Certificates

Supervisor Recommendation

This is to certify that the report entitled "Job Recommendation System using Cosine Similarity Algorithm" has been prepared under my supervision by **Bimal Shrestha (20227/075), Chetan Raj Budhathoki (20232/075) and Nabin Bhandari (20247/075)** in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology.

I have read and approved the report and recommend it for evaluation. The report meets the required standards for the degree program and demonstrates a high level of understanding and competence in the field of computer science and information technology.

……..……..…

Saroj Maharjan

Supervisor/Visiting Faculty

CSIT Department

Bhaktapur Multiple Campus

Dudhpati-02, Bhaktapur

Certificate of Approval

This is to certify that this project prepared by **Bimal Shrestha (20227/075), Chetan Raj Budhathoki (20232/075) and Nabin Bhandari (20247/075)** entitled "Job Recommendation System using Cosine Similarity Algorithm" in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology has been well studied. In our opinion, it is satisfactory in the scope and quality of a project for the required degree.

……..……..…

Saroj Maharjan
Supervisor/Visiting Faculty
CSIT Department
Bhaktapur Multiple Campus
Dudhpati-02, Bhaktapur

……..……..…

Sushant Paudel
Department Head
CSIT Department
Bhaktapur Multiple Campus
Dudhpati-02, Bhaktapur

……..……..…

Internal Examiner
CSIT Department
Bhaktapur Multiple Campus
Dudhpati-2, Bhaktapur

……..……..…

External Examiner
Central Department of
Computer Science and IT
Tribhuvan University
Kritipur, Nepal

Acknowledgment

The success and final outcome of this assignment required a lot of guidance and assistance from many people and we feel extremely fortunate to have got all this all along the completion of our assignment work. Whatever we have done is only due to such guidance and assistance and we would not forget to thank them. We take this opportunity to express our profound gratitude and deep regards to our supervisor **Mr. Saroj Maharjan** for his exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessing, help and guidance given by him from time to time shall carry us a long way in the journey of life on which we were about to embark.

We could not have completed this project without help from our college “Bhaktapur Multiple Campus” which gave us supportive hands before us academically, also regarding other activities related to Information Communication Technology as well as extracurricular activities that we got to participate in. It gave us a family-like environment.

We also take this opportunity to express a deep sense of gratitude to **Mr. Sushant Poudel**, coordinator, BMC College, for his cordial support, valuable information and guidance, which helped us in completing this task through various stages.

Project Team

Chetan Raj Budhathoki

Nabin Bhandari

Bimal Shrestha

Abstract

Job searching has become increasingly challenging in recent times, with manual job portals requiring users to spend countless hours searching for job vacancies, leading to inefficiency and time wastage. To tackle this problem, this report introduces a job recommendation system that utilizes content-based filtering and cosine similarity algorithms to suggest relevant job opportunities to job seekers based on their user profile and job vacancy requirements. The system aims to improve the job search experience for both job seekers and employers by providing a personalized and efficient approach to job searching.

This report discusses the design, implementation, and evaluation of the job recommendation system. The system was developed using PHP and a MYSQL database was used to store the job listings. The user interface was designed to be user-friendly and easy to navigate. A user study was conducted to evaluate the system's effectiveness, and the results showed that the system successfully recommended relevant jobs to job seekers.

The job recommendation system has the potential to significantly impact the job market by allowing employers to better manage their resources and receive applications from interested job seekers, while also providing job seekers with a more efficient way to search for relevant job opportunities. Overall, this system is a valuable tool for job seekers and employers alike, and has the potential to transform the job search experience.

Keywords: Recommendation system, Cosine similarity, Content-based filtering

List of Abbreviations

CSS	Cascading Style Sheets
CV	Curriculum Vitae
DBMS	Database Management System
ER	Entity Relationship
HTML	Hyper Text Markup Language
JS	JavaScript
PHP	Hypertext Preprocessor
SQL	Structured Query Language
TF-IDF	Term Frequency-inverse document frequency

Table of Contents

List of Figures

List of Tables

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

A Recommendation System is a subclass of information filtering system that seeks to predict the "preference" a user would give to an item. In most of the existing job portal systems, users or job seekers have to manually search for the different job vacancies and apply for those job vacancies. The Job Recommendation System has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and, in some case, it reduces the hardships faced by the existing system. Moreover, this system is designed for particular need of the company to carry out operations in a smooth and effective manner.

The application is reduced as much as possible to avoid errors while entering the data. It also provides error messages while entering invalid data. No formal knowledge is required for the user to use this system. Thus, by this all it proves that it is user friendly.

Job Recommendation System, as described above can lead to error free, reliable and fast management systems. It can assist the user to concentrate on their own other activities rather than concentrating on record keeping.

Every organization, whether big or small, has challenges to overcome and manage the information of vacancy, job, vacancy, job candidate, interview schedule. Through this system, every organization can have their own personal account, post the jobs, and view the applications sent by different job seekers for those jobs. And also, this system is very useful for those people who are seeking job opportunities. This system will ultimately allow employers to better manage resources. This system is equally advantageous for job seekers as they can manually search for the jobs in which they are interested and apply for that job. Another important feature of this system is it recommends the job seeker, a list of relevant jobs matching the content of user profile and job specification of job vacancies using content-based filtering and cosine similarity algorithm. Thus, this unique feature of the system makes it different from existing job portal system.

1.2 Problem statement

In existing Job Portal Systems such as merojob.com, indeed.com, SimplyHired.com etc., job seekers have to manually search for the job vacancies that are suitable to them.

So, our motive is to solve this problem by introducing the system in which job seekers can register their account and upload their resume and create their own profile based on their experience and skills. According to their profile, our system suggests the list of the jobs that are relevant to them.

Relevant job recommendations are therefore crucial for a good user experience. Here we present a method to compute the similarities between user profiles and job specification related to different jobs using cosine similarities and provide a list of jobs that are according to their similarities.

1.3 Objectives

The general objective of the project is to develop a web-based application where different organizations can post job vacancies and job seekers can search and apply for the job suitable for them as well as this system will also recommend the job relevant to their job.

- To provide personalized job recommendations to job seekers based on their skills, Experience, and preferences. This will help job seekers to find job opportunities that are relevant to their career goals and aspirations.

- To provide accurate job recommendations that match the job seeker's skills and qualifications with the job requirements and qualifications provided by the employer which will increase the chances of successful job placements for both job seekers and employers.

- To provide and efficient platform that connects job seekers with potential employers, reducing the time and effort required to find suitable job opportunities.

1.4 Scope and Limitation

1.4.1 Scope

Job recommendation systems have a wide scope in today's job market, since the technology advancement have changed the working principle of every organization and it has made general public to use the technology in a massive way. Job recommendation systems also have a wide scope in the context of Nepal, where the job market is rapidly growing and changing.

Our system ensures that the website is available, accessible, and reached to every Nepali job seeker and recruiter. In the near future this recommendation system will fulfill the necessity in a speedy manner.

The scope of the system can be listed as follow:
It can act as a platform for maintaining job seeker and employer records, posting job details and job application lists, providing right job for right person.

It can act as a system for the large data collection from the different users (job seekers and employers), that can be processed for different use cases.

- Maintain job seeker and employer records.
- Provide customized job postings.
- Maintain job posting details.
- Maintain the job applications lists.

1.4.2 Limitation

The limitations of the project are listed below:

1. Limited data: A job recommendation system relies heavily on data to generate recommendations. If there is limited data available, the system may not be able to provide accurate or relevant job recommendations.

2. Inability to handle context: Cosine similarity is a context-free measure of similarity and may not be able to handle context-specific job requirements. For example, a job that requires a certain level of experience may be recommended to job seekers with similar skills but without the required experience.

1.5 Development Methodology

Since the requirements were pretty much clear from the beginning and time was allocated likewise, we found waterfall model as the best fit to this system as the System Development Lifecycle (SDLC).

Here's how waterfall model was applied.

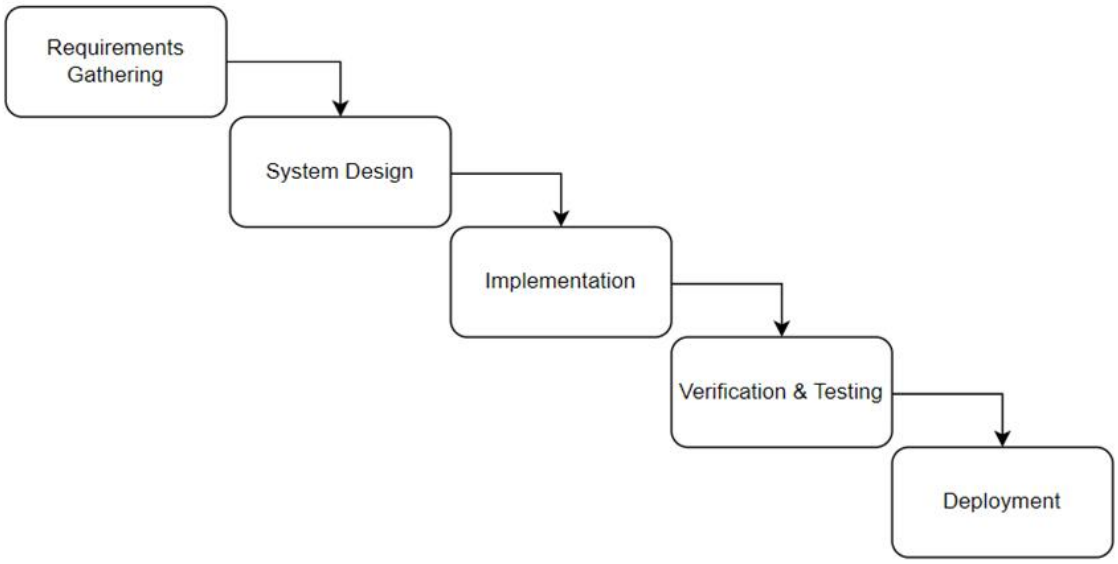


Figure 1: Methodology diagram of Project

1.5.1 Requirement Gathering

The first step in the waterfall model is to gather all the requirements for the job recommendation system. We tried understanding the user requirements, business goals and

system constraints. Clearly defining the objectives of the system and the kind of users interacting with the system like job seekers, recruiters, understanding their specific requirements and the level of personalization needed. Data sources were identified that will be used to provide job recommendations such as job postings, candidate profiles and user behavior data.

1.5.2 Design

Once the requirements were identified we designed the system architecture which involved creating a high-level design for the job recommendation system where we defined the data models, algorithms, and the overall system architecture. We also considered the user interface where that will be used by users to easily interact with the system.

1.5.3 Implementation

In this phase, the job recommendation system was built according to the design specifications. We implemented the algorithms and data models, as well as the user interface and other features that were specified in the design phase. We also conducted thorough testing to ensure the system is functioning as intended.

1.5.4 Testing

In the testing phase, we conducted comprehensive testing to ensure that the job recommendation system is free of bugs and meets all the requirements. We conducted unit testing, integration testing, and system testing to ensure that the system is working as expected.

1.5.5 Deployment

Once the job recommendation system was thoroughly tested and validated, we deployed it in a production environment. This involved installing the system on the production servers and configuring it for use, by end-users.

1.5.6 Maintenance

After the job recommendation system was deployed, we have been constantly monitoring it to ensure that it's working properly. Any bugs that may arise will be fixed and necessary updates to the system will be provided to keep it up-to-date with changing user needs.

1.6 Report Organization

The report is divided into seven chapters along with their descriptions, and also there include some additional sections of references and a conclusion.

1.6.1 Chapter 1: Introduction

In this chapter, the project "Job Recommendation System" is described. This chapter discusses the questions: what is the project about, what problems does the project address, what are the goals of the project, etc.?

1.6.2 Chapter 2: Background Study and Literature Review

In this chapter, the underlying concept and theories used in the project are discussed. As the project is to create a system (web application) As the project is to create a system (web application), that focuses on recommending jobs based on the cosine similarity between contents from different parties. Different research article done on recommending systems are mentioned and discussed.

1.6.3 Chapter 3: System Analysis

In this chapter, the analysis of the system is done in various aspects such as requirement analysis, various feasibility analyses, and also a brief analysis of the overall system is done.

1.6.4 Chapter 4: System Design

In this chapter, the main system design of the system is discussed and various respective system designs drawn are discussed. As the system is object-oriented based on the class, object, state, sequence, and activity diagrams, components diagrams, deployment diagrams, and also the major algorithm used in the project segment intersection, linear interpolation, and multi-layer perceptron neural network are discussed.

1.6.5 Chapter 5: Implementation and Testing

In this chapter, how the project was conducted, what programming languages, CASE tools, and technologies were used for completing the project are discussed, and also what testing methodologies and test cases conducted during the implementation period are discussed.

1.6.6 Chapter 6: Conclusion and Recommendation

In this chapter the summary of the overall project, what was achieved, and what future enhancement in the project can be done for the project is discussed.

CHAPTER 2
BACKGROUND STUDY AND LITERATURE REVIEW

2.1 Background Study

The fundamental theories, general concepts, and terminologies related to the project are discussed below:

2.1.1 Fundamental Concepts

2.1.1.1 Recommender systems

A recommender system is a subclass of information filtering system that seeks to predict the "rating" or "preference" that a user would give to an item. Recommender systems are widely used in various applications, including e-commerce, social media, and job search engines.

2.1.1.2 Job Matching

Job matching refers to the process of matching job seekers with job postings that best fit their skills, experience, and preferences. Job recommendation systems use a variety of techniques, such as machine learning and candidate matching algorithms, to provide accurate and relevant job recommendations.

2.1.2 Theories

2.1.2.1 Cosine Similarity

Cosine similarity is a mathematical technique used to measure the similarity between two vectors in a high-dimensional space. It is commonly used in information retrieval and recommendation systems to compare the similarity between documents or items based on their content or attributes.

The cosine similarity is calculated as the cosine of the angle between two vectors. It ranges from -1 to 1, with values closer to 1 indicating a higher degree of similarity between the two vectors.

In the field of information retrieval, cosine similarity is widely used to rank documents based on their relevance to a query. In a study by Manning et al. (2008), cosine similarity was found to be an effective measure for information retrieval tasks, such as document ranking and query expansion. The study showed that cosine similarity outperformed other similarity measures, such as Euclidean distance and Jaccard similarity, in terms of retrieval effectiveness.

2.1.2.1 Content-based filtering

Content-based filtering is a technique used in recommendation systems that recommends items to users based on their similarity to items that the user has previously shown interest in or interacted with. The system uses information about the content or attributes of the items to identify other items that are similar in some way.

2.1.3 Terminologies

2.1.3.1 Job recommendation system

A system that recommends job openings to job seekers based on their skills, experience, and preferences.

2.1.3.2 Content-based filtering

A technique that matches job seekers with job openings based on the similarity of their skills, experience, and job preferences.

2.1.3.3 Accuracy

A measure of how closely the recommended job openings match the job seeker's preferences.

2.1.3.4 Personalization

Personalization refers to the ability of job recommendation systems to provide personalized job recommendations based on a user's skills, experience, and preferences. Personalization can help increase user engagement and satisfaction with the system.

2.2 Literature Review

Recommender Systems have become an important research field since the emergence of the first paper on collaborative filtering in the mid-1990s. In general, these systems are stated as the support systems which help users to find content, products, or services (such as books, movies, music, TV programs, and websites). By gathering and examining suggestions from other users, which means reviews from various establishments, and users.

These systems are broadly classified into collaborative filtering (CF) and content-based filtering (CB). Content-Based recommender system [1] tries to guess the features or behavior of a user given the item's features, he/she reacts positively to.

In 2008 [2] Kleanthi Lakiotaki, Stelios Tsafarakis, and Nikolaos Matsatsinis proposed UTA-Rec. UTARec is a Recommender System that incorporates Multiple Criteria Analysis methodologies. The system's performance and capability of addressing certain shortfalls of existing Recommender Systems is demonstrated in the case of movie recommendations. UTARec's accuracy is measured in terms of Kendall's tau and ROC curve analysis and is also compared to a Multiple Rating Collaborative Filtering (MRCF) approach. Juan A.

Mucheol Kim & Sang Oh Park [3] in 2011 proposed an intelligent movie recommender system with a social trust model. The proposed system is based on a social network for analyzing social relationships between users and generating group affinity values with user profiles. In experiments, the performance of this system is evaluated with precision-recall and F-measures.

In 2012 Punam Bedi, Ravish Sharma [4] proposed a Trust based Ant Recommender System (TARS) that produces valuable recommendations by incorporating a notion of dynamic trust between users and selecting a small and best neighborhood based on the biological metaphor of ant colonies.

Along with the predicted ratings, displaying additional information for explanation of recommendations regarding the strength and level of connectedness in the trust graph from where recommendations are generated, items and number of neighbors involved in predicting ratings can help active users make better decisions. Also, new users can highly benefit from pheromone updating strategy known from ant algorithms as positive feedback in the form of aggregated dynamic trust pheromone defines ‘‘popularity’’ of a user as recommender over a period of time. The performance of TARS is evaluated using two datasets of different sparsity levels viz. Jester dataset and Movie Lens dataset (available online) and compared with traditional Collaborative Filtering based apps convince users about the product. It is an agent-based recommender system. It combines a hybrid recommender system with automated argumentation.

Recommender system is one of the applications which is being used by many vectors and online service providers to believe the necessity of online users. Thus, the recommender system is presented as an intelligent system, which identifies the user category on the basis of user information and then user interest analysis. Once such information is obtained, in the second stage, the analysis is performed to obtain the similarity group respective to necessity products and services. To perform such an analysis there are some existing techniques such as content based as well as collaborative recommender systems.

CHAPTER 3
SYSTEM ANALYSIS

3.1 System Analysis

System analysis was conducted to investigate the system and its components for effective goal determination of the system. In the system analysis, we conducted a requirement analysis of the project to gather functional and non-functional requirements to get a clear idea of the functionalities needed for the system, also feasibility study of the project was conducted to determine whether the project is possible or not before starting it and lastly after conducting the feasibility study basic system analysis was done where we drew system analysis diagrams.

3.1.1 Requirement Analysis

Firstly, during the system analysis phase, we first conducted the requirement analysis of the project where the functional and non-functional requirements were listed down. Proper requirement analysis was conducted to get a clear picture of what system functionalities are required and no missing functionalities arise during the middle of the project.

3.1.1.1 Functional Requirements

Functional requirements are the specific features or capabilities that a system must have in order to fulfill its intended purpose. In the context of a job recommendation system, some possible functional requirements might include:

1. User Registration: The system should allow job seekers to create an account and fill out a profile that includes personal information, work experience, education, and skills.

2. Job Search: The system should allow job seekers to search for job opportunities based on keywords and job type.

3. Job Matching: The system should use algorithms and data analysis techniques to match the job seeker's qualifications with the job requirements provided by the employers.

4. Employer Registration and Job Posting: The system should allow employers to register and create a profile that includes information about their company and job opportunities. Employers should be able to post job openings, including job requirements, qualifications, and job descriptions.

5. Feedback and Evaluation: The system should allow job seekers and employers to provide feedback and evaluate the job recommendations and job placements.

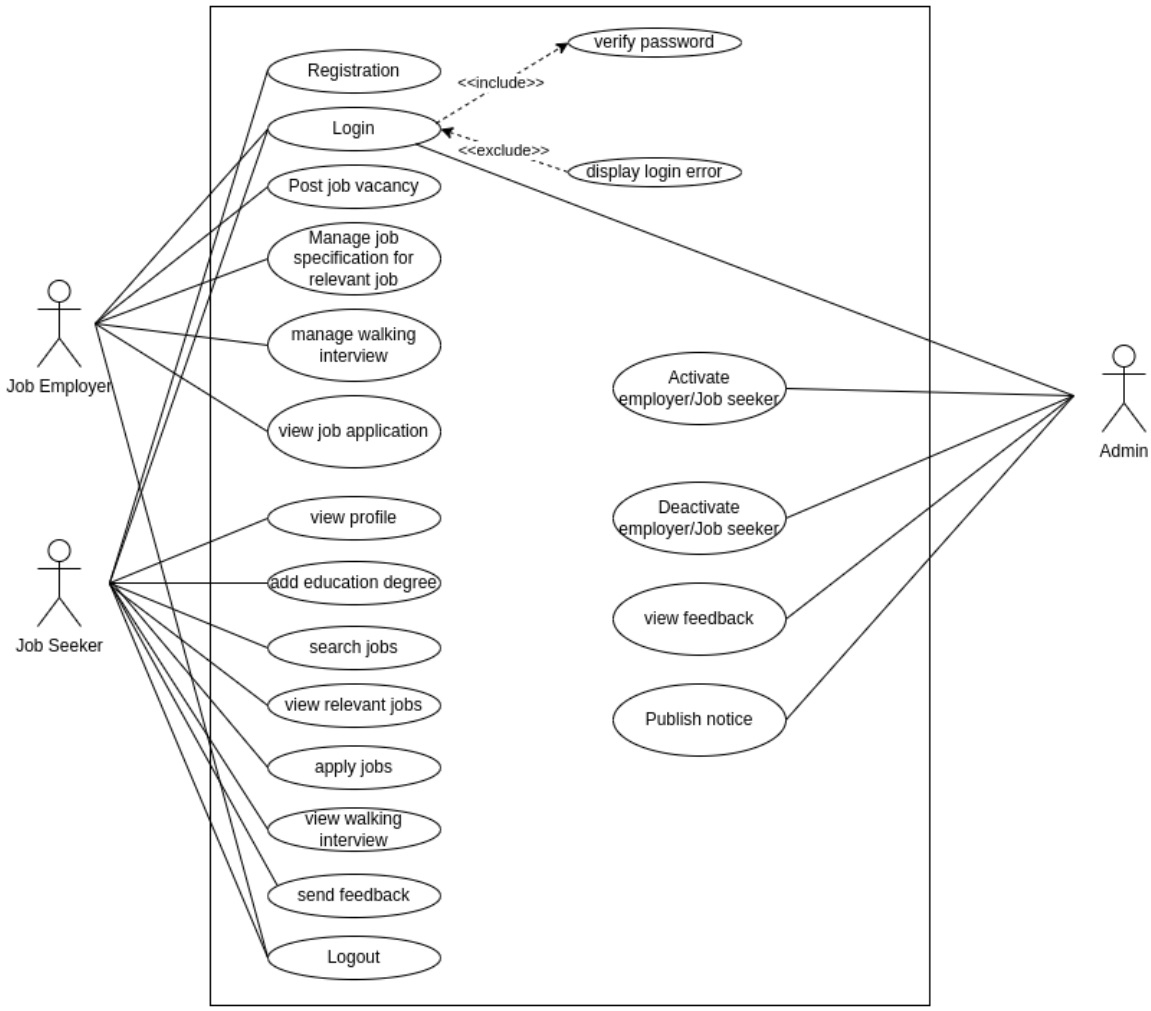


Figure 2: Functional requirement

3.1.1.2 Non-Functional Requirements

Non-functional requirements are not directly related to the functions performed by the system. Simultaneously, Non-functional requirements may relate not only to the software system itself some may relate to the technological process of creating software.

1. Usability: The system should be easy to use and intuitive for both job seekers and employers, with clear navigation and user-friendly interfaces.

2. Performance: The system should provide fast response times and be able to handle a high volume of traffic and user data.

3. Security: The system should be secure, protect user data, and prevent unauthorized access or breaches.

4. Scalability: The system should be able to handle a growing number of users and job opportunities without sacrificing performance or user experience.

5. Compatibility: The system should be compatible with multiple web browsers, operating systems, and devices.

6. Compliance: The system should comply with Nepalese labor laws and regulations and provide job recommendations that comply with them.

7. Maintainability: The system should be easy to maintain and upgrade, with clear documentation and efficient code.

3.1.2 Feasibility Analysis

After the proper requirement gathering was done, we conducted the feasibility analysis of the project to identify whether the project is feasible in every aspect and whether the project is possible to proceed in each aspect before starting the major work of the project. A feasibility study is an evaluation of a proposed project or system to determine whether it is practical, viable, and worth pursuing. In the context of a job recommendation system, a feasibility study might consider a range of factors, including technical, economic, operational, and legal considerations. Various feasibility analyses are discussed below:

3.1.2.1 Technical

From a technical perspective, a job recommendation system is feasible since the necessary technologies and infrastructure are available or can be developed. Since this system is able to handle large amounts of data and traffic, use similarity algorithms and data analysis techniques, and provide fast response times and reliable performance, this is technically feasible.

3.1.2.2 Operational

Operational feasibility refers to whether the job recommendation system can be integrated into the existing business processes and workflows. Since this system is highly needed by both the job seeker and employer and can benefit both parties, it can be considered highly operational. The system is easy to use and intuitive for both job seekers and employers. The system can also be made compatible with other relevant systems or applications, such as applicant tracking systems or social media platforms.

3.1.2.3 Economic

Our system is economic friendly and feasible since the system development, implementation costs, maintenance costs and other associated costs are comparatively cheap. It does not require large number of manpower and resources. It also helps both user parties on time and cost saving, increased efficiency and productivity. Overall, our system can self-sustain and is economically feasible.

3.1.2.4 Schedule

The schedule of the project was analyzed by drawing the Gantt chart through Microsoft Excel.

Table 1: Table of the task schedule

Task	Start Date	Duration (days)	End Date
Documentation	11th Dec	92	13th Mar
Requirement Gathering	11th Dec	5	16th Dec
System Design	17th Dec	14	31th Dec
Incremental Implementation	1st Jan	56	26th Feb
Verification & Testing	28th Feb	8	8th Mar
Deployment	9th Mar	2	11th Mar

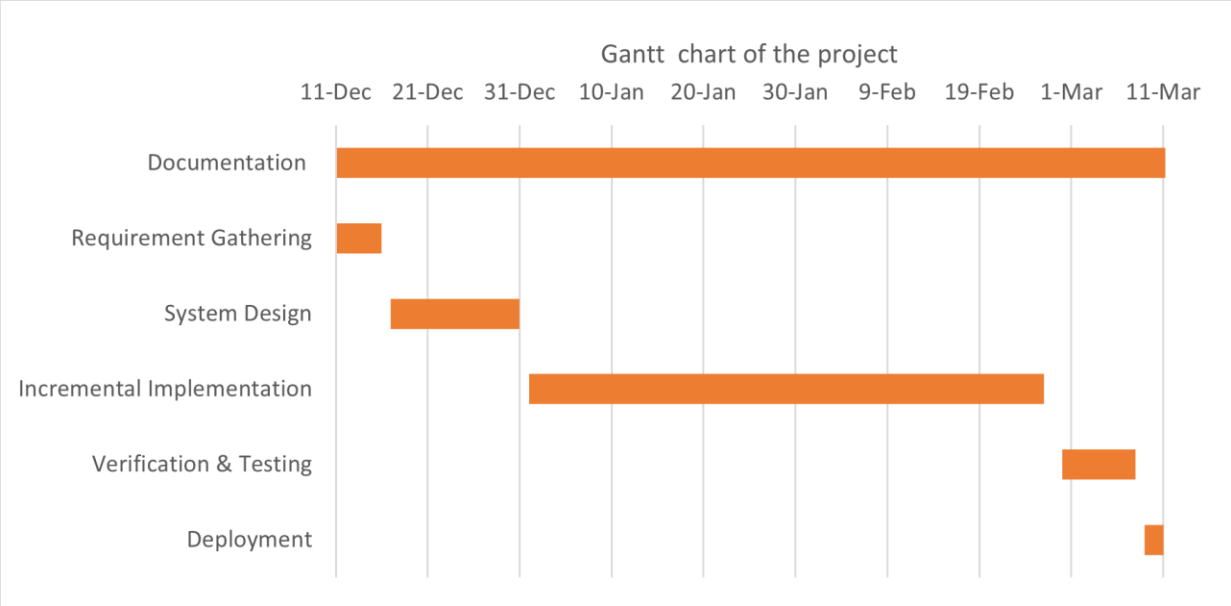


Figure 3: Gantt chart of the project

While analyzing the Gantt chart of the project it was found that the project can be completed on a fixed schedule so the project was found feasible on the time aspect.

3.1.3 Analysis

After we conducted the feasibility study, we identified that the project was feasible from every aspect and there may not arise any major hindrance during the execution of the project. After we got the assurance that the project was feasible, we conducted a thorough system

analysis of the project before starting the implementation of the project. When conducting the system analysis, the team decided to implement the project through structured approach though Object Oriented is applied for the algorithm part. Also in the analysis period, various system diagrams like ER Diagram, use case Diagram and DFD Diagram were sketched by the team to analyze the system to be developed.

3.1.3.1 Data Modeling (ER Diagram)

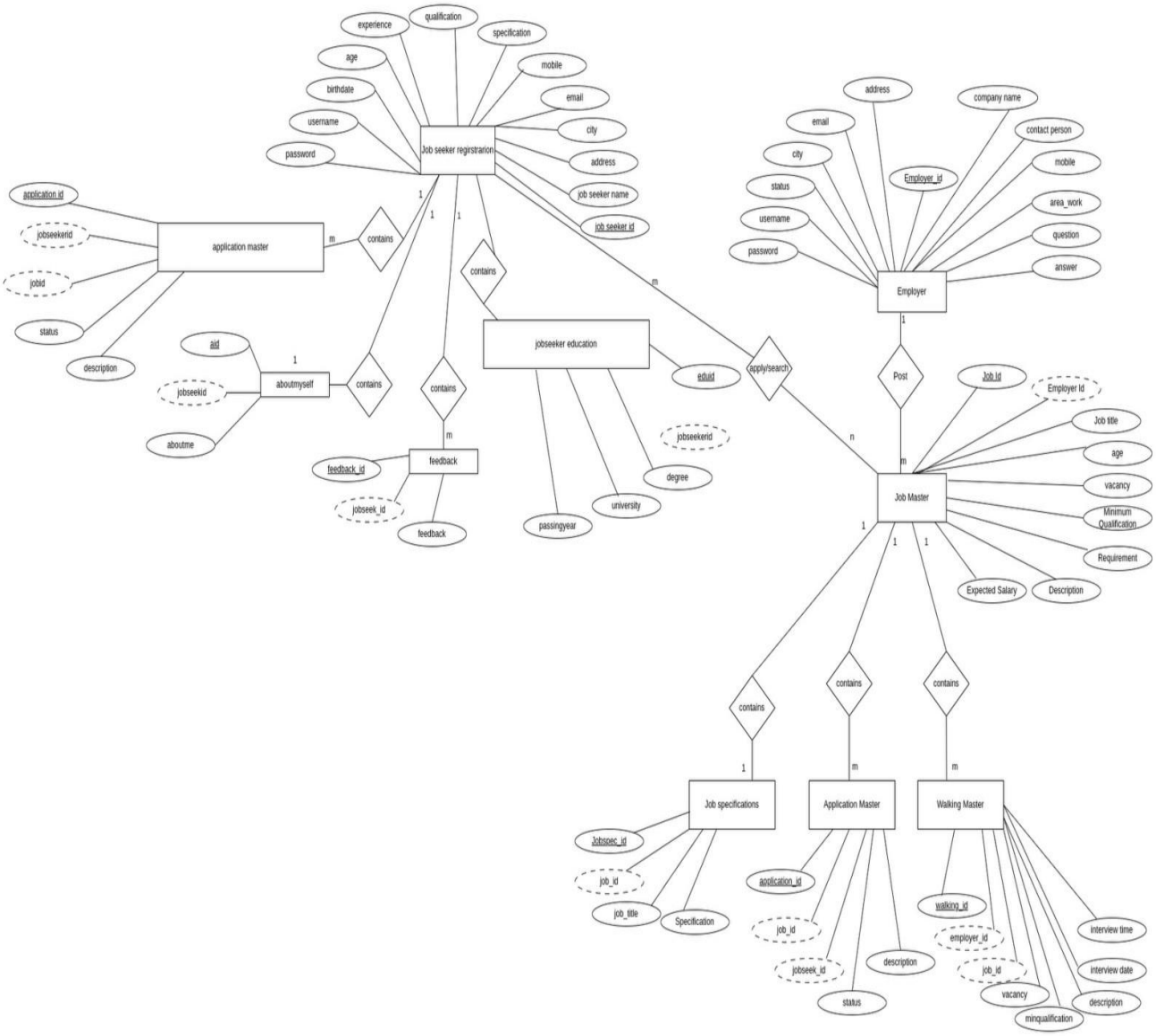


Figure 4: ER Diagram

The job recommendation project aims to develop a system that provides personalized job recommendations to job seekers based on their skills, experience, and preferences. The project utilizes an Entity-Relationship (ER) diagram to illustrate the logical structure of the data and the relationships between different entities within the system.

Entity Descriptions:

Employer:

The "Employer" entity represents the companies or organizations that post job openings. It contains information about the employers, such as their name, contact details, industry, and any other relevant information.

Job Seeker Registration:

The "Job Seeker Registration" entity captures the details of individuals who sign up as job seekers on the platform.

- It includes information such as the job seeker's name, contact information, educational background, work experience, skills, and preferences.
- The "Job Seeker Registration" entity allows for the creation and maintenance of user profiles, which serve as the basis for generating personalized job recommendations.

Job Master:

The "Job Master" entity stores the information related to individual job openings available on the platform.

- It contains details about each job, including the job title, description, required qualifications, job location, salary range, and other relevant attributes.

Application Master:

The "Application Master" entity records the applications submitted by job seekers for specific job openings.

- It includes information such as the application ID, job seeker ID, job ID, application status, and any additional details relevant to the application process.

Job Seeker Education:

The "Job Seeker Education" entity represents the educational background of job seekers.

- This entity provides insights into the educational qualifications of job seekers, allowing the system to match them with job openings that require specific educational criteria.

Job Specifications:

The "Job Specifications" entity contains detailed information about the requirements and qualifications for specific job openings.

- It includes attributes such as job title, job description, required skills, experience level, educational requirements, certifications, language proficiency, and any other relevant specifications.

Relationships Descriptions

"Employer" and "Job Master":

Relationship: One-to-Many (1:N)

Description: An employer can post multiple job openings, but each job opening belongs to a single employer.

Example: ABC Company (employer) has posted three job openings for different positions. Each job opening is associated with the ABC Company as the employer.

"Job Seeker Registration" and "Application Master":

Relationship: One-to-Many (1:N)

Description: A job seeker can submit multiple applications, but each application is submitted by a single job seeker.

Example: Ram Sharma (job seeker) has submitted applications for three different job openings. Each application is associated with John Doe as the applicant.

"Job Master" and "Application Master":

Relationship: One-to-Many (1:N)

Description: A job opening can receive multiple applications, but each application is specific to a single job opening.

Example: The "Software Developer" job opening has received applications from five different job seekers. Each application is associated with the "Software Developer" job opening.

"Job Seeker Registration" and "Job Master":

Relationship: Many-to-Many (M:N)

Description: A job seeker can be interested in and apply to multiple job openings, and each job opening can have multiple interested job seekers.

Example: Sulabh Gauchan (job seeker) is interested in and has applied to three different job openings, including "Marketing Coordinator," "Sales Representative," and "Project Manager." Each job opening has received applications from multiple job seekers, including Sulabh Gauchan.

Job Seeker Registration and Job Seeker Education:

Relationship: One-to-One (1:1)

Description: Each job seeker's education profile is associated with their registration information.

Example: Sita Khatiwoda(job seeker) has a corresponding education profile that includes her degree, educational institution, GPA, and certifications.

Job Specifications and Job Master:

Relationship: One-to-One (1:1)

Description: Each job opening has specific job specifications associated with it.

Example: The "Marketing Manager" job opening has detailed job specifications that outline the required skills, experience level, and educational qualifications.

Conclusion

The ER diagram for the job recommendation system incorporates the entities "Employer", "Job Seeker Registration", "Job Master", "Application Master", "Job Seeker Education", "Job Specifications". etc. These entities and their relationships enable the system to facilitate job matching and recommendation based on various criteria.

By leveraging this ER diagram, the job recommendation system can effectively match job seekers with suitable job openings based on factors such as education, skills, experience, and job specifications. It enables employers to post job openings, receive applications, and evaluate potential candidates, while job seekers can register, apply to job openings, and showcase their qualifications.

Overall, the ER diagram provides a visual representation of the entities and relationships within the job recommendation system, offering a solid foundation for the system's implementation and functionality.

3.1.3.2 Process modeling using DFD

Level 0 DFD Diagram:

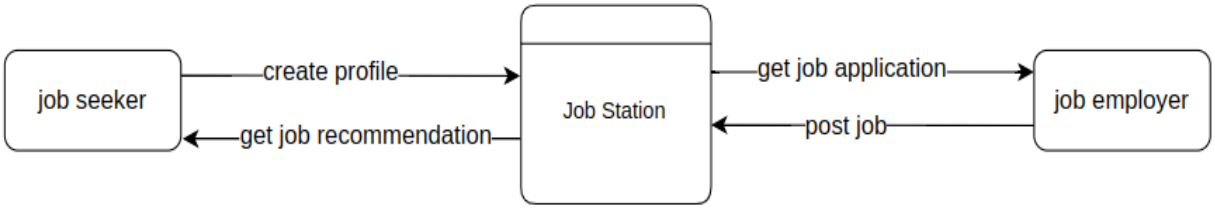


Figure 5: 0-level DFD

In this diagram, we have the following components:

1. External Entities

Job Seeker:

These are individuals seeking job recommendations. They interact with the system by providing their preferences and receiving job recommendations.

Job Employer:

These are individuals providing job details. They interact with the system by providing their job details and receiving job applications.

2. System:

Job Station:

This system represents the core functionality of the system. It receives use preferences and utilizes cosine similarity to recommend suitable jobs.

3. Data Flows:

Seeker Preferences:

This data flow represents the input from job seekers, including their skills, qualifications.

Employer Preferences:

This data flow represents the input from job employer, including job details.

Recommended jobs:

This data flow represents the output of the system.

Level 1 DFD Diagram:

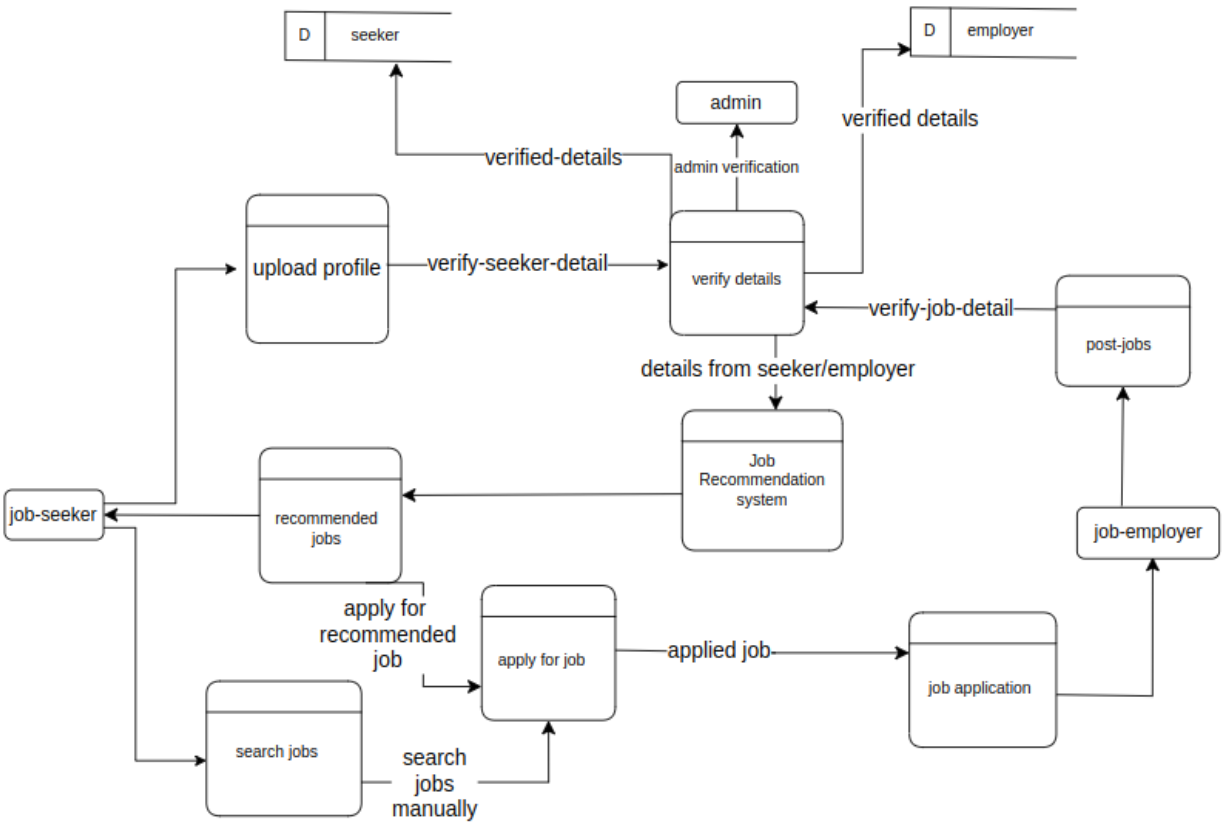


Figure 6: 1-level DFD

In this diagram, we have the following components:

1. External Entities:

Job Seeker:

These are individuals seeking job recommendations. They interact with the system by providing their preferences and receiving job recommendations.

Job Employer:

These are individuals providing job details. They interact with the system by providing their job details and receiving job applications.

2. Processes:

Job Recommendation System:

This process recommends the relative job to the job seeker by matching to the job seeker profile to the related job post.

Retrieve Seeker Preferences:

This sub-process retrieves the seekers preferences, including their skills, experience, and job preferences.

Fetch Job Data:

This sub-process fetches relevant job data from the job database.

Calculate Cosine Similarity:

This sub-process applies cosine similarity algorithm to measure the similarity between user preferences and job data.

Generate Job Recommendations:

This sub-process generates the final job recommendations based on the cosine similarity scores.

3. Data Flows:

Seeker Preferences:

This data flow represents the input from job seekers, including their skills, qualifications.

Job Data:

This data flow represents the input from employer.

Cosine Similarity Scores:

This data flow represents the scores calculated through the cosine similarity algorithm.

Recommended Jobs:

This data flow represents the data present to the seeker dashboard after similar jobs are found and recommended to the seeker.

Level 2 DFD Diagram:

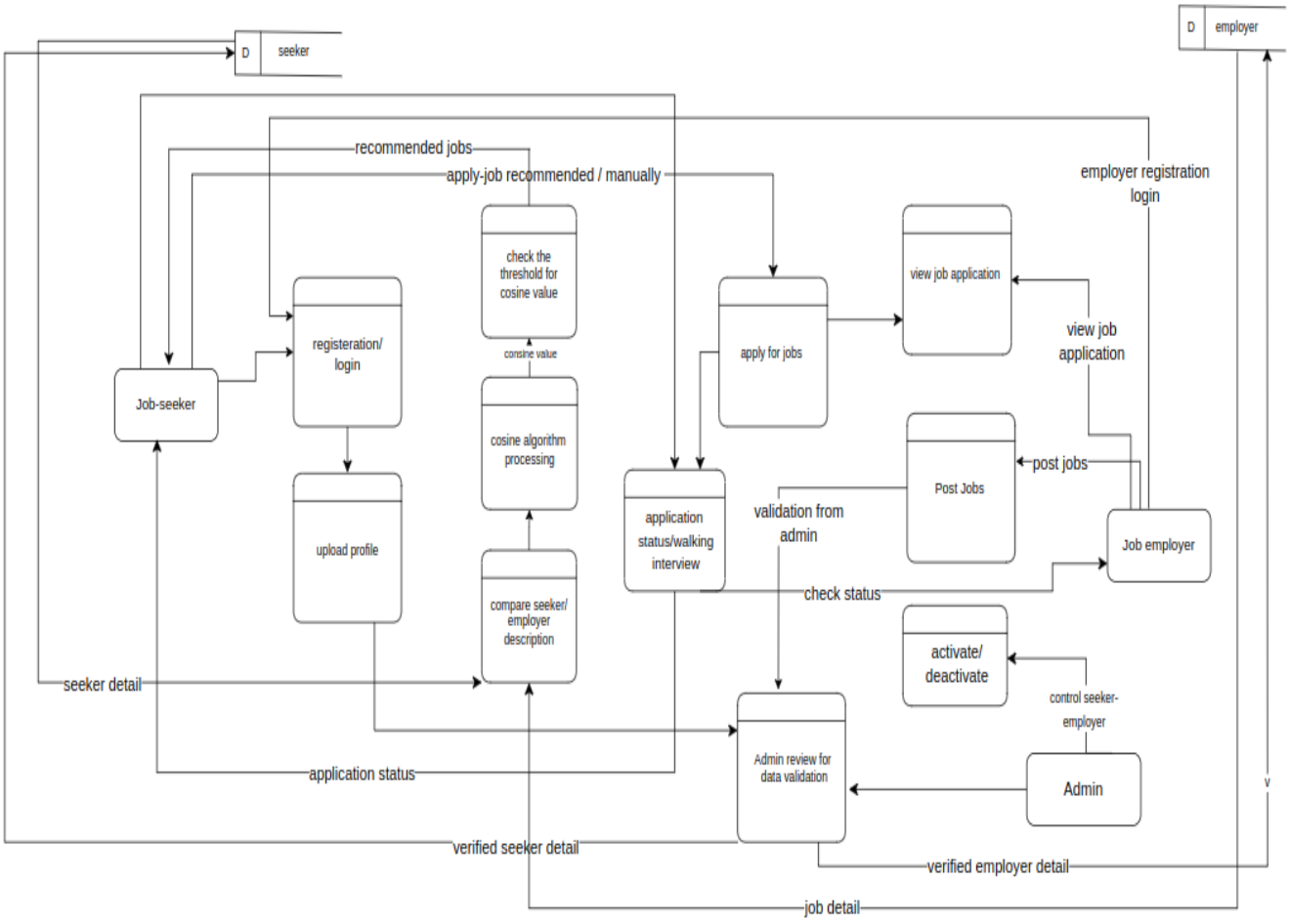


Figure 7: 2-level DFD

In this diagram, we have the following components:

1. External Entities

Job Seeker

These are individuals seeking job recommendations. They interact with the system by providing their preferences and receiving job recommendations.

Job Employer

These are individuals providing job details. They interact with the system by providing their job details and receiving job applications.

2. Processes

Retrieve Seeker Preferences

This sub-process retrieves the seekers preferences, including their skills, experience, and job preferences.

Fetch Job Data

This sub-process fetches relevant job data from the job database.

Calculate Cosine Similarity

This sub-process applies cosine similarity algorithm to measure the similarity between user preferences and job data.

Generate Job Recommendations

This sub-process generates the final job recommendations based on the cosine similarity scores.

3. Data Flows

Seeker Preferences

This data flow represents the input from job seekers, including their skills, qualifications.

Job Data

This data flow represents the input from employer which is presented to the seeker when searched manually or gets recommended by the system.

Cosine Similarity Scores

This data flow represents the scores calculated through the cosine similarity algorithm.

Admin

The data flow is controlled by the admin by validating the seeker's and employer's data.

Recommended Jobs

This data flow represents the data present to the seeker dashboard after similar jobs are found and are recommended to the seeker.

Job Applications

This data flow represents the data that is flowed to employer when seeker applies for job.

CHAPTER 4

SYSTEM DESIGN

4.1 Design

4.1.1 Database Design

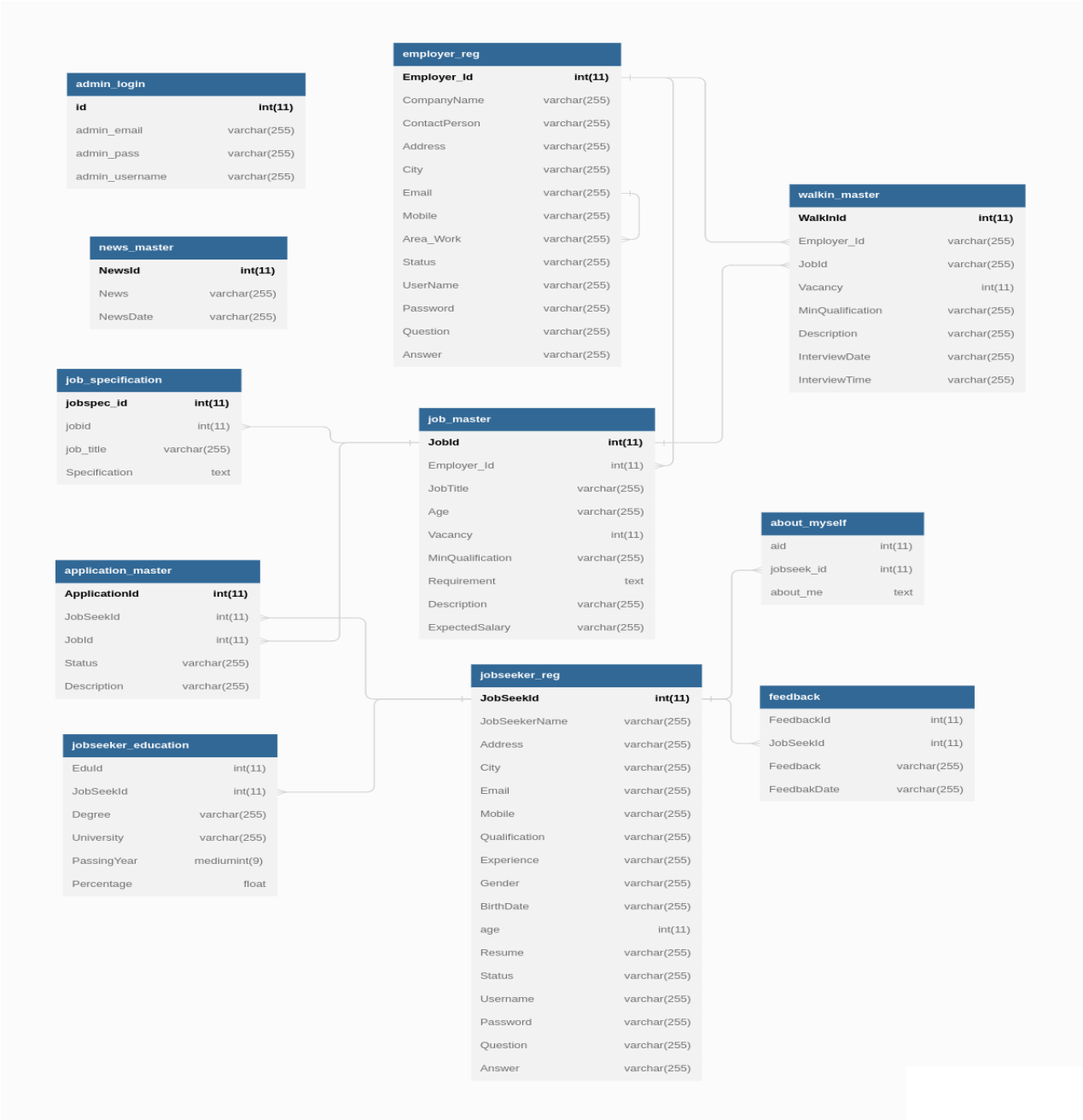


Figure 8: Database Design

Database design is the process of structuring and organizing data in a database system to meet specific requirements efficiently. It involves defining the tables, fields, relationships, and constraints that govern how data is stored, retrieved, and manipulated. The goal of database design is to create a well-organized and optimized database structure that ensures data integrity, minimizes redundancy, and enables efficient data access and manipulation operations**.**

1. Table Name: Employer

The "Employer" table stores information about the employers or companies posting job openings.

Table Description:

The "Employer" table serves as a central repository for employer data, allowing the system to manage and track employer details for job recommendations.

Table Columns:

- employer_id: (Primary Key, Integer): The unique identifier for each employer.
- username (String): The username chosen by the employer for login.
- password (String): The password associated with the employer's account.
- company_name (String): The official name of the employer's company.
- email (String): The email address of the employer for communication.
- contact_person (String): The name of the primary contact person for the employer.
- mobile (String): The contact number or mobile phone number of the employer.
- city (String): The city where the employer is located.
- address (String): The physical address or location of the employer's office.
- status (String): The status of the employer's account (e.g., active, suspended).

Column Descriptions:

- The employer_id uniquely identifies each employer and serves as the primary key for the table.
- The username and password columns allow employers to securely log in to their accounts.

- The company_name column stores the official name of the employer's company.
- The email column captures the employer's email address for communication purposes.
- The contact_person column holds the name of the primary contact person for the employer.
- The mobile column stores the contact number or mobile phone number of the employer.
- The city column represents the city where the employer is located.
- The address column captures the physical address or location of the employer's office.
- The status column indicates the current status of the employer's account.

Relationships:
The "Employer" table may have relationships with other tables, such as "Job Master" (via employer_id) and "Application Master" (via employer_id).

Data Examples:
- employer_id: 1
- username: abc_company
- password: *********
- company_name: ABC Company
- email: abc@example.com
- contact_person: Ram Sharma
- mobile: +977-8976567
- city: Lalitpur
- address: 123 kusunti
- status: Active

2. Table Name: Job Seeker
The "Job Seeker" table stores information about individuals who register as job seekers in the system.

Table Description:
The "Job Seeker" table serves as a repository for job seeker data, enabling the system to manage and track their registration details and qualifications.

Table Columns:

- job_seeker_id (Primary Key, Integer): The unique identifier for each job seeker.
- job_seeker_name (String): The name of the job seeker.
- address (String): The address of the job seeker.
- city (String): The city where the job seeker resides.
- email (String): The email address of the job seeker.
- mobile (String): The mobile number of the job seeker.
- qualification (String): The highest qualification attained by the job seeker.
- experience (Integer): The total years of experience of the job seeker.
- gender (String): The gender of the job seeker.
- birthdate (Date): The birthdate of the job seeker.
- age (Integer): The age of the job seeker.
- resume (Blob): The resume or CV of the job seeker.
- status (String): The status of the job seeker's registration (e.g., active, inactive).
- username (String): The username for the job seeker's account.
- password (String): The password for the job seeker's account.
- question (String): The security question for the job seeker's account.
- answer (String): The answer to the security question for the job seeker's account.

Column Descriptions:

- The job_seeker_id uniquely identifies each job seeker and serves as the primary key for the table.
- The job_seeker_name column stores the name of the job seeker.
- The address column holds the address of the job seeker.
- The city column represents the city where the job seeker resides.
- The email column stores the email address of the job seeker.
- The mobile column stores the mobile number of the job seeker.
- The qualification column represents the highest qualification attained by the job seeker.
- The experience column stores the total years of experience of the job seeker.
- The gender column indicates the gender of the job seeker.

- The birthdate column stores the birthdate of the job seeker.
- The age column represents the age of the job seeker.
- The resume column stores the resume or CV of the job seeker as a binary large object (BLOB).
- The status column indicates the registration status of the job seeker (e.g., active, inactive).
- The username column stores the username for the job seeker's account.
- The password column stores the password for the job seeker's account.
- The question column stores the security question for the job seeker's account.
- The answer column stores the answer to the security question for the job seeker's account.

Relationships:
The "Job Seeker" table may have relationships with other tables, such as "Application Master" (via job_seeker_id) , "Job Seeker Education" (via job_seeker_id) , “About Myself” (via job_seeker_id) and “Feedback”(via job_seeker_id).

Data Examples:
- job_seeker_id: 1
- username: johndoe92
- password: *********
- email: ramsharma@gmail.com
- full_name: John Doe
- mobile: +1 123-456-7890
- city: Los Angeles
- address: 456 Oak Street, Apt 202
- qualification: Bachelor's Degree
- skills: Java, Python, Project Management
- experience: 5
- status: Active

3. Table Name: Job Master

The "Job Master" table stores information about individual job openings posted by employers.

Table Description:

The "Job Master" table serves as a repository for job details, enabling the system to manage and track specific job openings and their requirements.

Table Columns:

- job_id (Primary Key, Integer): The unique identifier for each job.
- employer_id (Foreign Key, Integer): The identifier of the employer who posted the job.
- job_title (String): The title or name of the job.
- age (Integer): The age requirement or age limit for the job.
- vacancy (Integer): The number of vacancies available for the job.
- minimum_qualification (String): The minimum qualification required for the job.
- requirement (String): Additional requirements or qualifications for the job.
- description (String): A description or summary of the job responsibilities and duties.
- expected_salary (Integer): The expected salary or salary range for the job.

Column Descriptions:

- The job_id uniquely identifies each job and serves as the primary key for the table.
- The employer_id column stores the identifier of the employer who posted the job, establishing a relationship with the "Employer" table.
- The job_title column holds the title or name of the job.
- The age column represents the age requirement or age limit for the job.
- The vacancy column stores the number of vacancies available for the job.
- The minimum_qualification column specifies the minimum qualification required for the job.
- The requirement column captures any additional requirements or qualifications for the job.
- The description column provides a description or summary of the job responsibilities and duties.
- The expected_salary column stores the expected salary or salary range for the job.

Relationships:
The "Job Master" table has a relationship with the "Employer" table via the employer_id column.

Data Examples:
- job_id: 1
- employer_id: 123
- job_title: Software Engineer
- age: 25-40
- vacancy: 3
- minimum_qualification: Bachelor's Degree in Computer Science
- requirement: Proficiency in Java and SQL, 3+ years of experience
- description: Develop and maintain software applications, collaborate with cross-functional teams, participate in code reviews.
- expected_salary: $80,000 - $100,000 per year

4. Table Name: Job Specification
The "Job Specification" table stores information about specific requirements or specifications for each job.

Table Description:
The "Job Specification" table serves as a repository for job specifications, enabling the system to manage and track the specific requirements of different job titles.

Table Columns:
- job_id (Primary Key, Integer): The unique identifier for each job.
- job_title (String): The title or name of the job.
- specification (String): The detailed specification or requirements for the job.

Column Descriptions:
- The job_id uniquely identifies each job and serves as the primary key for the table.
- The job_title column holds the title or name of the job.
- The specification column stores the detailed specification or requirements for the job.

Relationships:

The "Job Specification" table may have a relationship with the "Job Master" table via the job_id column.

Data Examples:

- job_id: 1
- job_title: Software Engineer
- specification: Proficiency in Java, experience with Spring Framework, strong problem-solving skills, knowledge of database systems (SQL, NoSQL), familiarity with agile development methodologies.

5. Table Name: Job Seeker Education

The "Job Seeker Education" table stores information about the educational background of job seekers.

Table Description:

The "Job Seeker Education" table serves as a repository for job seekers' educational details, enabling the system to manage and track their qualifications and academic history.

Table Columns:

- edu_id (Primary Key, Integer): The unique identifier for each education record.
- job_seeker_id (Foreign Key, Integer): The identifier of the job seeker associated with the education record.
- degree (String): The degree or qualification attained by the job seeker.
- university (String): The university or institution from which the job seeker obtained their degree.
- passing_year (Integer): The year in which the job seeker passed or completed their degree.
- percentage (Decimal): The percentage or grade achieved by the job seeker in their degree.

Column Descriptions:

- The edu_id uniquely identifies each education record and serves as the primary key for the table.
- The job_seeker_id column stores the identifier of the job seeker associated with the education record, establishing a relationship with the "Job Seeker" table.
- The degree column holds the degree or qualification attained by the job seeker.
- The university column represents the university or institution from which the job seeker obtained their degree.
- The passing_year column stores the year in which the job seeker passed or completed their degree.
- The percentage column captures the percentage or grade achieved by the job seeker in their degree.

Relationships:

The "Job Seeker Education" table has a relationship with the "Job Seeker" table via the job_seeker_id column.

Data Examples:

- edu_id: 1
- job_seeker_id: 123
- degree: Bachelor of Science
- university: XYZ University
- passing_year: 2020
- percentage: 85.5

6. Table Name: Feedback

The "Feedback" table stores information about feedback provided by job seekers.

Table Description:

The "Feedback" table serves as a repository for job seekers' feedback, enabling the system to capture and track their comments or suggestions.

Table Columns:

- feedback_id (Primary Key, Integer): The unique identifier for each feedback entry.

- job_seeker_id (Foreign Key, Integer): The identifier of the job seeker who provided the feedback.
- feedback (String): The actual feedback or comment provided by the job seeker.
- feedback_date (Date): The date when the feedback was given.

Column Descriptions:
- The feedback_id uniquely identifies each feedback entry and serves as the primary key for the table.
- The job_seeker_id column stores the identifier of the job seeker who provided the feedback, establishing a relationship with the "Job Seeker" table.
- The feedback column holds the actual feedback or comment provided by the job seeker.
- The feedback_date column stores the date when the feedback was given.

Relationships:
The "Feedback" table has a relationship with the "Job Seeker" table via the job_seeker_id column.

Data Examples:
- feedback_id: 1
- job_seeker_id: 123
- feedback: "I had a positive experience using the job recommendation system. The suggested jobs were highly relevant, and the interface was user-friendly".
- feedback_date: 2023-05-10

4.1.2 Forms and Report Design

Form and report design in system design refer to the process of creating visually appealing and user-friendly interfaces for data input (forms) and data presentation (reports) in a software system.

Form Design

Form design involves creating the layout and structure of user input forms that allow users to enter data into the system. The design of forms aims to ensure a seamless and intuitive user experience. Key considerations in form design include:

- Field Placement
- Validation and Error Handling
- User-Friendly Elements
- Accessibility

Report Design:

Report design focuses on creating structured and informative presentations of data from the system. Reports provide a way to summarize, analyze, and present data in a meaningful format. Important aspects of report design include:

- Layout and Formatting
- Data Presentation
- Filtering and Sorting
- Summary and Aggregation
- Export and Printing

Both form and report design play a crucial role in enhancing the usability and effectiveness of a system by enabling efficient data entry and meaningful data presentation. Well-designed forms and reports contribute to a positive user experience and improve the overall functionality and usefulness of the system.

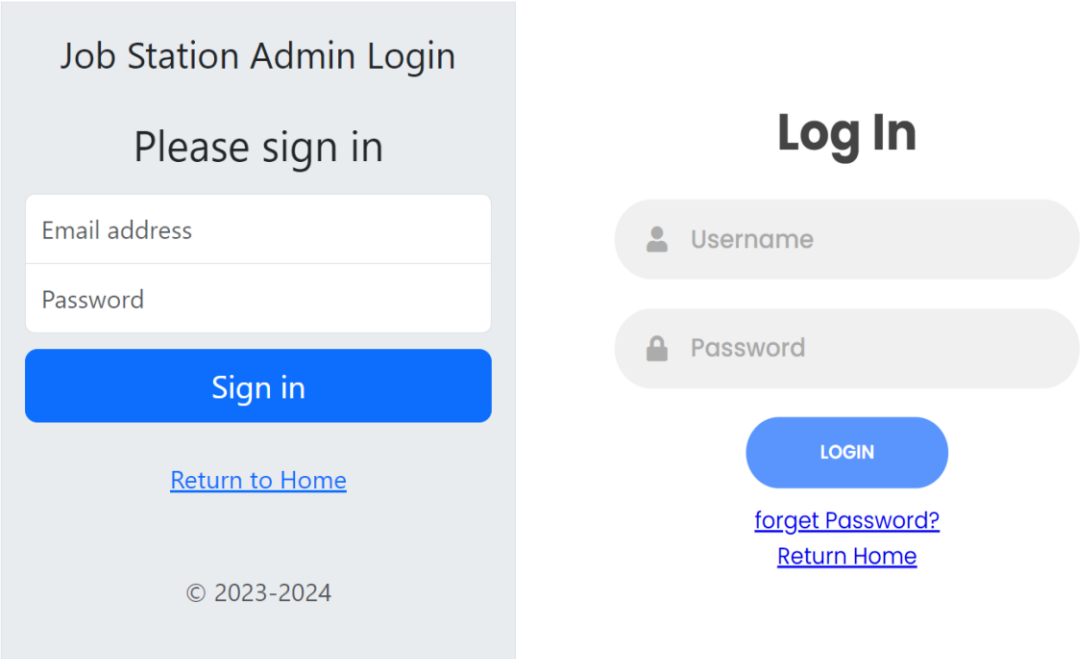


Figure 9: Form design by using prototyping tool

4.1.3 Interface and Dialogue Design

Interface and dialogue design in system design focuses on creating intuitive, user-friendly, and efficient interfaces for users to interact with a software system. It involves designing the visual elements, navigation, and interaction patterns that enable users to effectively communicate with the system.

Interface Design

Interface design encompasses the visual and graphical aspects of the system's user interface (UI). Key considerations in interface design include:

- Layout and Structure
- Visual Elements
- Navigation
- Responsive Design

Dialogue Design

Dialogue design focuses on defining the interactions and communication between the user and the system. It involves designing the flow of information, input mechanisms, and feedback mechanisms. Key considerations in dialogue design include:

- Input and Data Entry
- Feedback and Validation
- Error Handling
- System Response and Progress
- Help and Documentation

Interface and dialogue design aim to create a user interface that is visually appealing, easy to navigate, and supports effective communication between users and the system. By designing intuitive interfaces and clear dialogues, system designers enhance user satisfaction, reduce errors, and improve the overall usability and user experience of the system.

4.2 Algorithm Details

4.2.1 Cosine Similarity

This concept is used in this project to calculate the score between two document vectors. The similarity between two vectors is defined by the angle between them. If the angle between two vectors is zero and thus cosine=1, representing the perfect match. If those two vectors are perfectly dissimilar, then the angle between the vectors is perfect 90 degree and cosine=0, representing the perfect dissimilar. The inputs required for cosine similarity calculation must be represented in the form of a Vector Space Model. A Vector Space Model is the mathematical structure formed by a collection of vectors.

In this recommendation system we have implemented the cosine similarity between job seeker's bio data and job specifications that is related to different jobs. For calculating cosine similarity, we have to convert these two texts (bio, specification) into document vectors. In order to obtain document vectors, we use the concept of term frequency.

4.2.1.1 Term Frequency - Inverse Document Frequency (TF-IDF)

1. Term Frequency (TF)

Term frequency is a measure of how often a term is found in the collection of documents. A reasonable scoring mechanism is computed with document terms. It counts the frequency of the terms that match between the query terms and the document term list which is denoted by tf(t, d).

2. Inverse Document Frequency (IDF)

The IDF (Inverse Document Frequency) component of TF-IDF reflects the proportion of documents in a corpus that contain a specific term. Terms unique to a small percentage of documents, such as "javascript," "frontend," and "css," receive higher IDF values, indicating their importance in distinguishing and characterizing the documents in which they appear. Conversely, common words like "a," "the," and "and" have lower IDF values as they offer less discriminatory power. By incorporating IDF into TF-IDF calculations, the relevance of terms is adjusted based on their occurrence across the corpus, resulting in more nuanced and informative representations of term importance within documents and the overall corpus.

$$\text{IDF} = \log\left(\frac{number\ of\ document\ in\ the\ corpus}{number\ of\ documents\ in\ the\ corpus\ contain\ the\ term}\right)$$

2. Inverse Document Frequency (IDF)

The TF-IDF of a term is calculated by multiplying TF and IDF scores.

Term Frequency - Inverse Document Frequency (TF-IDF) = TF * IDF

CHAPTER 5
IMPLEMENTATION AND TESTING

5.1 Implementation

After completing the system design phase, where we have prepared all the necessary system designs, and studied all the necessary algorithms for the project, we implemented the blueprint and design sketches prepared from previous stages into a functional system code. The implementation details are described below:

5.1.1 Tools Used

Various tools used during the implementation of the project are further described below:

5.1.1.1 CASE tools

The various Computer Aided Software Engineering (CASE) Tools used for the implementation of the project are listed below:

1: Diagram Tools

For diagramming, various system designing tool during the project lifecycle, like draw.io and lucid chart were used.

2. Documentation Tools

For writing the documentation/report of the project, Microsoft word was used.

3. Design tools:

For designing various user interfaces of the web application for the project, the online web application-based Software, Figma was used.

4. Configuration Management Tools:

To track the changes occurred in the web application and maintaining the code version during the project lifecycle, Git was used. XAMPP for the server creation and database hosting.

5.1.1.2 Programming Languages:

For writing the code, Visual Studio Code Integrated Development Environment was used and for the web application to run and see the output, web browsers like Google Chrome were used. We have used HTML, CSS, JavaScript in Frontend, PHP as backend, MySQL as database

And XAMPP as a server for serving files and database.

5.1.2 Description of Implementation Modules of the System

1. Data collection module: Collects job seeker and job opening data from various sources such as user profiles, company profiles. Store the data in a database.

2. User profile module: Create user profiles based on job seekers' skills, experience, and job preferences. This can be done by calculating the cosine similarity between the job seeker's profile and the job opening profiles. The user profile can be represented as a vector of weighted attributes.

3.Job opening profile module: Create job opening profiles based on the requirements, responsibilities, and qualifications of the job openings. This can also be done by calculating the cosine similarity between the job opening's profile and the job seeker profiles. The job opening profile can also be represented as a vector of weighted attributes.

4. Recommendation engine module: Generate job recommendations for job seekers based on their user profiles and job opening profiles using cosine similarity. The recommendation engine can compare the cosine similarity values between the job seeker's profile and all job opening profiles in the database to generate a list of recommended job openings.

5. Evaluation module: Evaluate the performance of the recommendation engine by measuring the precision, recall, and accuracy of the recommendations generated using cosine similarity.

6. User interface module: Provide an interactive user interface for job seekers to view and apply for recommended job openings. It is done using a web development tools such as HTML, CSS, JS.

7. Database module: Store and manage the data collected, preprocessed, and generated by the system using a database management system such as MySQL or PostgreSQL.

Overall, the implementation module for our system involves collecting and preprocessing data, creating user and job opening profiles, generating recommendations using cosine similarity, evaluating the recommendation engine, providing a user interface for job seekers, and storing data in a database.

8. Calculation Process

In this recommendation system we have implemented the cosine similarity between job seeker's bio data and job specifications that is related to different jobs. For calculating cosine similarity, we have two part of text one is bio data of a job seeker and another is job specification or we can say requirement required by employer. Here is the actual calculation process of algorithm in our system.

Here we consider two inputs,
text1 = [bio data of a job seeker]
text2 = [job specification or requirement by employer]
i.e
$text1 = "Computer Engineer BscCSIT HTML CSS JS";
$text2 = "Front End Developer HTML CSS JS BscCSIT";

Step 1:

Using the explode function for conversion of text into Array:

$array_text1=['Computer', 'Engineer','BscCSIT','HTML','CSS','JS',];
$array_text2=['Front', 'End','Developer','HTML','CSS','JS','BscCSIT'];

Step 2: Concatenating $array_text1 and $array_text2

$array_text3 = $array_text1 . $array_text2;

Step 3: Removing dublicate value, proportion and other common words and gives 0 to every possible tags.

$array_text3 = array_unique(array_text3);

$array_text3 = array_fill_keys($array_text3, 0);

$array_text3=[‘Computer=>0’, ‘Engineer=>0’, ’BscCSIT=>0’, ’HTML=>0’, ’CSS =>0’, ’JS =>0’, ‘Front=>0’, ‘End=>0’, ‘Developer=>0’];

Step 4: Assigning key values for $array_text1 and $array_text2 using function

a$ = array_fill_keys($array_text1, 1) with key value 1.

$a = Array(

[computer]=>1

[Engineer]=>1

[BscCSIT]=>1

[HTML]=>1

[CSS]=>1

[JS]=>1)

b$ = array_fill_keys($array_text2, 1) with key value 1.

$a = Array(

[Front]=>1

[End]=>1

[Developer]=>1

[HTML]=>1

[CSS]=>1

[JS]=>1

[BscCSIT=>1

)

Step 5: Now, it again assigns the key value for $array_text1 and $array_text2 which is 1 and it is added with $a.

$array_text1=array_fill_keys($a,1) + $a;

$arrary_text2=arrary_fill_keys($b,1) + $a;

Output for $array_text1:

array(

[computer]=>1

[Engineer]=>1

[BscCSIT]=>1

[HTML]=> 1

[CSS] => 1

[JS] =>1

)

Output for $arrary_text2:

array(

[Computer]=> 0

[Engineer] =>0

[BscCSIT]=> 1

[HTML] =>1

[CSS]=>1

[JS]=>1

[front]=>1

[End]=>1

[Developer]=>1

)

Thus, we compute cosine similarity using output of array text 1 and array text 2 using following

formula:

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|}$$

If similarity is greater than 0.5 i.e 50%, system suggests corresponding job vacancy to the jobseeker.

9. Working of recommendation algorithm:

Calculating vectors of attribute of dimension of job details.

Vector value can be 0/1 depending upon whether the attributes are selected or not.

Table 2: Vector representation of the attribute

Attribute List	bio data of a job seeker	bio data of a job seeker
Frontend	1	1
Bootstrap	1	0
Scss	0	1
JavaScript	0	0
React	1	0
NextJs	0	1

1 represents true value whereas 0 represents false. Representing the attribute values in vector form where A = [1, 1, 0, 0, 1, 0] and B = [1, 0, 1, 0, 0, 1].

Calculation of cosine similarity of vector A and B.

A.B = (a1.b1) + (a2.b2) + (a3.b3) + (a4.b4) + (a5.b5) + (a6.b6)

=1+0+0+0+0+0

= 1

$$||A|| = \sqrt{1^2 + 1^2 + 0^2 + 0^2 + 1^2 + 0^2}$$
$$= \sqrt{1 + 1 + 1}$$
$$= \sqrt{3}$$

$$||B|| = \sqrt{1^2 + 0^2 + 1^2 + 0^2 + 0^2 + 1^2}$$
$$= \sqrt{1 + 1 + 1}$$
$$= \sqrt{3}$$

$$||A|| * ||B|| = \sqrt{3} * \sqrt{3}$$

Similarity of job specification or requirement by employer with bio data of a job seeker

$$= (A.B) / (||A|| * ||B||)$$
$$= 2/3$$
$$= 0.666$$

Generally, we have a threshold value likely 0.50 which filter out the result sets and returns the significant sets of job post. Values less than or equal to 0.50 are cut off.

5.2 Testing

Testing is a process of executing a program with the intent of finding bugs that makes the application fail to meet the expected behavior. Regardless of the development methodology, the ultimate goal of testing is to make sure that what is created does what it is supposed to do. Testing plays a critical role for assuring quality and reliability of the software. We have included testing as a part of the development process. The test cases should be designed with maximum possibilities of finding the errors or bugs. Various levels of testing are as follows.

5.2.1 Test Cases for Unit Testing

We have designed and executed a few test cases to check if the application meets the functional requirements.

Below are the test cases for Online Job Recommendation System application.

Table 3: Unit Test Cases for application

TEST MODULE	TEST CASE	EXPECTED RESULT	TEST RESULT
ADMIN	Username:bimal pass: bimal	User successfully logged in and directed to the admin dashboard page.	PASS
ADMIN	Username:bimal Pass: sabin	Displays error message	PASS
ADMIN	Upon successful login, click on the 'Manage Employer' tab	Display the list of employers whose account registration is to be verified	PASS
ADMIN	Upon successful login, click on the 'Manage Job Seeker' tab	Display the list of seekers whose account registrations to be verified	PASS
ADMIN	Upon successful login, click on the 'Public No' tab	Can publish new news and edit and delete existing news	PASS
ADMIN	Upon successful login, click on the 'Feedback' tab	Can view the feedbacks send by employers and seekers	PASS
EMPLOYER	Provide details for registration Username:Kalyan	Employer registration successfully	PASS

	Pass:kalyan	submitted	
EMPLOYER	Upon successful login, click on 'Manage Jobs' tab	Employer posts jobs with the required details and view the details of all the posted jobs	PASS
EMPLOYER	Upon successful login, click on 'Walking Interview' tab	Employers post job vacancies that have to be attended physically and also can view the list of previously posted jobs	PASS
EMPLOYER	Upon successful login, click on 'Profile' tab	Can view as well as edit information about themselves	PASS
EMPLOYER	Upon successful login, click on 'APPLICATION' tab	Can view the list of applications send by applicants or PASS seeker for a particular job as well as can reply all later message to the applicant	PASS
JOBSEEKER	Provide details for registration Eg: username:sanam pass: sanam	Job Seeker registration successfully submitted	PASS
JOBSEEKER	Enters invalid login credentials Username:sarita pass:hellow113	Display error message	PASS
JOBSEEKER	Upon successful login, click on ' Profile' tab	List details of jobseeker	PASS
JOBSEEKER	Upon successful login, click on 'Search Jobs' tab	Can search for different job vacancies that are available and also view status of previously applied jobs	PASS
JOBSEEKER	Upon successful login, click on 'Walking Interview' tab	Can view the list of vacancies which should be attended physically	PASS
JOBSEEKER	Upon successful login, click on 'Education' tab	Can add their educational degrees	PASS
JOBSEEKER	Upon successful login, click on 'Feedback' tab	Can send feedbacks to administrator	PASS

Test Objectives: Successfully submits the job application

Table 4: Test Case for applying job

TEST CONDITION	INPUT SPECIFICATION	OUTPUT SPECIFICATION	TEST RESULT
The user is currently on the jobs page	User clicks on the "APPLY" tab against a job post	Prompts a message as "Successfully applied to this job"	PASS

Test Objectives: User Checks for applied jobs.

Table 5: Test Case for listing all jobs

TEST CONDITION	INPUT SPECIFICATION	OUTPUT SPECIFICATION	TEST RESULT
The job seeker is currently logged into their account	The job seeker clicks on search job tab	Lists all the jobs that are applied	PASS

5.2.2 Test Cases for System Testing

System testing is a type of testing that evaluates the complete software system as a whole to ensure that it meets its functional and non-functional requirements and specifications. Below are the test cases for System Testing for our Job Recommendation System application.

Test Objectives: Test the system's ability to handle a large volume of job seekers and job postings

Table 6: Test Case for handling a large volume of data

TEST CONDITION	INPUT SPECIFICATION	OUTPUT SPECIFICATION	TEST RESULT
To handle a large volume of job seekers and job postings	By feeding large number of job seekers and job posts. Number of job seekers should be registered and number of jobs should be posted.	System is able to handle those large number of data	PASS

Test Objective: Test the accuracy of the job recommendations

Table 7: Test Case for the accuracy of the algorithm

TEST CONDITION	INPUT SPECIFICATION	OUTPUT SPECIFICATION	TEST RESULT
To check the accuracy of the job recommendations system.	By feeding relevant data to the system at jobseeker and job posting end.	If similarity is greater than 0.5 i.e 50%, system suggests corresponding job vacancy to the jobseeker.	PASS

5.2 Results Analysis

During the testing phase of the job recommendation system, we conducted various types of testing to ensure that the system meets its functional and non-functional requirements. The following are the key findings and analysis of the test results:

Summary of Test Results

The system performed well during testing, and no major defects were found. However, we identified some minor issues related to the search functionality and user authentication, which were promptly resolved.

Performance Analysis

The system was tested for its response time, throughput, and scalability. The average response time for the system was 2.5 seconds, which meets the system's requirement of responding within 3 seconds. The throughput of the system was also satisfactory, and the system was able to handle a large number of job seekers and job postings without any performance degradation.

User Feedback

During the testing, we gathered feedback from the users regarding the usability and user interface of the system. Based on the feedback, we identified some areas for improvement, such as making the search functionality more intuitive and user-friendly.

Reliability, Availability, and Maintainability

The system was tested for its reliability, availability, and maintainability. The system was found to be reliable, and it was able to handle a large number of concurrent users without any downtime. The system was also found to be easily maintainable, and it was easy to add new features and make changes to the system.

Recommendations

Based on the test results and analysis, we recommend the following enhancements to the system:

Improve the search functionality to make it more intuitive and user-friendly

Add more security features to ensure the privacy and confidentiality of user data

Enhance the system's scalability to handle even larger volumes of job seekers and job postings

Overall, the results of the testing phase indicate that the job recommendation system is functioning as intended and meets its functional and non-functional requirements. However, there is still room for improvement, and the recommendations outlined above will help to enhance the system's usability, security, and scalability.

CHAPTER 6
CONCLUSION AND FUTURE RECOMMENDATIONS

6.1 Conclusion

The major goal of this project is to provide a web-based system where Job Seekers can find the relevant jobs based on their profile specifications and employers can manage and select the right candidate for their jobs by viewing their profile and resumes.

It is a web-based application developed using Html, CSS, JavaScript, Bootstrap for its front end, PHP for its backend and MySQL for databases.

People are unable to get access to job opportunities due to inefficient distribution of information on job offers but now the Internet has made a huge impact on knowledge management and information dissemination all over the world. It is expected to use less budget and less resources and can be easily access by every user because of its user-friendly environments and also the system is not so complex for users. Any person who has access to the Internet can access information anywhere in the world. This system aims and objective is to provide a platform where recruiters and job seekers can communicate.

The development and implementation of the system provides the following features:

- User authentication
- CV or resume upload
- Feedback mechanism
- Find relevant jobs

6.2 Future Recommendations

This project fulfills the primary requirements of the job seekers and employers. It can be extended in several ways. It can provide recommendations and email updates for new job postings based on the job seeker's search history. Since the job seekers might be interested in building a strong resume, it can provide tips and information for the same. It can also provide templates for building the Resumes which might interest most applicants. The mobile application is developed fulfilling the functionalities of job seekers, it can be extended to support functionalities of Employers as well.

In this system, Cosine Similarity is calculated using the term frequency of each term in job seeker biodata. But Term frequency suffers from critical problems as all the terms are considered equally important. In fact, Certain terms have little or no selective power in determining relevance. And also, terms which appear very few in number have higher probability in determining the relevance. So, to overcome this problem we have to scale down the term weight of the terms with high collection frequency. For this we can use TF-IDF weighting instead of term frequency which can improve the outcome of our system.

Reference

[1] Abhjit Roy, “Introduction to recommender system-1:Content-Based Filtering And Collaborative Filtering”.[Online]
Available:https://towardsdatascience.com/introduction-to-recommender-systems-1-971bd274f421. [Accessed 09 Feb 2023].

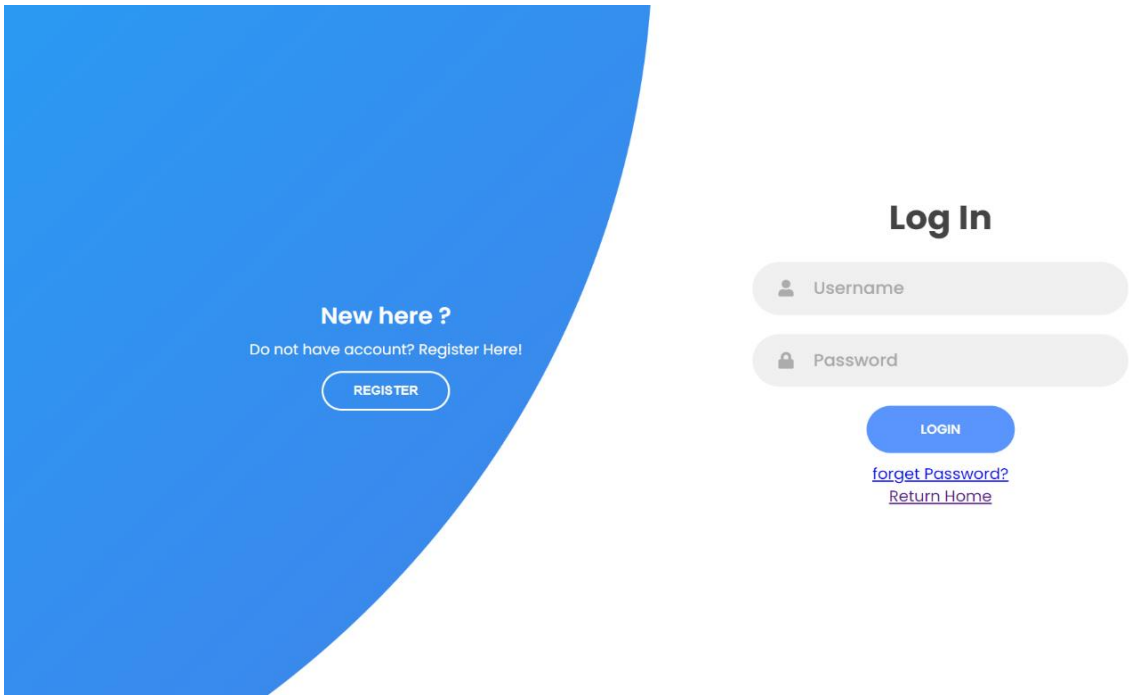
[2] K. Lakiotaki, ”UTA-Rec: A Recommender System based on Multiple Criteria Analysis,” RecSys’08, pp. 219-225, 2008.

[3] S. O. P. Mucheol Kim, “Group affinity based social trust model for an intelligent movie recommender system,” Multimedia Tools and Applications, vol. 64, no. 2, pp. 505-516, May, 2013.

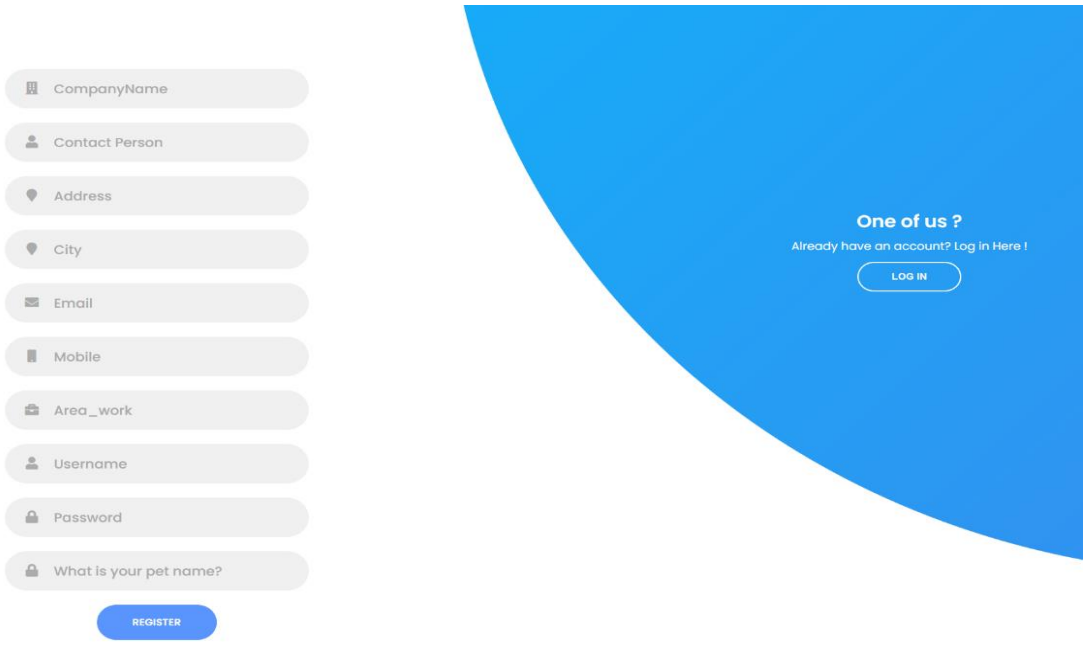
[4] R. S. Punam Bedi, “Trust based recommender system using ant colony for trust computation,” Expert Systems with Applications: An International Journal, vol. 39, no. 1, pp. 1183-1190, January, 2012.

APPENDICES

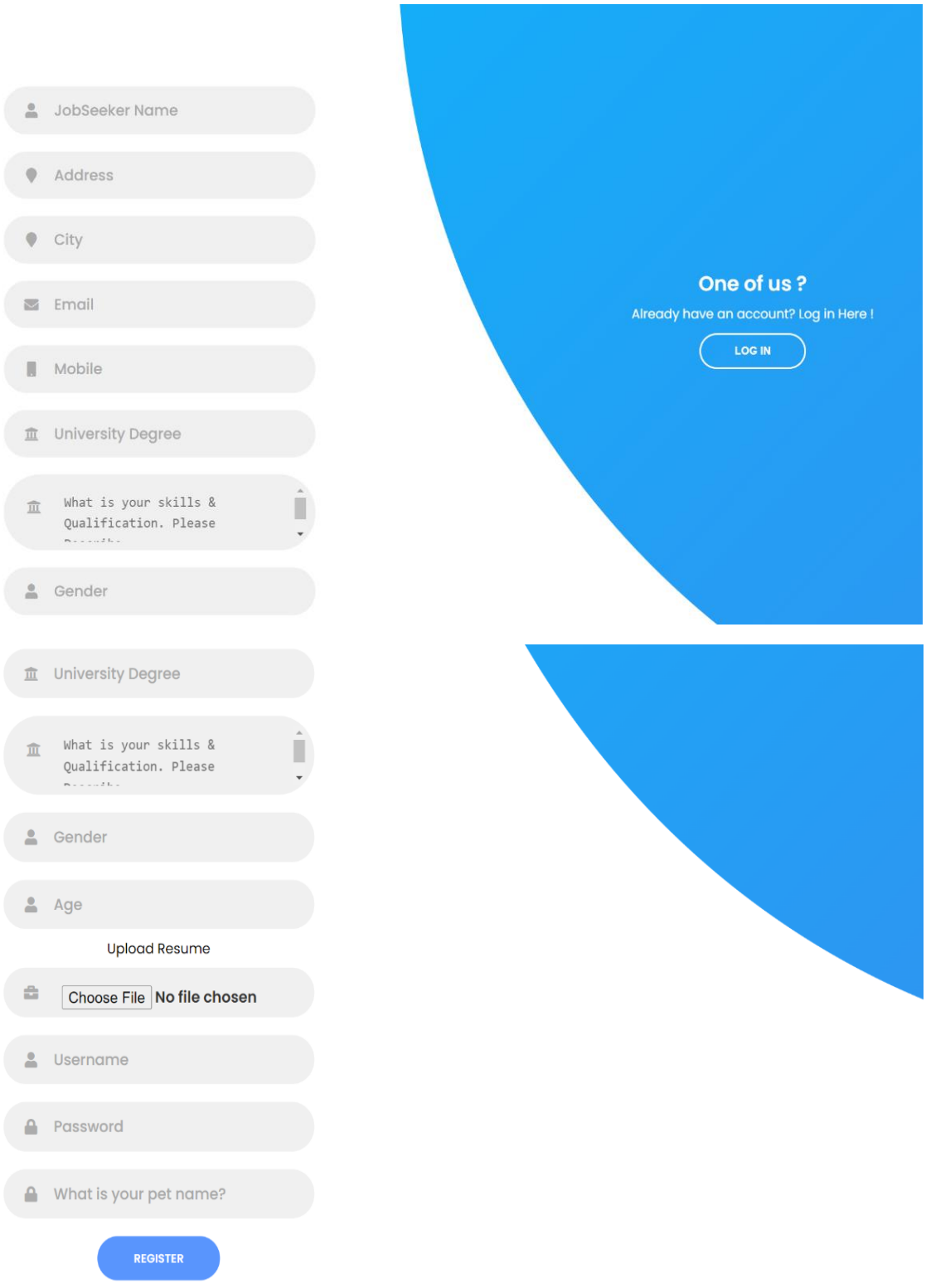
Screenshots



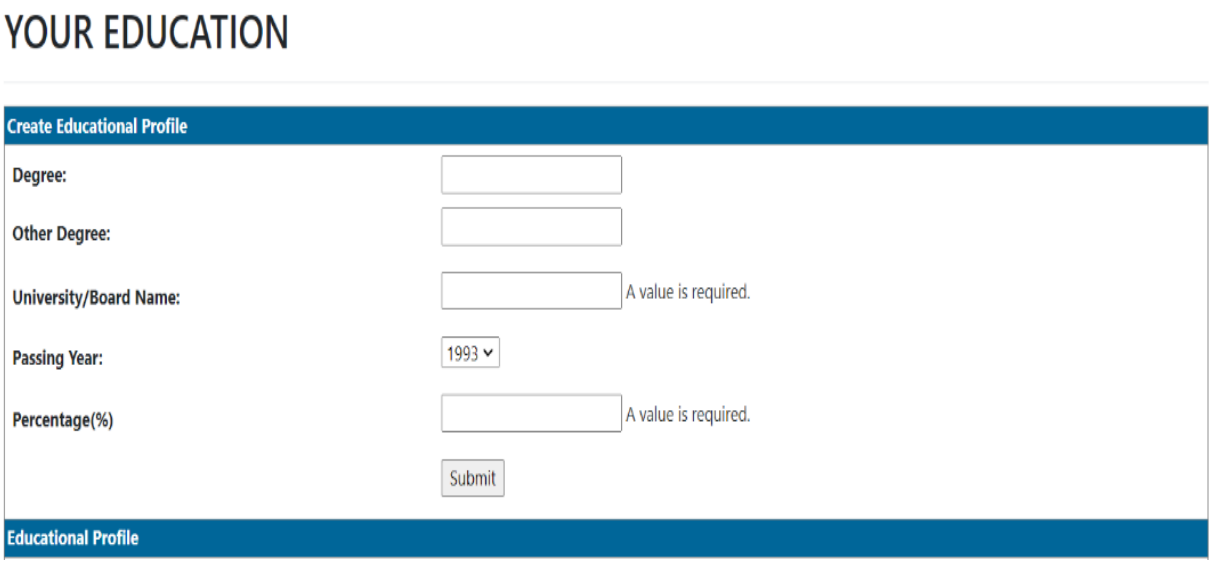
Screenshot 1: Employer Login Form



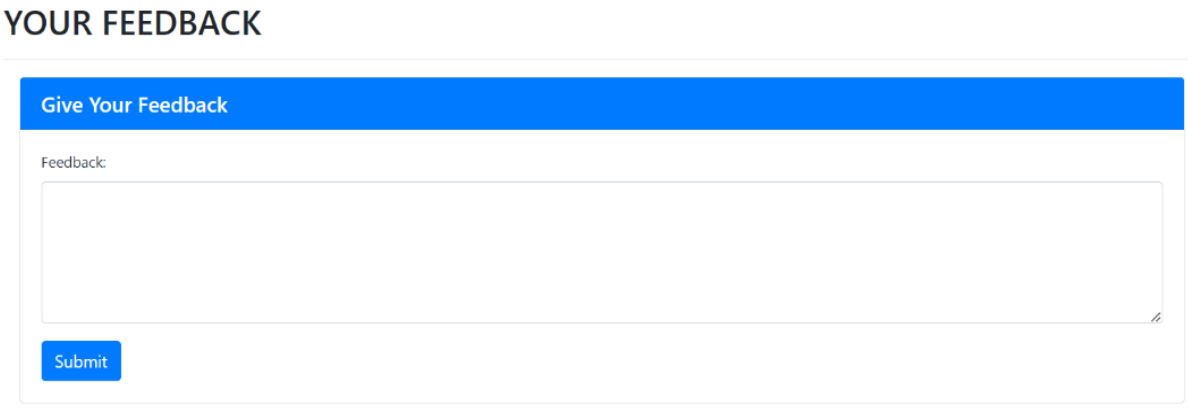
Screenshot 2: Employer Registration Form



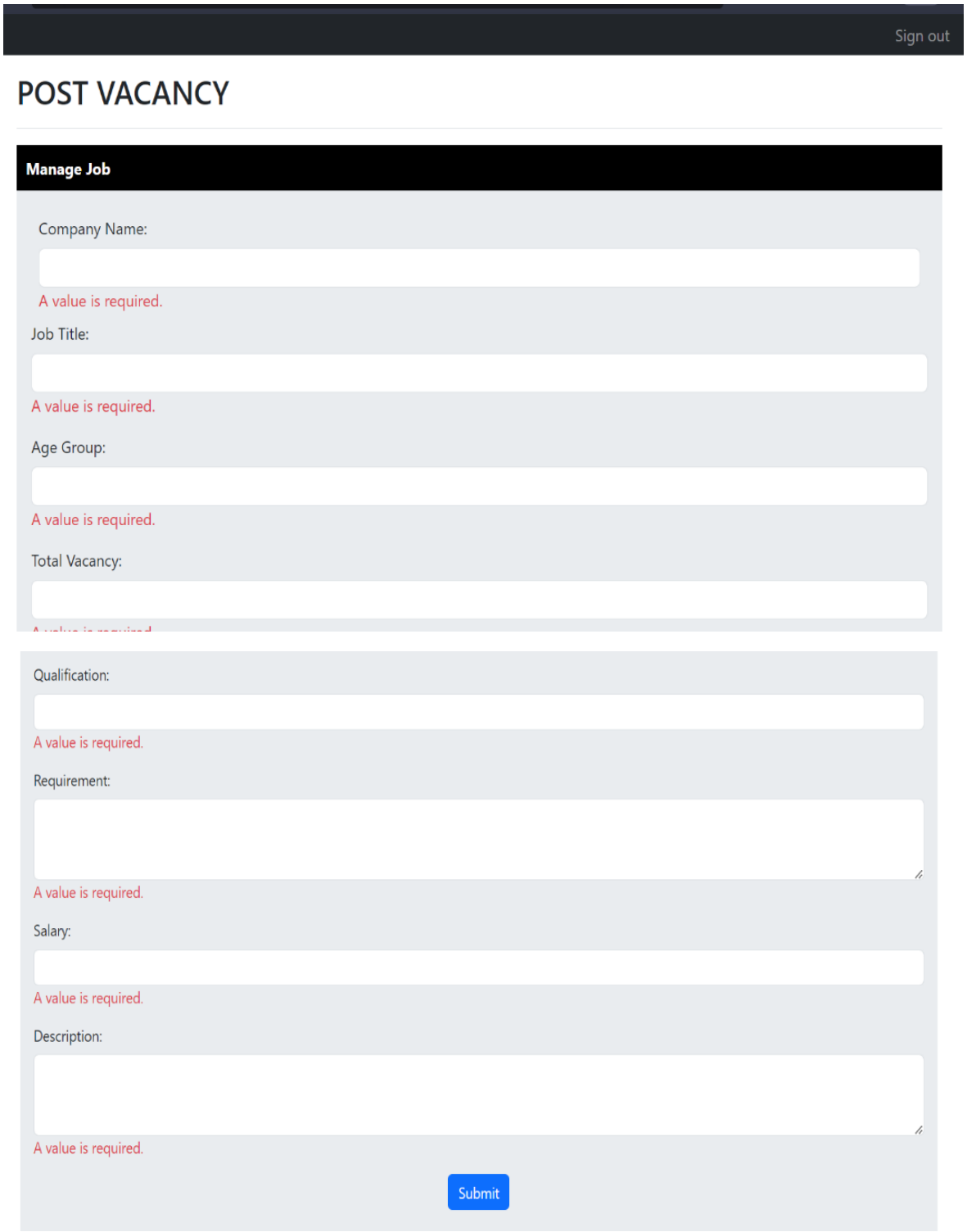
Screenshot 3: Job Seeker Registration Form



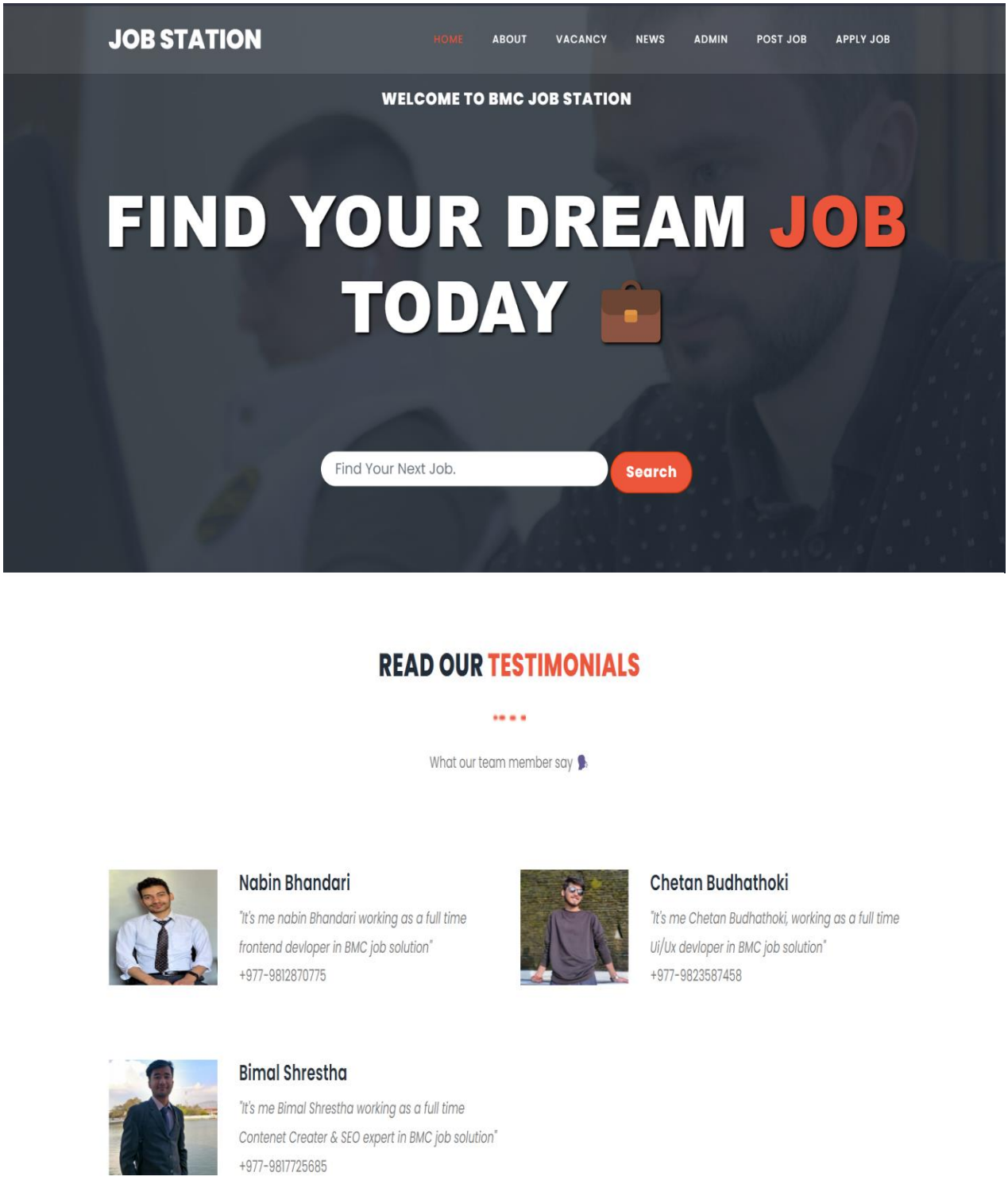
Screenshot 4: Job Seeker Education Form



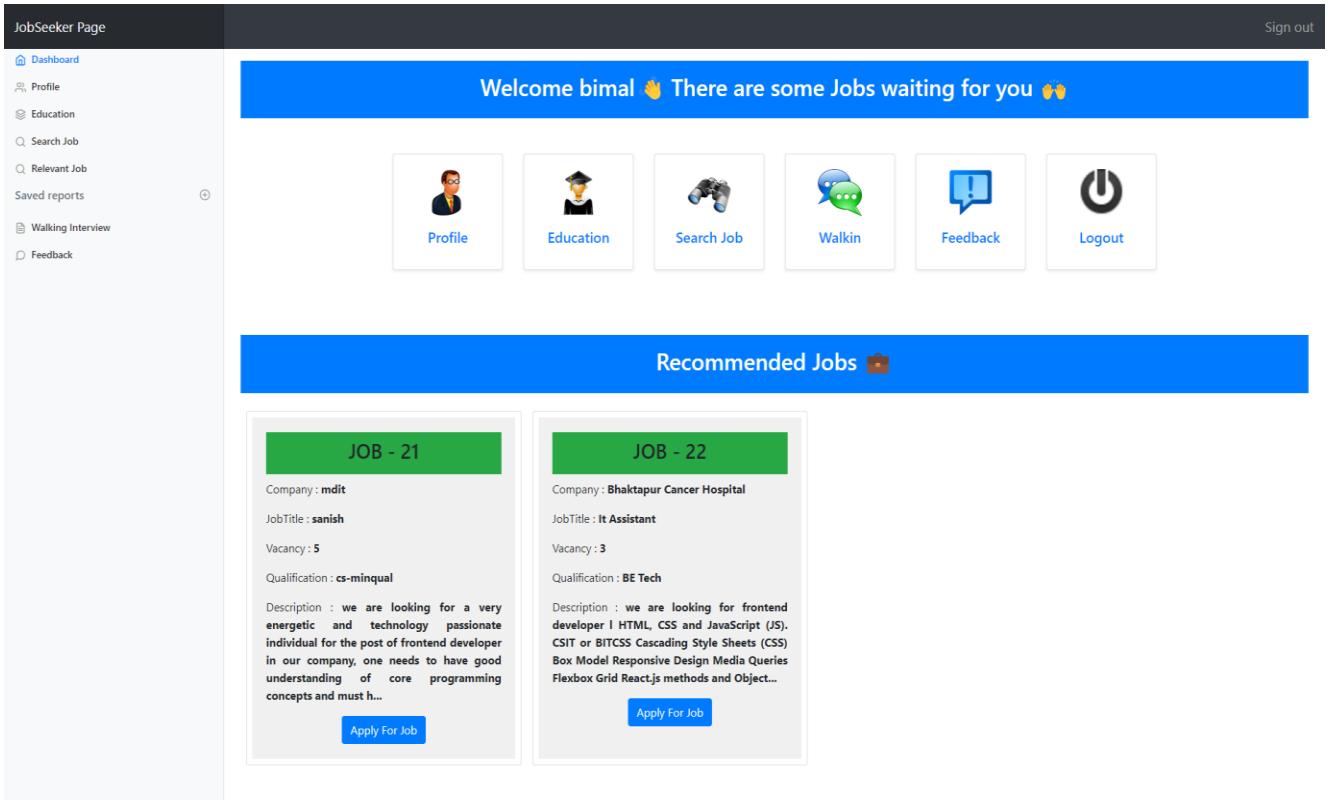
Screenshot 5: Job Seeker Feedback Form



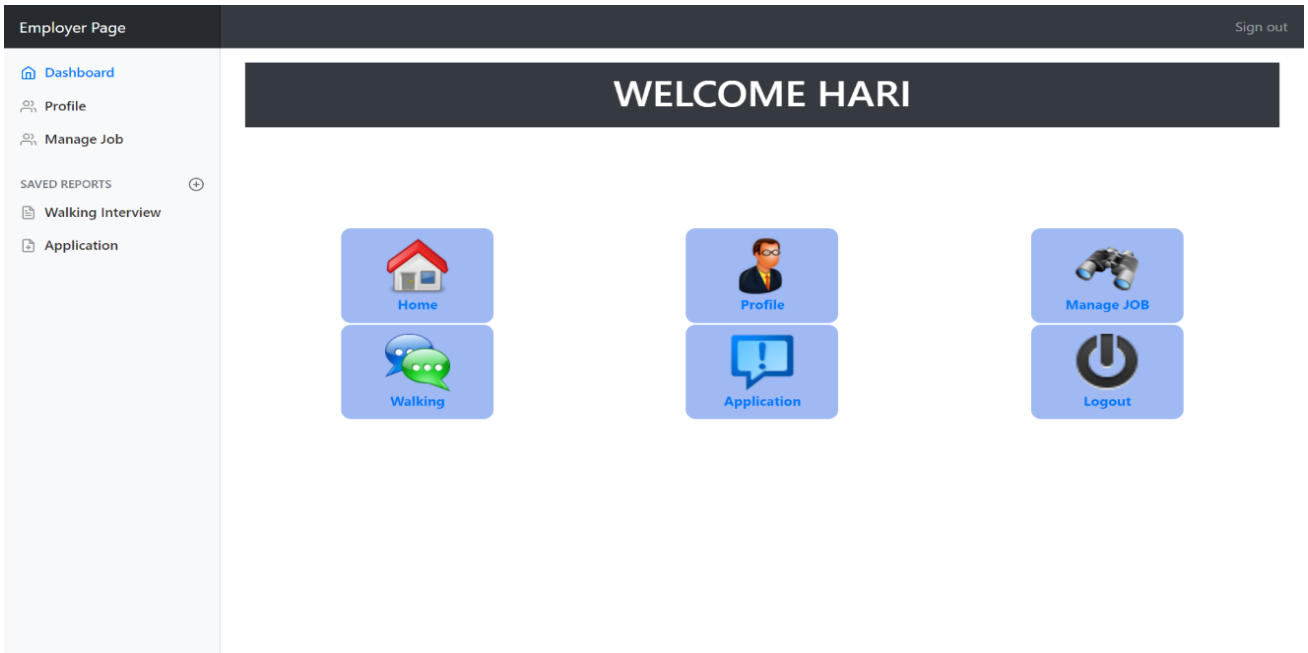
Screenshot 6: Job Employer new Job post Form



Screenshot 7: Home page



Screenshot 8: JobSeeker Hompage



Screenshot 9: Employer Hompage

Snippets of major source components

Similarity Class Snippet

```php
<?php
class Similarity{

    static public function dot($tags)
    {
        // print_r($tags);
        $tags = array_unique($tags);
        $tags = array_fill_keys($tags, 0);
        ksort($tags);
        return $tags;
    }

    protected function dot_product($a, $b){
        $products = array_map(function ($a, $b) {
            return $a * $b;
        }, $a, $b);
        return array_reduce($products, function ($a, $b) {
            return $a + $b;
        });
    }

    protected function magnitude($point){
        $squares = array_map(function ($x) {
            return pow($x, 2);
        }, $point);
        return sqrt(array_reduce($squares, function ($a, $b) {
            return $a + $b;
        }));
    }

    static public function cosine($a, $b, $base){
        $a = array_fill_keys($a, 1) + $base;
        $b = array_fill_keys($b, 1) + $base;

        ksort($a);
        ksort($b);


        $similarity = new Similarity();
        return $similarity->dot_product($a, $b) / ($similarity->magnitude($a) * $similarity-
                        >magnitude($b));
    }

    public static function checka($a, $base){
        $a = array_fill_keys($a, 1) + $base;
        ksort($a);
        return $a;
    }

    public static function checkb($b, $base){
        $b = array_fill_keys($b, 1) + $base;
        ksort($b);
        return $b;
    }
}
```

LOGS OF THE VISIT TO SUPERVISOR

S.N.	Date	Topic Discussed	Signature of Supervisor	Remarks
1	Dec 10, 2023	Team formation and Project Idea Discussion		
2	Dec 23,2022	Project Proposal Submission		
3	Jan 19,2023	Discussion of the project topic with the supervisor		
4	Feb 6, 2023	Discussion of the initial development of the project topic with the supervisor		
5	Mar 4, 2023	Discussion of the system development during the mid-phase and discussion of the core algorithms with the supervisor		
6	May 11, 2023	Predefence with the supervisor before the final defence.		