TRIBHUVAN UNIVERSITY

INSTITUTE OF SCIENCE AND TECHNOLOGY

BHAKTAPUR MULTIPLE CAMPUS

A Project Report On

"Job Recommendation System using Hybrid Approach"

In partial fulfillment of the requirements for Bachelor Degree in Computer science and Information Technology

Under the supervision of

Mr. Sushant Paudel

Department of CSIT

Bhaktapur Multiple Campus

Submitted By:

Suman Gole, 23274/076

Suman Shrestha, 23276/076

Sushmita Khadka, 23280/076

Submitted To:

Institute of Science and Technology

Tribhuvan University

Kirtipur, Nepal

4th April 2024

SUPERVISOR’S RECOMMENDATION

This is to recommend that **Suman Gole, Suman Shrestha, and Sushmita Khadka** carry out project work entitled **“Job Recommendation System using Hybrid Approch**” for the requirement for the project work in Bachelor of Science (B.Sc.) degree in Computer Science and Information Technology (CSIT) under my supervision in the Department of CSIT, Bhaktapur Multiple Campus, Institute of Science and Technology (IoST), Tribhuvan University (T.U.), Nepal. To my knowledge, this work has not been submitted for any other degree. They have fulfilled all the requirements laid down by the Institute of Science and Technology (IoST), Tribhuvan University (T.U.), Nepal for the submission of the project work for the partial fulfillment of Bachelor of Science (B.Sc.) CSIT degree.

Mr. Sushant Paudel

Supervisor

Department of CSIT

Bhaktapur Multiple Campus

TU

4th April 2024

CERTIFICATE OF APPROVAL

This project work entitled **"Job Recommendation System using Hybrid Approach"** by **Sushmita Khadka, Suman Gole, and Suman Shrestha,** under the supervision of Sushant Poudel in the Department of CSIT, Bhaktapur Multiple Campus, Institute of Science and Technology (IoST), Tribhuvan University (T.U.) is hereby submitted for the partial fulfillment of the Bachelor of Science (B.Sc.) degree in CSIT. This report has been accepted and forwarded to the Controller of Examination, Institute of Science and Technology, Tribhuvan University, Nepal for the legal procedure.

Mr. Sushant Paudel

Supervisor

Department of CSIT:

Bhaktapur Multiple Campus:

TU

HOD

Department:

Bhaktapur Multiple Campus:

TU

External Examiner

Internal Examiner

Department:

Bhaktapur Multiple Campus:

TU

ACKNOWLEDGMENT

We would like to express sincere gratitude to our project supervisor, **Mr. Sushant Paudel**, for his proper guidance and consistent support and input throughout this course of our group project. The way he supported us with his crucial suggestions and recommendations has been invaluable. This experience and expertise in the field of computer science and Information Technology have been instrumental in shaping our ideas and refining the project to its fullest potential. His constructive feedback and his consistent commitment to fostering collaborative learning and developing an environment have been essential for us to accomplish this project.

Furthermore, we would also like to thank, the CSIT department at Bhaktapur Multiple Campus for providing us with the necessary equipment, environment, and coordination with necessary resources. Without such open and unwavering support from the Campus, it would not have been possible for us to accomplish this project. This collaboration has surely enriched our academic journey and also helped us to visualize and realize what it feels like to work in the real Information technology industry.

ABSTRACT

With the rise in job platforms that connect job seekers and employers, there have been increasing challenges for both counterparts to find the right match. Considering these issues, our project **“Job Recommendation System (JRS) using Hybrid Approach”** presents a comprehensive way of recommending jobs to the job seeker as per their skills. In particular, our system uses Hybrid Approach (content-based and collaborative filtering) to increase the precision of the job recommendations. This is done by analyzing the textual connection to the job descriptions and the user profiles.

The algorithmic foundation of our recommendation system utilizes the TFIDF and cosine similarity that ensures the recommendations are based on Keyword matches. This approach does show decent accuracy and effectiveness while suggesting the job which would eventually allow the job seeker and employers to find what they are looking for. Through the implementation of JRS, we aim to streamline the job search which shall allow users to more efficient way of discovering career opportunities that align with their experience and expertise.

Keywords: **Recommendation System, TFIDF, relevant job suggestions, Hybrid Approach**

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

CBF	Content Based Filtering
CF	Collaborative Filtering
CSIT	Computer Science and Information Technology
CSS	Cascading Style Sheet
DFD	Data Flow Diagram
ER	Entity Relationship
HTML	Hyper Text Markup Language
IOST	Institute of Science and Technology
JRS	Job Recommendation System
JS	JavaScript
SQL	Structured Query Language
TF-IDF	Term Frequency-Inverse Document Frequency
TU	Tribhuvan University

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 Introduction

With the rise of the internet, there has been a significant increase in the number of job portals in Nepal and the global way. And, it is not always easy to find a trustworthy and dependable application. However, recommendation systems have made this task relatively easy for both the job seeker and the employer. Recommendation systems find what the user desires to find and with its algorithms, it recommends the relevant job openings to job seekers. Similarly, the employer can also explore job seekers through such a system.

Job Recommendation System (JRS) is a website that is built to provide relevant jobs to job seekers. Similarly, the employer can also post their opening in our system which would make things more interactive in our web platform. As the user fills in the details like skills and preferred industry while signing in, the system would then recommend the available job openings to the user. Our system will collect data about the candidates like personal information, skills, experience, and preferences which would then be fed to the algorithms to recommend the relevant job to the candidate. The recommendation system is one of the main parts of our system. Recommendation systems are a special type of information filtering system. They are software applications that aim to support users in their decision-making while interacting with a vast amount of information. They recommend items of interest to the users based on preferences they have expressed, either explicitly or implicitly [1].

For this, we have used content-based filtering to efficiently extract the necessary information. We utilize the TFIDF, cosine similarity, and person correlation algorithms to recommend the most suitable jobs for the candidates. While there are still some limitations to this system, the algorithms used do show promising results for further developments to make a sustainable system.

1.2 Problem Statement

Even though different job platforms are active in Nepal, we can still see there is a huge gap between job seekers and employers. But the thing is, many companies are hiring teams and not finding one. It is not that there are no employment opportunities; it is that the hirers are not finding a perfect fit for the company, and the seekers aren't able to grab such opportunities [2].

Considering the present context of the job market, it seems that the recommendation of higher-level jobs to entry-level candidates, lack of openness about hiring policies and remunerations, and lack of incentives for the intern or trainee candidates seem major issues as well.

Similarly, many job seeker are still not aware of the working mechanism of the job portals due to their complex signup and navigation pages. As such, the job seeker can be quite disappointed with the portals at times. Similarly, many job seekers and companies are still using the trading way of personal recommendations and visiting the office techniques which greatly reduces the selections of deserving candidates as often there are unprofessional ethics and misuse of the power of high-level employees of the company. As such, at present, even well-skilled personnel are facing difficulties while finding a suitable job on their horizon. In addition to that limited job listing, lack of proper user interface and experience design as well as lack of transparency in the hiring process are among the most disappointing issues for the general job seeker in Nepal in the present context.

1.3 Objectives

JRS is a recommendation system created with the vision of making the job-finding process simplistic, efficient, and more user-friendly. It aims to recommend suitable and relevant jobs to job seekers as per their skills and preferences industries by implementing TFIDF and cosine similarly. Similarly, our system will also allow employers to post their vacancies in a more efficient way. Some of the primary goals of JRS are:

- To recommend the suitable job to the job seeker as per their skills using the Hybrid method including Cosine similarity and Pearson correlation.
- To connect the candidates and employer more efficiently and transparently through responsive web application.
- To enhance the user experience while searching for the job through easy navigation using a simple yet robust user interface through style components.

1.4 Scope and Limitations

1.4.1 Scope

Considering the way our current job platform has been lagging behind in connecting the best-fit candidates and employers, there is huge scope for a JRS to resolve the prevalent issues with its simplicity and efficiency. The usage of content-based filtering along with TFIDF and cosine similarity is what makes JRS a simplistic yet robust system that can also have sustainable business prospects in the future. In addition to our future vision of adding collaborative filtering and making the system hybrid is something that would expand our horizon and become more compelling and competitive to the market.

In addition to the searching and filtering process, as per the skills, remunerations, and preferred industry are going to make the user experience more engaging, and with the feedback options that we have built-in, there would be a continuation of making our system more and more efficient and user friendly to lure more job seeker and employer.

1.4.2 Limitations

JRS relies heavily on the availability and quality of textual information within the user profiles and job descriptions which could impact the precision of recommendations in cases of poorly articulated data. In addition, the user can only be recommended a suitable job if the user has filled in the profile with the required details, else the system's ability to recommend will potentially decrease. Moreover, there can be issues while handling semantic relationships within the job and skill descriptions. The usage of synonyms could still bring down the accuracy of recommendations.

1.5 Development Methodology

To develop JRS, we have used the Agile development methodology. As this methodology allows us to foster flexibility, collaboration, and iterative progress, we did find an enhancement in the development and collaboration speed. Similarly, as Agile development focuses more on adaptive planning, evolutionary development, and customer feedback it properly aligns with our visions for the development of our system.

Reason for Selecting Agile Development Methodology

In traditional software-development process models the requirement analysis and designing of the product is limited/restricted to the initial phase/stage of the process and usually, they follow a sequential order. Adding new requirements at a later phase in the development can increase the cost of the project exponentially [3]. However, in agile methodology, there is a way for us to have consistent feedback, short-cycled iterative improvement, and incremental-delivery mechanism which really helped us to become more productive and efficient while developing this project.

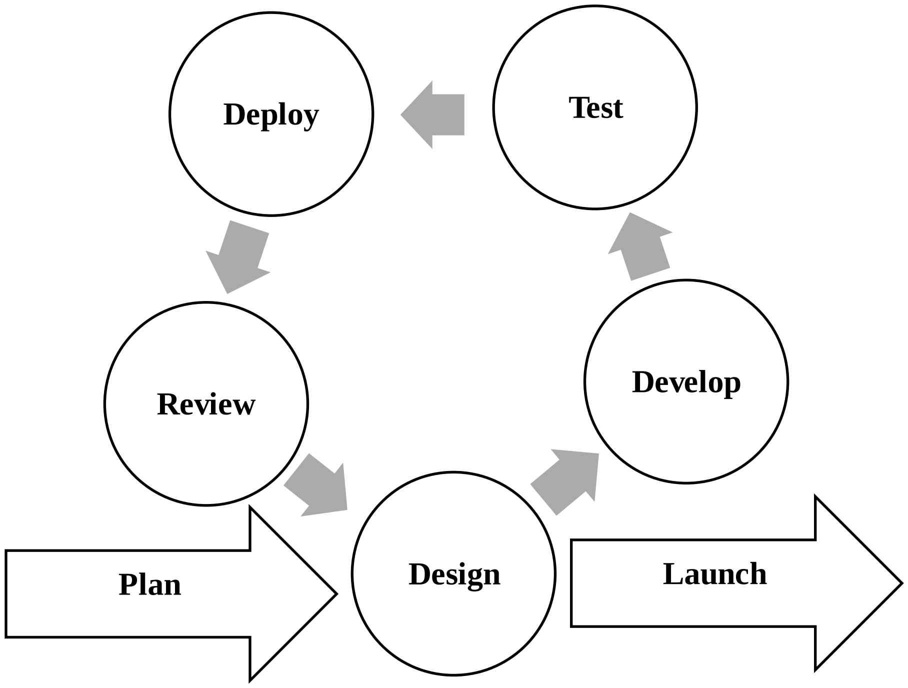


Figure 1: Agile Model

As seen in the figure, first we plan for the projects by gathering the requirements and then brainstorming for the best possible solutions. After that, we design and develop the model for that requirement. Subsequently, we test the design module before deploying it locally. Then after that review is done with the supervisor for further feedback. This process was iteratively followed till we launched JRS in a production line.

Moreover, the key principle of the Agile, iterative, and incremental development process has allowed us to review our steps in the development process and also mage the project into smaller and more manageable interactions or sprints. Similarly, continuous feedback ensures the adjustments efficient in response to the suggested changes. On top of that, the regular sprint reviews facilitated the collaboration between members, stakeholders, and the project supervisor. This shall allow us to maintain a shared understanding of overall project goals and visualize its implementation.

1.6 Report Organization

This report has been organized as per the guidelines provided by TU. The report is separated into different chapters. Each chapter consists of various sub-chapters with its content. The preliminary section of the report consists of a Title Page, Acknowledgments, Abstract, Table of contents, and List of Abbreviations. List of Figures. The main report is divided into 6 chapters which include:

1. **Chapter 1: Introduction**
   It consists of an introduction and background of the project, its objective, scopes, and limitations. This gives a general overview of the whole project and its focus points.
2. **Chapter 2: Background Study and Literature Review**
   This chapter focuses on the background study, problem statement, as well Literature Review. Likewise, the study of existing systems is also included in this chapter. Different documents that were used in reviewing the existing technology are cited in the literature review.
3. **Chapter 3: System Analysis**
   It contains system design with architecture design, database design, and activity design with all necessary diagrams.
4. **Chapter 4: System Design**
   It consists of Implementation and testing with development methodology, Functional block diagram, Algorithm, and testing process implemented throughout the development.
5. **Chapter 5: Implementation and Testing**
   Different test cases are listed here.
6. **Chapter 6: Conclusion and Future Recommendations**
   It presents the conclusion of the project with future enhancements possible for the future of the project
7. **References:**
   It contains the citations for the study material that we have used in the software development process.
8. **Appendices:** It consists of snapshots of our projects showcasing the functionality and demonstrations of recommendations.

CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW

2.1 Background Study

The comfort-seeking nature of humans has driven us with all kinds of innovations and developments that have made our lives easier and easier over time. From primitive times, selecting the suitable and best possible options has been crucial for human survival and the extension of our civilizations. In the past, humans did have recommendations and suggestions from the older generations. The accuracy of those suggestions was generally vital for their survival.

In the present context, the globalization of industries has made the world a single entity. As such, there is a vast number of options and alternatives for any type of service or product that we might need. Considering the dilemma that the people faced while choosing suitable options recommendations systems were innovated. Elaine Rich created the first recommender system in 1979, called Grundy. She looked for a way to recommend a user a book she might like. Her idea was to create a system that asks the user specific questions and assigns her stereotypes depending on her answers. Depending on a user's stereotype, she would then get a recommendation for a book she might like [4].

After the creation of the recommendations system, people became more and more eager to get suggestions and recommendations about the products and services that they desired. As it eases their indecisiveness and optimizes their productivity, people try to depend more and more on the recommended system. This type of system has a greater impact on almost all types of industries at present. The job market is one of the most essential entities that has a huge scope to leverage the recommendations system to connect employers with suited job seekers.

Job searching in Nepal has always been challenging due to the lack of transparency in the hiring process, limited job listing, and incomplete job information as well as lack of work-life balance. There are still a lot of skilled human resources that are not finding desired jobs. Similarly, there are promising companies that are not finding suitable employees. And, to fill this gap many platforms have already developed. Merojob, Kumarijob, JobsNepal, and

Hamrojob are some promising job portals that have been giving opportunities to connect employers with candidates. However, there are still challenges in those systems due to complex navigation on some of their websites as well as a lack of transparency about the benefits and remunerations. Similarly, one of the main concerns is that people are still getting unrelated jobs even after filling up complex forms.

2.2 Literature Review

Unemployment has been one of the most concerning issues for Nepal and many other developing countries. The report from the Department of Foreign Employment (DoFE) of Nepal shows that 7,71,000 young individuals ventured abroad for foreign employment in the fiscal year 2022/23 only [5]. This number indicates the only surfacing issues that we are facing at present. But, if we were to delve deep into it with some vision for the future, it is a very disturbing hurdle that we are going to face. This shows that there would lack of skilled human resources in Nepal slowing its development process in every aspect.

Not only in Nepal, the global unemployment rate stood at 5.1 percent in 2023, a modest improvement from 2022 when it stood at 5.3 percent, which is still quite concerning and shows that there are still a lot of gaps to be filled between the job market and the job seeker [6]. As such, to enhance the connectivity of the job seeker with the employer more relatively and suitably, Job recommendation systems still hold the scope to make this world a better place by increasing the employment rate globally.

From the start of the commercialization of the internet in the late 1980s, the question was raised of how this technology could be leveraged in employee recruitment to enhance job seeker-vacancy matching [7]. This has led to several approaches to innovate and develop recommendation systems that are specialized in the e-recruitment market. From simplistic content-based filtering to the collaborative and more advanced system, there is a wide range of possibilities to choose from while delivering a job recommendations system.

However, in our view rather than complex and inaccurate systems leveraging simple systems with better accuracy with be more beneficial for general job seekers. As such, JRS is developed with content-based filtering. This would allow the users to simply update their details with their preferred industry and their skills and get a suitable job more efficiently and reliably.

Existing System

i. MeroJob

MeroJob has been one of the most prominent online job portals for Nepal since 2009. It has been bridging the gap between employers and candidates through the use of a web application that is available at merojob.com. It has been fairly intuitive user interface. However, there are points that can be improved. The recommendations of jobs through email can be more personalized. Junior candidates are also getting job recommendations for job openings that are for 2-3 years of experience.

ii. JobsNepal

JobsNepal is a locally focused employment website that offers services to the recruiting and job-seeking community. It claims that the services are 100% free, but they also provides some premium job openings that are associated with some cost. Either way, it is also one of the most prominent e-recruitment sites in Nepal. It is available on jobsnepal.com.

iii. KumariJob

In comparison to Mero Job and Jobs Nepal, Kumari Job seems to have a more comprehensive range of services, from employee outsourcing and training to permanent and temporary recruitment services. This company was established in 2014 and it is available through kumarijob.com. similarly, it also has relatively better transparency and user interfaces relative to the above-listed portals.

While the existing online job portals do seem to be showing some promising results. There are still some limitations to them. Some job portals contain inaccurate information about job postings, such as outdated or expired listings, leading to frustration for job seekers. Moreover, there is a lack of transparency in the hiring process on many job portals, making it difficult for job seekers to determine the status of their application [8].

CHAPTER:3 SYSTEM ANALYSIS

3.1 System Analysis

Before building any system, analysis of its requirements, scope, and limitations is essential as it helps us to create a boundary that can help her to remain within our goals. While developing JRS, proper planning and system analysis were done by physical and Google Meet. With such collaborations, we were able to set the scopes, functionalities, and also the constraints of JRS.

3.1.1 Requirement Analysis

To gather the requirements of JRS, we have used a multifaceted approach. We have taken oral interviews with our classmates and we have also searched online media platforms like Reddit and Twitter to gather how the existing system is not fulfilling the needs of general job seekers. These such processes, we have listed the function and non-functions requirements below:

i. Function Requirement

- This system has both a user and admin login module. Admin login is used to give access to the admin panel and the user login module is used by general users to access jobs details and also view other user's information as well.
- Registered users are allowed to post their preferences. They can view the recommended jobs based on those preferences.
- This system has an admin function that is in charge of maintaining the database of the system. The main role of the admin is to post the jobs of companies and manage them.

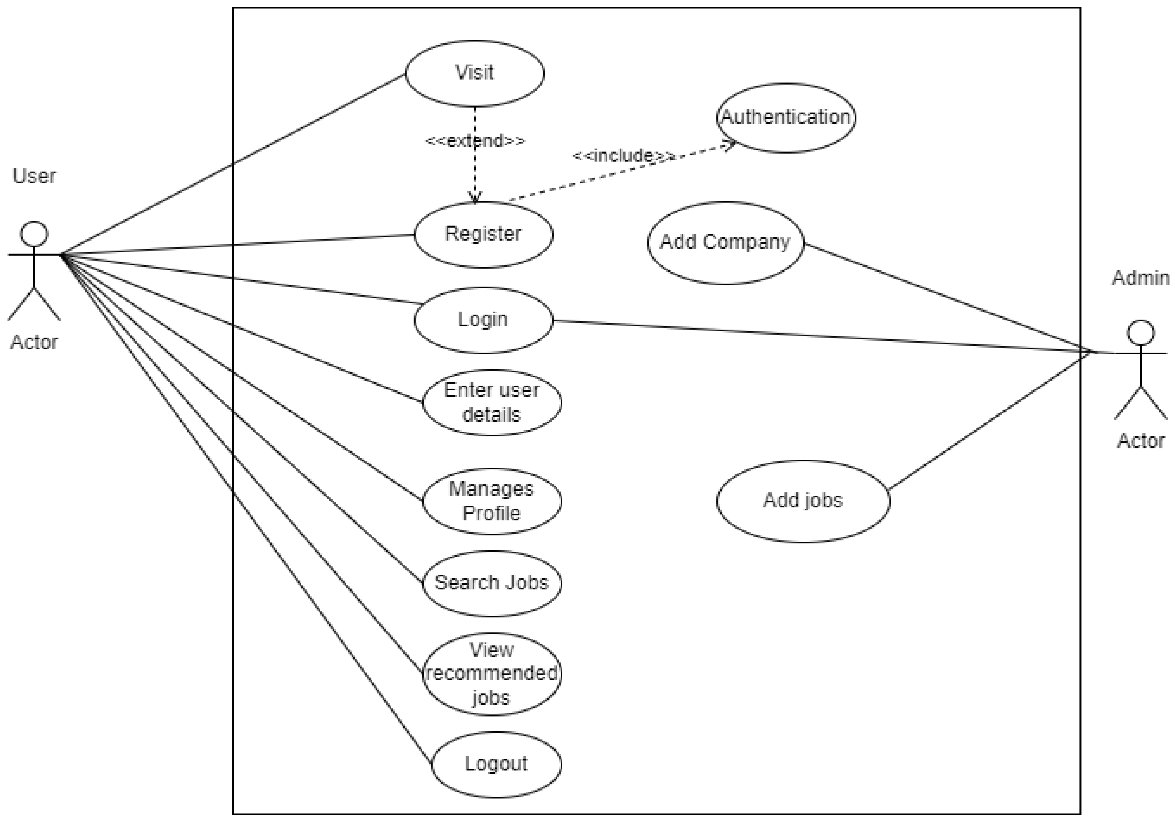


Figure 3.1: Use Case Diagram

ii) Non-Function Requirement

- **Availability:**

  JRS will be available for all users from any geographical location which shall facilitate hiring people through a totally online process.

- **Usability:**

  Along with simplicity, user experience is also one of our focus areas. JRS uses react for the front end to make the system interactive and user-friendly.

- **Reliability:**

  To make the system more reliable we have tried to automate most of the things that concern a date as such the users don't get any outdated or irrelevant data. Similarly, the system will be active all the time which would help to collaborate with people around the world in the future.

- **Performance:**

  Rather than using complex models and confusing navigations in the system, we have implemented a simple yet robust and efficient system. The speed and user experience of JRS would be one of the most important factors to engage our users.

3.1.2 Feasibility Study

After gathering the requirements for JRS, a proper feasibility study was done where we determined how achievable our system is. Building the system with time constraints was indeed a bit challenging. But, with proper planning and study, we are able to attain your milestones along the way to eventually have the system ready in time. Along with time constraints operational feasibility, economic feasibility, and technical feasibility were also studied.

i. Technical Feasibility

The system can be built using web development technologies such as Python as a core programming language. The backend will use Django and the algorithm was implemented in Python. The frontend was developed using React. The technologies that are required for the development of the system were readily available.

ii. Operation Feasibility

Since the proposed system can be accessed using a web browser which is available on both desktop computers as well as mobile devices, thus, it is operationally feasible.

iii. Economical Feasibility

The system will be built using the tools that are freely available on the internet as well as open-source images, so, this system is economically feasible.

iv. Schedule

Schedules were managed by using Gantt's chat which is presented below.

Gantt Chart

Table 3.1: Sprint Schedule Table

Process	**Sprints**	**Task**	**Start**	**Finish**	**Duration (days)**
Requirement Gathering	Sprint 1	-Group discussion about possible topics -Research potential projects -Prepare proposal document	2080/06/16	2080/06/29	14
Planning	Sprint 2	-Read Research papers -Research System Design -Feasibility Analysis	2080/07/15	2080/08/08	23
Designing	Sprint 3	-System Design -Write Database Schema -Design Webpages	2080/08/15	2080/09/03	18

Coding	Sprint 4	-Setup environment for coding -Prepare home page -Collect seed data -Make authentication system	2080/09/04	2080/09/11	7
	Sprint 5	-API for home page -Code for Vectorization and weighting -Report writing	2080/09/12	2080/09/19	8
	Sprint 6	-implement cosine similarity -implement correlation coefficient -Integration of algorithm with database and backend -Report writing	2080/09/20	2080/10/10	19
	Sprint 7	-Integration of web app and backend -Implement APIs for recommendation -Report writing	2080/10/12	2080/10/28	16
Testing and Debugging	Sprint 8	-Find and fix bugs -Review the report	2080/10/11	2080/11/11	30

		-Unit and System testing -Report writing			
Documentation		-Group Discussion for final report -Presentation Preparation	2080/06/16	2080/12/04	19

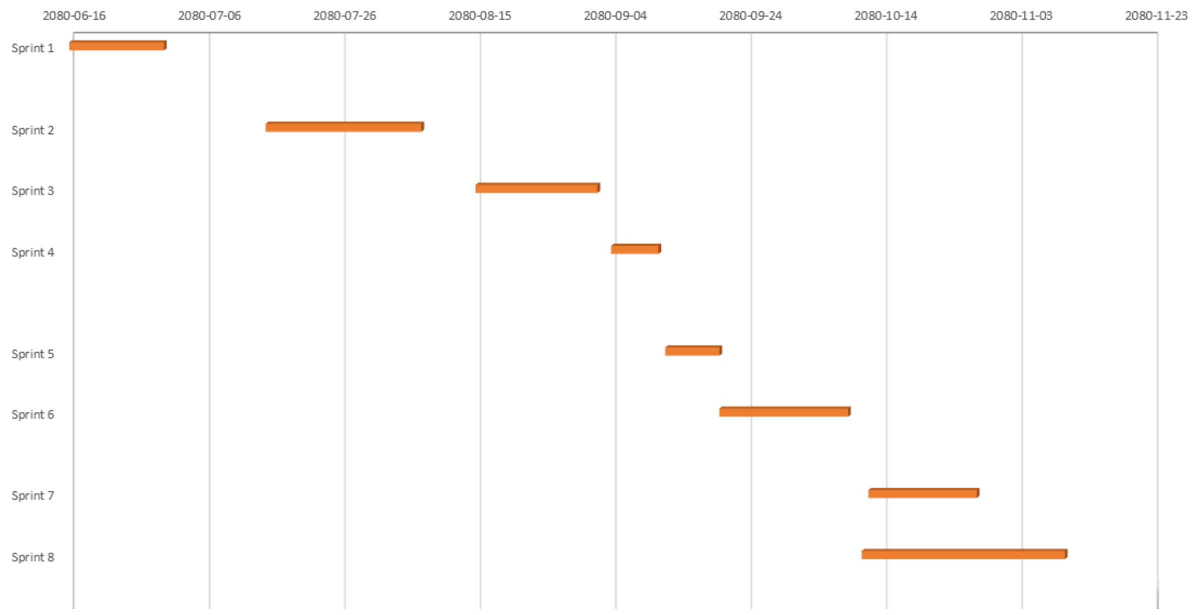


Figure 3.2: Gantt's Chart

3.1.3 Analysis

- **Data Modelling using ER Diagrams**

The ER diagram shows how the entities are related to each other. This system consists of mainly three entities i.e., company, user profile, and jobs. Admin manages the jobs and companies and users search existing job details. User consists of attributes like id, username, and password. Similarly, a job consists of job_id, address, city, contact, and email. Similarly, a company consists of company_id, user_id, name, website, description, etc. ER diagram clearly illustrates the relationship between all the entities residing in the system which provides a clear vision of the system.

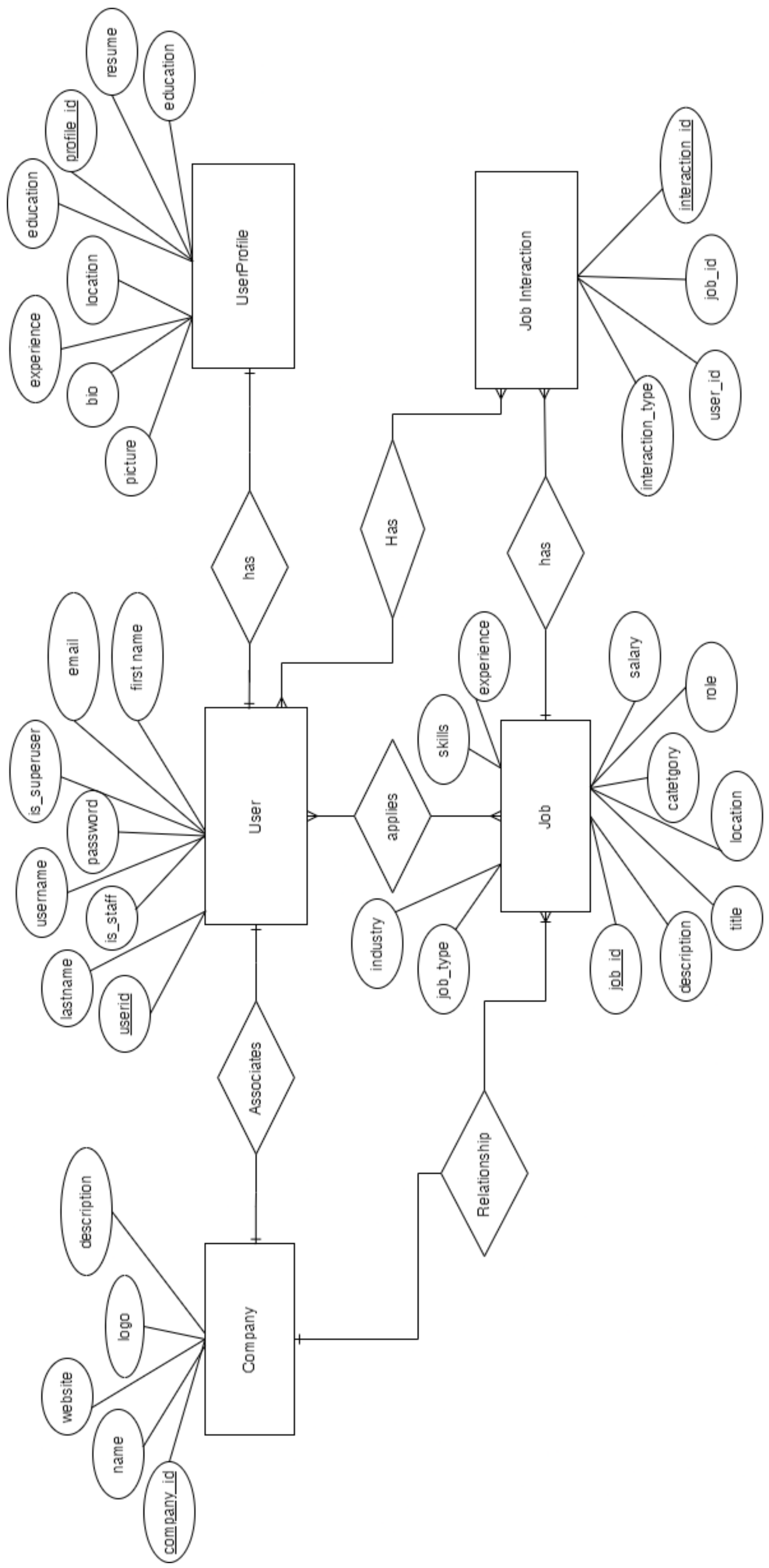


Figure 3.2: ER Diagram

This ER diagram has entities like Company, Job, User (representing candidates), and Job Interaction. Companies can have many open Jobs, each with details like title and description. Users can apply to various Jobs, and their applications (including resumes and interview scheduling) are captured in the Job Interaction entity. This structure provides a foundation for modeling applicant tracking, job postings, and the overall recruitment process.

i) Level 0 DFD

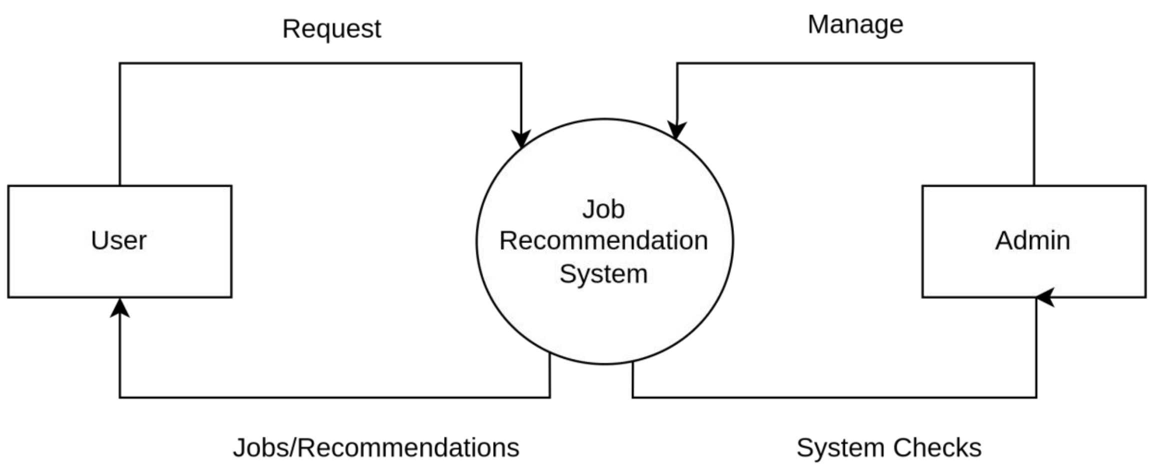


Figure 3.3: Level 0 DFD

This level 0 data flow diagram (DFD) for the job recommendation system shows three external entities: User, Admin, and Manage. These components along with the arrow lines show the surficial overview and working of the JRS. Firstly, the user requests the Job recommendation (done automatically) while the user searches for a job with a keyword or when the user is logged in. Moreover, the Admin is authorized to manage the listed jobs and verify the company and its profiles so that only the valid job gets through the system.

ii) Level 1 DFD

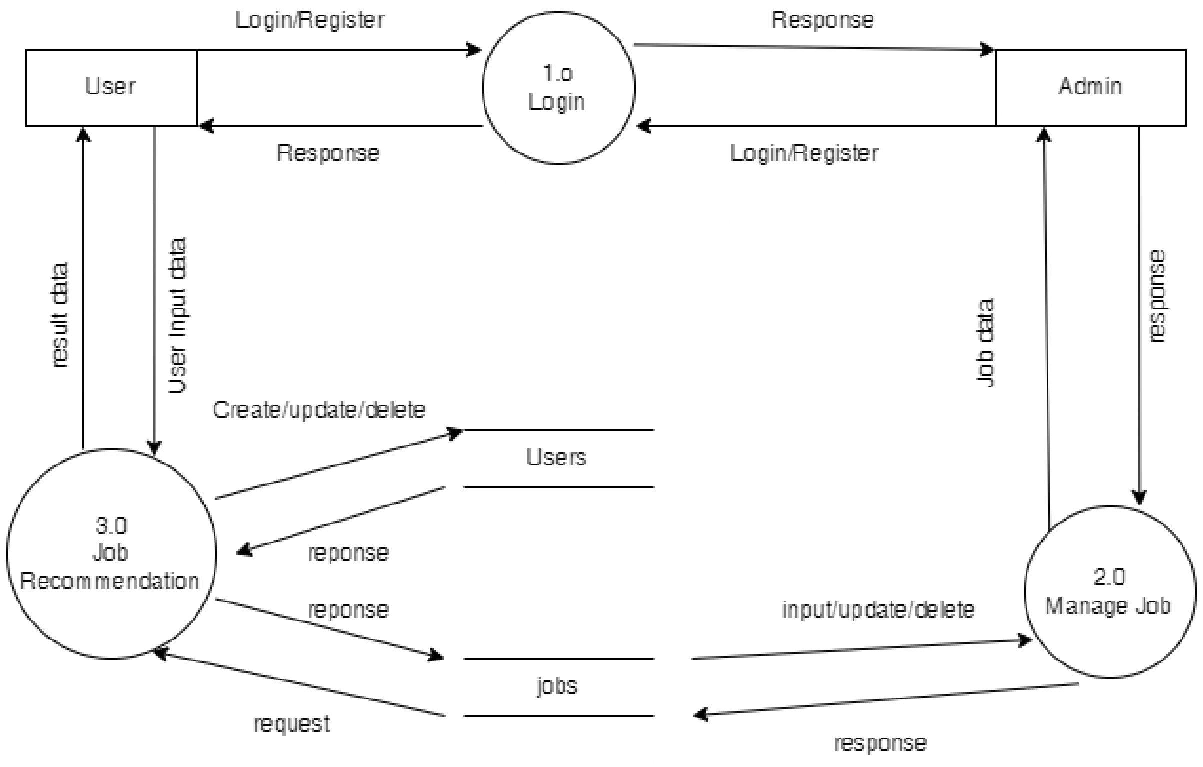


Figure 3.4: Level 1 DFD

This level 1 Data Flow Diagram (DFD) depicts deeper into the processes of the job recommendation system. It separates the level 0 "User" interaction into three processes: " Login", "Manage job" and “Job Recommendation”. During login, the system validates the user's username and password. Once logged in, the user can provide their preferences or search criteria. The "Job Recommendation" process takes this user input and calculates recommendations, by matching it against job data stored in the database. Finally, the "Manage Jobs" process allows administrators to add, update, or delete job postings within the system. This level 1 DFD provides a clearer picture of how user login, preference gathering, recommendation generation, and admin control over job postings work together in the system.

iii) Level 2 DFD

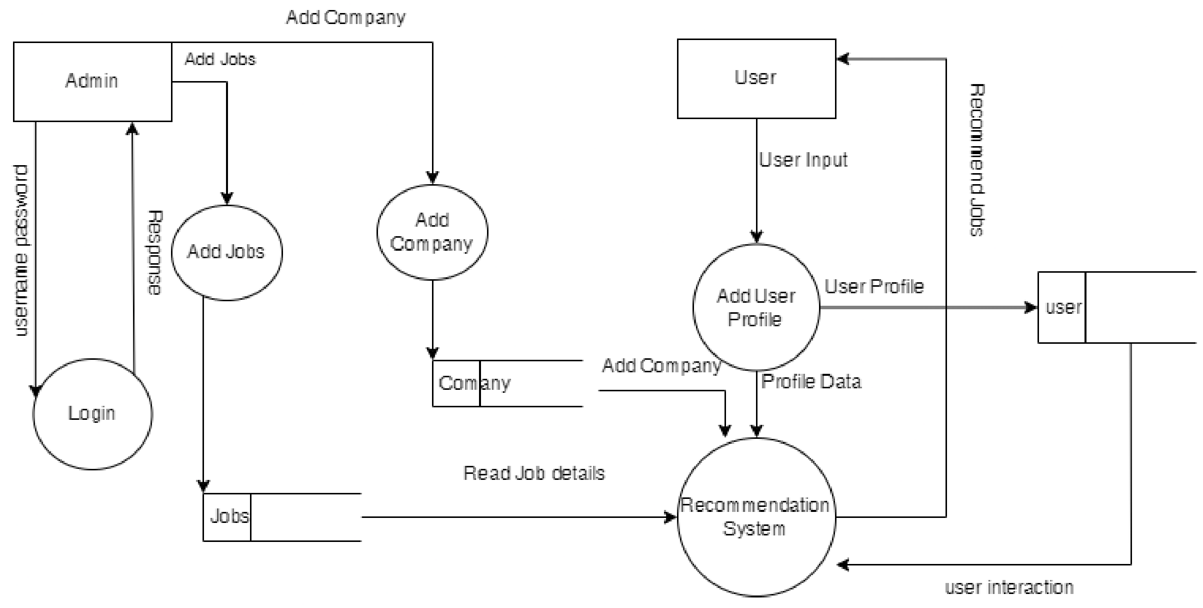


Figure 3.5: Level 2 DFD

This level 2 DFD zooms in on the more detailed view of the system. It includes 5 processes which are extended from the level 1 DFD. Here, the admin is authenticated with a username and password. After authentication, the admin is able to manage and perform crud operations on listed jobs, company as well, and users. Moreover, the user entity is also given access to add and modify profiles. Similarly, add profile process is connected to the user store which is the table to store all the user and their interaction as well. From this as well as the user profiles, the recommendation system applies the hybrid approach to recommend the job to the user.

CHAPTER 4: SYSTEM DESIGN

4.1 Design

System Design includes the means and methodologies to improve the management and control of the software development process. It is the process of defining the elements of a system such as the architecture, modules, and components, the different interfaces of those components, and the data that goes through that system. It is meant to satisfy the specific needs and requirements of a business or organization through the engineering of a coherent and well-running system.

i) Working Mechanism of Proposed System

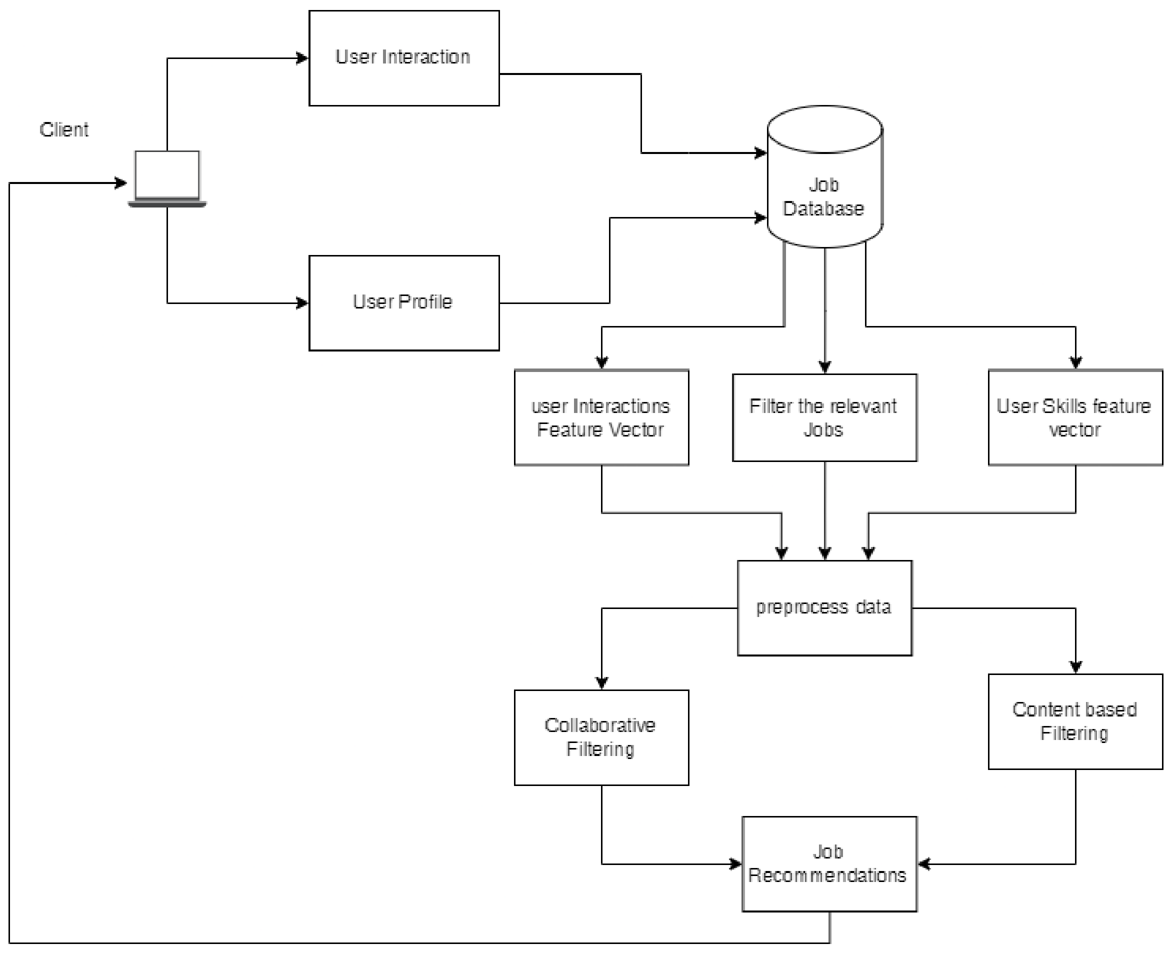


Figure 4.1: System Design

Initially, the system was proposed to only use the content-based filtering for the recommendation. However, as user seeks “crowd wisdom” to find better job prospects and companies to work, we have added the collaborative filtering to make JRS a hybrid system. The system first takes the user profile details and also start to store the user interaction as the candidate explores through the JRS. When the user makes a query or tries to look for a job, the server matches the similarity between the searched keyword, user profile details, and user similarities to make the relevant recommendation.

To prepare for analysis, the "Preprocess Data" component cleans and extracts features from both user profiles and job descriptions. This processed data is then fed into the recommendation model. This model analyzes the data and generates personalized recommendations, returning the top N most relevant job openings for each user. Finally, the recommended jobs are sent back to the client application for user display. This design leverages a database for efficient storage of user-profiles and job postings.

ii) System Flowchart

The flow chart given below illustrates a process for checking user status and displaying jobs. It starts with a login prompt and user input. The system validates the login and checks user data. If the login is valid and the data is sufficient, then the system checks if the user has crossed the interaction limitations. If yes, the system will recommend the user through the hybrid filtering. Otherwise, recommendation is given through content-based approach. In case of an invalid login, the user is directed to the login screen again. This cycle continues until a valid login occurs with enough data to display all jobs.

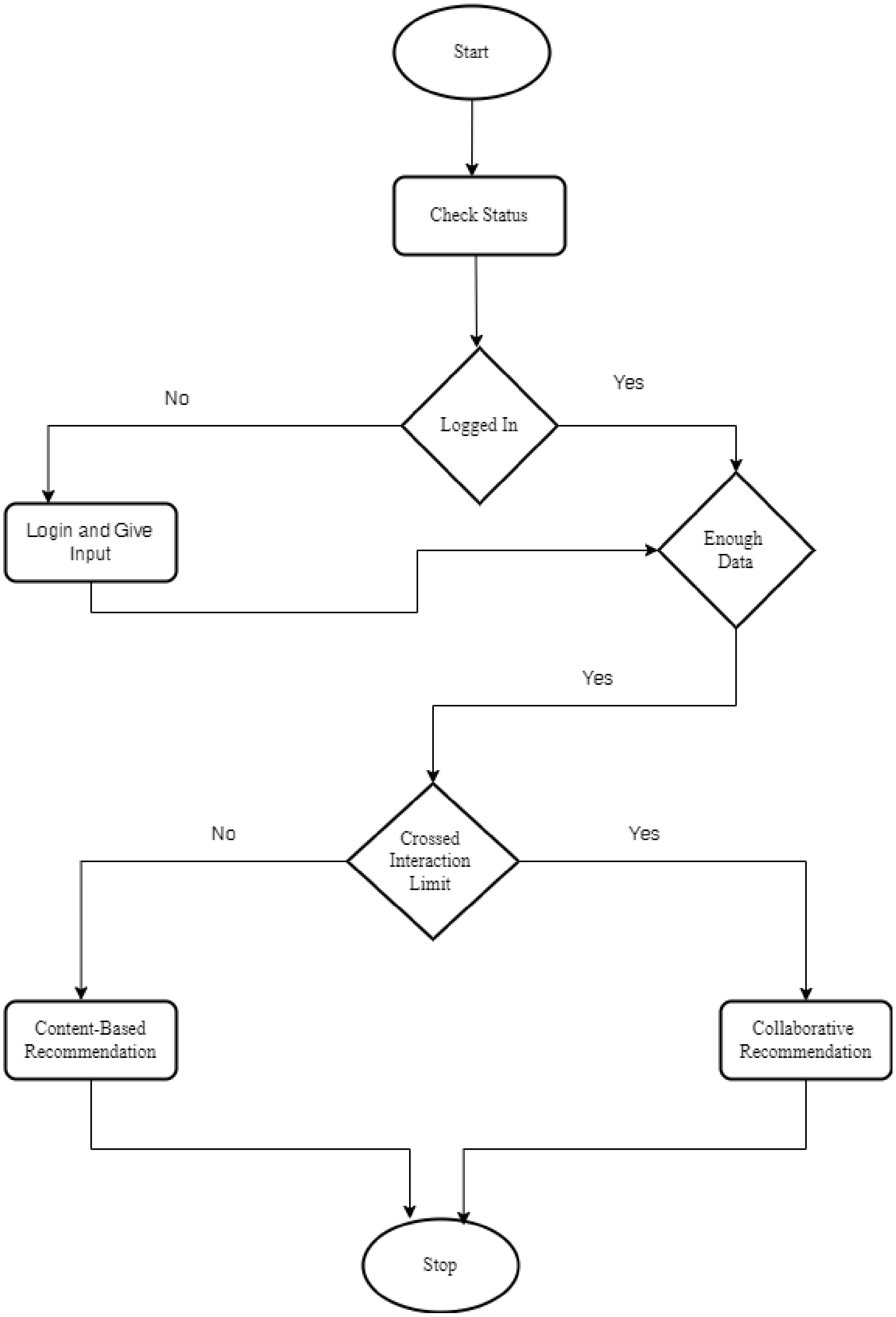

Start
Check Status
Logged In
No
Yes
Login and Give Input
Enough Data
Yes
Crossed Interaction Limit
No
Yes
Content-Based Recommendation
Collaborative Recommendation
Stop

Figure 4.2: System Flow chart

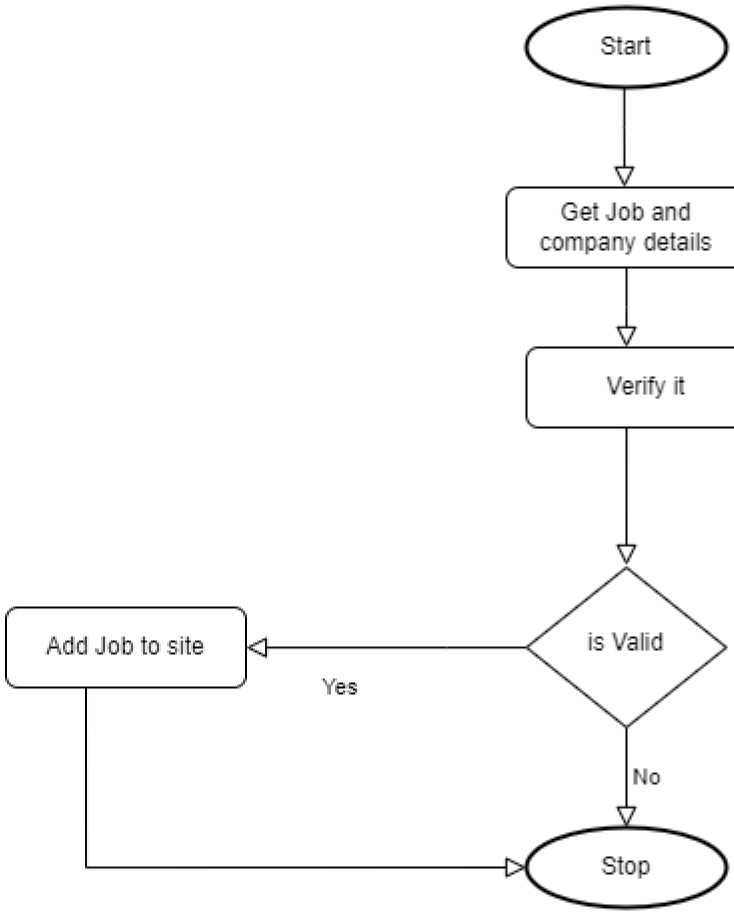


Figure 4.3: Admin Flow Chart

This admin flow chart details the process for adding a job posting to a website. It starts with the verification of the listed job and company as well as the users. Then after checking if the entity is verified, it is added to the site. Otherwise, if the information is invalid, the flowchart stops, indicating the user needs to revise the job details before submission.

4.1.1 Database Design

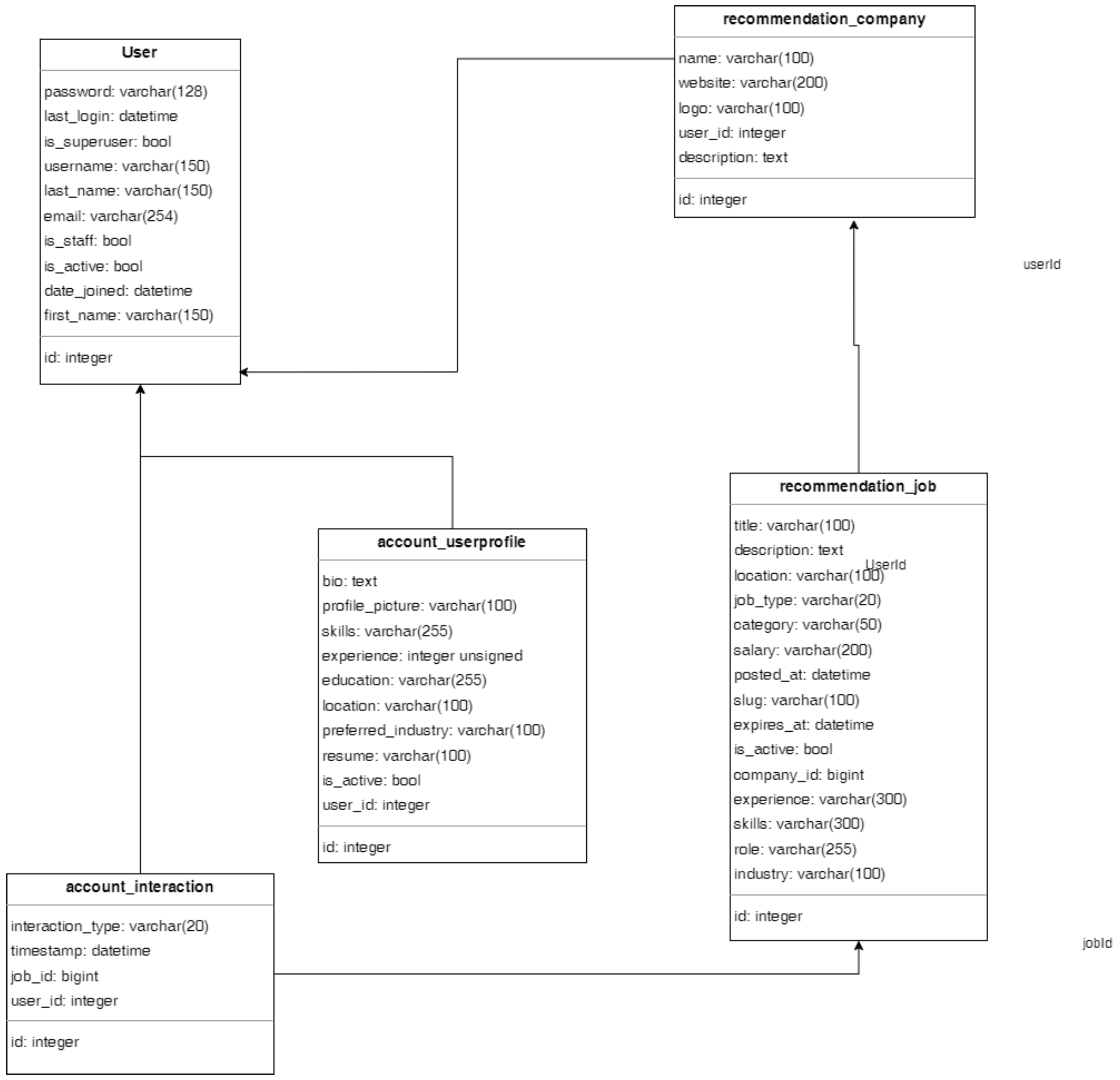


Figure 4.1: Database Schema

The above database schema contains tables for users, jobs, and interactions between them.

Users: This table stores user information like name, email, password, and other profile details. It also includes flags indicating whether the user is a customer or an admin.

Jobs: This table stores details about job details including the title, description, location, category, and other relevant information. It also has fields for the salary, experience level required, and the company offering the job.

Recommendation_jobs: This table acts as a link between users and jobs. It stores the user ID and the job ID, indicating which jobs have been recommended to which users. There are also fields for the timestamp of the recommendation and potentially additional flags or ratings.

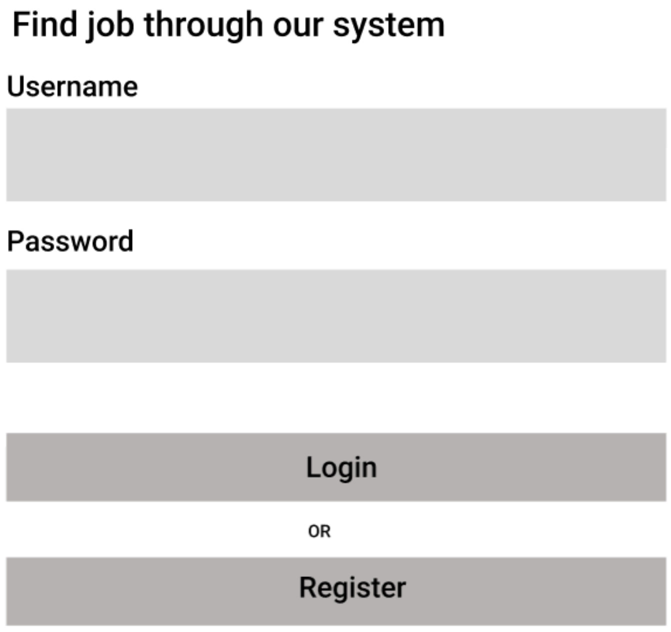
Account_interaction: This table stores details about user interactions with the system, including the type of interaction (e.g., clicking a recommended job), the timestamp, and the job ID for specific interactions.

This schema allows the system to track users, manage job postings, and record user interactions with recommendations. This data can be used to personalize recommendations, improve the system's accuracy over time, and analyze user behavior.

4.1.2 Forms and Report Design

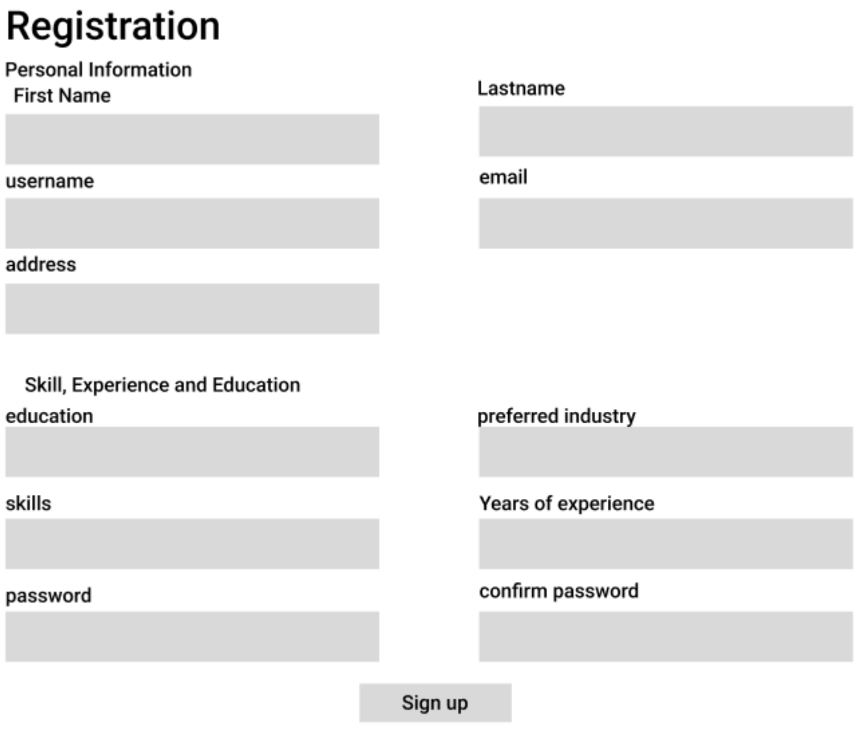
Forms Design

Forms are essential in JRS to collect user information in the registration and login process. Below we can see a login portal with a form asking for a username and password to authenticate the registered user.



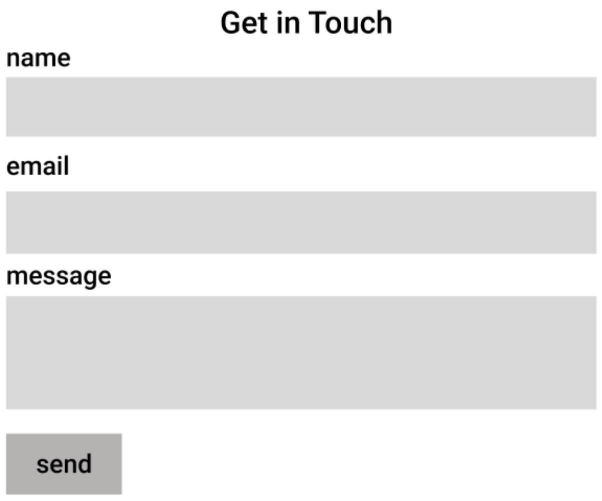
Login Form

If a user is already, registered the user can directly log in to the system. This would give the user access to more jobs as per the user profile. While, a guest user can also explore the JRS jobs, the guest user can not have the recommendation that the logged-in user can have.



Registration Form

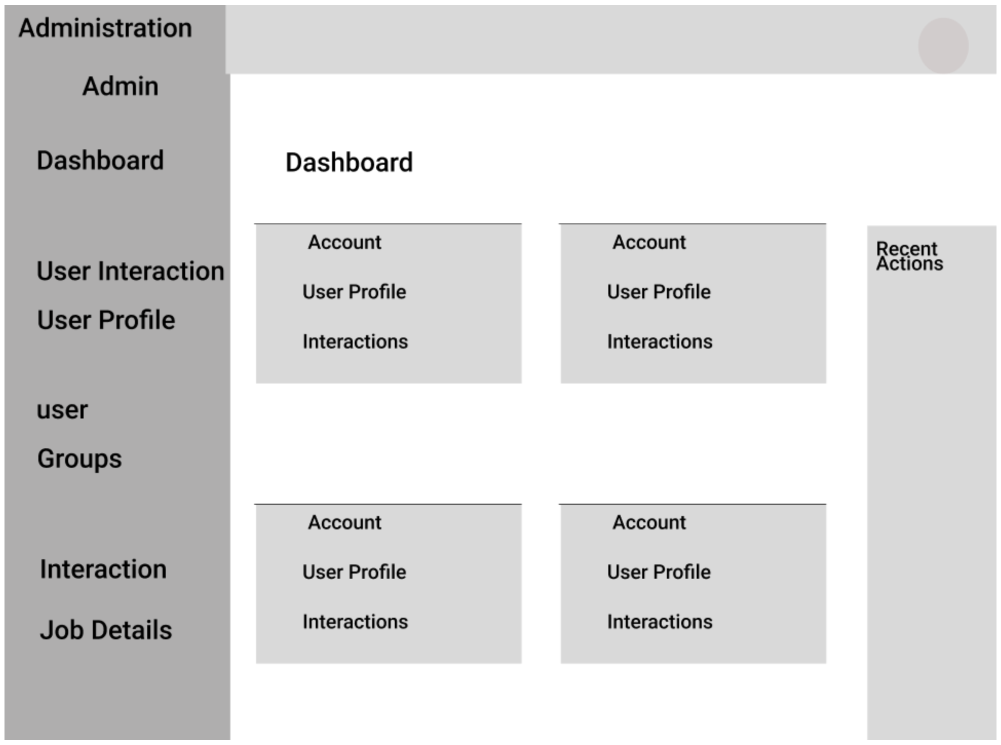
Here, the user or candidates can fill up the form for the registration. The user must fill up all the * given fields to proceed for the registration. Moreover, the skills and the preferred industry are the field from which the system take data to recommend the user using the content-based approach.



Enquiry Form

Here, the user can simply give their name, email and enquiry about any things that they would like to know about your system. This can include enquiry for the collaboration, job listing or even suggestion for the company.

Report Design

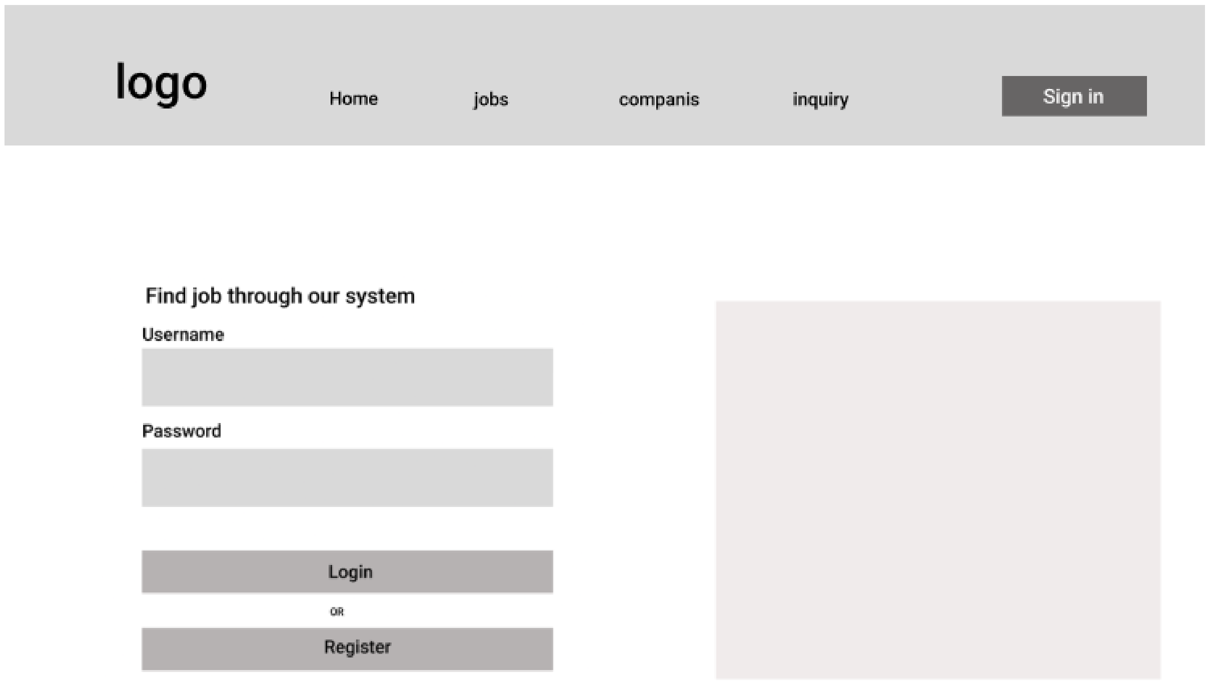


Report Design

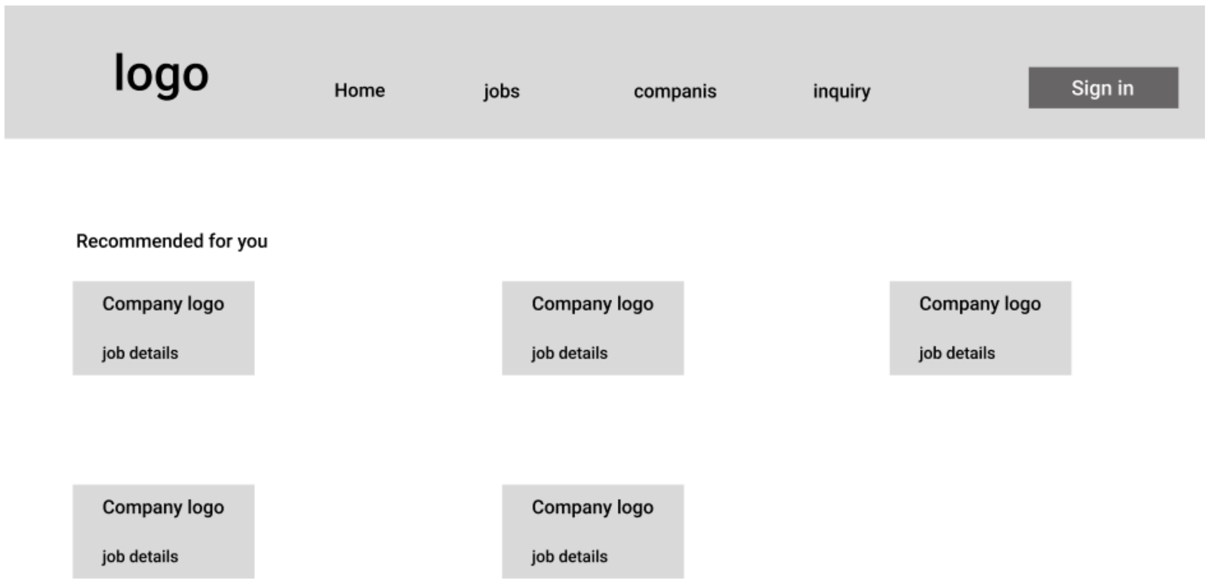
We have implemented the reporting of your user, job, and company through the Django administration. In the above screenshot, we can see that there is reporting interface that helps the admin to see interaction of user in the system. It can also be used to change the privilege level of the user as per the necessity.

4.1.3 Interface and Dialogue Design

The User interface of JRS is developed using the React JavaScript framework. we have focused on simplicity while developing the interface of the JRS aiming to make our system usable to all groups of people including technical and non-technical.

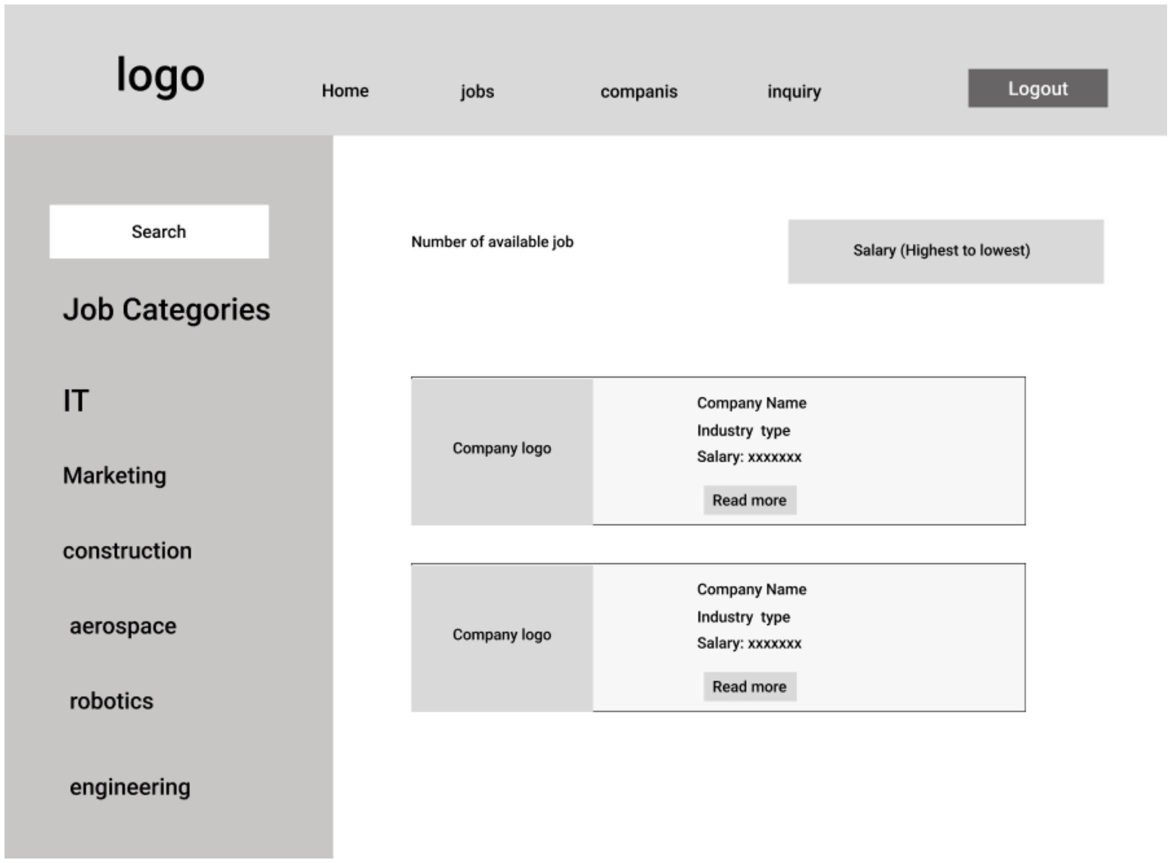


Interface of Home Page



Recommended Job for Logged in user

After the user register and log in to the system, JRS recommends the job to the user as per the skills of the user in the profile and also the preferred industry. Initially, JRS starts to recommend the user with content-based approach.



Interface for Searching and Sorting Jobs

Here, the user can search desired job in the search bar and also sort the job as per the salary range from high to low. We have used merge sort for sorting the jobs as per their salary.

4.2 Algorithm Details

4.2.1 Content-Based Filtering

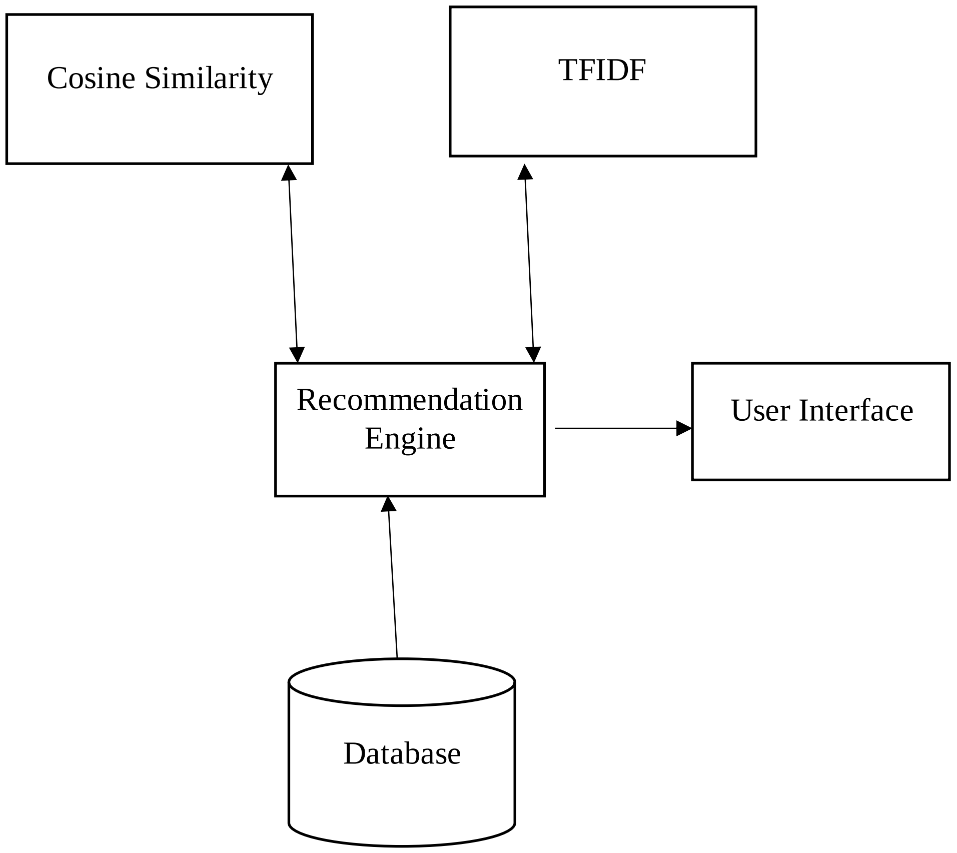


Figure 4.2: Content-Based Filtering

Content-based filtering is a popular algorithm used in job recommendation systems. This algorithm recommends jobs to users based on the content or characteristics of the jobs and the user's preferences. In this approach, similarities are calculated using the metadata of the product, and the details of the user profile but not the past interactions between the user and the items. Content-based filtering is the best when it comes to tackling the cold start problem which is when there is no data between the interaction between the user and the item. However, there is a lack of diversity of the recommended jobs to the users which may dissatisfy the users.

4.2.2 Collaborative Filtering

In JRS collaborative filtering is used to calculate recommend job based on the similarity between users and jobs which makes the recommendation of jobs more diverse. As such, the user feels free to explore their options. While content-based filtering is simpler and more efficient to implement and get the required recommendations in simpler cases, it does have some limitations. Even though collaborative approach is quite complex than content-based filtering, it does have a wider scope for scalability and accuracy. It user-job detail similarity to make recommendations and filter out unrelated jobs. However, when there is a lack of details in user profiles and user interactions, the system may face a cold start problem. This can be solved using the Hybrid Filtering approach.

4.2.3 Hybrid Filtering

Considering how content-based and collaborative filtering has its limitations, we have used the hybrid filtering approach to enhance the quality of the job recommendation in JRS. First, the system starts with a content-based approach and then with enough users and data, it starts to recommend users in a collaborative approach. As a hybrid approach is a recommendation system that combines both techniques the user gets better recommendations and exploration of jobs that align with their experience and expectations. Here is a brief overview of how the hybrid approach works:

- **Combining Techniques**: Hybrid approach used in JRS combines content-based and collaborative filtering techniques to enhance the strength and quality of the system's recommendation. This combination would allow the user to get the best results for the user to explore through the recommendations.
- **Addressing Limitations**: Hybrid approach used in JRS can solve the issues of content-based systems like lack of diversity in job recommendation and also the cold start problem that the collaborative system might face. As such, a hybrid system would be more robust while allowing the user to explore their interested jobs.

4.2.2 Cosine Similarity

Cosine Similarity is a metric, helpful in determining, how similar the data objects are irrespective of their size. In cosine similarity, data objects in a dataset are treated as a vector. The formula to find the cosine Similarity between two vectors is:

where Ai and Bi are components of vectors A and B respectively.

• A. B = product (dot) of the vector's 'A' and 'B'.

• ||A|| and ||B|| = length of the two vectors 'A' and 'B'.

• ||A|| * ||B|| = cross product of the two vectors ‘A’ and ‘B’.

The resulting similarity ranges from −1 meaning perfectly dissimilar, to 1 meaning exactly similar, with 0 indicating no similarity, while in-between values indicate intermediate similarity or dissimilarity.

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

4.2.2 TF-IDF

Term Frequency - Inverse Document Frequency (TF-IDF) is a widely used statistical method in NLP and information retrieval. It measures how important a term is within a document relative to the collection of documents.

Term Frequency: TF of a term or word is the number of times the term appears in a document compared to the total number of words in the document.

TF = number of times a term appears in the document / total number of terms in the document

$$TF_{i,j} = \frac{n_{i,j}}{\Sigma_k \, n_{i,j}}$$

Inverse Document Frequency: IDF of a term reflects the proportion of documents in the corpus that contain the term. Words that are unique to a small percentage of documents receive higher importance values than words common across all documents.

For example: The stop words that occur most of the time in the document get less weight than the words that occur rarely.

IDF = log (number of documents in corpus/number of documents in corpus that contain the term)

$$IDF\ (wt) = 1 + log(\frac{N}{df_i})$$

TF-IDF score: is the product of the Term Frequency and Inverse Data Frequency.

Final Formula:

$$TF - IDF(wt) = TF_{i,j} \times IDF(wt)$$

4.2.3 Merge Sorting

We have used the Merge sort algorithm to search and filter the jobs in JRS. As it is one of the highly efficient sorting algorithms that use the divide-and-conquer approach, it does show efficient results and better user experience as well. It starts the process by dividing the array of the objects into smaller subarrays recursively until each subarray contains only one element. After that, it starts to merge the array in a sorted way. Eventually combining them to form the array that contains all the elements. Its time complexity is O (n log n), making it suitable for sorting large datasets that we have in our system. This also allows us to become more scalable.

4.2.4 Algorithm Steps

1. Insert all the required data in user profile form by the user
2. Take the skills field content which is in comma-separated values from the logged-in user
   Example: User Skills = HTML, CSS, Javascript, Python
3. Get all the skills field content from all the jobs from the jobs table

   Example: Skills in all the jobs = ['HTML, CSS, Javascript, Typescript, Angular', 'HTML, CSS, javascript, Python']
4. Pass the User skill and job skills to the recommender function

   Example: User Skills = [HTML, CSS, Javascript, Python]

   All jobs Skills = ['HTML, CSS, Javascript, Typescript, Angular', 'html, CSS, javascript, Python']
5. Perform the text preprocessing

a) Split the text by commas values of user skills and insert into list of user skills

   Example: User skills = HTML, CSS, Javascript, Python

b) Convert the skills of the jobs into the "list" type

   Example: JobSkills =['HTML, CSS, Javascript, Typescript, Angular',

   'html, css, javascript, Python']

c) Normalize the text of user skills: Convert the user skill set to lowercase

   Example: User skills = ['html', 'css', 'javascript', 'python']

d) Convert each job skills into list of skills by splitting by commas

   Example: [['HTML', 'CSS', 'Javascript', 'Typescript', 'Angular'], ['html', 'css', 'javascript', 'Python’]]

e) Create a single list of all the jobs skills

   Example: Job skills = ['HTML', 'CSS', 'Javascript', 'Typescript', 'Angular', 'html', 'css', 'javascript', 'Python’]

f) Convert single list of job skills into lowercase

   Example: Job skills = ['html', 'css', 'javascript', 'typescript', 'angular', 'html', 'css', 'javascript’,’python’]

g) Create unique list of job skills only

   Example: Job skills = ['html', 'css', 'javascript', 'typescript', 'angular', 'python']

h) Convert each job skills into lowercase

   Example: Job Skills 1 = [['html', 'css', 'javascript', 'typescript', 'angular']]

   Job Skills 2 = [['html', 'css', 'javascript', 'python']]

i) Create unique list containing user and job skills

   Example: ['html', 'css', 'javascript', 'typescript', 'angular', 'python']

j) Create a list of job skills in vector form by using the tf-idf

For example [0.33, 0.33, 0.32, 0.33, 0.38, 0]

k) Create a list of skills of all jobs (both jobs in this case) in vector form

   Example: Job Skills List = [[0.33, 0.33, 0.32, 0.33, 0.38, 0], [[0.5, 0.5, 0.5, 0, 0]]

l) Create a list of user skills in vector form by using tf-idf again.

a) Calculate cosine similarity between User skill and each job skills

b) Calculate the dot product between the user skill and each job skill

c) Calculate the magnitude of user skill and each job skill

d) Use cosine similarity formula to calculate similarity between user skills and skills in each job:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \sqrt{\sum\limits_{i=1}^{n} B_i^2}}$$

Where,

A = list of user skills

B = list of each job skill

n = length of the list of job skills

7. Get the list of cosine similarity values between the coder skill and each job skills

Example: List of Cosine similarity of user skills with skills Job 1 and Job 2 = [0.6708, 1.0]

8. Assign the cosine similarity value between them in the cosine value field of job skill.

9. Display the recommended jobs for the user in their personal dashboard which has cosine similarity values of skills between the user skill and job skill in descending order.

10. When the user crosses the interaction limits defined in the system, the model is switched to hybrid approach applying Pearson correlation.

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1. Implementation

The project implies various development technologies for system development. React and Django are used for interface design and backend processing.

5.1.1. Tools Used

a. Figma was used to create the wireframes of the interfaces, report and design.
b. JavaScript and React – JavaScript and React were used to handle client-side logic.
c. HTML, CSS, and styled-components – CSS and styled component was used to stylize the website.
d. Python and Django – Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Python was used to implement the cosine algorithm as well as to complete the overall development.
e. Draw.io – It was used to draw the diagrams used in the document.

5.1.2. Implementation Details of Modules

i) TF-IDF

```
def calculate_term_frequency(self, terms_list):
    term_freq_dict = {}
    total_terms = len(terms_list)
    for term in terms_list:
        term = self.clean_special_characters(term)
        if term not in term_freq_dict:
            term_freq_dict[term] = 0
        term_freq_dict[term] += 1 / total_terms
    return term_freq_dict


  def calculate_inverse_document_frequency(self, term):
    term = self.clean_special_characters(term).strip().lower()
    num_docs_with_term = sum(
        1 for doc in self.cleaned_documents if term in doc.lower()
    )
    return 1 + math.log(
        len(self.cleaned_documents) / num_docs_with_term + 1
    )  # 1 is added for smoothing
```

The TFIDF class is designed to compute the TF-IDF (Term Frequency-Inverse Document Frequency) matrix for a collection of jobs and user profile details. It preprocesses each term in the documents by removing special characters and stopwords, then calculates the TF and IDF values for each term. Finally, it generates a TF-IDF matrix, where each row represents a document and each column represents a term, with values representing the importance of each term in each document relative to the entire document collection. This module facilitates text analysis and information retrieval tasks by quantifying the relevance of terms in documents within a corpus.

ii) Cosine Similarity

```
def cosine_similarity(self, vector1, vector2):
    dot_product = np.dot(vector1, vector2)
    norm_vec1 = np.linalg.norm(vector1)
    norm_vec2 = np.linalg.norm(vector2)

    # Check for zero division
    if norm_vec1 == 0 or norm_vec2 == 0:
        return 0.0
    else:
        return dot_product / (norm_vec1 * norm_vec2)
```

The CosineSimilarity class calculates the cosine similarity between two document vectors, representing documents as numerical vectors using TF-IDF values. It offers methods to compute the dot product of vectors, their magnitudes, and the cosine similarity between documents. This similarity metric measures the cosine of the angle between the vectors, providing a value between 0 and 1 to indicate the degree of similarity. The class facilitates tasks such as document comparison, information retrieval, and recommendation systems by quantifying the similarity between documents, allowing for efficient document clustering and relevant document retrieval.

iii) Pearson Correlation

```
def pearson_similarity(self, vector1, vector2):
    mean_vec1 = np.mean(vector1)
    mean_vec2 = np.mean(vector2)

    # Compute Pearson correlation coefficient
    numerator = np.sum((vector1 - mean_vec1) * (vector2 - mean_vec2))
    denominator = np.sqrt(np.sum((vector1 - mean_vec1) ** 2)) * np.sqrt(
        np.sum((vector2 - mean_vec2) ** 2)
    )
    if denominator == 0:
        return 0.0
    else:
        return numerator / denominator
```

Here PearsonCorrelation class implements the Pearson correlation coefficient, a measure of the linear correlation between two variables. It computes the correlation between two arrays by first aligning their lengths and then calculating the means, covariances, and variances of the arrays. The Pearson correlation coefficient is then computed using these statistics, providing a value between -1 and 1, where 1 indicates a perfect positive linear relationship, -1 indicates a perfect negative linear relationship, and 0 indicates no linear relationship. If the covariance is zero, indicating no linear relationship, the method returns 0. This class is useful for analyzing relationships between variables in user profiles and job details, identifying patterns, and making predictions based on observed correlations.

iv) Job Details View

```
class RecommendationView(views.APIView):
    def get(self, request, format=None):
        try:
            user_id = request.user.id

            if not user_id:
                return Response({"data": None}, status=status.HTTP_200_OK)

            cached_data = CACHE.get(user_id)
            if cached_data is None or self.is_cache_expired(user_id):
                # take the top 3 interactions that are recently recorded
                interactions = Interaction.objects.filter(user_id=user_id).order_by(
                    "-timestamp"
                )[:INTERACTION_THRESHOLD]
                user_profile = UserProfile.objects.filter(user_id=user_id).first()
                job_listings = Job.objects.all()
                job_listings_dict = {}

                # Filter jobs based on each interaction title and add them to job_listings_dict

                if interactions.count() > INTERACTION_LIMIT:
                    interaction_history = [
                        interaction.job.title + "," + interaction.job.description
                        for interaction in interactions
                    ]
                    for interaction in interaction_history:
                        titles = text_analyzer.preprocess_document(interaction)
                        for title in titles:
                            similar_jobs = job_listings.filter(
                                Q(title__icontains=title)
                                | Q(description__icontains=title)
                            )[:10]
                            for job in similar_jobs:
                                job_listings_dict[str(job.id)] = (
                                    job.title + "," + job.description
                                )

                    top_jobs = get_top_jobs(
                        model="pearson",
                        filtered_jobs=job_listings,
                        job_listings_dict=job_listings_dict,
                        data=interaction_history,
                    )

                # If no interactions found or job_listings_dict is empty, filter based on user skills
                if not job_listings_dict:
                    if user_profile.skills:
                        user_skills = user_profile.skills.split(",")
                        for skill in user_skills:
                            similar_jobs = job_listings.filter(
                                Q(title__icontains=skill)
                                | Q(description__icontains=skill)
                            )
                            for job in similar_jobs:
                                job_listings_dict[str(job.id)] = (
                                    job.title + "," + job.description
                                )
                        top_jobs = get_top_jobs(
                            filtered_jobs=job_listings,
                            model="cosine",
                            job_listings_dict=job_listings_dict,
                            data=user_profile.skills,
                        )

                    else:
                        # If neither interactions nor skills are available, return empty response
                        return Response({"data": []}, status=status.HTTP_200_OK)

                if top_jobs:
                    # put the top jobs in the cache if there are any
                    CACHE[user_id] = {
                        "data": top_jobs,
                        "timestamp": datetime.now(),  # Update timestamp
                    }
                # give the response from the cache

                serializer = JobDetailsSerializer(CACHE.get(user_id).get("data"), many=True)
                return Response({"data": serializer.data}, status=status.HTTP_200_OK)

            except Exception as e:
                print(e)
                return Response({"data": "Internal Server Error"})

        def is_cache_expired(self, user_id):
            """
            Method to check if the cache is expired. If cache is older than 10 seconds then it is
        considered expired

            Args:
                user_id (int): user identifier

            Returns:
```

```
    Bool: Returns True if the cache is expired otherwise False
"""
cached_data = CACHE.get(user_id)
if cached_data:
    timestamp = cached_data.get("timestamp")
    if timestamp:
        return datetime.now() - timestamp > timedelta(seconds=10)
return True
```

Here, the view method handles the retrieval of the job details and also provides recommendations based on user interactions and profile skills. Initially, it extracts the user and job ids from the request and also gets the details for the original job. If there is a job that matches the details, it logs a user-click interaction and starts to generate recommendations. The recommendations are based on user interaction history using the Pearson correlation-based model or through filtered job-matching user skills. If there are no recommendations available, it returns the original job details.

5.2. Testing

After building the project, the following testing measures were applied, and the results as shown below:

5.2.1. Test Cases for Unit Testing

Table 5. 1: Test Case for User Registration

Test Case	Input	Expected Result	Test Result
Successful User Registration	firstName=Sushmita, lastName=Khadka, username=sushmita email address= suhsmita@gmail.com password=admin confirm password=admin	The user should be registered successfully.	The user is registered successfully.

User Registration Fail	firstName=Sushmita, lastName=Khadka, username=sushmita email address= suhsmita@gmail.com password=admin confirm password=admin123	The user should not be registered.	The user is not registered.

Table 5. 2: Test Case for User Login

Test case	Input	Expected Result	Test Result
Successful User Login	username= sushmita password = admin	The login page should be redirected to page with recommended jobs.	The login page is redirected to a page with recommended jobs.
User Login Fail	username =sushmita password = sushmita	Should not be able to log in.	Unable to log in.

Table 5.3: Test Case for Recommendation

Test case	Input	Expected Result	Test Result
Content based recommendation after new user login	User5 was registered with Java as skills	The system should show recommend job related to Java and Dotnet	Java jobs were shown to the user.
Collaborative filtering after user cross 10 interaction limit	User consistently clicks on read more and applies for job related to Marketing and Management	The system should recommend job related to Marketing and Management	Job related to Management and marketing were recommended.

Table 5. 4: Test Case for Job Search

S.N.	Test case	Input	Expected Result	Test Result
1	Searching	Search query= PHP	All the jobs with PHP as a title should be displayed	All the jobs with PHP as a title is displayed
2	Searching	Search query= Aeronautics	No jobs should be shown as there are no job for aeronautics in the database	No jobs with Aeronautics were shown

Table 5. 5: Test Case for Job Category Browsing

S.N.	Test case	Input	Expected Result	Test Result
1	Job Browsing based on Categories	The specific category is clicked	Only the job belonging to the clicked category should be displayed.	Only the job belonging to the clicked category is displayed.

5.2.1. Test Cases for System Testing

In System testing, we have tested if the units that working together are performing after the integrations of the whole system. The table given below has a detailed overview of the system overview.

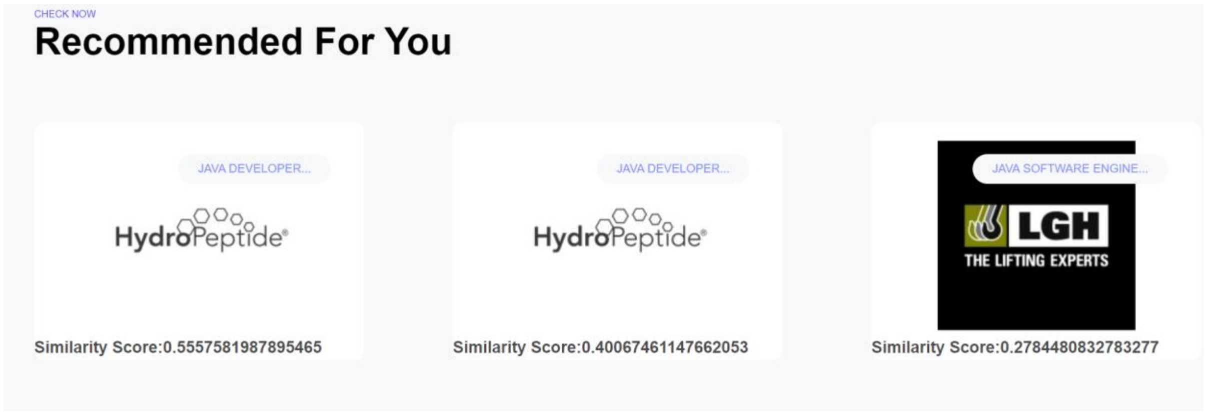
Table 5.6: System Testing

Test Case	Expected Results	Actual Results	Pass/ Fail
Open the application click "register".	The register screen should appear.	The register screen appeared.	Pass

Enter a valid Required details for the registration	The user was registered andtaken to the homepage.	The user was registered taken to the homepage.	Pass
Enter an invalid username and password and click "login".	The user should see an error message and remainin the login screen.	The user saw an error message and remained onthe login screen.	Pass
Check if the recommended jobs for registered user matched the user skills and preferred industry	The user should have relevant job recommendation as per their skill	The user got relevant suggestion as per the skills that they updated in their profile	pass
Sort the Salary range	The system should sort the salary range of listed job	The system displayed Sorted list of job as per the salary	Pass
Click on a specific search result.	The system should display the details of the selected job.	The system displayed thedetails of the selected job.	Pass
Log out of the system.	The user should be loggedout and taken to the login screen.	The user was logged outand taken to the login screen.	Pass

5.3 Result Analysis

The system accurately shows that relevant job to the user as per the skills that they used when registering in the system. In the example above recommendation test case, user who registered using Java as skill is given relevant recommendation using content-based filtering.


CHECK NOW

Recommended For You

JAVA DEVELOPER...

Similarity Score:0.5557581987895465

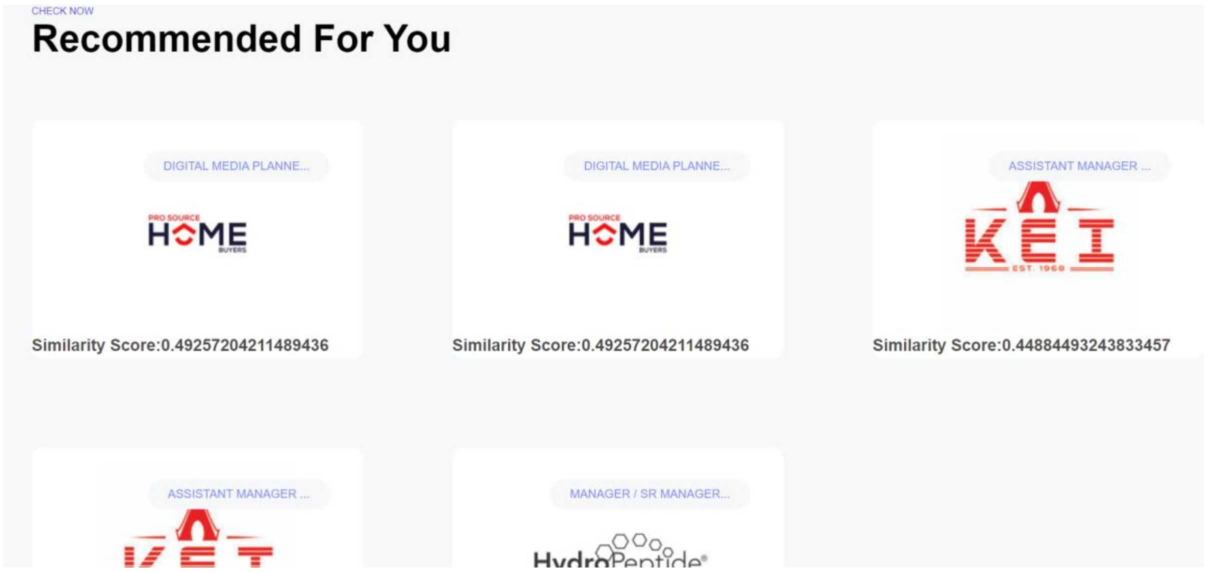
JAVA DEVELOPER...

Similarity Score:0.40067461147662053

JAVA SOFTWARE ENGINE...

Similarity Score:0.2784480832783277

Then, the user consistently interacts with the job related to management and marketing. After the user crosses the interaction limit, user is shown job related to management and Marketing as shown in the above recommendation test case.


CHECK NOW

Recommended For You

DIGITAL MEDIA PLANNE...

Similarity Score:0.49257204211489436

DIGITAL MEDIA PLANNE...

Similarity Score:0.49257204211489436

ASSISTANT MANAGER ...

Similarity Score:0.44884493243833457

ASSISTANT MANAGER ...

MANAGER / SR MANAGER...

This clearly shows that the recommendation is capable of switching to the collaborative filtering after the user crosses the interaction limit.

CHAPTER 6: CONCLUSION AND FUTURE RECOMMENDATION

6.1. Conclusion

JRS makes it easy for persons to showcase their portfolios and provides a convenient way to search for jobs related to the information and technology field. This project uses an approach of cosine similarity of content-based filtering to recommend jobs to the people that are suitable for them according to their skills. It diminishes the way to go through the entire job description to find out whether the job is well suited for them or not.

6.2. Future Recommendations

While JRS is almost a complete system that can recommend jobs to the users as per their skills and interaction with the listed jobs, it still has a lot of features that can be added in the future to make it more robust. Here, there is a possibility of adding a feature that can recommend certain types of courses as per the skills of the users to enhance their profile and seek better opportunities. Similarly, we can also add a feature that can predict the salary and benefits of a certain user with the available details of the profile as well as the current trends and salary ranges of the job ecosystem. In addition, there is a possibility for us to add a feature for seeking a job within a certain radius of the user location.

REFERENCES

[1] S. Karagianndis, S. Antaris, C. Zisopoulos, "Content-Based Recommendation Systems," November 2008. [Online]. Available: https://www.researchgate.net/publication/236895069_Content-Based_Recommendation_Systems. [Accessed 2 January 2024].

[2] P. Tamrakar, "job-vacan-nepal," [Online]. Available: https://prashannat.com.np/job-vacancy-nepal/. [Accessed 2 September 2023].

[3] J. Shah. P. Nanavati, G. Waja, "AGILE SOFTWARE DEVELOPMENT," *International Journal of Engineering Applied Sciences and Technology,* vol. 5, no. 12, pp. 73-78, 2021.

[4]"Recommender system," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Recommender_system. [Accessed 29th November 2023].

[5] S. Khadka, "myRebulica," 23 July 2023. [Online]. Available: https://myrepublica.nagariknetwork.com/news/over-771-000-youths-sought-foreign-employment-in-fy-2022-23/. [Accessed 6 October 2023].

[6] "Internation Labour Organization," 10 January 2024. [Online]. Available: https://www.ilo.org/global/about-the-ilo/newsroom/news/WCMS_908068/lang--en/index.htm#:~:text=The%202023%20global%20unemployment%20rate,rates%20also%20improved%20in%202023.. [Accessed 15 January 2024].

[7] S. Bhulai, C. Ruijt, "Job Recommender Systems: A Review," Faculty of Science, Vrije Universiteit Amsterdam, Amsterdam, the Netherlands, 2021.

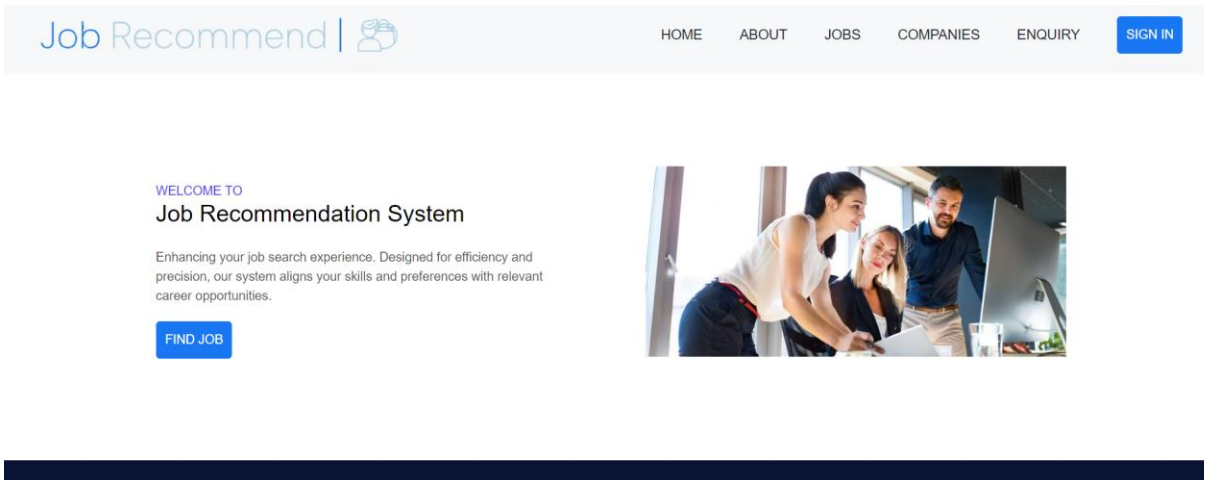
[8] U. Siddique, "Quora," [Online]. Available: https://www.quora.com/What-are-the-problems-and-limitations-of-Nepali-job-portal-sites-in-Nepal. [Accessed 29 February 2024].

Log of visits to Supervisor

Table 6.1: Supervisor meet log

Date	**Remarks**
2080/06/03	Initial Proposal submission for review &supervision. (Physical meetup)
2080/06/26	Resubmission of proposal with changed topic (Google Meet)
2080/08/21	Discussion of project status and asked for suggestions. (Google Meet)
2080/9/15	Shown project status (Google Meet)
2080/10/27	Sought suggestion for the Report and Presentation Tactics and also shown project status (Google Meet)
2080/12/13	Submitted final defense report for review and supervision
2080/12/22	Report Evaluations and Further Discussion

APPENDICES

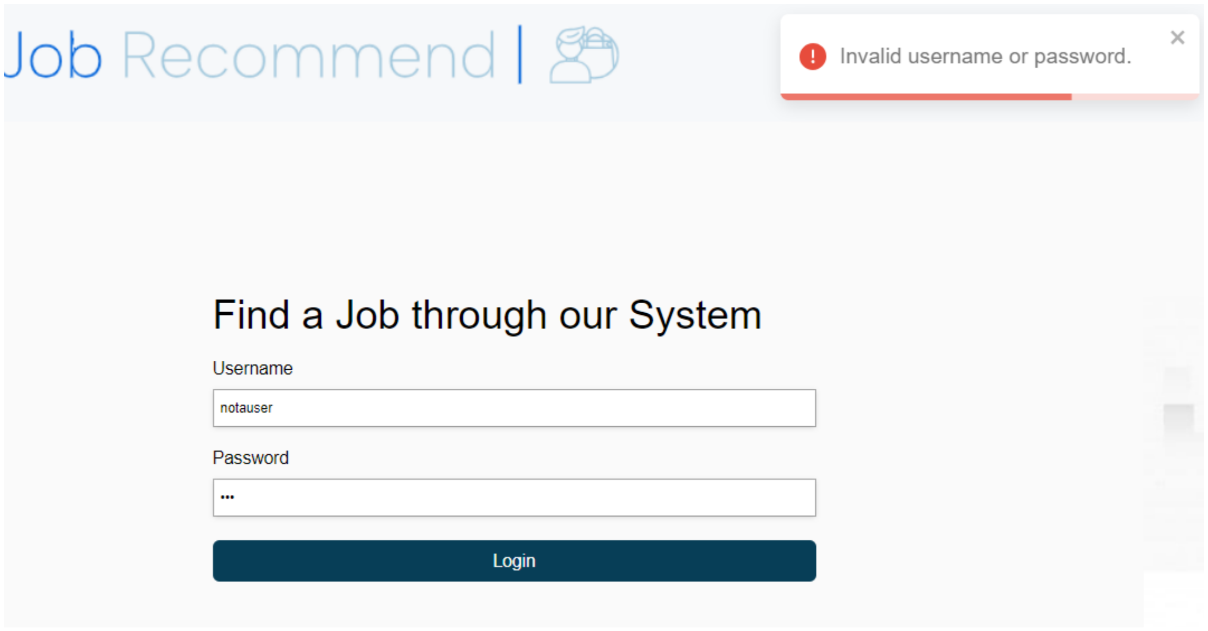


WELCOME TO

Job Recommendation System

Enhancing your job search experience. Designed for efficiency and precision, our system aligns your skills and preferences with relevant career opportunities.

Home Page



Invalid username or password.

Find a Job through our System

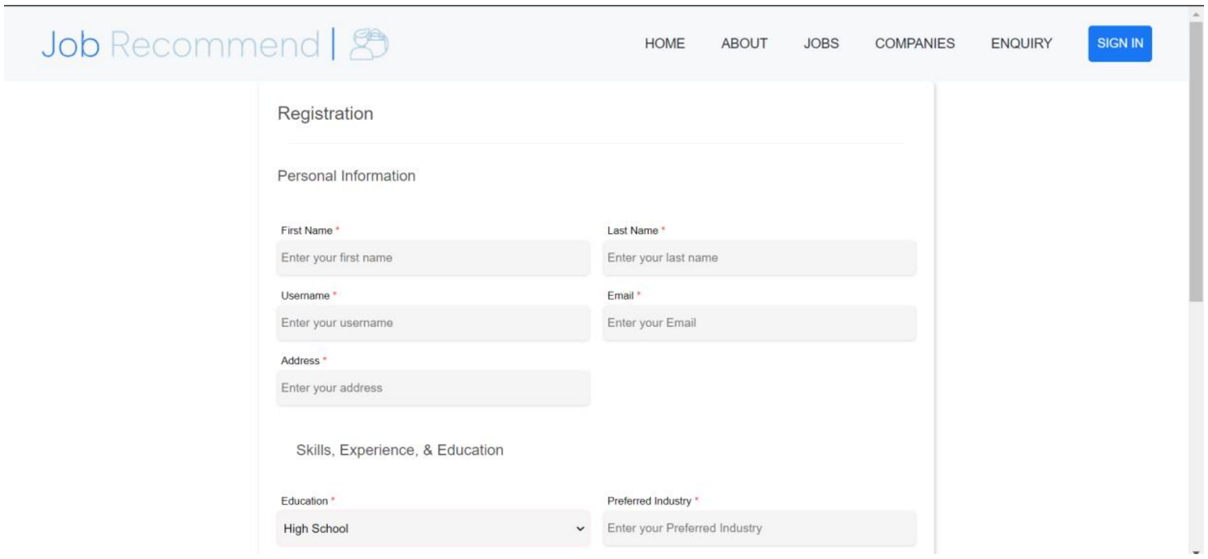
Username

notauser

Password

•••

Denied Access to Unregistered User



Registration

Personal Information

First Name *
Enter your first name

Last Name *
Enter your last name

Username *
Enter your username

Email *
Enter your Email

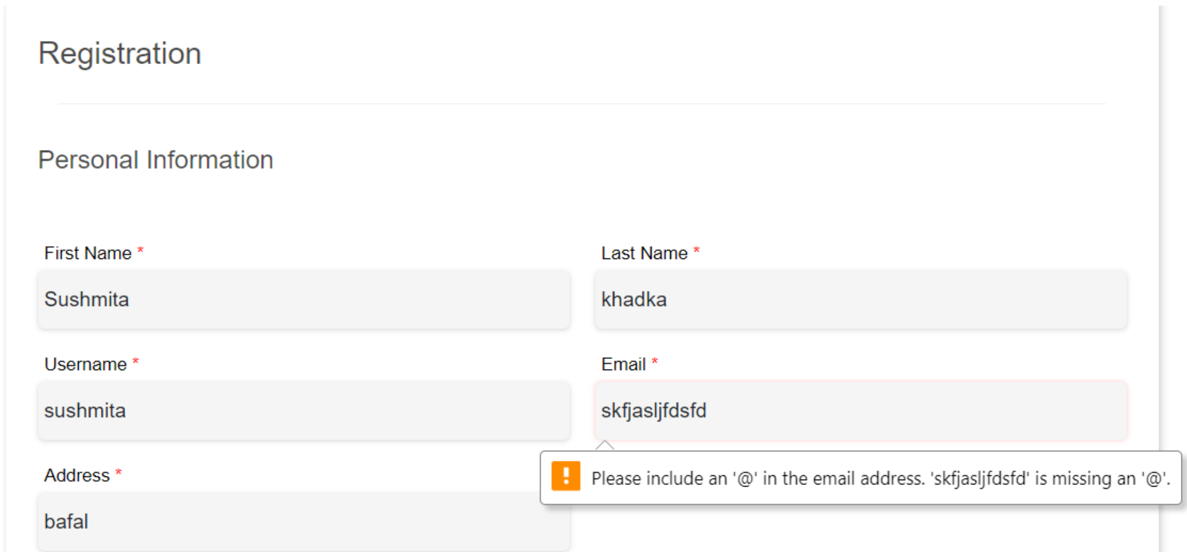
Address *
Enter your address

Skills, Experience, & Education

Education *
High School

Preferred Industry *
Enter your Preferred Industry

User Registration Page



Registration

Personal Information

First Name *
Sushmita

Last Name *
khadka

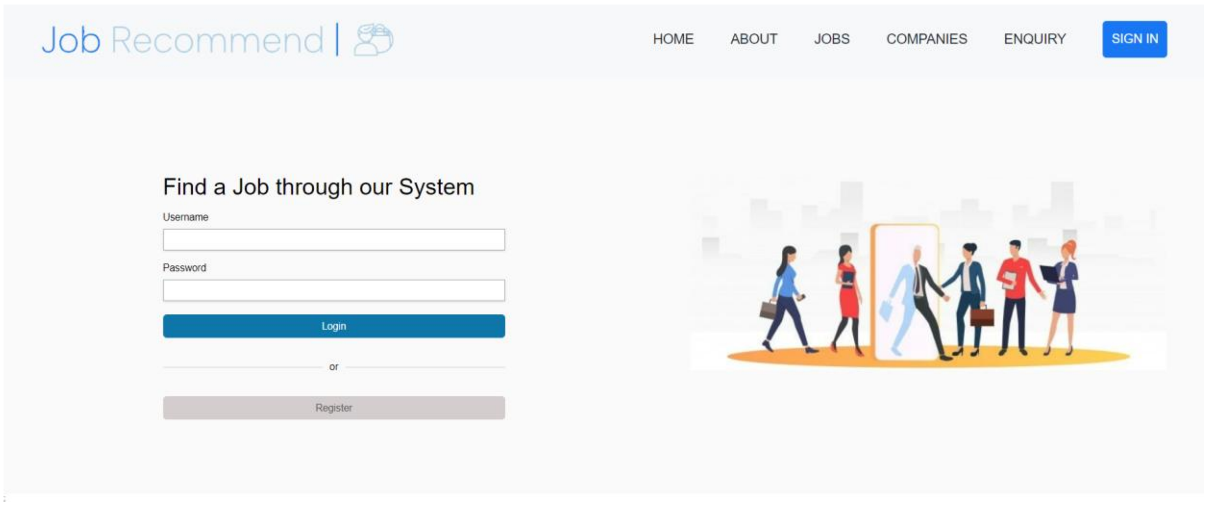
Username *
sushmita

Email *
skfjasljfdsfd

Please include an '@' in the email address. 'skfjasljfdsfd' is missing an '@'.

Address *
bafal

Form Validation

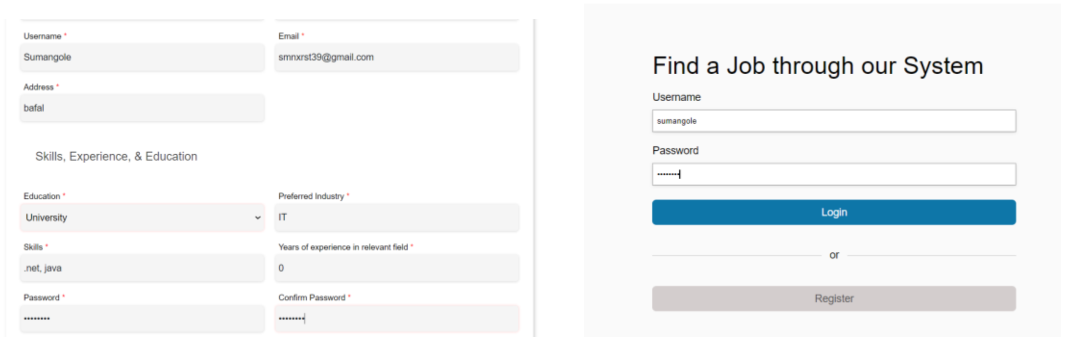


Find a Job through our System

Username:

Password:

User Login page



Username *: Sumangole

Email *: smnxrst39@gmail.com

Address *: bafal

Skills, Experience, & Education

Education *: University

Preferred Industry *: IT

Skills *: .net, java

Years of experience in relevant field *: 0

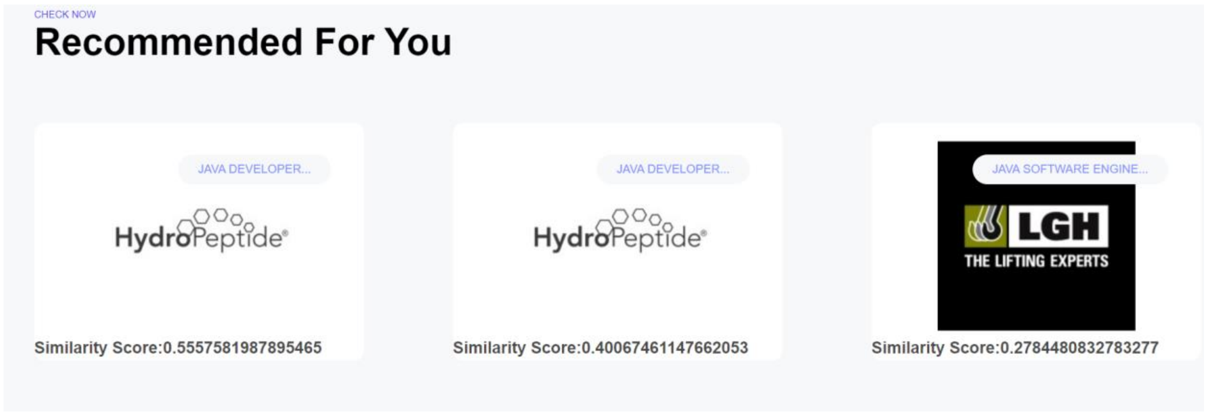
Password *: ••••••••

Confirm Password *: ••••••••

Find a Job through our System

Username: sumangole

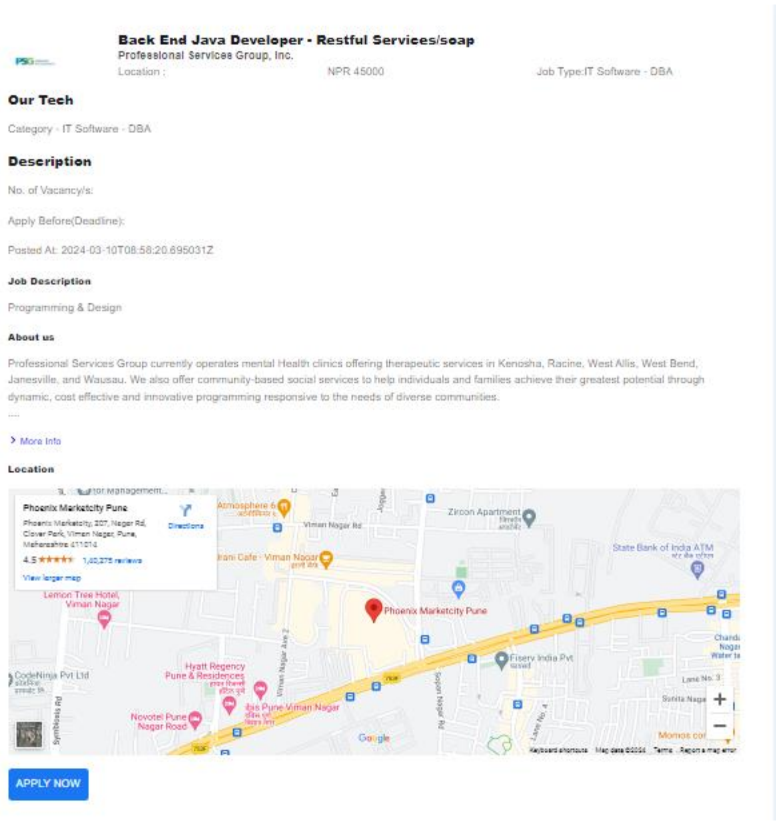
Password: •••••••



Recommended For You

JAVA DEVELOPER...	JAVA DEVELOPER...	JAVA SOFTWARE ENGINE...
Similarity Score:0.5557581987895465	Similarity Score:0.40067461147662053	Similarity Score:0.2784480832783277

Job Recommendation as per User Skills



Back End Java Developer - Restful Services/soap
Professional Services Group, Inc.
Location : NPR 45000 Job Type:IT Software - DBA

Our Tech

Category - IT Software - DBA

Description

No. of Vacancy/s:

Apply Before(Deadline):

Posted At: 2024-03-10T08:58:20.695031Z

Job Description

Programming & Design

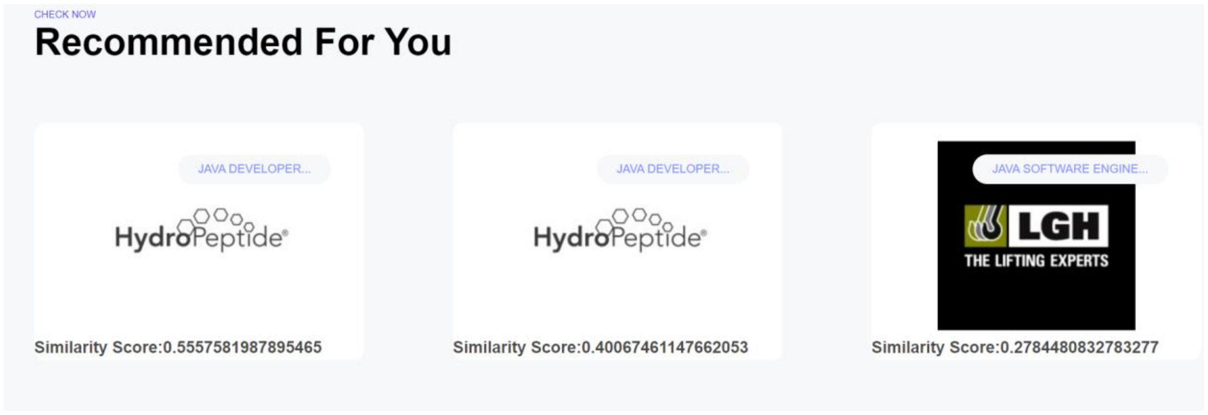
About us

Professional Services Group currently operates mental Health clinics offering therapeutic services in Kenosha, Racine, West Allis, West Bend, Janesville, and Wausau. We also offer community-based social services to help individuals and families achieve their greatest potential through dynamic, cost effective and innovative programming responsive to the needs of diverse communities.

.....

> More Info

Location



CHECK NOW

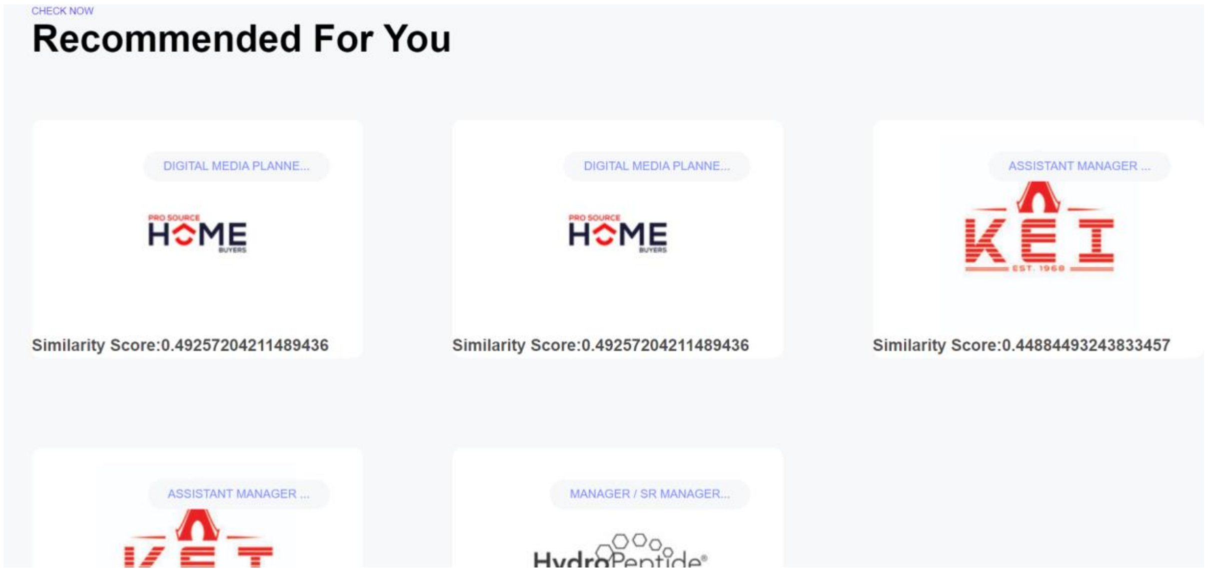
Recommended For You

Similarity Score:0.5557581987895465

Similarity Score:0.40067461147662053

Similarity Score:0.2784480832783277

Recommendation after Clicking on a Job Listing


CHECK NOW

Recommended For You

DIGITAL MEDIA PLANNE...

Similarity Score:0.49257204211489436

DIGITAL MEDIA PLANNE...

Similarity Score:0.49257204211489436

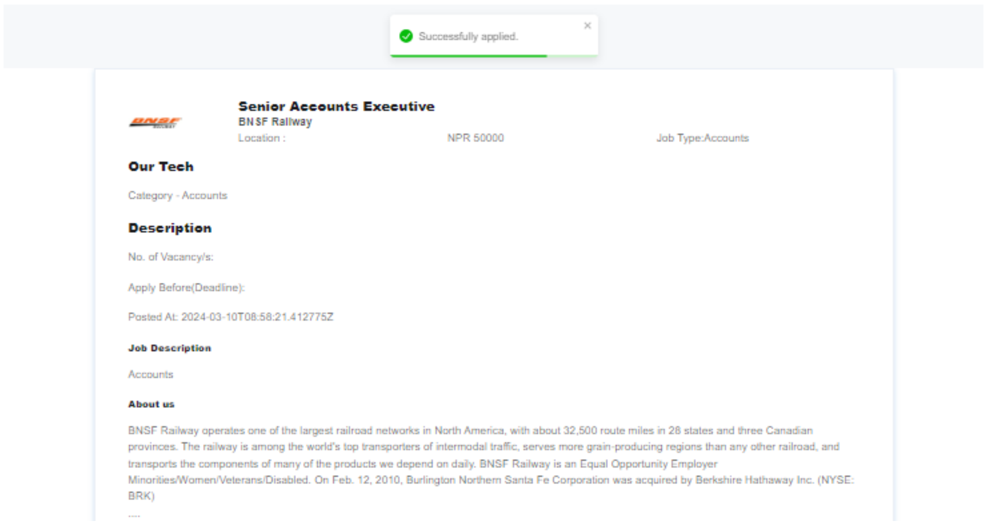
ASSISTANT MANAGER ...

Similarity Score:0.44884493243833457

ASSISTANT MANAGER ...

MANAGER / SR MANAGER...

Recommendation after Crossing the Interaction Limit


Successfully applied.

Senior Accounts Executive
BN SF Railway
Location : NPR 50000 Job Type:Accounts

Our Tech

Category - Accounts

Description

No. of Vacancy/s:

Apply Before(Deadline):

Posted At: 2024-03-10T08:58:21.412775Z

Job Description

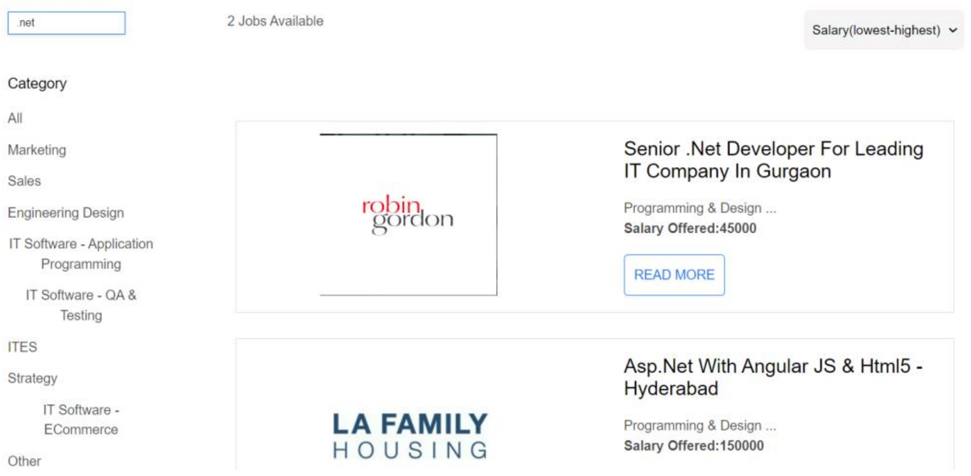
Accounts

About us

BNSF Railway operates one of the largest railroad networks in North America, with about 32,500 route miles in 28 states and three Canadian provinces. The railway is among the world's top transporters of intermodal traffic, serves more grain-producing regions than any other railroad, and transports the components of many of the products we depend on daily. BNSF Railway is an Equal Opportunity Employer Minorities/Women/Veterans/Disabled. On Feb. 12, 2010, Burlington Northern Santa Fe Corporation was acquired by Berkshire Hathaway Inc. (NYSE: BRK)

....

Job Application Acceptance


.net

2 Jobs Available

Salary(lowest-highest)

Category

All

Marketing

Sales

Engineering Design

IT Software - Application Programming

IT Software - QA & Testing

ITES

Strategy

IT Software - ECommerce

Other

Senior .Net Developer For Leading IT Company In Gurgaon

Programming & Design ...

Salary Offered:45000

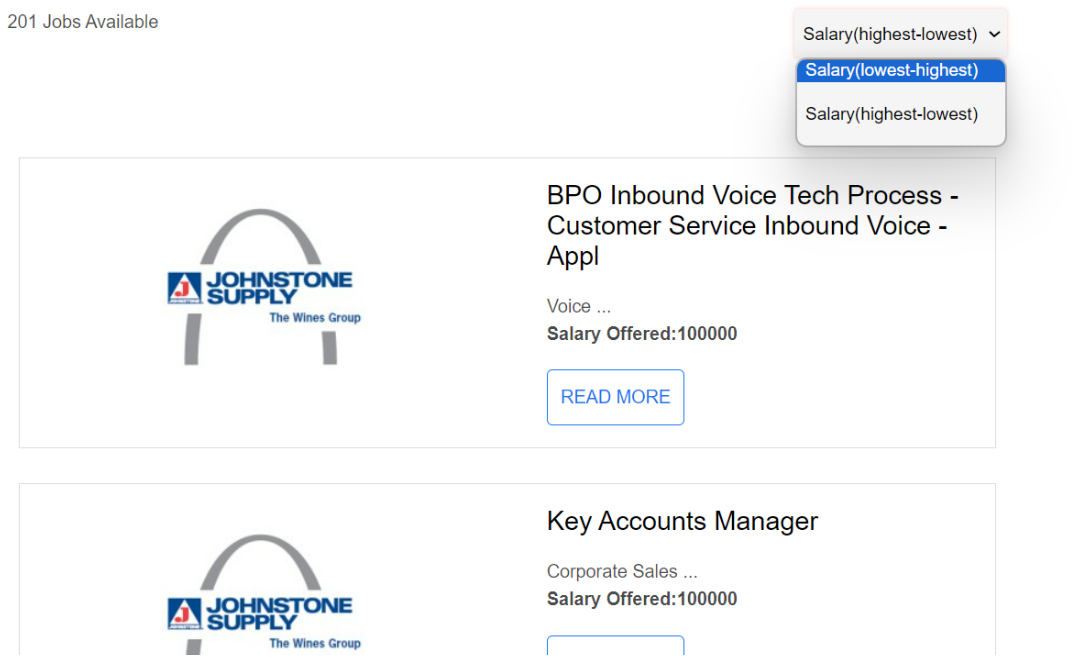
READ MORE

Asp.Net With Angular JS & Html5 - Hyderabad

Programming & Design ...

Salary Offered:150000

Searching Job


201 Jobs Available

Salary(highest-lowest)

Salary(lowest-highest)

Salary(highest-lowest)

BPO Inbound Voice Tech Process - Customer Service Inbound Voice - Appl

Voice ...

Salary Offered:100000

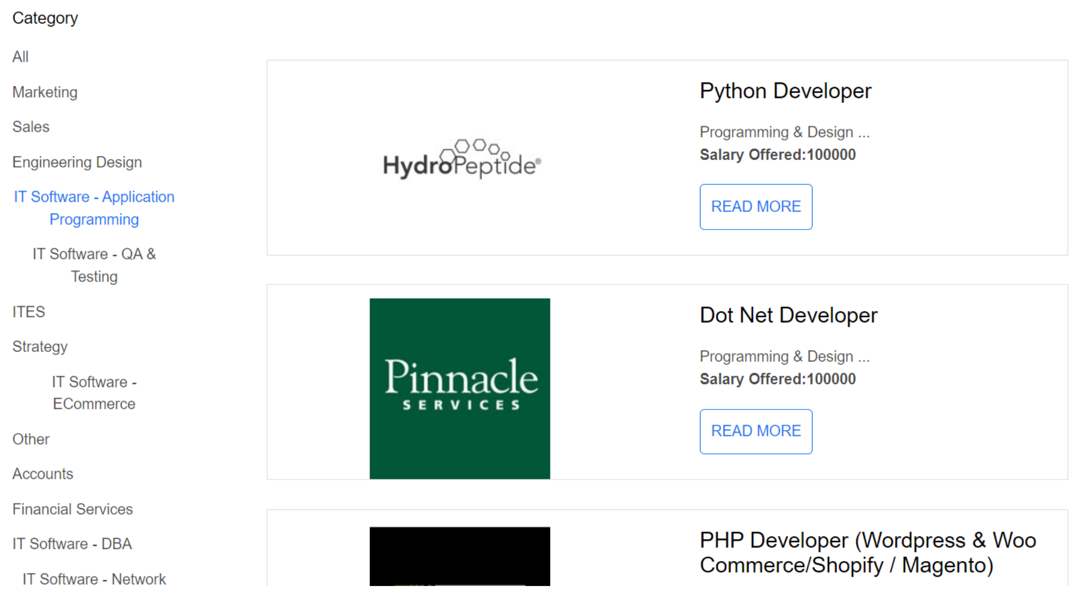
READ MORE

Key Accounts Manager

Corporate Sales ...

Salary Offered:100000

Sorting Renumeration


Category

All

Marketing

Sales

Engineering Design

IT Software - Application Programming

IT Software - QA & Testing

ITES

Strategy

IT Software - ECommerce

Other

Accounts

Financial Services

IT Software - DBA

IT Software - Network

Python Developer

Programming & Design ...

Salary Offered:100000

READ MORE

Dot Net Developer

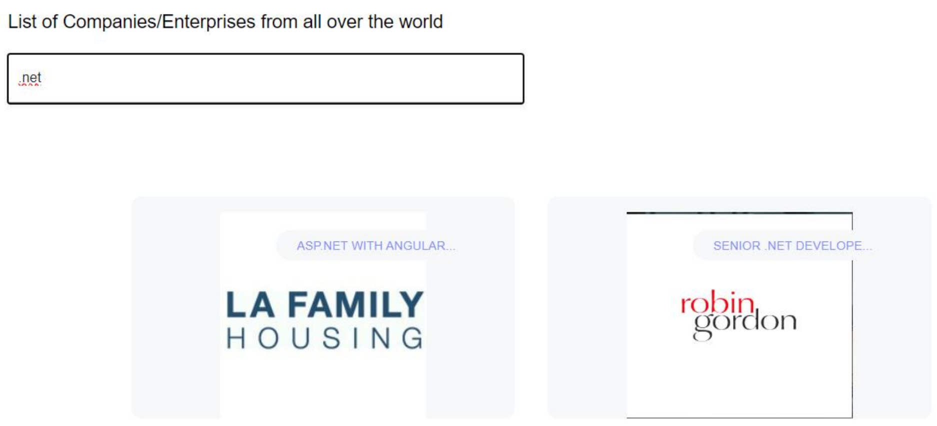
Programming & Design ...

Salary Offered:100000

READ MORE

PHP Developer (Wordpress & Woo Commerce/Shopify / Magento)

Filtering

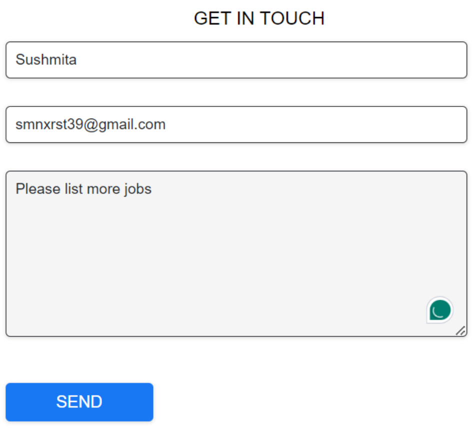

List of Companies/Enterprises from all over the world

.net

ASP.NET WITH ANGULAR...

SENIOR .NET DEVELOPE...

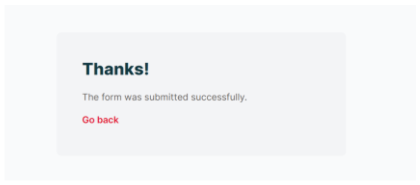
Job Search in Company


GET IN TOUCH

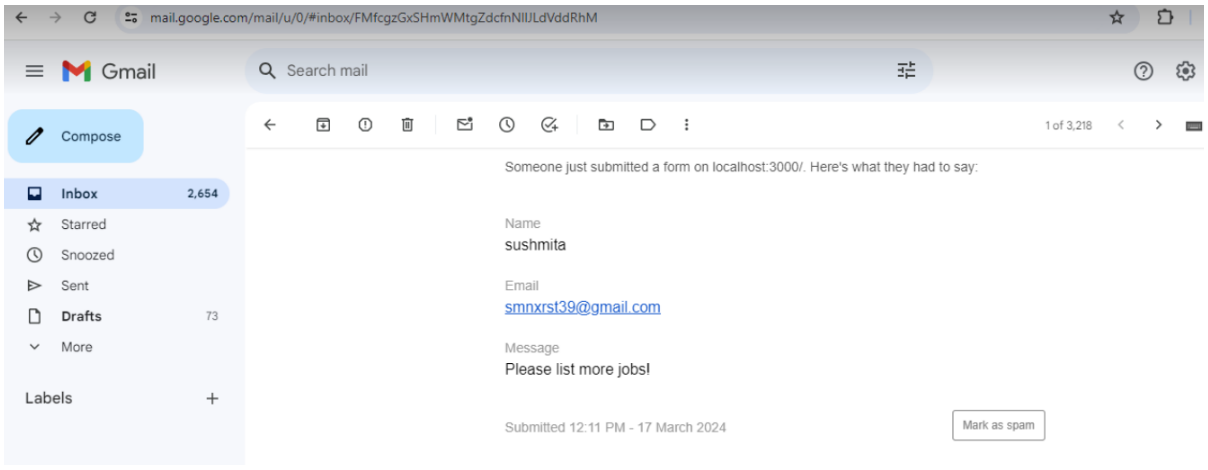
Sushmita

smnxrst39@gmail.com

Please list more jobs


Thanks!

The form was submitted successfully.



Someone just submitted a form on localhost:3000/. Here's what they had to say:

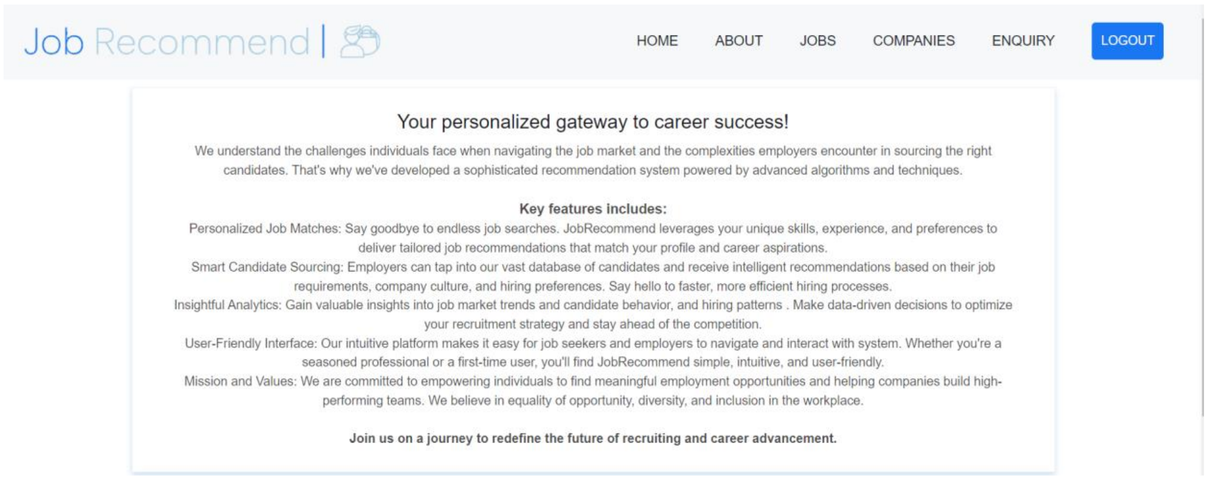
Name
sushmita

Email
smnxrst39@gmail.com

Message
Please list more jobs!

Submitted 12:11 PM - 17 March 2024

Enquiry Submission



Your personalized gateway to career success!

We understand the challenges individuals face when navigating the job market and the complexities employers encounter in sourcing the right candidates. That's why we've developed a sophisticated recommendation system powered by advanced algorithms and techniques.

Key features includes:

Personalized Job Matches: Say goodbye to endless job searches. JobRecommend leverages your unique skills, experience, and preferences to deliver tailored job recommendations that match your profile and career aspirations.

Smart Candidate Sourcing: Employers can tap into our vast database of candidates and receive intelligent recommendations based on their job requirements, company culture, and hiring preferences. Say hello to faster, more efficient hiring processes.

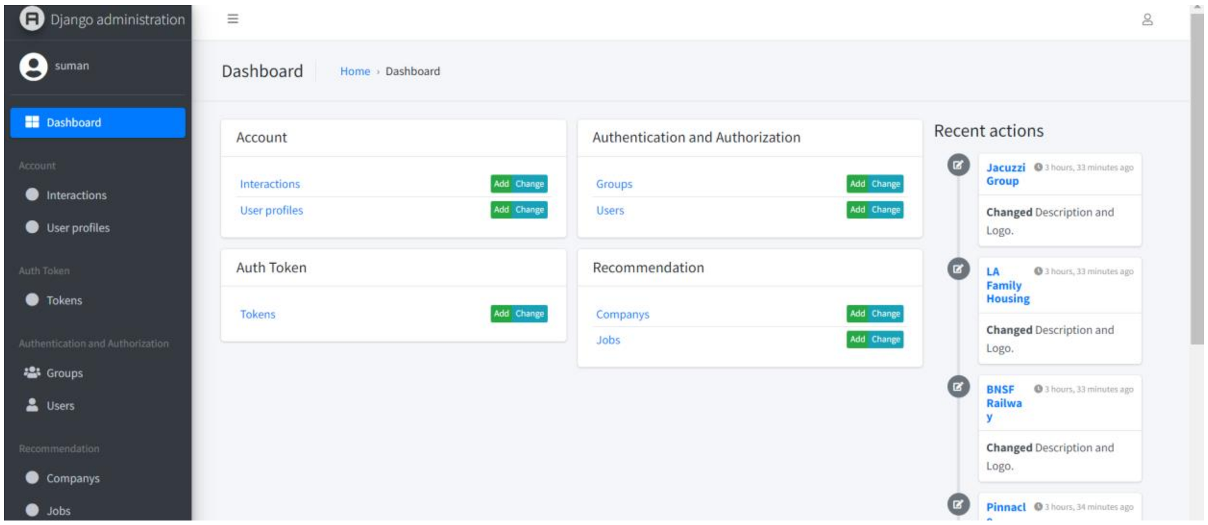
Insightful Analytics: Gain valuable insights into job market trends and candidate behavior, and hiring patterns . Make data-driven decisions to optimize your recruitment strategy and stay ahead of the competition.

User-Friendly Interface: Our intuitive platform makes it easy for job seekers and employers to navigate and interact with system. Whether you're a seasoned professional or a first-time user, you'll find JobRecommend simple, intuitive, and user-friendly.

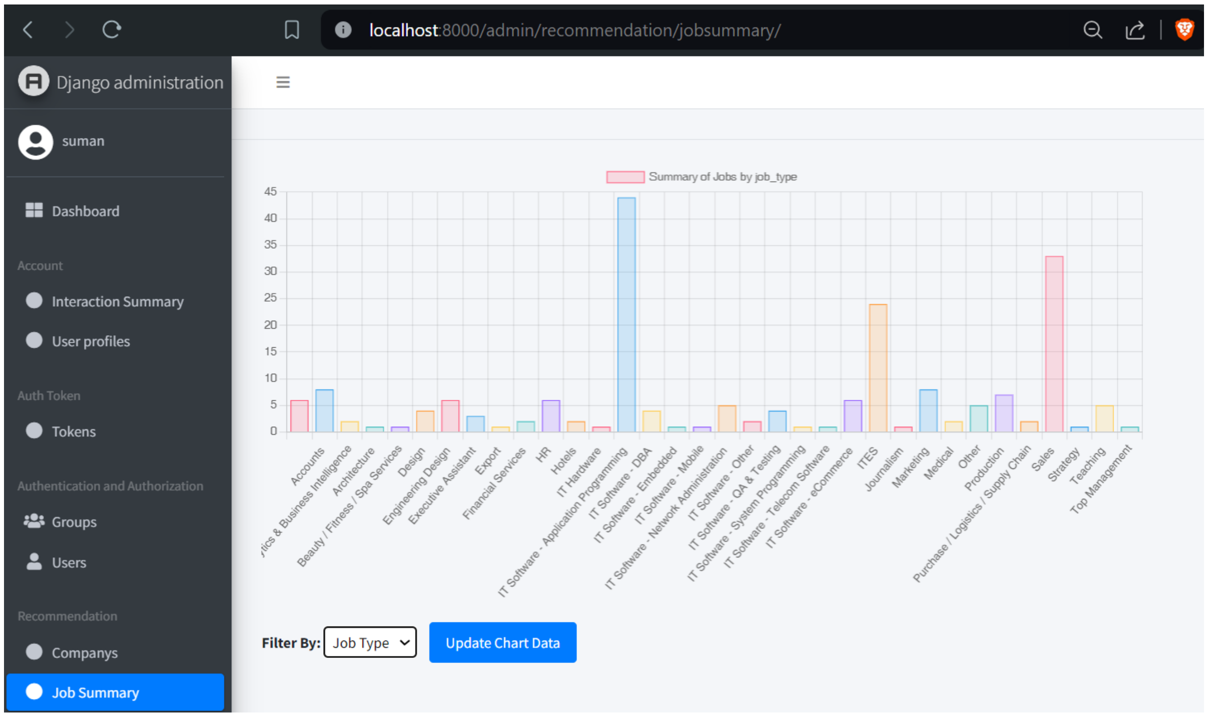
Mission and Values: We are committed to empowering individuals to find meaningful employment opportunities and helping companies build high-performing teams. We believe in equality of opportunity, diversity, and inclusion in the workplace.

Join us on a journey to redefine the future of recruiting and career advancement.

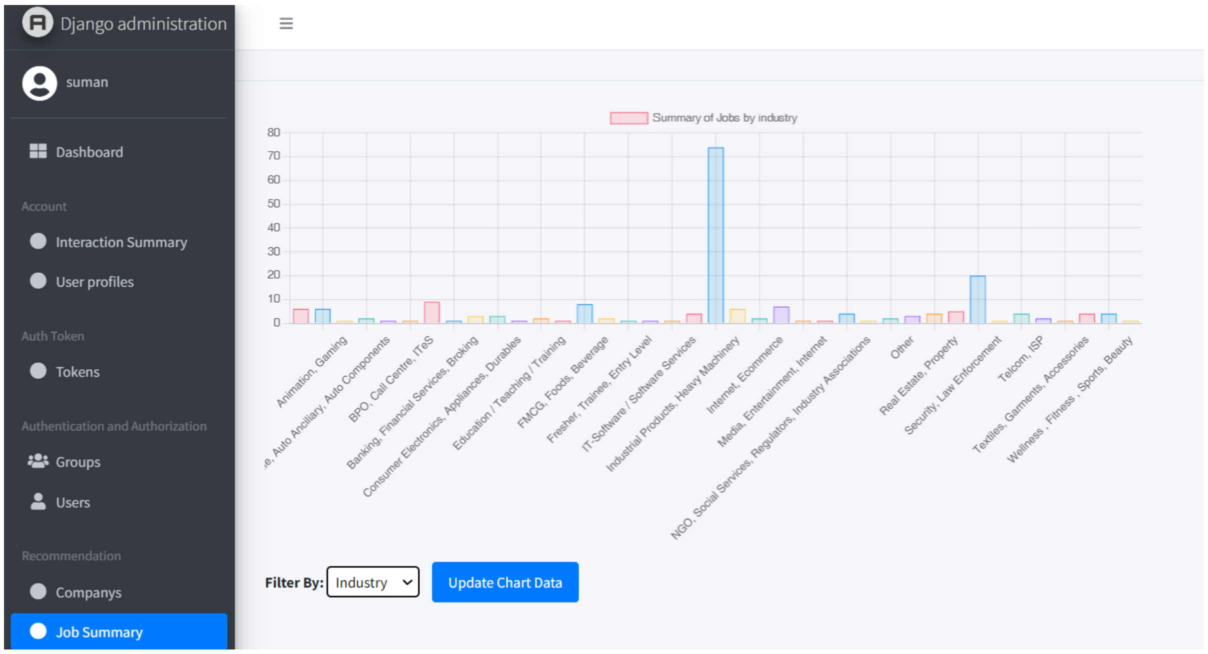
About Page



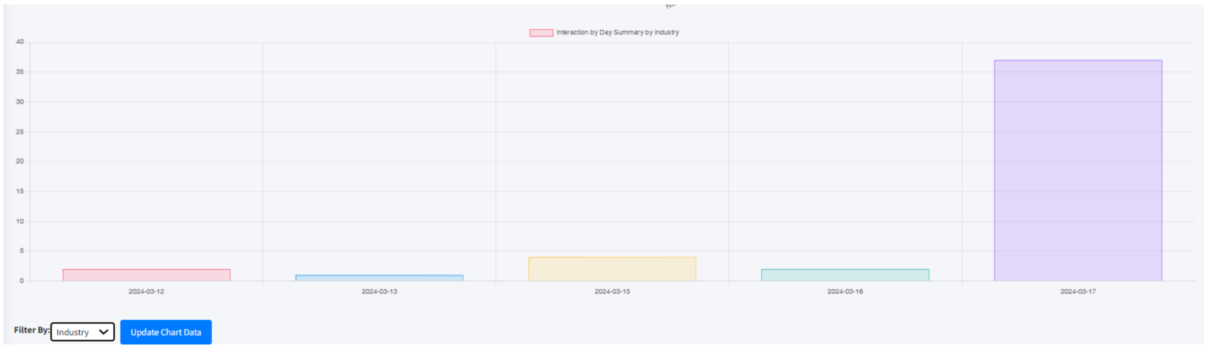
Django Administration Panel



Bar Graph of Job Type shown Admin Dashboard


Django administration
suman
Dashboard
Account
Interaction Summary
User profiles
Auth Token
Tokens
Authentication and Authorization
Groups
Users
Recommendation
Companys
Job Summary
Summary of Jobs by industry
80
70
60
50
40
30
20
10
0
Animation, Gaming
e, Auto Anciliary, Auto Components
BPO, Call Centre, ITeS
Banking, Financial Services, Broking
Consumer Electronics, Appliances, Durables
Education / Teaching / Training
FMCG, Foods, Beverage
Fresher, Trainee, Entry Level
IT-Software / Software Services
Industrial Products, Heavy Machinery
Internet, Ecommerce
Media, Entertainment, Internet
NGO, Social Services, Regulators, Industry Associations
Other
Real Estate, Property
Security, Law Enforcement
Telcom, ISP
Textiles, Garments, Accessories
Wellness , Fitness , Sports, Beauty
Filter By:
Industry
Update Chart Data

Bar Graph of Jobs as per Industry



Bar Graph of user Interaction by day