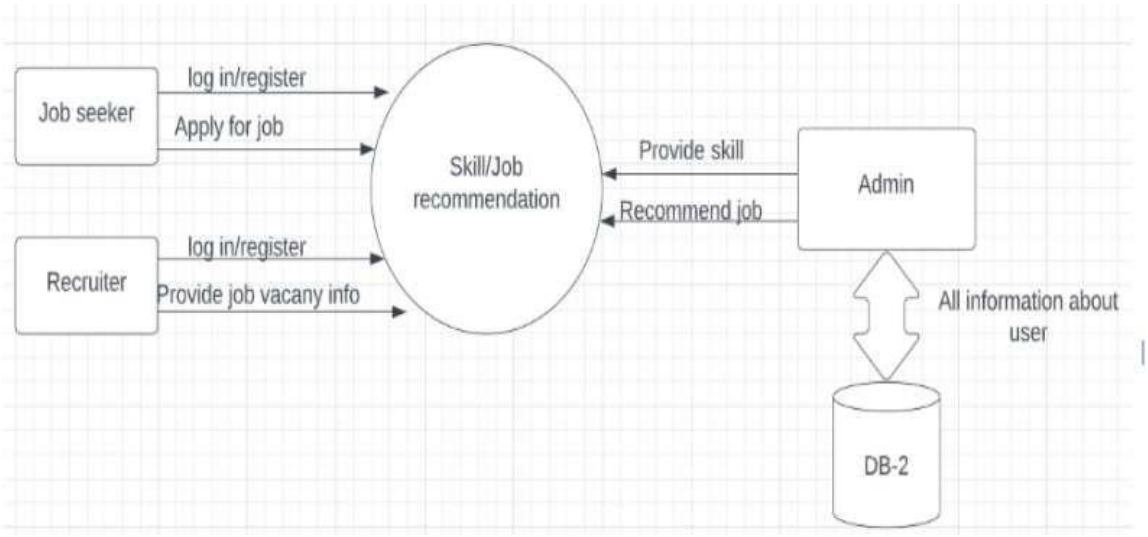
4.2 Non-functional Requirements:

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Helps candidates who are searching for jobs based on their skill set
NFR-2	**Security**	Provides separate platform for job recruiters and job seekers
NFR-3	**Reliability**	Open source ,feel free to use. Many job opportunities are provided for the candidates
NFR-4	**Performance**	Quicker response and makes less time to provide result
NFR-5	**Availability**	For particular jobs openings it provides job offers
NFR-6	**Scalability**	Response time s faster compared to other applications

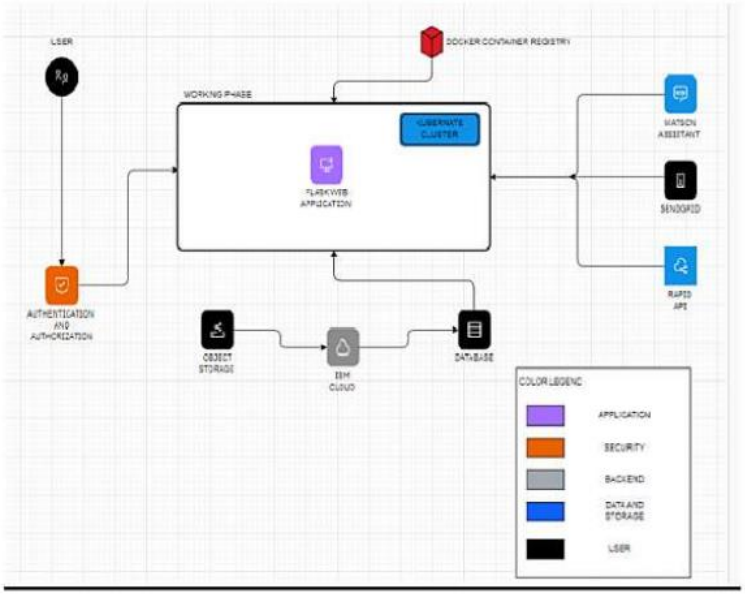
5.PROJECT DESIGN

5.1 Data flow diagram

Data Flow Diagrams: A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the rightamount of the system requirement graphically. It shows how data enter and leaves the system, what changes the information, and where data is stored.



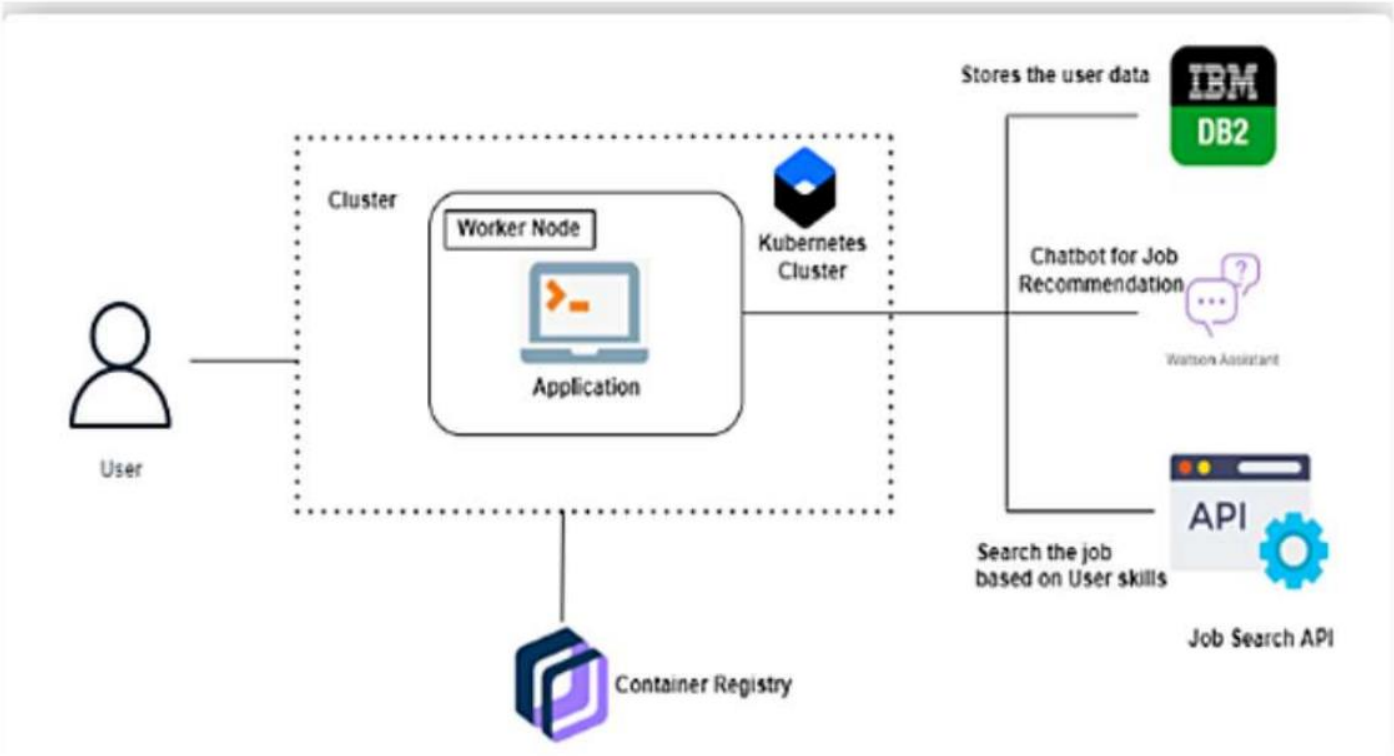
Flow Diagram:



Guidelines:

1. Registration using form, Gmail
2. Confirmation using OTP, gmail
3. flask app -using python library
4. first - Homepage with login and register
5. In Homepage showing post job and apply job
6. Login page- login and confirmation message
7. register page- register an confirm using OTP or email
8. after Login-dashboard showing more jobs and can search specific job
9. Login and register database are stored in IBM DB2
10. OTP Messages are sent through Send grid
11. Rapid api is connected to display jobs and to search jobs
12. Files can be stored in IBM Storage
13. Services are received from IBM Cloud account

5.2 Solution and Technical Architecture



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Registration	USN-1	As a user, I can register for the website 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 web	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the website through Google	I can register & access the dashboard with Google Login	Low	Sprint-2
		USN-4	As a user, I can register for the website through G-mail.	I can register through Google mail	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the website by entering email & password	I can log in and access my account	High	Sprint-1
	Dashboard	USN-6	As a user, I enter the dashboard and learn new skills.	I can access my dashboard and learn new skill.	High	Sprint-3
Administrator	Recommendation	USN-7	As an administrator, I recommend new job	I can give job recommendation	High	Sprint-4
	Maintenance	USN-8	As an administrator, I can access and store all the details about users in DB-2.	I can access the database	High	Sprint-4

Activate Windows

6 PROJECT PLANING & SCHEDULING

6.1 Sprint Planning & Estimation

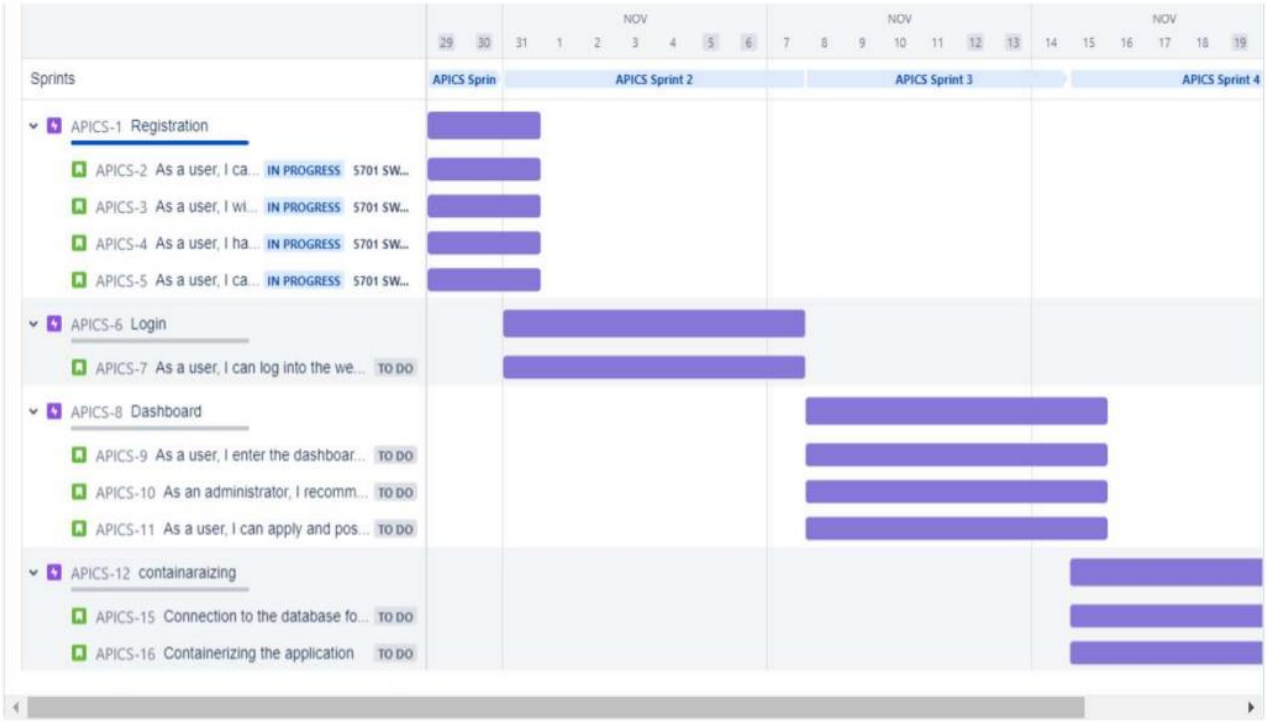
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	AshwinVishal Joshva
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	Suryamoorthi
Sprint-2		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Vasanthkumar
Sprint-1		USN-4	As a user, I can register for the application through Gmail	I can receive confirmation email & click confirm	Medium	Joshva
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	I can access my account / dashboard	High	Suryamoorthi Vasantkhumar
Sprint-1	Dashboard	USN-6	Create a model set that contains those models, then assign it to a role.	Assign that group to the appropriate roles on the Roles page	High	AshwinVishal
Sprint-1	Identity-Aware	USN-7	Open, public access, User-authenticated access, Employee-restricted access.	Company public website. App running on the company intranet. App with access to customer private information.	High	Vasanthumar Joshva

Sprint-1	Communication	USN-8	A customer care executive is a professional responsible for communicating the how's and why's regarding service expectations within a company.	For how to tackle customer queries.	Medium	Joshva
Sprint-1	Device management	USN-9	You can Delete/Disable/Enable devices in Azure Active Directory butyou cannot Add/Remove Users in the directory.	Ease of use.	Medium	AshwinVishal

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	5 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports From JIRA



7 CODING & SOLUTIONING

7.1 Cloud Features:

1. Self-service On-Demand

This is one of the most essential and significant characteristics of cloud computing. This means that cloud computing enables clients to regularly monitor the abilities, allotted network storage, and server uptime. Therefore, it is one of the most fundamental features of cloud computing that helps clients control various computing abilities as per their requirements.

2. Resources Pooling

This is also a fundamental characteristic of cloud computing. Pooling resources means that a cloud service provider can distribute resources for more than one client and provide them with different services according to their needs. Resource Pooling is a multi-client plan useful for data storing, bandwidth services and data processing services. The provider administers the data stored in real-time without conflicting with the client's need for data.

3. Easy Maintenance

This is one of the best cloud characteristics. Cloud servers are easy to maintain with low to almost zero downtime. Cloud Computing powered resources undergo several updates frequently to optimize their capabilities and potential. The updates are more viable with the devices and perform quicker than the previous versions.

4. Economical

This kind of service is economical as it efficiently reduces IT costs and data storage expenditure. Moreover, most cloud computing services are free. Even if there are paid plans, it's only to expand storage capacity, and these costs are often very nominal. This is a massive advantage of using cloud computing services.

5. Rapid Elasticity and Scalability

The best part of using cloud storage is that it can easily handle all the workload and data load concerning storage. Furthermore, as it is fully automated, businesses and organizations can save heavily on manual labor and technical staffing as cloud services are elastic, scalable and automated. This is one of the significant advantages of using cloud services.

API Features:

1. HTTPS/SSL certificates

- Programming cheat sheets
- TrY- for free: Red Hat Learning Subscription
- eBook: An introduction to programming with Bash
- Bash Shell Scripting Cheat Sheet
- eBook: Modernizing Enterprise Java

The gold standard for the web is HTTPS using SSL certificates, and Let's Encrypt can help you achieve this. It is a free, automated, and open certificate authority from the non-profit Internet Security Research Group (ISRG).

2. Cross-origin resource sharing

CORS is a browser-specific security policy preflight check. If your API server is not in the same domain as the requesting client's domain, you will need to deal with CORS. For example, if your server is running on api.domain-a.com and gets a client request from domain-b.com, CORS sends an HTTP precheck request to see if your API service will accept client-side requests from the client's domain.

3. Authentication and JSON Web Tokens

There are several approaches to validate an authenticated user in your API, but one of the best ways is to use JSON Web Tokens (JWT). These tokens are signed using various types of well-known cryptographic libraries.

When a client logs in, an identity-management service provides the client with a JWT. The client can then use this token to make requests to the API. The API has access to a public key or a secret that it uses to verify the token.

There are several libraries available to help verify tokens, including jsonwebtoken. For more information about JWT and the libraries that support it in every language, check out JWT.io.

```
import jwt from 'jsonwebtoken'

export default function (req, res, next) {
    // req.headers.authorization Bearer token
    const token = extractToken(req)
    jwt.verify(token, SECRET, { algorithms: ['HS256'] },
(err, decoded) => {
        if (err) { next(err) }
        req.session = decoded
        next()
    })
}
```

4. Authorizations and scopes

Authentication (or identity verification) is important, but so is authorization, i.e., *does the verified client have the privilege to execute this request?* This is where scopes are valuable. When the client authenticates with the identity management server and a JWT token is created, having the identity management service provide the scopes for the given authenticated client can enable the API service to determine if this verified client request can be performed without having to perform an additional costly lookup to an access control list.

7.2 FEATURE 2

Docker Features:

1. Faster and Easier configuration:

It is one of the key features of Docker that helps you in configuring the system in a faster and easier manner. Due to this feature, codes can be deployed in less time and with fewer efforts. The infrastructure is not linked with the environment of the application as Docker is used with a wide variety of environments.

2. Application isolation:

Docker provides containers that are used to run applications in an isolated environment. Since each container is independent, Docker can execute any kind of application.

3. Increase in productivity:

It helps in increasing productivity by easing up the technical configuration and rapidly deploying applications. Moreover, it not only provides an isolated environment to execute applications, but it reduces the resources as well.

4. Swarm:

Swarm is a clustering and scheduling tool for Docker containers. At the front end, it uses the Docker API, which helps us to use various tools to control it. It is a self-organizing group of engines that enables pluggable backends.

5. Services:

Services is a list of tasks that specifies the state of a container inside a cluster. Each task in the Services lists one instance of a container that should be running, while Swarm schedules them across the nodes.

Kubernetes Features:

1. Auto-scaling. Automatically scale containerized applications and their resources up or down based on usage
2. Lifecycle management. Automate deployments and updates with the ability to:
    a. Rollback to previous versions
    b. Pause and continue a deployment

3. Declarative model. Declare the desired state, and K8s works in the background to maintain that state and recover from any failures.

4. Resilience and self-healing. Auto placement, auto restart, auto replication and auto scaling provide application self-healing

5. Persistent storage. Ability to mount and add storage dynamically

6. Load balancing. Kubernetes supports a variety of internal and external load balancing options to address diverse needs.

7. DevSecOps support. DevSecOps is an advanced approach to security that simplifies and automates container operations across clouds, integrates security throughout the container lifecycle, and enables teams to deliver secure, high-quality software more quickly. Combining DevSecOps practices and Kubernetes improves developer productivity.

7.3 DATABASE SCHEMA

Username: Varchar (32)

Email: Varchar (32)

Phone Number: Varchar (32)

Password: Varchar (32)

Pin: Varchar (32)

8 TESTING

8.1 TEST CASES

8.1.1 Verify that after registration users are navigated to login page

8.1.2 Verify the UI elements in Login/Signup popup

8.1.3 Verify user is able to log into application with Valid credentials

8.1.4 Verify that categories of news are shown in homepage

8.1.5 Verify that news is displayed in homepage

8.1.6 Verify that when clicked on news it is redirected to correct page

8.2 USER ACCEPTANCE TESTING

Purpose of Document

The purpose of this documents to briefly explain the test coverage and open issues of the Skills/Job Recommender Application project at the time of the release to User Acceptance Testing (UAT).

1. Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	3	2	2	1	8
Duplicate	1	0	3	0	4

External	2	0	0	1	3
Fixed	5	2	4	7	18
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	11	9	13	11	44

2. Test Case Analysis

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

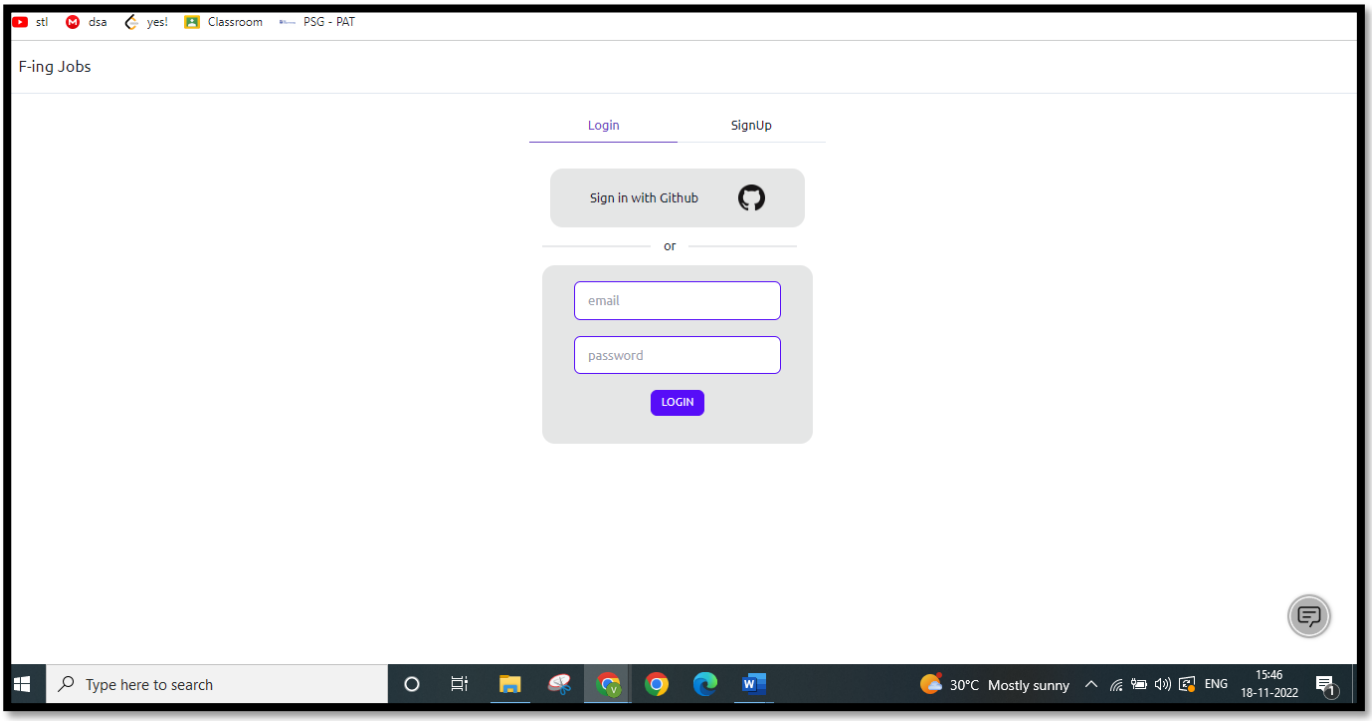
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	4	0	0	4
Security	3	0	0	3
Outsource Shipping	7	0	0	7
Exception Reporting	6	0	0	6
Final Report Output	3	0	0	3
Version Control	2	0	0	2

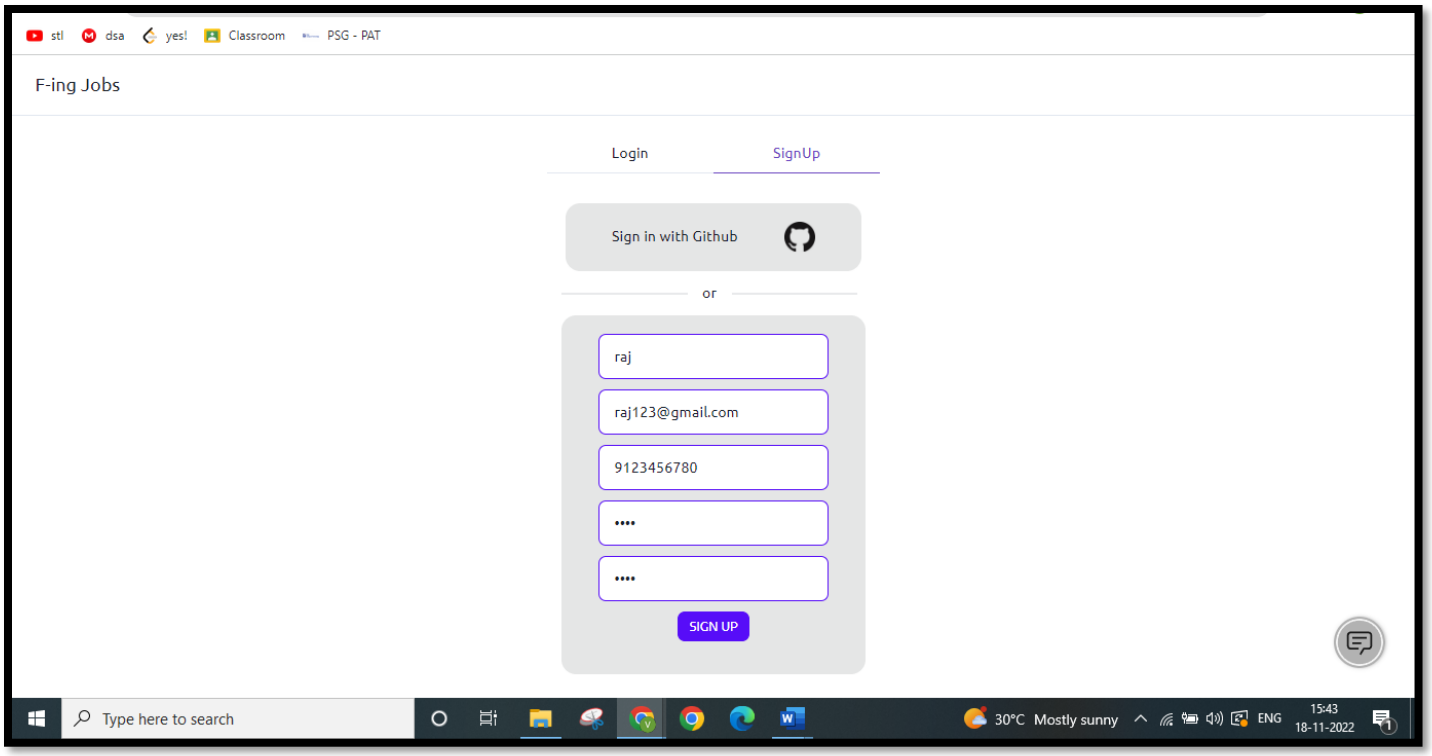
9.RESULT

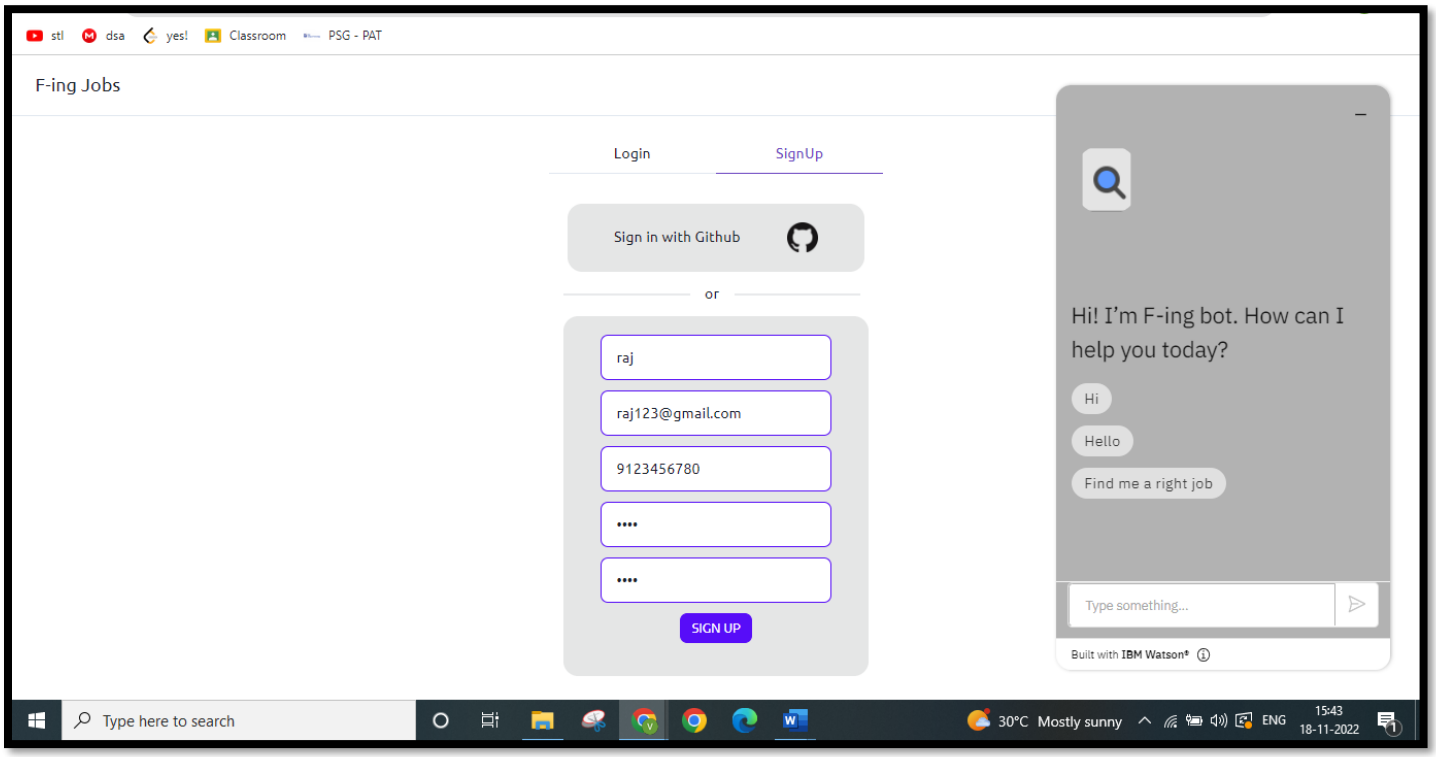
9.1. PERFORMANCE METRICS

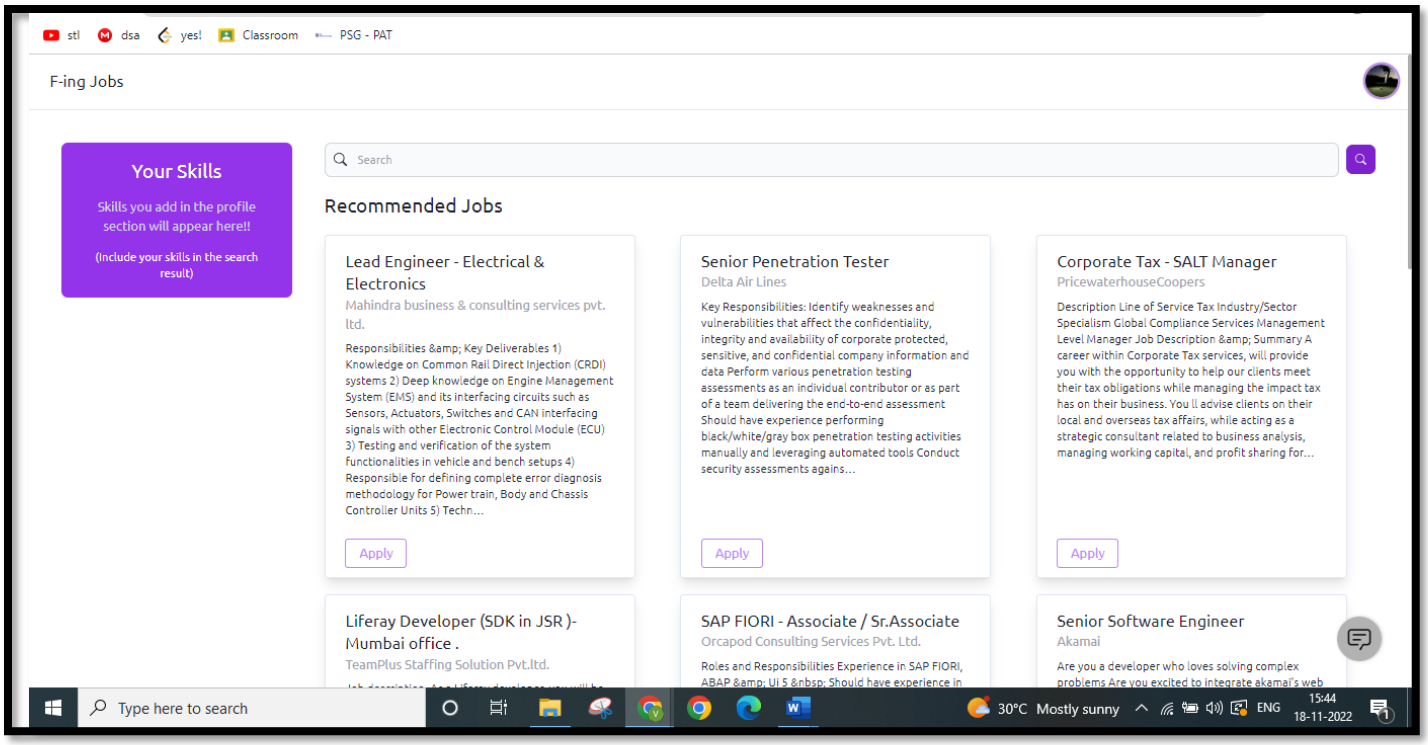
The application performance index, or Apdex score, has become an industry standard for tracking the relative performance of an application. It works by specifying a goal for how long a specific web request or transaction should take. Those transactions are then bucketed into satisfied (fast), tolerating (sluggish), too slow, and failed requests. A simple math formula is then applied to provide a score from O to 1.

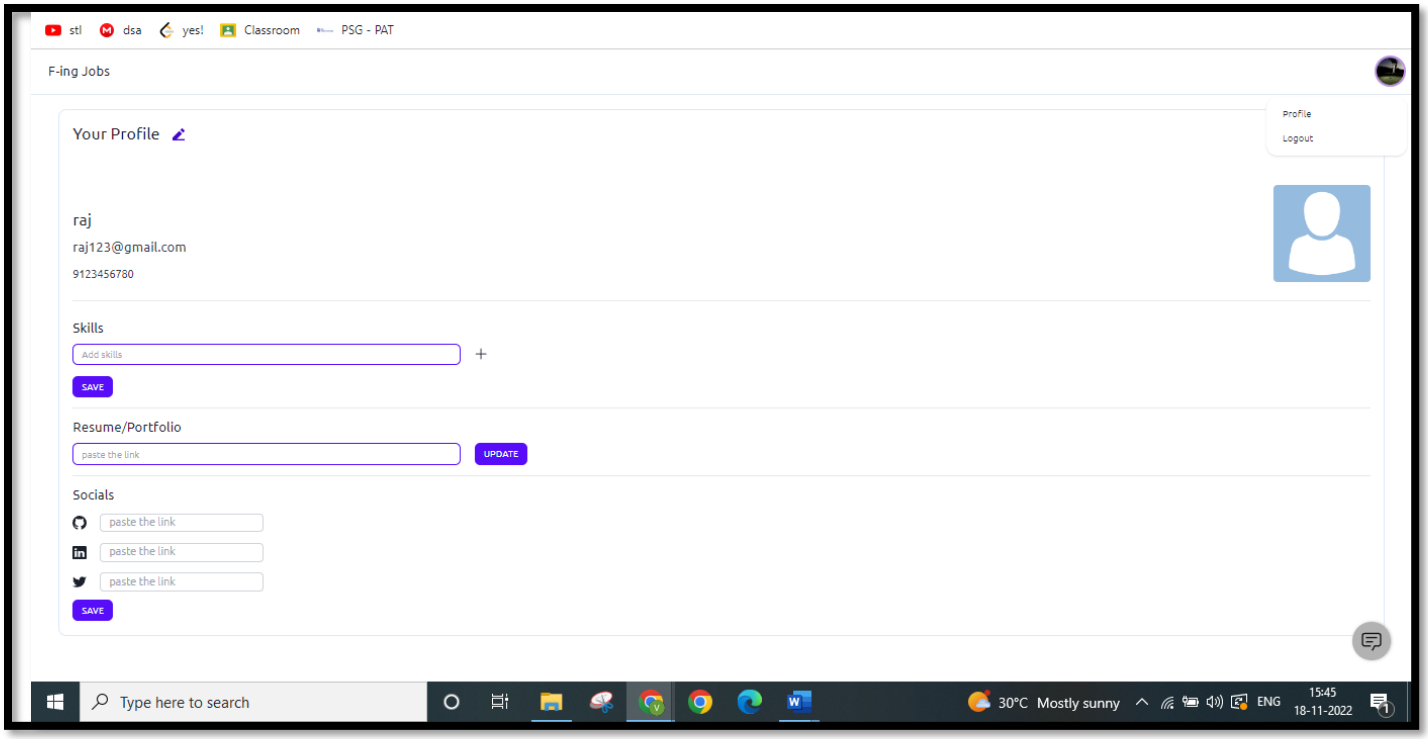
SCREENSHOTS

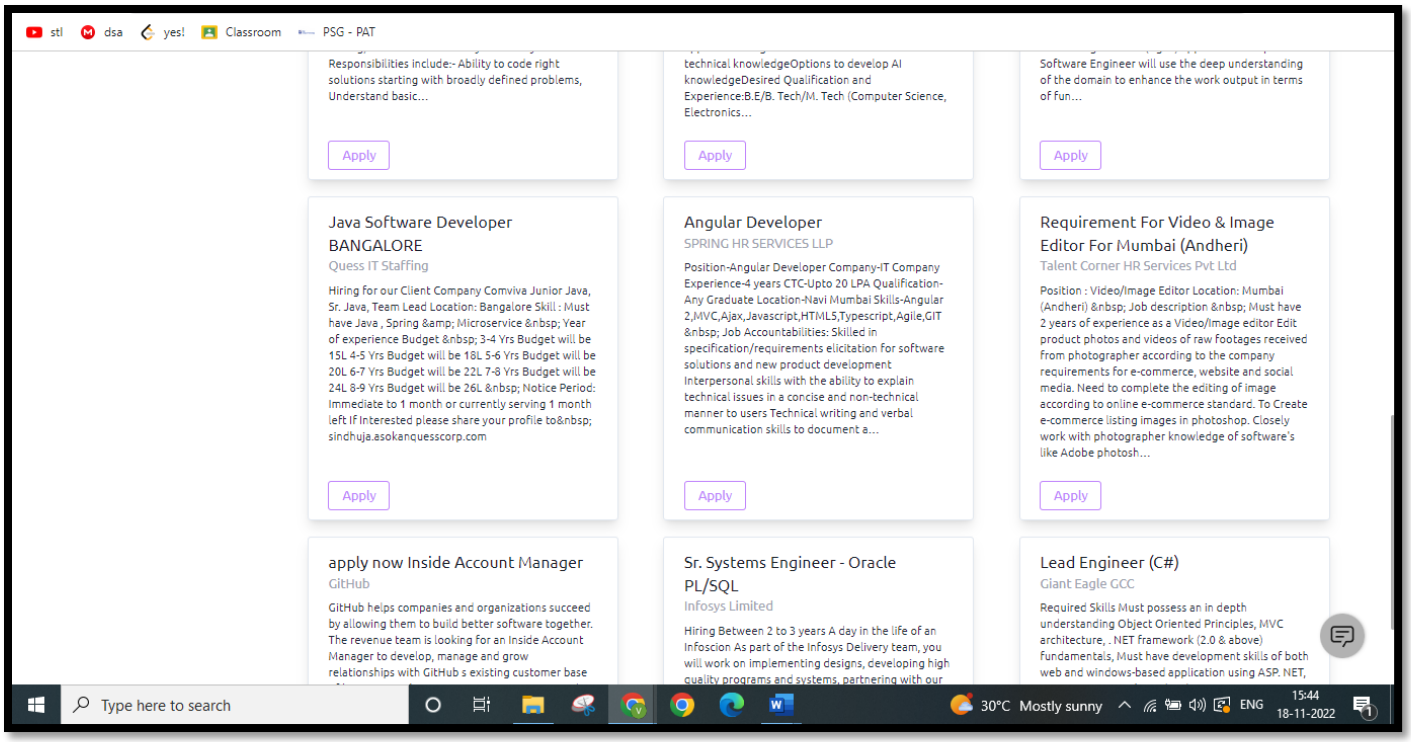


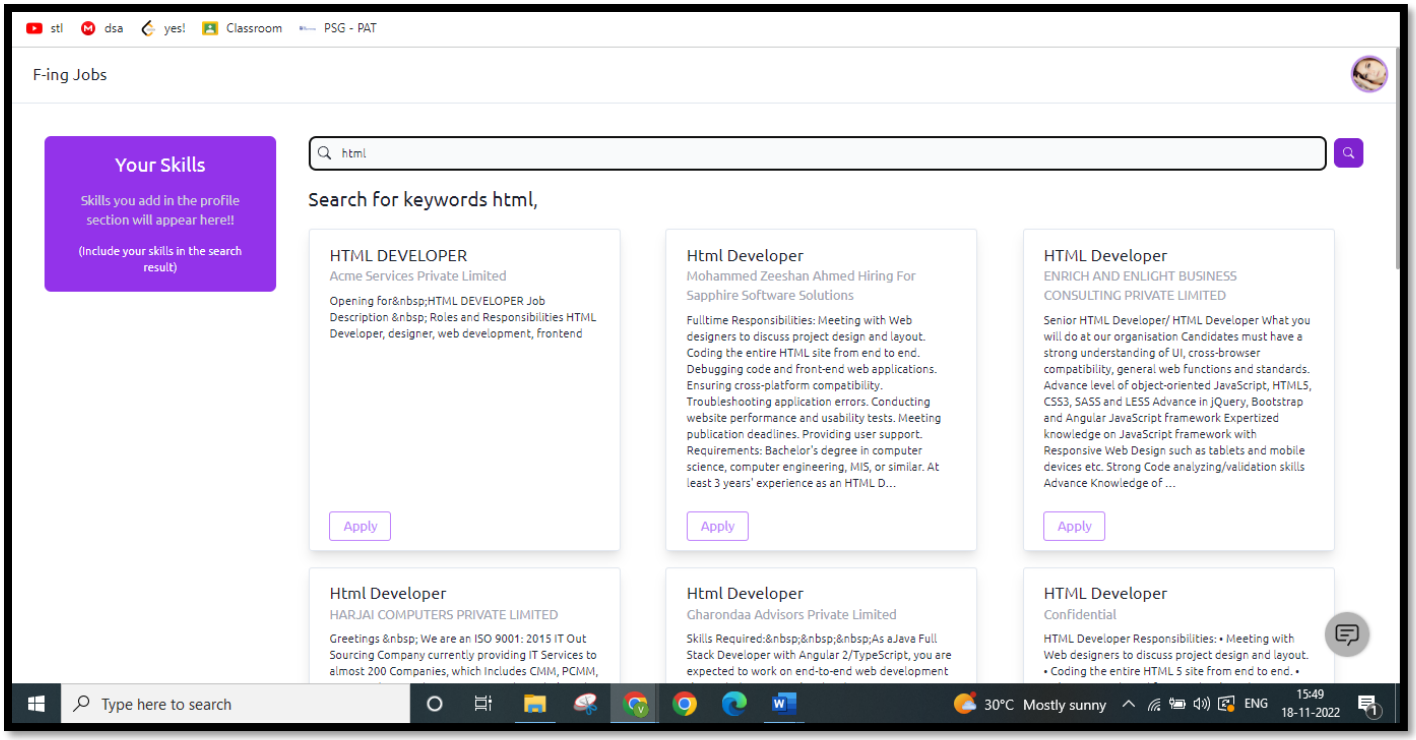

stl
dsa
yes!
Classroom
PSG - PAT
F-ing Jobs
Login
SignUp
Sign in with Github
or
raj
raj123@gmail.com
9123456780
SIGN UP
Type here to search
30°C Mostly sunny
ENG
15:43
18-11-2022

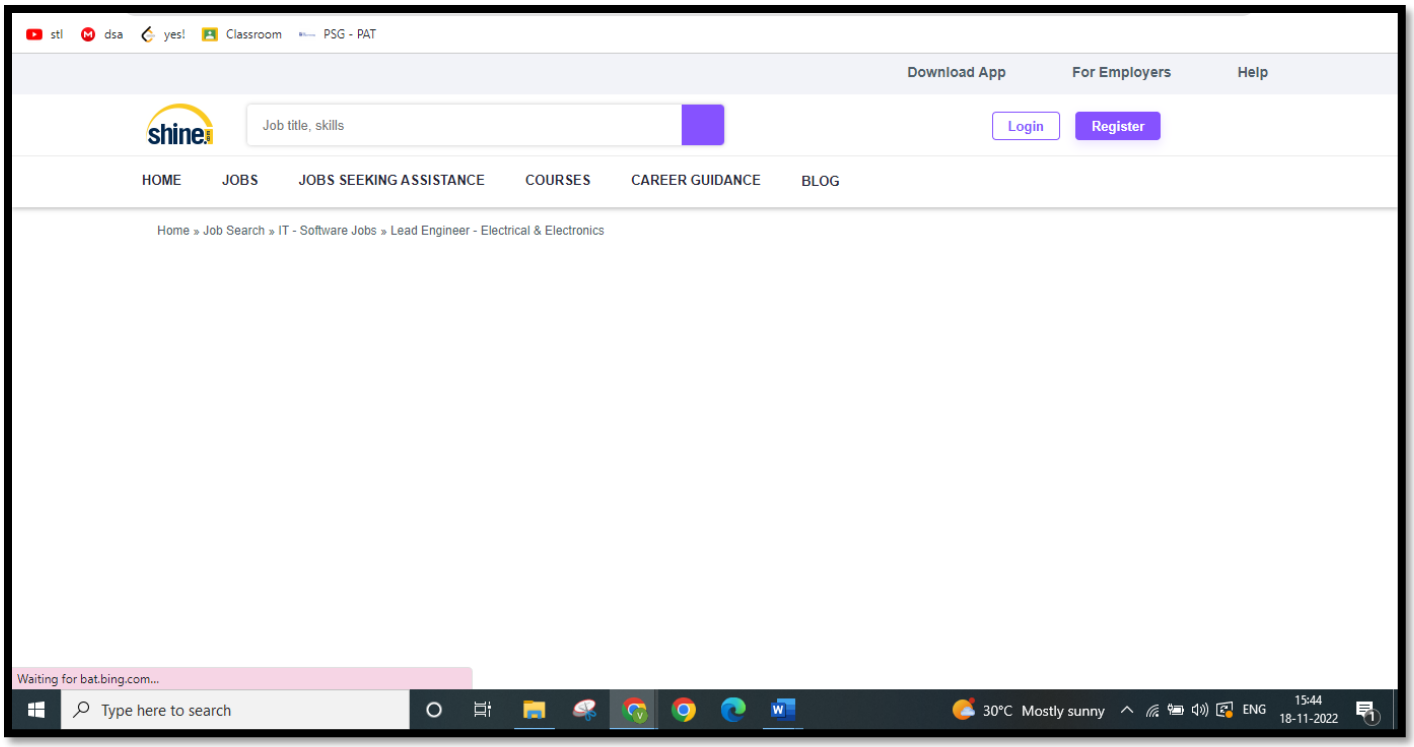

stl
dsa
yes!
Classroom
PSG - PAT
F-ing Jobs
Login
SignUp
Sign in with Github
or
raj
raj123@gmail.com
9123456780
SIGN UP
Hi! I'm F-ing bot. How can I help you today?
Hi
Hello
Find me a right job
Type something...
Built with IBM Watson®
Type here to search
30°C Mostly sunny
ENG
15:43
18-11-2022


stl
dsa
yes!
Classroom
PSG - PAT
F-ing Jobs
Your Skills
Skills you add in the profile section will appear here!!
(Include your skills in the search result)
Search
Recommended Jobs
Lead Engineer - Electrical & Electronics
Mahindra business & consulting services pvt. ltd.
Responsibilities & Key Deliverables 1) Knowledge on Common Rail Direct Injection (CRDI) systems 2) Deep knowledge on Engine Management System (EMS) and its interfacing circuits such as Sensors, Actuators, Switches and CAN interfacing signals with other Electronic Control Module (ECU) 3) Testing and verification of the system functionalities in vehicle and bench setups 4) Responsible for defining complete error diagnosis methodology for Power train, Body and Chassis Controller Units 5) Techn...
Apply
Senior Penetration Tester
Delta Air Lines
Key Responsibilities: Identify weaknesses and vulnerabilities that affect the confidentiality, integrity and availability of corporate protected, sensitive, and confidential company information and data Perform various penetration testing assessments as an individual contributor or as part of a team delivering the end-to-end assessment Should have experience performing black/white/gray box penetration testing activities manually and leveraging automated tools Conduct security assessments agains...
Apply
Corporate Tax - SALT Manager
PricewaterhouseCoopers
Description Line of Service Tax Industry/Sector Specialism Global Compliance Services Management Level Manager Job Description & Summary A career within Corporate Tax services, will provide you with the opportunity to help our clients meet their tax obligations while managing the impact tax has on their business. You ll advise clients on their local and overseas tax affairs, while acting as a strategic consultant related to business analysis, managing working capital, and profit sharing for...
Apply
Liferay Developer (SDK in JSR)- Mumbai office .
TeamPlus Staffing Solution Pvt.ltd.
SAP FIORI - Associate / Sr.Associate
Orcapod Consulting Services Pvt. Ltd.
Roles and Responsibilities Experience in SAP FIORI, ABAP & UI 5   Should have experience in
Senior Software Engineer
Akamai
Are you a developer who loves solving complex problems Are you excited to integrate akamai's web
Type here to search
30°C Mostly sunny
ENG
15:44
18-11-2022


stl
dsa
yes!
Classroom
PSG - PAT
F-ing Jobs
Profile
Logout
Your Profile
raj
raj123@gmail.com
9123456780
Skills
Add skills
SAVE
Resume/Portfolio
paste the link
UPDATE
Socials
paste the link
paste the link
paste the link
SAVE
Type here to search
30°C Mostly sunny
ENG
15:45
18-11-2022


stl
dsa
yes!
Classroom
PSG - PAT
Responsibilities include:- Ability to code right solutions starting with broadly defined problems, Understand basic...
Apply
technical knowledgeOptions to develop AI knowledgeDesired Qualification and Experience:B.E/B. Tech/M. Tech (Computer Science, Electronics...
Apply
Software Engineer will use the deep understanding of the domain to enhance the work output in terms of fun...
Apply
Java Software Developer BANGALORE
Quess IT Staffing
Hiring for our Client Company Comviva Junior Java, Sr. Java, Team Lead Location: Bangalore Skill : Must have Java , Spring & Microservice   Year of experience Budget   3-4 Yrs Budget will be 15L 4-5 Yrs Budget will be 18L 5-6 Yrs Budget will be 20L 6-7 Yrs Budget will be 22L 7-8 Yrs Budget will be 24L 8-9 Yrs Budget will be 26L   Notice Period: Immediate to 1 month or currently serving 1 month left If Interested please share your profile to sindhuja.asokanquesscorp.com
Apply
Angular Developer
SPRING HR SERVICES LLP
Position-Angular Developer Company-IT Company Experience-4 years CTC-Upto 20 LPA Qualification-Any Graduate Location-Navi Mumbai Skills-Angular 2,MVC,Ajax,Javascript,HTML5,Typescript,Agile,GIT   Job Accountabilities: Skilled in specification/requirements elicitation for software solutions and new product development Interpersonal skills with the ability to explain technical issues in a concise and non-technical manner to users Technical writing and verbal communication skills to document a...
Apply
Requirement For Video & Image Editor For Mumbai (Andheri)
Talent Corner HR Services Pvt Ltd
Position : Video/Image Editor Location: Mumbai (Andheri)   Job description   Must have 2 years of experience as a Video/Image editor Edit product photos and videos of raw footages received from photographer according to the company requirements for e-commerce, website and social media. Need to complete the editing of image according to online e-commerce standard. To Create e-commerce listing images in photoshop. Closely work with photographer knowledge of software's like Adobe photosh...
Apply
apply now Inside Account Manager
GitHub
GitHub helps companies and organizations succeed by allowing them to build better software together. The revenue team is looking for an Inside Account Manager to develop, manage and grow relationships with GitHub s existing customer base
Sr. Systems Engineer - Oracle PL/SQL
Infosys Limited
Hiring Between 2 to 3 years A day in the life of an Infoscion As part of the Infosys Delivery team, you will work on implementing designs, developing high quality programs and systems, partnering with our
Lead Engineer (C#)
Giant Eagle GCC
Required Skills Must possess an in depth understanding Object Oriented Principles, MVC architecture, . NET framework (2.0 & above) fundamentals, Must have development skills of both web and windows-based application using ASP. NET,
Type here to search
30°C Mostly sunny
ENG
15:44
18-11-2022


stl
dsa
yes!
Classroom
PSG - PAT
F-ing Jobs
Your Skills
Skills you add in the profile section will appear here!!
(Include your skills in the search result)
html
Search for keywords html,
HTML DEVELOPER
Acme Services Private Limited
Opening for HTML DEVELOPER Job Description   Roles and Responsibilities HTML Developer, designer, web development, frontend
Apply
Html Developer
Mohammed Zeeshan Ahmed Hiring For Sapphire Software Solutions
Fulltime Responsibilities: Meeting with Web designers to discuss project design and layout. Coding the entire HTML site from end to end. Debugging code and front-end web applications. Ensuring cross-platform compatibility. Troubleshooting application errors. Conducting website performance and usability tests. Meeting publication deadlines. Providing user support. Requirements: Bachelor's degree in computer science, computer engineering, MIS, or similar. At least 3 years' experience as an HTML D...
Apply
HTML Developer
ENRICH AND ENLIGHT BUSINESS CONSULTING PRIVATE LIMITED
Senior HTML Developer/ HTML Developer What you will do at our organisation Candidates must have a strong understanding of UI, cross-browser compatibility, general web functions and standards. Advance level of object-oriented JavaScript, HTML5, CSS3, SASS and LESS Advance in jQuery, Bootstrap and Angular JavaScript framework Expertized knowledge on JavaScript framework with Responsive Web Design such as tablets and mobile devices etc. Strong Code analyzing/validation skills Advance Knowledge of ...
Apply
Html Developer
HARJAI COMPUTERS PRIVATE LIMITED
Greetings   We are an ISO 9001: 2015 IT Out Sourcing Company currently providing IT Services to almost 200 Companies, which Includes CMM, PCMM,
Html Developer
Gharondaa Advisors Private Limited
Skills Required: As a Java Full Stack Developer with Angular 2/TypeScript, you are expected to work on end-to-end web development
HTML Developer
Confidential
HTML Developer Responsibilities: • Meeting with Web designers to discuss project design and layout. • Coding the entire HTML 5 site from end to end. •
Type here to search
30°C Mostly sunny
ENG
15:49
18-11-2022


stl
dsa
yes!
Classroom
PSG - PAT
Download App
For Employers
Help
shine.in
Job title, skills
Login
Register
HOME
JOBS
JOBS SEEKING ASSISTANCE
COURSES
CAREER GUIDANCE
BLOG
Home » Job Search » IT - Software Jobs » Lead Engineer - Electrical & Electronics
Waiting for bat.bing.com...
Type here to search
30°C Mostly sunny
ENG
15:44
18-11-2022

10.ADVANTAGES AND DISADVANTAGES

Advantages:

1. Employment Opportunities:

The foremost advantage of having a profile in our application is that it is your doorway into employment opportunities worldwide. Before the advent of online job applications , students would get jobs through connections. However, now your job opportunities have increased magnanimously. Students who have attained education abroad can put in their area of specialization and find an appropriate job. Apart from this, if there is a particular company that you're interested in, you can make applications for the same.

2. Easy Job Applications:

The traditional recruiting process has taken a back seat and online job application has become paramount. Gone are the days, where you would have to run around with copies of your resume. With the ease of uploading the necessary information on your profile, not only will the recruiters peruse through your profile but you can update your skills regularly. The initial stress of a job application is reduced because the recruiter is already aware of your skills and wants to explore them further. This gives you an excellent opportunity to capitalize on the same and use the app to its fullest.

3. Initiate Connections:

Apart from receiving a job offer, the connections you establish on your profile help you in the long run. For instance, you may start by connecting with your school and college friends and eventually shift to your colleagues. An alumnus from your university is good connections to have. Having an illustrious list of connections speaks to your strong profile. Having a connection who is working at your dream company can be your pathway to the same. Initiating connections will allow you to analyze industry trends and be at the top of the game.

4. Endorsement and Connections

Collecting endorsements and connections is an excellent way of adding social backing to your profile. As mentioned earlier, having illustrious connections will add value to your profile. Upon receiving endorsement for your skills, employers receive extra confidence in your profile. The trick now is to not only have relevant skills but also make your profile stand out.

Disadvantages:

1. Risk of identity theft

There are loads of personal information that you have to display on your profile for prospective employers to see. Hence, in a case whereby Linkedln servers develop an issue, you stand a risk of losing important information to the public, resulting in identity theft.

2. Incomplete profile challenge

LinkedIn like other social network websites required you to put up an attractive profile. That is a profile that is appealing to employers and prospective recruiters. People however find it hard to fill out profile details completely due to one reason or the other.

3. Tons of spam messages

There's a saying that among 12 disciples there will always be a Judas. Think of how many Judas will be available on a website with over 1200 million people. LinkedIn is filled with spam messages from recruiters, employers, and even job seekers. All just to seek attention, mislead, and extort money, etc.

4. Premium package can be expensive

Good thing they say doesn't come cheap. Although, LinkedIn allows you to join the platform without paying. But the LinkedIn premium packages are charged for. For example, the "medium-sized career" price is just about $29.99/month. There are so many added benefits that this offer brings but can still be very costly for a starter or medium-sized business.

11.CONCLUSION

We proposed an application for job recommendation task. This application facilitates the opportunities 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.

12.FUTURE SCOPE

The future is and will remain unknown to us , but fact is that. Till there is a issue of unemployment in the world of job market, the job portals will exist and will grow in proportionate with demand .

13.APPENDIX

Source code:

Index.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8" />
  <link rel="icon" type="image/svg+xml" href="cv.png" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Job Search</title>
</head>

<body>
  <div id="root"></div>
  <script type="module" src="/src/main.jsx"></script>
</body>

</html>
```

Main.py

```
from backend import create_app
import os

app = create_app()

port = os.environ.get("PORT", 5000)


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

index.css

```
@import url("https://fonts.googleapis.com/css2?family=Ubuntu&display=swap");

@tailwind base;
@tailwind components;
@tailwind utilities;

:root {
  font-family: Inter, Avenir, Helvetica, Arial, sans-serif;
  font-size: 16px;
  line-height: 24px;
```

```
  font-weight: 400;

  color-scheme: light;
  /* color: rgba(255, 255, 255, 0.87);
  background-color: #242424; */

  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;
}

#profile-card {
  background-image: url("data:image/svg+xml,%3csvg xmlns='http://www.w3.org/2000/svg' version='1.1'
xmlns:xlink='http://www.w3.org/1999/xlink' xmlns:svgjs='http://svgjs.com/svgjs' width='1440' height='560'
preserveAspectRatio='none' viewBox='0 0 1440 560'%3e%3cg
mask='url(%26quot%3b%23SvgjsMask1024%26quot%3b)' fill='none'%3e%3cpath
d='M1140.1901296559531 116.34762677598815L1129.4188620722516-6.7685250727561765
1006.3027102235073 4.0027425109452395 1017.0739778072088 127.11889435968956z'
fill='rgba(200%2c 217%2c 241%2c 0.4)' class='triangle-float2'%3e%3c/path%3e%3cpath d='M298.23
260.8 a157.93 157.93 0 1 0 315.86 0 a157.93 157.93 0 1 0 -315.86 0z' fill='rgba(200%2c 217%2c 241%2c
0.4)' class='triangle-float3'%3e%3c/path%3e%3cpath d='M778.59 516.07 a117.52 117.52 0 1 0 235.04 0
a117.52 117.52 0 1 0 -235.04 0z' fill='rgba(200%2c 217%2c 241%2c 0.4)' class='triangle-
float1'%3e%3c/path%3e%3cpath d='M468.9084559539883 415.4002443076678L357.01761862323065
435.12961782314227 376.7469921387052 547.0204551539 488.6378294694628 527.2910816384255z'
fill='rgba(200%2c 217%2c 241%2c 0.4)' class='triangle-float2'%3e%3c/path%3e%3cpath d='M33.91
344.01 a115.57 115.57 0 1 0 231.14 0 a115.57 115.57 0 1 0 -231.14 0z' fill='rgba(200%2c 217%2c 241%2c
0.4)' class='triangle-float1'%3e%3c/path%3e%3cpath d='M-1.96 18.7 a116.99 116.99 0 1 0 233.98 0
a116.99 116.99 0 1 0 -233.98 0z' fill='rgba(200%2c 217%2c 241%2c 0.4)' class='triangle-
float2'%3e%3c/path%3e%3cpath d='M1091.07 104.37 a170.17 170.17 0 1 0 340.34 0 a170.17 170.17 0 1 0
-340.34 0z' fill='rgba(200%2c 217%2c 241%2c 0.4)' class='triangle-float2'%3e%3c/path%3e%3cpath
d='M694.1712056255251 443.99968549638584L565.7613309753356 394.70775287477215
516.4693983537218 523.1176275249617 644.8792730039114 572.4095601465754z' fill='rgba(200%2c
217%2c 241%2c 0.4)' class='triangle-float2'%3e%3c/path%3e%3cpath d='M1156.75 235.79 a127.2 127.2 0
1 0 254.4 0 a127.2 127.2 0 1 0 -254.4 0z' fill='rgba(200%2c 217%2c 241%2c 0.4)' class='triangle-
float2'%3e%3c/path%3e%3c/g%3e%3cdefs%3e%3cmask id='SvgjsMask1024'%3e%3crect width='1440'
height='560' fill='white'%3e%3c/rect%3e%3c/mask%3e%3cstyle%3e %40keyframes float1 %7b
0%25%7btransform: translate(0%2c 0)%7d 50%25%7btransform: translate(-10px%2c 0)%7d
100%25%7btransform: translate(0%2c 0)%7d %7d .triangle-float1 %7b animation: float1 5s infinite%3b
%7d %40keyframes float2 %7b 0%25%7btransform: translate(0%2c 0)%7d 50%25%7btransform:
translate(-5px%2c -5px)%7d 100%25%7btransform: translate(0%2c 0)%7d %7d .triangle-float2 %7b
animation: float2 4s infinite%3b %7d %40keyframes float3 %7b 0%25%7btransform: translate(0%2c
0)%7d 50%25%7btransform: translate(0%2c -10px)%7d 100%25%7btransform: translate(0%2c 0)%7d
%7d .triangle-float3 %7b animation: float3 6s infinite%3b %7d %3c/style%3e%3c/defs%3e%3c/svg%3e");
```

```
  background-size: contain;
  background-repeat: repeat-x;
  background-position: center;
}
```

Main.jsx

```
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App'
import './index.css'

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
)
```

App.jsx

```
import { useEffect } from "react";
import { HashRouter, Route, Routes } from "react-router-dom";
import SignUp from "../src/screens/Signup";
import Alert from "./components/Alert";
import Navbar from "./components/Navbar";
import { AppProvider } from "./context/AppContext";
import Dashboard from "./screens/Dashboard";
import Login from "./screens/Login";
import Profile from "./screens/Profile";

function App() {
  useEffect(() => {
    window.watsonAssistantChatOptions = {
      integrationID: "9dc98458-3a56-424f-92f2-d5bbaff3915b", // The ID of this integration.
      region: "au-syd", // The region your integration is hosted in.
      serviceInstanceID: "8d893bea-6198-4677-aca6-2e871cac49db", // The ID of your service instance.
      onLoad: function (instance) {
        instance.render();
      },
    };
    setTimeout(function () {
      const t = document.createElement("script");
      t.src =
        "https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
        (window.watsonAssistantChatOptions.clientVersion || "latest") +
        "/WatsonAssistantChatEntry.js";
      document.head.appendChild(t);
    });
  }, []);
```

```
  return (
    <HashRouter>
      <AppProvider>
        <Navbar />
        <Alert />
        <Routes>
          <Route path="/" element={<Login />} />
          <Route path="/signup" element={<SignUp />} />
          <Route path="/dashboard" element={<Dashboard />} />
          <Route path="/profile" element={<Profile />} />
        </Routes>
      </AppProvider>
    </HashRouter>
  );
}

export default App;
```

Appcontext.jsx

```
import { createContext, useEffect, useState } from "react";
import { useNavigate } from "react-router-dom";

export const AppContext = createContext();

export const AppProvider = ({ children }) => {
  const navigate = useNavigate();

  const [skills, setSkills] = useState([]);

  const [user, setUser] = useState(null);

  const [showAlert, setShowAlert] = useState(null);

  useEffect(() => {
    let temp_user = JSON.parse(localStorage.getItem("user"));
    if (!temp_user) {
      navigate("/");
    } else {
      setUser(temp_user);
    }
  }, []);

  return (
    <AppContext.Provider
      value={{ user, setUser, showAlert, setShowAlert, skills, setSkills }}
    >
      {children}
    </AppContext.Provider>
  );
};
```

Backend_api.js

```
import { BASE_URL } from "../utils/helper";

export const loginUser = async (inputs) => {
  try {
    const response = await fetch(`${BASE_URL}/auth/login`, {
      method: "POST",
      body: JSON.stringify(inputs),
      headers: {
        "Content-Type": "application/json",
      },
    });
    const data = await response.json();
    return data;
  } catch (error) {
    console.error(error);
  }
};

export const registerUser = async (inputs) => {
  try {
    const response = await fetch(`${BASE_URL}/auth/signup`, {
      method: "POST",
      body: JSON.stringify(inputs),
      headers: {
        "Content-Type": "application/json",
      },
    });
    const data = await response.json();
    return data;
  } catch (error) {
    console.error(error);
  }
};

export const getUserSkills = async (token) => {
  try {
    const response = await fetch(`${BASE_URL}/user/skills`, {
      method: "GET",
      headers: {
        Authorization: `Bearer ${token}`,
        "Content-Type": "application/json",
      },
    });
    if (response.ok) {
      const { skills } = await response.json();
      return skills;
    } else {
```

```
      return null;
    }
  } catch (error) {
    console.error(error);
  }
};

export const saveUserSkills = async (skills, token) => {
  try {
    const response = await fetch(`${BASE_URL}/user/skills`, {
      method: "POST",
      body: JSON.stringify({ skills }),
      headers: {
        Authorization: `Bearer ${token}`,
        "Content-Type": "application/json",
      },
    });
    if (response.ok) {
      return true;
    } else {
      return false;
    }
  } catch (error) {
    console.error(error);
  }
};

export const removeUserSkills = async (skills, token) => {
  try {
    const response = await fetch(`${BASE_URL}/user/skills`, {
      method: "DELETE",
      body: JSON.stringify({ skills }),
      headers: {
        Authorization: `Bearer ${token}`,
        "Content-Type": "application/json",
      },
    });
    if (response.ok) {
      return true;
    } else {
      return false;
    }
  } catch (error) {
    console.error(error);
  }
};

export const updateUserDetails = async (inputs, token) => {
  try {
    const response = await fetch(`${BASE_URL}/user/profile`, {
      method: "POST",
```

```
      body: JSON.stringify(inputs),
      headers: {
        Authorization: `Bearer ${token}`,
        "Content-Type": "application/json",
      },
    });
    if (response.ok) {
      const data = await response.json();
      return data;
    } else {
      return null;
    }
  } catch (error) {
    console.error(error);
  }
};
```

Dashboard.jsx

```
import axios from "axios";
import React, { useContext, useEffect, useState } from "react";
import { Link } from "react-router-dom";
import JobCard from "../components/JobCard";
import SearchBar from "../components/SearchBar";
import Skill from "../components/Skill";
import { AppContext } from "../context/AppContext";
import { getUserSkills } from "../proxies/backend_api";

const Dashboard = () => {
  const { user, skills, setSkills } = useContext(AppContext);

  const [filterUsingSkills, setFilterUsingSkills] = useState(false);

  const [query, setquery] = useState("");
  const [posts, setposts] = useState(null);
  const id = import.meta.env.VITE_ADZUNA_API_ID;
  const key = import.meta.env.VITE_ADZUNA_API_KEY;
  const baseURL =
`http://api.adzuna.com/v1/api/jobs/in/search/1?app_id=${id}&app_key=${key}&results_per_page=15&wh
at=${query}&what_and=${skills.join(
    " "
  )}&&content-type=application/json`;

  const baseURL2 =
`http://api.adzuna.com/v1/api/jobs/in/search/1?app_id=${id}&app_key=${key}&results_per_page=15&wh
at=${query}&content-type=application/json`;

  const searchJobsFromQuery = async () => {
    const { data } = await axios.get(filterUsingSkills ? baseURL : baseURL2);
    setposts(data.results);
```

```
};

useEffect(() => {
  if (user) {
    (async () => {
      setSkills(await getUserSkills(user.token));
    })();
  }
}, [user]);

useEffect(() => {
  searchJobsFromQuery();
}, [filterUsingSkills]);

return (
  <div className="flex gap-10 my-10 lg:my-24 mx-20 lg:mx-40">
    <div className="hidden lg:block bg-primary w-1/5 p-10 h-3/6 rounded-lg">
      <div className="text-xl text-white capitalize font-extrabold mb-6">
        Your skills
      </div>
      <ul className="list-none text-gray-200 flex flex-col gap-2">
        {skills.map((skill, ind) => (
          <Skill skill={skill} key={ind} checked={filterUsingSkills} />
        ))}
      </ul>
      <button
        className="p-2 bg-white text-primary rounded mt-5"
        onClick={() => {
          setFilterUsingSkills(!filterUsingSkills);
          searchJobsFromQuery();
        }}
      >
        Include your skills
      </button>
    </div>

    <div className="mx-auto">
      <SearchBar setquery={setquery} onClick={searchJobsFromQuery} />
      {query === "" ? (
        <h2 className="text-2xl mt-5">Recommended Jobs</h2>
      ) : (
        <h2 className="text-2xl mt-5">
          Search for keywords {query}
          {filterUsingSkills && `,${skills.join(",")}`}
        </h2>
      )}

      <div className="mt-10 grid grid-cols-1 lg:grid-cols-3 md:grid-cols-2">
        {query !== null ? (
          posts?.map((post, ind) => (
            <JobCard
```

```
            key={ind}
            title={post.title}
            company={post.company.display_name}
            description={post.description}
            link={post.redirect_url}
          />
        ))
      ) : (
        <></>
      )}
    </div>
  </div>
 </div>
);
};

export default Dashboard;
```

login.jsx

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

const Login = () => {
 const { setShowAlert, 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;
  };

  const handleLogin = async () => {
    if (checkInputErrors()) {
      const data = await loginUser(inputs);
      if (data.error) {
        setShowAlert({ type: "error", message: data.error, duration: 3000 });
        return;
      }
      setUser(data);
      setShowAlert({
        type: "success",
        message: `Welcome back ${data.name}`,
        duration: 3000,
      });
      localStorage.setItem("user", JSON.stringify(data));
      navigate("/dashboard");
    }
  };

  return (
    <div className="flex flex-col justify-center items-center gap-10 mt-5">
      <div>
        <button className="bg-base-300 rounded-box flex flex-row justify-evenly items-center gap-10 px-10
py-5 w-fit 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
```

```
        onSubmit={(e) => e.preventDefault()}
        className="card bg-base-300 rounded-box flex flex-col justify-center items-center gap-5 px-10 py-5
w-fit mx-auto"
      >
        <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 font-medium">
              {error.email}
            </p>
          )}
        </div>
        <div>
          <input
            value={inputs.password}
            type="password"
            name="password"
            placeholder="password"
            className="input input-bordered input-primary w-full"
            onChange={handleChange}
          />
          {error.password !== "" && (
            <p className="text-sm text-red-500 mt-1 font-medium">
              {error.password}
            </p>
          )}
        </div>
        <div className="text-center">
          <button
            type="submit"
            onClick={handleLogin}
            className="btn btn-sm btn-primary mb-4"
          >
            Login
          </button>
          <p>
            Don't have an account?{" "}
            <Link className="text-blue-400" to="/signup">
              Sign up
            </Link>
          </p>
        </div>
      </form>
    </div>
```

```
    </div>
  );
};

export default Login;
```

Profile.jsx

```
import React, { useContext, useEffect, useState } from "react";
import { AiOutlineClose } from "react-icons/ai";
import { BsLinkedin } from "react-icons/bs";
import { GoMarkGithub } from "react-icons/go";
import { MdDeleteForever } from "react-icons/md";
import { RiEdit2Fill } from "react-icons/ri";
import { TfiTwitterAlt } from "react-icons/tfi";
import { VscAdd } from "react-icons/vsc";
import { AppContext } from "../context/AppContext";
import {
  getUserSkills,
  removeUserSkills,
  saveUserSkills,
  updateUserDetails,
} from "../proxies/backend_api";

const Profile = () => {
  const { user, setShowAlert, setUser, skills, setSkills } =
    useContext(AppContext);

  const [addSkill, setAddSkill] = useState("");

  const [newSkills, setNewSkills] = useState([]);

  const [removedSkills, setRemovedSkills] = useState([]);

  const [isEditingEnabled, setIsEditingEnabled] = useState(false);

  const [userInfo, setUserInfo] = useState({
    name: "",
    phone_number: "",
  });

  const handleUserInfoChange = ({ target: { name, value } }) => {
    setUserInfo((prev) => ({ ...prev, [name]: value }));
  };

  const changeSkills = () => {
    if (
      addSkill !== "" &&
      !skills.find((item) => item.toLowerCase() === addSkill.toLowerCase())
    ) {
```

```
    setNewSkills((prev) => [...prev, addSkill.trim()]);
    setSkills((prev) => [...prev, addSkill.trim()]);
  }
  setAddSkill("");
};

const removeSkills = (skill_name) => {
  setRemovedSkills((prev) => [...prev, skill_name]);
  setSkills((prev) => prev.filter((item) => item !== skill_name));
  setNewSkills((prev) => prev.filter((item) => item !== skill_name));
};

const updateSkills = async () => {
  let skillsAdded = false,
    skillsRemoved = false;
  if (newSkills.length !== 0) {
    skillsAdded = await saveUserSkills(newSkills, user.token);
  }
  if (removeSkills.length !== 0) {
    skillsRemoved = await removeUserSkills(removedSkills, user.token);
  }
  if (skillsAdded || skillsRemoved) {
    setShowAlert({
      type: "success",
      message: "Profile updated!",
      duration: 3000,
    });
  }
  setNewSkills([]);
  setRemovedSkills([]);
};

const updateUserInfo = async () => {
  const data = await updateUserDetails(userInfo, user.token);
  if (data) {
    setUser((prev) => {
      prev = { ...prev, name: data.name, phone_number: data.phone_number };
      localStorage.setItem("user", JSON.stringify(prev));
      return prev;
    });
    setShowAlert({
      type: "success",
      message: "Profile updated!",
      duration: 3000,
    });
  }
  setIsEditingEnabled(false);
};
```



```
useEffect(() => {
  if (user) {
```

```
    (async () => {
      let data = await getUserSkills(user?.token);
      if (data) setSkills(data);
    })();
    setUserInfo({
      name: user.name,
      phone_number: user.phone_number,
    });
  }
}, [user]);

return (
  <div className="my-5 mx-10">
    <div className="border-2 border-blue-100 w-full h-fit rounded-xl p-5 flex flex-col gap-3">
      <div className="flex justify-between w-full min-h-[25vh]">
        <div className="flex flex-col justify-between">
          <h1 className="md:text-2xl  text-xl font-medium flex items-center gap-4">
            Your Profile{" "}
            <button>
              {isEditingEnabled ? (
                <AiOutlineClose
                  color="#ff8977"
                  onClick={() => setIsEditingEnabled(!isEditingEnabled)}
                />
              ) : (
                <RiEdit2Fill
                  color="#4506cb"
                  onClick={() => setIsEditingEnabled(!isEditingEnabled)}
                />
              )}
            </button>
          </h1>
          <div className="flex flex-col gap-3">
            {isEditingEnabled ? (
              <>
                <input
                  name="name"
                  value={userInfo.name}
                  className="input input-bordered w-full input-xs p-3 text-lg input-primary"
                  type="text"
                  placeholder="name"
                  onChange={handleUserInfoChange}
                />
                <input
                  disabled
                  value={user?.email}
                  className="input input-bordered w-full input-xs p-3 text-lg input-primary"
                  type="text"
                  placeholder="name"
                />
                <input
```

```
            name="phone_number"
            value={userInfo.phone_number}
            className="input input-bordered w-full input-xs p-3 text-lg input-primary"
            type="number"
            placeholder="phone number"
            onChange={handleUserInfoChange}
          />
          <button
            className="btn btn-xs btn-outline btn-primary"
            onClick={updateUserInfo}
          >
            Update
          </button>
        </>
      ) : (
        <>
          <h2 className="md:text-2xl xl:text-2xl sm:text-xl">
            {user?.name}
          </h2>
          <p className="md:text-xl sm:text-md text-gray-700">
            {user?.email}
          </p>
          <span className="text-gray-700">{user?.phone_number}</span>
        </>
      )}
    </div>
  </div>
  <div className="flex flex-col justify-end w-fit gap-4">
    <img
      src="avatar.webp"
      alt="profile"
      className="md:w-36 w-20 rounded-md object-contain"
    />
    {/* <button className="btn btn-outline btn-active btn-sm">
      change
    </button> */}
  </div>
</div>
<div className="divider my-2"></div>
<div className="flex flex-col">
  <div className="flex justify-between gap-2 flex-col">
    <h4 className="text-xl">Skills</h4>
    <form
      className="flex gap-5 items-center"
      onSubmit={(e) => e.preventDefault()}
    >
      <input
        autoComplete="off"
        value={addSkill}
        type="text"
        name="addSkill"
```

```
      placeholder="add addSkill"
      onChange={(e) => setAddSkill(e.target.value)}
      className="input input-bordered w-full input-primary max-w-xl input-sm"
    />

    <button
      className="hover:rotate-90 transition-all"
      onClick={changeSkills}
    >
      <VscAdd size={20} />
    </button>
  </form>
  <ul className="flex gap-2 flex-wrap">
    {skills?.map((addSkill, ind) => (
      <li
        className="bg-indigo-100 rounded p-2 flex gap-2 items-center"
        key={ind}
      >
        {addSkill}
        <MdDeleteForever
          color="#ff8977"
          onClick={() => removeSkills(addSkill)}
          size={20}
        />
      </li>
    ))}
  </ul>
  <button
    className="btn btn-sm w-fit btn-primary"
    type="button"
    onClick={updateSkills}
  >
    Save
  </button>
</div>
<div className="divider my-2"></div>
<div className="flex justify-between gap-2 flex-col">
  <h4 className="text-xl">Resume/Portfolio</h4>
  <div className="flex gap-5">
    <input
      className="input input-bordered w-full input-primary max-w-xl input-sm"
      type="text"
      placeholder="paste the link"
    />
    <button className="btn btn-primary btn-sm">update</button>
  </div>
</div>
<div className="divider my-2"></div>
<div className="flex gap-2 flex-col">
  <h3 className="text-xl">Socials</h3>
  <div className="flex flex-col gap-2">
```

```
            <div className="flex gap-5 items-center">
              <GoMarkGithub size={20} />
              <input
                type="text"
                placeholder="paste the link"
                className="border-2 border-gray-300 rounded-md px-3 my-1 max-w-md"
              />
            </div>
            <div className="flex gap-5 items-center">
              <BsLinkedin size={20} />
              <input
                type="text"
                placeholder="paste the link"
                className="border-2 border-gray-300 rounded-md px-3 my-1 max-w-md"
              />
            </div>
            <div className="flex gap-5 items-center">
              <TfiTwitterAlt size={20} />
              <input
                type="text"
                placeholder="paste the link"
                className="border-2 border-gray-300 rounded-md px-3 my-1 max-w-md"
              />
            </div>
            <button className="btn btn-primary btn-sm max-w-fit">save</button>
          </div>
        </div>
      </div>
    </div>
  </div>
 );
};

export default Profile;
```

Signup.jsx

```
import React, { useContext, useState } from "react";
import { Link, useNavigate } from "react-router-dom";
import { AppContext } from "../context/AppContext";
import { registerUser } from "../proxies/backend_api";
import { emailRegex } from "../utils/helper";

const SignUp = () => {
  const { setShowAlert, setUser } = useContext(AppContext);

  const navigate = useNavigate();

  const [inputs, setInputs] = useState({
    name: "",
    email: "",
```

```
  phone_number: "",
  password: "",
  confirm_password: "",
});

const [error, setErrors] = useState({
  name: "",
  email: "",
  phone_number: "",
  password: "",
  confirm_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.name.trim() === "") {
    setErrors((prev) => {
      return { ...prev, name: "Enter a valid name" };
    });
    status = false;
  }

  if (inputs.phone_number.trim() === "") {
    setErrors((prev) => {
      return { ...prev, phone_number: "Enter a valid phone number" };
    });
    status = false;
  }

  if (inputs.confirm_password.trim() === "") {
    setErrors((prev) => {
      return { ...prev, confirm_password: "Enter a valid  password" };
    });
    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;
    }

    if (inputs.password.trim() !== inputs.confirm_password.trim()) {
      setErrors((prev) => {
        return { ...prev, confirmPassword: "Password don't match" };
      });
      status = false;
    }
    return status;
  };

  const handleSignUp = async () => {
    if (checkInputErrors()) {
      const data = await registerUser(inputs);
      if (data.error) {
        setShowAlert({ type: "error", message: data.error, duration: 3000 });
        return;
      }
      setUser(data);
      setShowAlert({
        type: "success",
        message: `Your journey starts here ${data.name}`,
        duration: 3000,
      });
      localStorage.setItem("user", JSON.stringify(data));
      navigate("/profile");
    }
  };

  return (
    <div className="flex flex-col justify-center items-center gap-10 mt-5">
      <div>
        <button className="bg-base-300 rounded-box flex flex-row justify-evenly items-center gap-10 px-10
py-5 w-fit 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>
        <div className="card bg-base-300 rounded-box flex flex-col justify-center items-center gap-3 px-10
py-5 w-fit mx-auto">
```

```
<div>
  <input
    value={inputs.name}
    type="text"
    name="name"
    placeholder="name"
    className="input input-bordered input-primary w-full"
    onChange={handleChange}
  />
  {error.name !== "" && (
    <p className="text-sm text-red-500  font-medium">{error.name}</p>
  )}
</div>
<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  font-medium">{error.email}</p>
  )}
</div>
<div>
  <input
    value={inputs.phone_number}
    type="text"
    name="phone_number"
    placeholder="phone number"
    className="input input-bordered input-primary w-full"
    onChange={handleChange}
  />
  {error.phone_number !== "" && (
    <p className="text-sm text-red-500  font-medium">
      {error.phone_number}
    </p>
  )}
</div>
<div>
  <input
    value={inputs.password}
    type="password"
    name="password"
    placeholder="password"
    className="input input-bordered input-primary w-full"
    onChange={handleChange}
  />
  {error.password !== "" && (
```

```
            <p className="text-sm text-red-500  font-medium">
              {error.password}
            </p>
          )}
        </div>
        <div>
          <input
            value={inputs.confirm_password}
            type="password"
            name="confirm_password"
            placeholder="confirm password"
            className="input input-bordered input-primary w-full"
            onChange={handleChange}
          />
          {error.confirm_password !== "" && (
            <p className="text-sm text-red-500  font-medium">
              {error.confirm_password}
            </p>
          )}
        </div>
        <div className="text-center">
          <button
            onClick={handleSignUp}
            className="btn btn-sm btn-primary mb-4"
          >
            Sign Up
          </button>
          <p>
            Already have an account?{" "}
            <Link className="text-blue-400" to="/">
              Sign in
            </Link>
          </p>
        </div>
      </div>
    </div>
  </div>
 );
};

export default SignUp;
```

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"
  }
}
```

Dockerfile

```
# Build step #1: build the React front end
FROM node:16-alpine as react-builder
WORKDIR /app
ENV PATH /app/node_modules/.bin:$PATH
COPY package.json ./
COPY ./src ./src
COPY ./public ./public
COPY ./index.html ./vite.config.js ./postcss.config.cjs ./tailwind.config.cjs ./.env  ./
RUN npm install
RUN npm run build

# Build step #2: build the API with the client as static files
FROM python:3.10
WORKDIR /app
COPY --from=react-builder /app/dist ./dist
COPY main.py ./main.py

RUN mkdir ./backend
COPY backend/ ./backend/
RUN pip install -r ./backend/requirements.txt

EXPOSE 5000
ENTRYPOINT ["python","main.py"]
```

Alert.jsx

```
import React, { useContext, useEffect, useState } from "react";
import { AppContext } from "../context/AppContext";

const alert_options = {
  success: {
    bg: "bg-green-100  dark:bg-green-200",
    text: "text-green-700 dark:text-green-800",
    icon: (
      <svg
        xmlns="http://www.w3.org/2000/svg"
        fill="none"
        viewBox="0 0 24 24"
        className="stroke-current flex-shrink-0 w-6 h-6"
      >
        <path
          strokeLinecap="round"
          strokeLinejoin="round"
          strokeWidth="2"
          d="M13 16h-1v-4h-1m1-4h.01M21 12a9 9 0 11-18 0 9 9 0 0118 0z"
        ></path>
      </svg>
    ),
  },
  error: {
    bg: "bg-red-100  dark:bg-red-200",
    text: "text-red-700 dark:text-red-800",
    icon: (
      <svg
        xmlns="http://www.w3.org/2000/svg"
        className="stroke-current flex-shrink-0 h-6 w-6"
        fill="none"
        viewBox="0 0 24 24"
      >
        <path
          strokeLinecap="round"
          strokeLinejoin="round"
          strokeWidth="2"
          d="M10 14l2-2m0 0l2-2m-2 2l-2-2m2 2l2 2m7-2a9 9 0 11-18 0 9 9 0 0118 0z"
        />
      </svg>
    ),
  },
  warning: {
    bg: "bg-yellow-100  dark:bg-yellow-200",
    text: "text-yellow-700 dark:text-yellow-800",
    icon: (
      <svg
        xmlns="http://www.w3.org/2000/svg"
        className="stroke-current flex-shrink-0 h-6 w-6"
```

```
        fill="none"
        viewBox="0 0 24 24"
      >
        <path
          strokeLinecap="round"
          strokeLinejoin="round"
          strokeWidth="2"
          d="M12 9v2m0 4h.01m-6.938 4h13.856c1.54 0 2.502-1.667 1.732-3L13.732 4c-.77-1.333-2.694-
1.333-3.464 0L3.34 16c-.77 1.333.192 3 1.732 3z"
        />
      </svg>
    ),
  },
  info: {
    bg: "bg-blue-100  dark:bg-blue-200",
    text: "text-blue-700 dark:text-blue-800",
    icon: (
      <svg
        xmlns="http://www.w3.org/2000/svg"
        className="stroke-current flex-shrink-0 h-6 w-6"
        fill="none"
        viewBox="0 0 24 24"
      >
        <path
          strokeLinecap="round"
          strokeLinejoin="round"
          strokeWidth="2"
          d="M9 12l2 2 4-4m6 2a9 9 0 11-18 0 9 9 0 0118 0z"
        />
      </svg>
    ),
  },
};

const Alert = () => {
  const { showAlert, setShowAlert } = useContext(AppContext);
  const [animateClasses, setAnimateClasses] = useState("opacity-0 h-0");
  const [styleClasses, setStyleClasses] = useState("");

  useEffect(() => {
    if (showAlert) {
      setAnimateClasses("opacity-100  mt-5 ");
      setTimeout(() => {
        setAnimateClasses("opacity-0 mt-0");
        setTimeout(() => {
          setShowAlert(null);
          setAnimateClasses("opacity-0 h-0 mt-0");
        }, 10);
      }, showAlert.duration);
    }
  }, [showAlert]);
```

```
  return (
    <div
      className={`absolute top-14 right-5 transition-all duration-850 ease-in-out flex p-4 rounded-lg
${animateClasses} ${
        alert_options[showAlert?.type]?.bg
      }`}
      role="alert"
    >
      {alert_options[showAlert?.type]?.icon}
      <div
        className={`ml-3 text-sm font-medium ${
          alert_options[showAlert?.type]?.text
        }`}
      >
        {showAlert?.message}
      </div>
    </div>
  );
};

export default Alert;
```

JobCard.jsx

```
import React from "react";

const JobCard = ({ title, company, description, link }) => {
  return (
    <div className="max-w-sm flex flex-col rounded overflow-hidden shadow-lg">
      <div className="px-6 py-4">
        <div className="font-bold text-xl">{title}</div>
        <div className="text-lg mb-2 text-gray-400">{company}</div>
        <p className="text-ellipsis overflow-hidden text-gray-800 text-base">
          {description}
        </p>
      </div>
      <div className="px-6 pt-4 pb-2 mt-auto mb-2">
        <a
          href={link}
          target="__blank"
          className="bg-transparent hover:bg-primary text-primary font-semibold hover:text-white py-2
mb-0 mt-4 px-4 border border-primary hover:border-transparent rounded"
        >
          apply
        </a>
      </div>
    </div>
  );
};
```

```
export default JobCard;
```

Navbar.jsx

```
import React, { useContext } from "react";
import { Link, useNavigate } from "react-router-dom";
import { AppContext } from "../context/AppContext";

const Navbar = () => {
  const navigate = useNavigate();
  const { user, setUser, setSkills } = useContext(AppContext);

  const logout = () => {
    setUser(null);
    setSkills([]);
    localStorage.removeItem("user");
    navigate("/");
  };

  return (
    <div className="navbar bg-base-100 border-b-2">
      <div className="flex-1">
        <a
          className="btn btn-ghost normal-case text-xl"
          onClick={() => navigate(user ? "/dashboard" : "/")}
        >
          F-ing Jobs
        </a>
      </div>
      {user && (
        <div className="flex-none gap-2">
          <div className="dropdown dropdown-end">
            <label tabIndex={0} className="btn btn-ghost btn-circle avatar ">
              <div className="w-10 rounded-full ring ring-opacity-50 ring-purple-700">
                <img src="https://placeimg.com/80/80/people" />
              </div>
            </label>
            <ul
              tabIndex={0}
              className="mt-3 p-2 shadow menu menu-compact dropdown-content bg-base-100 rounded-box
w-52"
            >
              <li>
                <a
                  className="justify-between"
                  onClick={() => navigate("/profile")}
                >
                  Profile
                </a>
              </li>
```

```
            <li>
              <a onClick={logout}>Logout</a>
            </li>
          </ul>
        </div>
      </div>
    )}
  </div>
 );
};

export default Navbar;
```

SearchBar.jsx

```
import React from "react";
import { BsSearch } from "react-icons/bs";

const SearchBar = ({ setquery, onClick }) => {
 const handlesubmit = (e) => {
  e.preventDefault();
  onClick();
 };

 return (
  <form className="flex items-center" onSubmit={handlesubmit}>
   <label htmlFor="simple-search" className="sr-only">
    Search
   </label>
   <div className="relative w-full">
    <div className="flex absolute inset-y-0 left-0 items-center pl-3 pointer-events-none">
     <BsSearch />
    </div>
    <input
     onChange={(e) => setquery(e.target.value)}
     name="search"
     type="text"
     id="simple-search"
     className="bg-gray-50 border border-gray-300 text-gray-900 text-sm rounded-lg focus:ring-blue-
500 focus:border-blue-500 block w-full pl-10 p-2.5  dark:bg-gray-700 dark:border-gray-600
dark:placeholder-gray-400 dark:text-white dark:focus:ring-blue-500 dark:focus:border-blue-500"
     placeholder="Search"
     required=""
    />
   </div>
   <button
    type="submit"
    className="p-2.5 ml-2 text-sm font-medium text-white bg-purple-700 rounded-lg border border-
purple-700 hover:bg-purple-800 focus:ring-4 focus:outline-none focus:ring-purple-300"
   >
```

```
      <BsSearch />
      <span className="sr-only">Search</span>
    </button>
  </form>
 );
};

export default SearchBar;
```

Skill.jsx

```
import React from "react";
import { TiTick } from "react-icons/ti";

const Skill = ({ skill, checked }) => {
 return (
  <li className="hover:text-white cursor-pointer flex gap-1 items-center">
    {skill}
    {checked && <TiTick />}
  </li>
 );
};

export default Skill;
```

Git Hub Link and Project demo Link

Git Hub Link

https://github.com/IBM-EPBL/IBM-Project-37463-1660310204

Project demo link

https://drive.google.com/file/d/1LNXEEc7wz12FnWcpildVyjiUX6NlwXC5/view?usp=drivesdk