A

Project Report on

INTELLIGENT JOB MATCH

Submitted in partial fulfillment of the

Requirements for the award of the degree of

Bachelor of Technology

in

COMPUTER SCIENCE AND ENGINEERING

By

Kodumuri Likhitha

(20EG105225)

S. Nikhila Reddy

(20EG105245)

Dutta Saurabh

(20EG105212)

Under the guidance of

Dr. V. Rama Krishna

Assistant Professor

Department of Computer Science and Engineering

Venkatapur(V), Ghatkesar(M), Medchal(D), Telangana-500088

2023-24

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project report entitled **Intelligent Job Match** that is being submitted by Kodumuri Likhitha bearing the hall ticket number **20EG105225**, S. Nikhila Reddy bearing hall ticket number **20EG105245** and Dutta Saurabh bearing hall ticket number **20EG105212** in partial fulfillment of the requirements for the award of the degree of the **Bachelor of Technology** in **Computer Science and Engineering** in **Anurag University** is a record of bonafide work carried out by them under my guidance and supervision from academic year 2023 to 2024.

The results presented in this project have been verified and found to be satisfactory. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree.

Internal Guide
Dr. V. Rama Krishna
Assistant Professor, CSE

Dean, CSE
Dr. G. Vishnu Murthy

External Examiner

DECLARATION

We hereby declare that the Report entitled **Intelligent Job Match** submitted in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** from **Anurag University** is a record of our original work done by us under the guidance of **Dr. V. Rama Krishna, Assistant Professor, Department of CSE** and this project work has not been submitted to any University for the award of any other degree.

Kodumuri Likhitha
20EG105225

S. Nikhila Reddy
20EG105245

Dutta Saurabh
20EG105212

Place: Hyderabad

Date:

ACKNOWLEDGMENT

We would like to express our sincere thanks and deep sense of gratitude to project supervisor **Dr. V. Rama Krishna,** Asst. Professor, Dept. of Computer Science and Engineering, Anurag University for his constant encouragement and inspiring guidance without which this project could not have been completed. His critical reviews and constructive comments improved our grasp of the subject and steered us to the fruitful completion of the work. His patience, guidance, and encouragement made this project possible.

We would like to express our special thanks to **Dr. V. Vijaya Kumar**, Dean School of Engineering, Anurag University, for his encouragement and timely support of our B. Tech program.

We would like to acknowledge our sincere gratitude for the support extended by **Dr. G. Vishnu Murthy**, Dean, Dept. of CSE, Anurag University. We also express our deep sense of gratitude to **Dr. V V S S S Balaram**, Acadamic co-ordinator,
Dr. Pallam Ravi, Project in-Charge, **Dr. V. Rama Krishna,** Project Co-ordinator and Project review committee members, whose research expertise and commitment to the highest standards continuously motivated us during the crucial stage our project work.

ABSTRACT

In today's digital era, the sheer volume of resumes received by organizations has surged significantly, necessitating the need for automated resume analysis systems. This project offers a robust and efficient solution for parsing and analyzing resumes using natural language processing (NLP) techniques and machine learning algorithms. The system harnesses the capabilities of various NLP libraries such as spaCy, NLTK (Natural Language Toolkit), and scikit-learn to extract crucial information from resumes, including contact details, skills, work experience, education, and certifications. By employing these tools, the system can accurately identify and categorize the relevant sections of each resume, facilitating streamlined processing. Moreover, the system integrates advanced algorithms like TF-IDF (Term Frequency-Inverse Document Frequency) and cosine similarity to assess the suitability of candidates for specific job roles. By comparing the content of resumes against predefined job descriptions, it generates relevancy scores and ranks candidates based on their skill matches and alignment with job requirements. This approach ensures that candidates whose profiles closely match the desired criteria are prioritized in the selection process. To enhance user experience and facilitate interaction, the system offers an intuitive user interface. Through this interface, users can easily upload resumes, input job descriptions, and obtain ranked lists of potential candidates tailored to their requirements. This feature simplifies the recruitment process, enabling recruiters to efficiently identify and shortlist suitable candidates with minimal effort.

In terms of performance evaluation, the system employs standard metrics such as accuracy, precision, and recall to assess its effectiveness. By continuously monitoring these metrics, recruiters can gauge the system's performance and fine-tune parameters as needed, ensuring high-quality candidate shortlisting and efficient recruitment workflows. Overall, this system serves as a valuable tool for organizations seeking to streamline their resume screening and analysis processes. By automating tedious tasks and leveraging advanced NLP and machine learning techniques, it saves time and resources while improving the quality of candidate selection, ultimately enhancing the efficiency of recruitment operations.

Keywords *- Resume Analysis, nltk, Job Description Parsing, Skill Extraction, Experience Level Determination, Candidate Ranking, Similarity Score Calculation, TF-IDF (Term Frequency-Inverse Document Frequency), Cosine Similarity.*

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Intelligent Job Match represents a significant leap in recruitment technology, leveraging state-of-the-art Natural Language Processing (NLP) techniques to redefine how organizations approach talent acquisition. By deeply analyzing resumes, this cutting-edge system assigns nuanced scores to job roles, evaluating candidates based on a thorough assessment of their skills and professional backgrounds. Through sophisticated NLP algorithms, the platform identifies and categorizes keywords within resumes, creating insightful clusters that reflect candidates' expertise across various sectors and domains. Unlike traditional resume parsers, Intelligent Job Match goes beyond surface-level assessments, providing not just an overall suitability score but also offering targeted insights, predictions, and analytics. For job seekers, the platform acts as a valuable guide, highlighting areas for skill improvement and suggesting strategies to enhance their resumes. On the recruiter's side, it streamlines the hiring process by providing comprehensive candidate evaluations and facilitating data-driven decisions. This technology empowers both parties with actionable intelligence, promoting better matches and elevating the overall quality of talent acquisition processes in today's competitive job market. Intelligent Job Match heralds a new era of recruitment efficiency, where data-driven insights and intelligent recommendations drive better hiring outcomes.

1.1 OVERVIEW

The intelligent job matching system revolutionizes recruitment by harnessing advanced technologies like artificial intelligence (AI), machine learning (ML), and natural language processing (NLP). Its primary aim is to efficiently connect qualified candidates with suitable job opportunities, adapting to the dynamic needs of talent acquisition. At its core, a powerful matching algorithm compares candidate profiles with job descriptions. Analyzing factors such as skills, qualifications, experience, and preferences from candidate data alongside job requirements from employers, the algorithm identifies correlations to recommend the most suitable matches. Machine learning models continuously refine this process, enhancing accuracy over time. The system begins by profiling candidates, extracting information from resumes and profiles including skill sets, work history, education, and certifications. Similarly, it processes job descriptions, extracting essential requirements using advanced NLP techniques.

Operating as a recommender system, it provides personalized suggestions to candidates for relevant job openings and presents employers with curated lists of potential candidates. A feedback mechanism ensures continuous improvement, refining the algorithm based on input from candidates and employers. With an intuitive interface, candidates can easily create profiles and receive tailored recommendations, while employers can post job openings and manage the recruitment process efficiently. Ultimately, the intelligent job matching system optimizes recruitment, improving outcomes for both candidates and employers.

1.2 PROBLEM STATEMENT

The traditional resume screening process is inefficient, time-consuming, and prone to human error. Recruiters struggle to manually review and assess the vast number of resumes they receive, often overlooking qualified candidates due to keyword biases or inconsistencies in formatting. This leads to missed opportunities for both employers and potential employees. To address this issue, we are introducing an intelligent resume screening system that leverages natural language processing to streamline and enhance the recruitment process, providing an efficient and unbiased evaluation of candidates.

For Recruiters:

Imagine a company receives numerous resumes for a Data Analyst position. The HR team faces the challenge of manually reviewing each resume to identify qualified candidates. The process is time-consuming, and there's a risk of overlooking relevant skills.

For Applicants:

Many job seekers face the challenge of identifying the most suitable roles based on their skills and qualifications. With numerous job listings and varying requirements, it's time-consuming and confusing for applicants to match their expertise with the right job opportunities. This often results in applying to multiple positions, leading to a lack of focus and reduced chances of landing an ideal job.

1.3 OBJECTIVE

The objective of our work is to revolutionize the hiring process through the development and implementation of the Intelligent Job Match, leveraging advanced Natural Language Processing (NLP), the tool parses resume, identifies crucial keywords, and organizes information into sectors. The primary goal is to enhance recruitment efficiency by automating resume screening, providing tailored recommendations, and delivering insightful analytics to both applicants and recruiters.

1.4 PROJECT SCOPE

The objective of this project is to develop an automated resume parsing and analysis system that utilizes natural language processing (NLP) techniques and machine learning algorithms to streamline the recruitment process. The system aims to efficiently extract key information from resumes and match candidate profiles with job requirements. The scope of the project includes the following components:

1. Resume Parsing: The system will employ advanced techniques to accurately parse resumes and extract relevant information such as contact details, skills, work experience, education, and certifications. Libraries such as spaCy and NLTK will be utilized for text processing and entity recognition.

2. **Data Preprocessing:** Textual data from resumes will undergo preprocessing steps such as tokenization, stop-word removal, and stemming/lemmatization to enhance quality and efficiency in subsequent processing. This ensures that the system operates on clean and standardized data.

3. **NLP Techniques:** Advanced NLP techniques will be leveraged to extract meaningful insights from resume content. This includes entity recognition, sentiment analysis, and topic modeling to identify key elements and sentiments within the text.

4. **Machine Learning Algorithms:** Machine learning models will be developed to analyze resume content and identify relevant patterns and associations. Supervised learning techniques may be employed for tasks such as classification and prediction.

5. **Matching Algorithm:** A sophisticated matching algorithm will be designed to assess the compatibility of candidate profiles with job descriptions. This algorithm will consider factors such as skill overlap, experience relevance, and educational background to generate accurate matches.

6. **User Interface:** The system will feature an intuitive user interface for recruiters, allowing them to upload resumes, input job descriptions, and view ranked lists of potential candidates. Interactive visualizations may be incorporated to enhance user engagement.

7. **Evaluation Metrics:** The system will be evaluated using metrics such as accuracy, precision, and recall to quantify performance in candidate matching. This will provide insights into the effectiveness of the system and areas for improvement.

8. **Scalability and Performance:** The system will be engineered to handle large volumes of resumes and perform efficiently in real-world scenarios. Strategies for optimization, parallel processing, and distributed computing will be implemented as needed.

9. **Documentation and Deployment:** Comprehensive documentation will be provided, covering the system architecture, algorithms, implementation details, and usage instructions. The system will be deployed in a production environment for accessibility to recruiters and hiring managers.

By addressing each component within the project scope, the system aims to deliver a robust and efficient solution for automating resume parsing, analysis, and candidate matching, thereby enhancing the recruitment process for organizations.

2. LITERATURE SURVEY

A study conducted by Sujit Amin, Nikita Jayakar, Pheba Babu, and M. Kiruthika explored the application of machine learning classification algorithms in recruitment processes. Their approach aimed to enhance efficiency, offer objective evaluation criteria, improve candidate experience, and allow for customization. However, they acknowledged potential challenges such as reliance on data quality and the risk of overdependence on algorithms, which could limit adaptability and overlook qualitative aspects of candidate assessment.

In another study, D Jagan Mohan Reddy, Sirisha Regella, and Srinivasa Reddy Seelam investigated the use of statistical measures and feature selection machine learning algorithms in recruitment. Their method focused on enhancing efficiency, minimizing costs, and predicting candidate commitment. Despite its potential benefits, the approach relies heavily on historical data, raising concerns about algorithmic bias and the exclusion of qualitative nuances in candidate evaluation.

Subhajit Maity, Sujan Sarkar, Avinaba Tapadar, Ayan Dutta, Sanket Biswas, Sayon Nayek, and Pritam Saha proposed a structured resume analysis and business intelligence processing approach to optimize HR processes within IT companies. While this method offers increased efficiency and effectiveness, there's a risk of dependency on technology leading to oversight of nuanced human aspects essential in HR management.

Rasika Ransing, Akshaya Mohan, Nikita Bhrugumaharshi, and Kailas Mahavarkar explored the use of KNN, Linear SVC, and XGBoost algorithms for automated resume screening. Their research aimed to enhance efficiency and accuracy in talent acquisition. However, concerns were raised regarding potential biases and limitations in algorithmic decision-making, which could impact fairness and inclusivity in the hiring process.

Lastly, Tumula Mani Harsha, Gangaraju Sai Moukthika, Dudipalli Siva Sai, and Satish Anamalamudi proposed a method employing machine learning algorithms and NLP for matching candidate qualifications with job requirements. While this approach saves time and effort, there are potential biases in algorithmic decision-making and limitations in contextual understanding that need to be addressed for more robust and equitable recruitment processes.

Resume Classification and Ranking using KNN and Cosine Similarity: The KNN Algorithm is used to categorize the resumes into the appropriate categories, and Cosine Similarity is used to determine how well the candidate's resume matches the job description. The resumes are then sorted in accordance with their classifications.

Web Application for Screening Resume: Semi- supervised learning is used by the suggested system, which is now being put into use, to attain high accuracy. The recruiter's workload will be lighter because they will not have to manually go through the extensive pool of applicants' resumes.

A Machine Learning approach for automation of Resume Recommendation system: The screening and shortlisting processes may be greatly facilitated by an automated method of Resume Classification and Matching, and the candidate selection and decision- making processes would undoubtedly be sped up.

Intelligent Recruitment System Using NLP: The paper's main objective is to extract data from resumes and conduct the necessary analysis on the data to turn it into information that recruiters can use. This intelligent recruitment system uses Natural Language Processing. As a result, the Resume Parser would assist recruiters in quickly choosing the most qualified individuals, saving them both time and effort.

ResumeNet-A Learning-Based Framework for Automatic Resume Quality Assessment: A neural- network model that incorporates text processing techniques is created in the proposed system to forecast each resume's quality. We suggest many modifications of the model to address the label deficiency problem in the dataset, either by using the pair/triplet-based loss or by introducing some semi-supervised learning strategy to take use of the large amount of unlabeled data. The proposed basic paradigm and its variations are both universal and simple to use.

An Empirical Study of Artificial Intelligence and its Impact on Human Resource Functions: In the Delhi/NCR region, 115 HR experts from various IT sectors participated in this study. The association between these two variables was found to be positive using the multiple regression. These regression models help in deciding the better resume.

A Hybrid Approach to Conceptual Classification and Ranking of Resumes and Their Corresponding Job Post: What has been presented is a hybrid approach that uses conceptual- based classification of resumes and job postings and automatically ranks candidate resumes (that fall under each occupational category) to their corresponding job postings in order to address the issues with the previously highlighted techniques.

Design and Development of Machine Learning based Resume Ranking System: The model uses cosine similarity to find the resumes that are the most comparable to the job description supplied and the KNN algorithm is used to pick and rank the resumes based on job descriptions in huge quantities. KNN stores all the available data and classifies a new data point based on the similarity. The resumes are ranked according to the nearest similarity to the job description.

Natural Language Processing based Jaro: The Interviewing Chatbot: By presenting chatbot that conducts interviews by evaluating the candidates resumes, the suggested system, JARO, expedites the interview process towards an objective decision-making process. The chatbot then develops a series of questions to be asked to the candidate. The technology will have functions like automated interviewing and resume analysis for analyzing the resumes better.

3. ANALYSIS

3.1 EXISTING SYSTEM

The existing system uses Naïve Bayes algorithm. This method uses traditional machine learning and has lower accuracy rates. The potential of people may be lost due to this system. The recruiters cannot differentiate between the similarity of the resumes of the candidates as they might have taken a copy of someone else's resume. There are other approaches being tried, such KNN, however they are ineffective for huge datasets. Another popular approach is text parsing; however, it can only be applied to resumes with structure. If the outcome is positive, the resume matched the job description, then the test of hypothesis gives a statistical representation by comparing the variable data to the specified data. Manually checking the resumes can be time- consuming for the recruiters. Although semi-supervised learning based on machine learning can be utilized, it can only forecast the quality of the resume based on the job description. Linear classification is also used for screening the resumes but it just finds the most similar resume to the job description by drawing the regression graph. The positive resumes on the regression graph are scored accordingly.

3.1.1 DRAWBACKS OF EXISTING SYSTEM

It is less effective and inaccurate data. Of course, like anything related to technology, resume screening tools are smart, but they are not perfect. It is unlikely that a rushed and inconsistent process like this enables hiring organizations to identify the best talent in their pipeline. In some cases, overwhelmed recruiters may ignore a portion of applications altogether. While the system can eliminate those who are not a fit, some applicants may know how to manipulate the algorithm, resulting in a bad fit landing on the desk of a busy recruiter. After all, most of the top results for "resume scanner" are written to explain how to game the system. annual screening requires the recruiter to go through each application to sort out the 88% of applicants that are unqualified. Manual screening also creates opportunities for human failure, such as biases, tiredness, forgetfulness, etc. An automated system has no way to measure the intangibles, like the level of writing or design of the resume. It usually does not identify qualified military veterans. Applicants might also try to game the system, putting in key words solely because they were in the job description.

3.2 PROPOSED SYSTEM AND ADVANTAGES

Our approach integrates advanced machine learning and natural language processing (NLP) techniques to comprehensively evaluate resumes, transcending traditional keyword-based screening methods. Following resume parsing, our software conducts real-time candidate assessments aligned with the recruiter's job requirements. This interactive online application organizes resumes by intelligently comparing them to specified job descriptions, using NLP for instantaneous comparison and ranking. Unlike conventional approaches, we prioritize contextual understanding, extracting abilities and relevant criteria directly from resume content.

To ensure accuracy and relevance, we will implement several key components:
Keyword Extraction and Resume Parsing: We will leverage PDF parsing libraries and advanced NLP techniques to extract text content from resumes. This process will involve identifying and extracting relevant keywords and skills from both job descriptions and resumes. Additionally, we will implement a scoring system based on the relevance of these skills to different job roles.

Matching Algorithm: We will develop a sophisticated matching algorithm to compare the extracted keywords from job descriptions with the content of each resume. This algorithm will assign a score or weight to each matched keyword based on its relevance to the specific job role. By considering the context and importance of each skill, we aim to provide a more accurate assessment of candidate suitability.

Threshold Setting: To streamline the screening process, we will define a scoring threshold to categorize resumes as qualified or unqualified. This threshold will be adaptable to project goals, allowing users to fine-tune the screening process according to their preferences and the evolving needs of the hiring process. This flexibility ensures that the screening criteria can be adjusted to meet the specific requirements of each recruitment campaign.

Real-time Recommendation System: Once the system is trained, it will seamlessly integrate into a real-time job matching platform. This platform will continuously assess incoming resumes, match them to suitable roles based on the predefined criteria, and offer intelligent recommendations, predictions, and analytics for both applicants and recruiters.

Example:

The real-time resume screening system is developed to efficiently identify qualified candidates for a Data Scientist position. Its primary function is to match resumes with a predefined job description, enabling swift and effective assessment of candidate suitability.

In this scenario, the job description sets out specific requirements including essential skills such as Python, Machine Learning, and Data Analysis, as well as desired experience of 3+ years and a Master's degree in Computer Science. The sample resume provided showcases a candidate's skills, experience, and educational background, which will be evaluated against these criteria.

To evaluate resumes, the system employs a comprehensive scoring mechanism. This scoring system comprises three main components: keyword matching, experience matching, and education matching. Each component contributes to determining the overall suitability of a candidate for the Data Scientist position.

Keyword matching involves comparing the keywords extracted from the job description with the skills listed in the resume. For each matched keyword, the system assigns a score based on its relevance to the job requirements. In the provided example, skills like Python and Data Analysis match the job description, earning the candidate points in keyword matching.

Experience matching assesses the candidate's relevant work experience against the required experience specified in the job description. If the candidate's experience exceeds the specified requirement, they receive a full score for this criterion. In the example, the candidate's 4 years of experience in machine learning projects exceed the 3+ years required, earning them a full score in experience matching.

Education matching compares the candidate's educational background with the required qualifications. If the candidate possesses the exact educational qualification specified in the job description, they receive a full score for education matching. In this scenario, the candidate holds a Master's degree in Computer Science, matching the requirement precisely.

Upon evaluating each criterion, the system calculates an overall score for the resume. In the provided example, the resume achieves a high score of 4 out of 5, indicating a strong match for the Data Scientist position. This comprehensive evaluation ensures that only the most qualified candidates are identified for further consideration by recruiters, streamlining the hiring process and improving efficiency.

In conclusion, our solution combines advanced technology with customizable features to optimize the resume evaluation process. By implementing keyword extraction, a sophisticated matching algorithm, threshold setting, and a real-time recommendation system, we aim to enhance efficiency, accuracy, and effectiveness in talent acquisition.

3.3 SYSTEM REQUIREMENTS

For the resume screening system described, implemented as a web-based application, the system requirements would encompass hardware, software, functionality, and security aspects. Here is a detailed breakdown of the system requirements:

1. **Software Requirements:**

Pyparser: This software tool is required for parsing and extracting information from PDF resumes.

Anaconda or Python: The application relies on Python programming language or the Anaconda distribution, which includes Python and additional data science libraries.

PyCharm IDE: PyCharm is an integrated development environment (IDE) that facilitates coding and debugging Python applications.

Streamlit: Streamlit is a Python library used for building interactive web applications for data science and machine learning projects.

PDFMiner: PDFMiner is a library for extracting text and metadata from PDF documents.

Natural Language Toolkit (NLTK): NLTK is a platform for building Python programs to work with human language data.

2. **Hardware Requirements:**

Processor: Intel i5 5th Gen or higher processor is recommended for optimal performance.

RAM: Minimum 4 GB of RAM is required to ensure smooth operation of the application.

Storage: At least 256 GB of storage space is recommended to store application files and data.

Device Compatibility: The application should be compatible with various devices including laptops, mobile phones, and tablets/PCs to ensure accessibility for users.

These system requirements ensure that the application can be effectively developed, deployed, and used by users to parse and process PDF resumes, apply natural language processing techniques, and generate recommendations based on the content of the resumes.

3.4 FUNCTIONAL REQUIREMENTS

Functional Requirements for the Intelligent Job Matching System:

1. **User Registration and Authentication:** Users should be able to register for an account on the platform to access job matching services securely. Authentication mechanisms should ensure secure login and logout functionalities for user accounts.

Features:

i. User registration form with fields for username, email, password, etc.

ii. Password hashing for secure storage and login/logout functionalities.

2. **Job Seeker Profile Creation:** Job seekers should have the ability to create and manage their profiles, including details such as skills, experience, education, and preferences.

Features:

i. Profile creation form with fields for skills, experience, education, etc.

ii. Options to update and edit profile information as needed.

3. **Job Posting Submission:** Employers should be able to submit job postings, including details such as job title, description, requirements, and location.

Features:

i. Job posting submission form with fields for job title, description, requirements, etc.

ii. Validation checks to ensure completeness and accuracy of job postings.

4. **Intelligent Matching Algorithm:** The system should utilize an intelligent matching algorithm to analyze job seeker profiles and job postings and identify potential matches based on compatibility.

Features:

i. Integration of machine learning algorithms for job matching.

ii. Matching algorithm to identify compatibility between job seeker profiles and job postings.

iii. Automated matching process triggered when new job postings or job seeker profiles are submitted.

5. **Recommendation Generation:** Based on the matching algorithm results, the system should generate recommendations for job seekers, highlighting relevant job postings.

Features:

i. Recommendation system to suggest job postings based on job seeker profiles.

ii. Real-time updates to keep job seekers informed about new recommendations.

6. **Application Submission:** Job seekers should have the ability to apply to job postings directly through the platform, submitting their resumes and cover letters.

Features:

i. Application submission form with fields for resume, cover letter, etc.

ii. Options for job seekers to track the status of their applications.

These functional requirements form the core features of the Intelligent Job Matching System, enabling job seekers to efficiently find and apply to relevant job opportunities based on their skills, experience, and preferences within a user-friendly web-based application.

3.5 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements for the Intelligent Job Matching System:

1. **Performance:** Performance requirements ensure efficient system operation, minimal latency, and response times, even under high load conditions.

Features:

i. **Response Time:** The system should respond to user interactions (e.g., searching for job postings, submitting applications) within a specified time limit (e.g., less than 3 seconds).

ii. **Throughput:** The system should support a minimum number of concurrent users or transactions per second (e.g., 100 users/transactions per second).

iii. **Scalability:** The system should be designed to scale horizontally to handle increased user traffic and data volume without performance degradation.

2. **Security:** Security requirements ensure the confidentiality, integrity, and availability of user data and system resources.

Features:

i. **Data Encryption:** User data, including passwords and personal information, should be encrypted using secure encryption algorithms to prevent unauthorized access.

ii. **Access Control:** Role-based access control (RBAC) should be implemented to restrict access to sensitive functionalities and data based on user roles and permissions.

iii. **Secure Authentication:** Secure authentication mechanisms, such as password hashing and session management, should be implemented to prevent unauthorized access.

iv. **Secure Communication:** All communication between client devices and the server should be encrypted using HTTPS protocol to prevent eavesdropping and data tampering.

3. **Usability:** Usability requirements ensure that the system is intuitive, easy to use, and provides a positive user experience.

Features:

i. **User-friendly Interface:** The user interface should be intuitive, with clear navigation, consistent layout, and informative feedback.

ii. **Accessibility:** The system should adhere to accessibility standards (e.g., WCAG) to accommodate users with disabilities.

iii. **Responsiveness:** The system should provide real-time feedback to user actions (e.g., loading indicators, success messages) to enhance usability.

4. **Reliability:** Reliability requirements ensure consistent and reliable system operation without unexpected failures or downtime.

Features:

i. **Availability**: The system should be available for use 24/7, with minimal downtime for maintenance or updates (e.g., 99.9% uptime).

ii. **Fault Tolerance:** The system should be resilient to failures, with built-in redundancy and failover mechanisms to ensure continuous operation.

iii. **Data Integrity:** Data integrity checks should be implemented to detect and prevent data corruption or loss.

5. **Scalability:** Scalability requirements ensure that the system can accommodate growth in user traffic and data volume without performance degradation.

Features:

i. **Horizontal Scalability:** The system architecture should support horizontal scaling by adding more servers or resources to handle increased load.

ii. **Elasticity:** The system should dynamically scale resources up or down based on demand to ensure efficient resource utilization.

iii. **Load Balancing:** Load balancing mechanisms should distribute incoming traffic evenly across multiple server instances to prevent overloading.

4. SYSTEM DESIGN

4.1 System Analysis

The Intelligent Job Match Project and Resume Ranking system aims to streamline the process of connecting job seekers with suitable job opportunities by leveraging advanced AI algorithms. The system incorporates features for job seeker registration, resume submission, job posting, resume ranking, and intelligent job matching. Job seekers register on the platform, upload their resumes, and specify their preferences and qualifications. Recruiters post job listings with detailed descriptions. The system employs natural language processing (NLP) and machine learning techniques to analyze resumes and job postings, facilitating accurate matching based on skills, experience, and job requirements.

4.2 Architecture Diagram

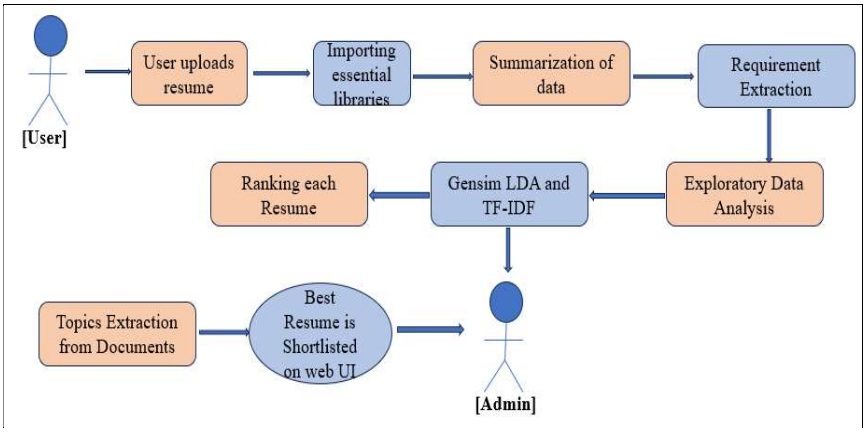


Figure 4.2 .1: *Block diagram of resumes and job descriptions undergoing parsing, keyword extraction, and matching for rapid job recommendations.*

User uploads resume: Job seekers interact with the User Interface of the Intelligent Job Match system to upload their resumes. They provide detailed information about their skills, experiences, education, and other relevant qualifications. This step is crucial as it serves as the initial input for the system to match job seekers with suitable job opportunities.

Importing essential libraries: Upon receiving the resume upload request, the application server imports essential libraries and dependencies required for processing the resume data. These libraries may include natural language processing (NLP) tools, machine learning frameworks, and other specialized packages for data analysis and manipulation.

Summarization of data: The uploaded resumes undergo a summarization process where key information is extracted and summarized. This involves techniques such as text summarization, where lengthy resumes are condensed into shorter, more concise representations. Additionally, keyword extraction and sentence embedding methods may be applied to identify and highlight important skills, experiences, and qualifications mentioned in the resume.

Requirement extraction: Simultaneously, the system extracts job requirements and preferences from job listings posted by recruiters. These requirements include desired skills, experience levels, educational backgrounds, and any other specific qualifications sought by employers. By extracting and structuring these requirements, the system prepares for comparison with the content of the resumes to identify suitable matches.

Exploratory data analysis: The system performs exploratory data analysis (EDA) on resumes and job requirements to uncover insights, such as skill, experience, and qualification distributions. EDA identifies trends and correlations, like common skill sets or industry-specific keywords, informing the matching process.

Gensim LDA and TF-IDF: The system applies advanced techniques such as Latent Dirichlet Allocation (LDA) and Term Frequency-Inverse Document Frequency (TF-IDF) to further analyze and extract insights from the resume and job description data. LDA is a topic modeling algorithm that identifies underlying themes or topics present in the text data. TF-IDF, on the other hand, measures the importance of words or terms in a document relative to a corpus of documents. These techniques help uncover latent topics, themes, and keywords within the resumes and job listings, facilitating better understanding and comparison.

Admin: An admin interface is provided to facilitate the management and oversight of the job matching process. The admin interface allows administrators or recruiters to review and assess the shortlisted resumes, make adjustments to the matching criteria, and manually intervene if necessary. It provides a centralized platform for monitoring and controlling the job matching workflow.

Best resume is shortlisted on web UI: Based on the analysis and comparison of resumes with job requirements, the system shortlists the best-matched resumes for each job listing. The shortlisted resumes are then displayed on the web UI, accessible to recruiters for further review and selection. This user-friendly interface allows recruiters to conveniently evaluate and assess candidate profiles, making informed hiring decisions.

Ranking each resume: Finally, the system ranks each resume based on its suitability and compatibility with the job requirements. Various factors such as the relevance of skills, experiences matching the job criteria, educational background, and other qualifications are considered in the ranking process. This ranking enables recruiters to prioritize and focus on the most promising candidates, streamlining the recruitment process and enhancing efficiency.

4.3 Data Flow diagram (DFD)

DFD graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system. The visual representation makes it a good communication tool between User and System designer. Structure of DFD allows starting from a broad overview and expand it to a hierarchy of detailed diagrams.

4.3.1 LEVEL-0 DIAGRAM

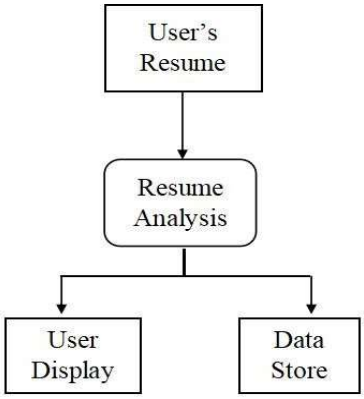


Figure 4.3.1 .1: *Level-0 DFD Diagram*

In this data-flow diagram, we can see that how system is associated with the Resume Analysis function and the database. We can see the data flow from resume to system, system to user and database. DFD Level 0 is also called a Context Diagram. It is a basic overview of the whole system or process being analyzed or modeled. It is

designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers. The context diagram must fit in one page. The process name in the context diagram should be the name of the information system. For example, Grading System, Order Processing System, Registration System. The context level diagram gets the number 0 (level zero). DFD Level 0 is also called a Context Diagram. It's a basic overview of the whole system or process being analysed or modelled. It is designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.

4.3.2 LEVEL-1 DIAGRAM

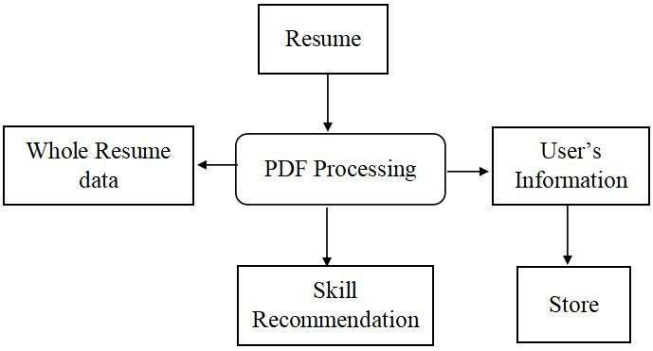


Figure 4.3.2 .1: *Level-1 DFD Diagram*

In this diagram, we can see that how PDF processing is working into the user's resume, it will fetch the text data from the resume. DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system, as you break down the high-level process of the Context Diagram into its sub processes.

4.3.3 LEVEL-2 DIAGRAM

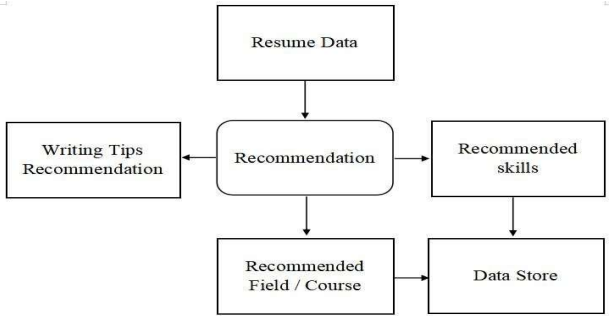


Figure 4.3.3 .1*: Level-2 DFD Diagram*

In this diagram, we can see that how recommendation is working into our system. A

level 2 data flow diagram (DFD) offers a more detailed look at the processes that make up an information system than a level 1 DFD does. It can be used to plan or record the specific makeup of a system. 2-level DFD goes one step deeper into parts of 1-level DFD. It can be used to plan or record the specific/necessary detail about the system's functioning.

4.4 UML Diagrams

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML comprises two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

4.4.1 Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor.

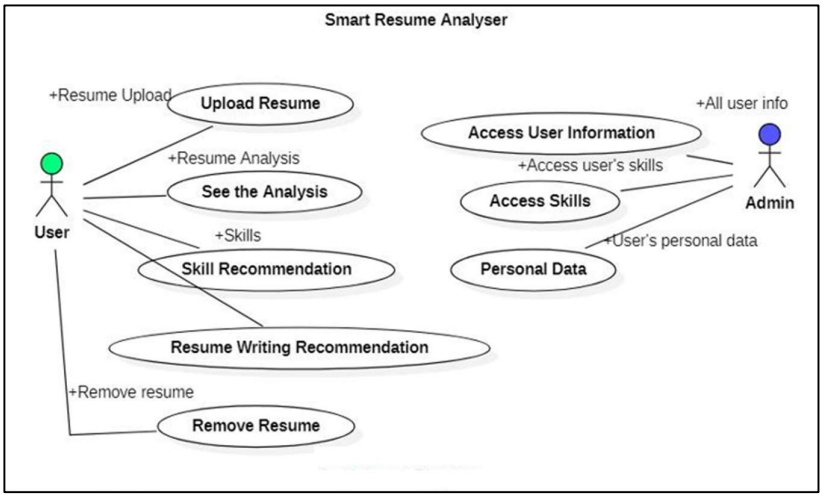


Figure 4.4.1 .1: *Use Case Diagram*

In this use case diagram, the actors are user and admin. They have various functionalities such as user has to upload resume, see analysis, skills are recommended, resume writing recommendations and can remove resume. Whereas, admin can access user information, access skills and user's personal data. We have seen what functionality the user and admin can use. We can see user and admin have the different access of the functionality. The purpose of use case diagram is to capture the dynamic aspect of a system. However, this definition is too generic to describe the purpose, as other four diagrams (Activity, Sequence, Collaboration, and State chart) also have the same purpose. We will look into some specific purpose, which will distinguish it from other four diagrams.

4.4.2 Class Diagram

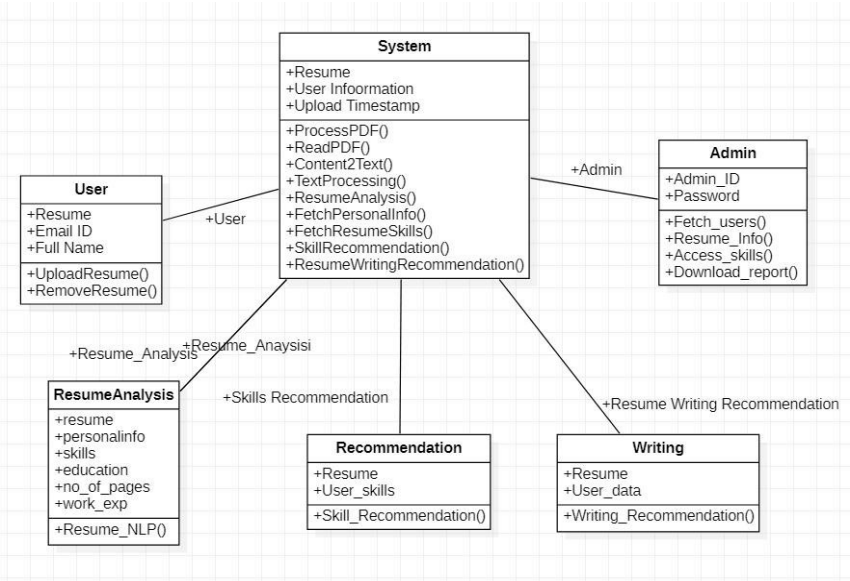


Figure 4.4.2 .1: *Class Diagram*

In the class diagram, the attributes are system, user, and admin. We see different classes and functions according to the functionality. We can see how system is associated with the user & admin functionality. The course chart is the most building piece of object-oriented modeling. It is utilized for common conceptual modeling of the structure of the application, and for nitty gritty modeling interpreting the models into programming code. Course charts can moreover be utilized for information modeling. The classes in a class diagram speak to both the most components and intelligent within the application, and the classes to be modified. The name of the class diagram should be meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance. Responsibility (attributes and methods) of each class should be clearly identified. For each class, minimum number of properties should be specified, as unnecessary properties will make the diagram complicated. Use notes whenever required to describe some aspect of the diagram. At the end of the drawing, it should be understandable to the developer/coder. Finally, before making the final version, the diagram should be drawn on plain paper and reworked as many times as possible to make it correct.

4.4.3 ACTIVITY DIAGRAM

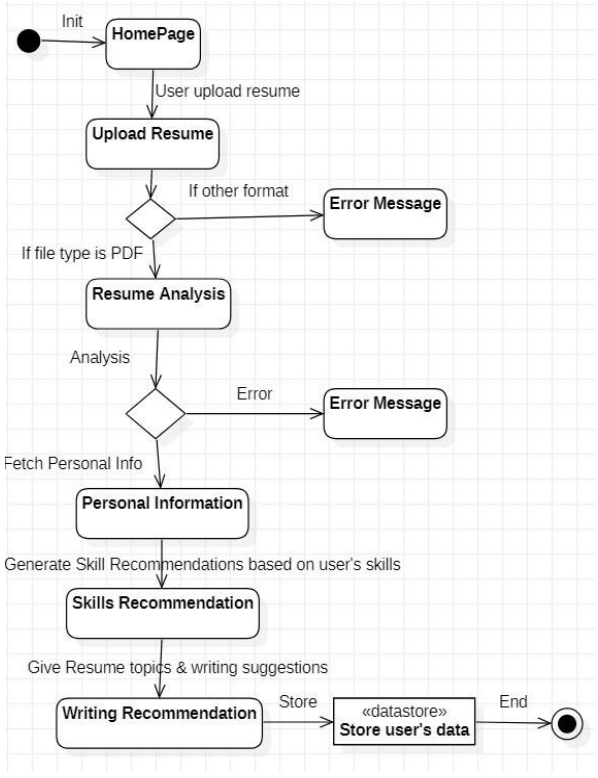


Figure 4.4.3 .1: *Activity Diagram*

In this diagram, we can see that the activity of Smart Resume Analyzer, if everything works well, we will get the output, otherwise it will be error. The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part. It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

5 PROPOSED METHOD

The suggested method is a resume ranking software that makes use of Cosine Similarity, and Overlapping Coefficient Natural Language as well as Natural Language Processing and Machine Learning. After filtering, it offers a ranking and suggests the best resume for a specified text job description. Rather than processing the entire document, the supplied information is summarised after data cleansing. Exploratory data analysis, or EDA, is a method used to develop trends and visual representations for the supplied data collection. To determine the frequency of terms that are used to meet the job description, TF-IDF is employed. The Web UI is used to extract the subjects from the document collection and display them. The user side of the dashboard allows users to upload their resumes in PDF format and access information on the abilities needed to land a job. Even the courses and advice for improved development are given to the users. The administrator side logs into the portal using the credentials and has access to data on the applicants' skill levels and resume scores. Both the user and the admin benefit from a hassle-free resume screening due to the complete process.

Keyword Extraction and Resume Parsing: In the first step of the process, user-uploaded PDF resumes are automatically processed to extract text content. This data extraction utilizes PDF parsing libraries and advanced Natural Language Processing (NLP) techniques such as Named Entity Recognition (NER) and syntactic parsing. These techniques help identify and categorize relevant details like skills, experiences, and education from the resumes. Following data extraction, a scoring mechanism is employed to assess the relevance of extracted skills to specific job roles, improving candidate screening efficiency. Integrated NLP models like spaCy and NLTK play a crucial role in accurately identifying key entities within the resume content. This approach streamlines resume analysis, facilitates informed decision-making in candidate selection, and optimizes recruitment workflows to handle large volumes of resumes effectively.

Matching Algorithm: The matching algorithm compares keywords extracted from the job description with content from each resume, evaluating their relevance. It assigns a score or weight to each matched keyword, reflecting its importance and alignment with job requirements. This process involves analyzing both the frequency and context of keywords within the resume content. The algorithm may utilize techniques like TF-IDF (Term Frequency-Inverse Document Frequency) to measure

keyword importance. By considering the specificity and uniqueness of matched keywords, the algorithm enhances accuracy in identifying suitable candidates. This approach optimizes the screening process by quantifying the degree of match between resume content and job expectations. It aids recruiters in efficiently shortlisting candidates who closely match the desired skill set and experience criteria. Overall, the matching algorithm improves the effectiveness and precision of candidate evaluation in the recruitment workflow.

Threshold Setting: The matching algorithm compares keywords extracted from the job description with content from each resume, evaluating their relevance. It assigns a score or weight to each matched keyword, reflecting its importance and alignment with job requirements. This process involves analyzing both the frequency and context of keywords within the resume content. The algorithm may utilize techniques like TF-IDF (Term Frequency-Inverse Document Frequency) to measure keyword importance. By considering the specificity and uniqueness of matched keywords, the algorithm enhances accuracy in identifying suitable candidates. This approach optimizes the screening process by quantifying the degree of match between resume content and job expectations. It aids recruiters in efficiently shortlisting candidates who closely match the desired skill set and experience criteria. Overall, the matching algorithm improves the effectiveness and precision of candidate evaluation in the recruitment workflow.

Real-time Recommendation System: The real-time recommendation system integrates seamlessly with job matching platforms, leveraging its trained model to evaluate incoming resumes swiftly. By applying advanced algorithms, it accurately matches candidate profiles with relevant job roles in real-time. This system provides intelligent recommendations based on skills, experience, and qualifications, enhancing the efficiency of the recruitment process. For applicants, it offers personalized job suggestions that align with their expertise and career aspirations. Simultaneously, recruiters benefit from predictive analytics, gaining insights into candidate-job fit and potential hiring trends. The system's real-time capabilities ensure timely and informed decision-making, improving overall recruitment outcomes and enhancing user experience on the platform.

5.1 DATABASE DICTIONARY

In this, we are going to understand the different tables of our database.

5.1.1 DATA DICTIONARY

DataBase Name: CV

5.1.2 USER DATA TABLE

Table Name: user_table

Column Name	Data Type	Description
ID	Auto Increment	Unique User ID
User Name	Varchar (50)	Full Name of user
Email ID	Varchar (50)	User Email ID
Resume_score	Varchar (10)	User's auto generated score
Time_stamp	Varchar (50)	Time stamp
Page_no	Varchar (5)	Number of pages
User_Level	Varchar (10)	User Experience
Actual_skills	Varchar (300)	User's actual skills
Recommended_skills	Varchar (300)	User Recommended skills
Recommended_courses	Varchar (600)	Recommended courses

Table 5.1.2.1: *User Data Table*

The above table describes about the database we are using to save data of uploaded resumes. Data gets automatically saved to the connected database. This SQL table has various constraints that are filled with the information. The database can be accessed by the admin and perform various operations on them. The admin can use the SQL functionalities to select the required candidates. For example, selecting the candidates who have better user experience or searching for the prominent domain in the actual skills.

5.2 ALGORITHMS

TF-IDF: The most popular technique for calculating word frequencies is called TF-IDF. This stands for "Term Frequency - Inverse Document" Frequency, which is one of the factors used to calculate a word's ultimate score [10]. Without getting into the numbers, TF-IDF word frequency scores seek to highlight more intriguing phrases, such as those that are repeated inside a text but not across texts. The TF-IDF Vectorizer can encode new frequency weightings, learn new terminology, tokenize texts, and invert frequency weightings.

Term Frequency: How frequently a word appears in a document is referred to as its term frequency.

Inverse Document Frequency: Downscale phrases that regularly appear in documents are referred to as having an inverse document frequency.

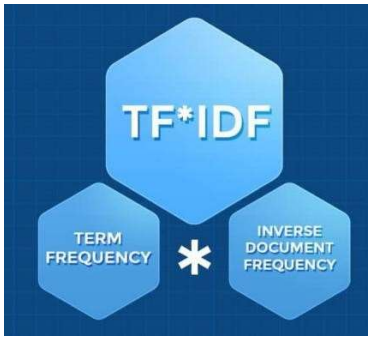


Figure 5.2 .1: *TF-IDF*

$$F - DF(,) = F(,) * DF(,) \quad (1)$$

$$F(,) = \frac{rq(,)}{\sum rq(i,)} \quad (2)$$

$$DF() = o(\frac{}{on()}) \quad (3)$$

Where, freq (t, d) is the number of times the word t appears in document d

N is the count of unique words in document d

Using equation (1) (2) and (3) the TF-IDF is calculated; a term's higher TF-IDF score, which is determined using the formulas above, indicates that it is more relevant to the content. We modelled the resumes and JD into a vector space in our system. This is done by assembling and translating a dictionary of terminology from the papers. A dimension of the vector space is assigned to each sentence. We created the TF- IDF matrix for the resumes and the job query using the Count Vectorizer and the TF- IDF matrix.

COSINE SIMILARITY: Cosine similarity is a metric for comparing two non-zero vectors in an inner product space. Its value is the same as the inner product of the identical vectors normalized to have the same length, or the cosine of the angle between them. The cosine of 0 degrees is 1, and any angle between (0,] radians) has a cosine that is less than 1. Thus, rather of considering magnitude, the comparison is based on orientation: two vectors with the same orientation have a cosine similarity of 1, two vectors oriented at 90 degrees from one another have a similarity of 0, and two vectors opposite have a similarity of -1, regardless of magnitude. The cosine similarity is quite helpful in positive space since the result is cleanly constrained in presentation style [0,1] [0,1].

The name is derived from the phrase "direction cosine": in this example, unit vectors are maximally "dissimilar" if they are orthogonal and maximally "similar" if they are

parallel. This is equivalent to the cosine, which has a value of unity when the segments subtend a zero angle and a value of zero when the segments are perpendicular.

$$\cos = \frac{\vec{\equiv}}{\|\vec{\equiv}\|} = \frac{\sum^{n} ii}{\sqrt{\sum_{i-1}^{n} i^2 . \sum^{n} i_{-1}^{2}}} \quad (4)$$

The below Figure represents cosine similarity where, ai and bi, respectively, are the parts of the vectors A and B. Below is a representation of the general Cosine Distance/Similarity formula [12]. In this instance, X1 represents the resume, X2 represents the job description that was provided by the resume.

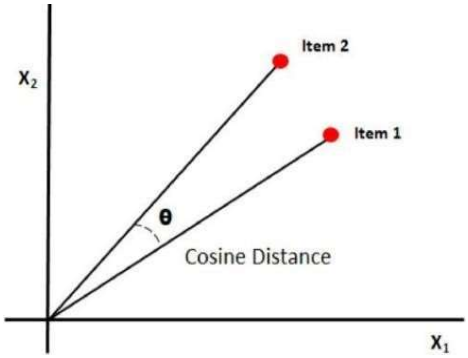


Figure 5.2 .2: *Cosine Similarity*

NLP: The JSON file is then sent through the NLP pipeline after the text annotations in the pdf file are converted to JSON format. This NLP pipeline is then used to train the model [13]. Using the NLP framework SpaCy, it can be trained. SpaCy is a framework that was developed for general data rather than for particular datasets, like a resume. In this method, rather than manually entering every word to create the dataset, semi-supervised learning is utilized to label the significant data in the ZIP file of PDF resumes. Homonyms, homophones, sarcasm, idioms, metaphors, exceptions to the rules of grammar and usage, and changes in sentence structure are just a few examples of the irregularities in human language that take humans years to learn but that programmers must teach natural language-driven applications to recognize and understand accurately from the beginning if those applications are to be useful.

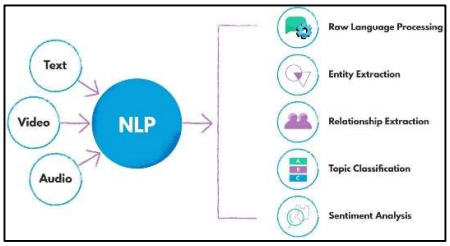


***Figure 5.2 .3**: NLP*

5.3 WORKFLOW

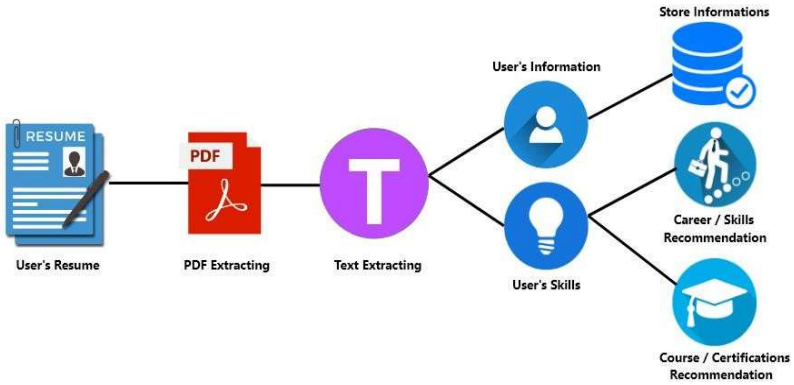


***Figure 5.3 .1:** Working of Intelligent Job Match*

We are going to see how actually our system is working behind, we have divided our work in separate tasks, let us understand each step of it. This shows the entire workflow of the system, how the resumes undergo various processes to give the resultant.

PDF EXTRACTING

PDF Extracting is module in which it will automatically get the resume from the user. The condition is that user's resume should be in PDF format. This module will automatically extract the user's data from the resume.

TEXT EXTRACTING

Text Extracting is the module in which it will fetch the text information from the resume. This text data will be used as language processing to further tasks like recommendations and fetching.

USER'S DATA

After the text extraction, there is next module is fetching the user information, just like for X person find the full name, contact, email, mobile, and skills from the text extraction data.

CAREER/SKILLS RECOMMENDATIONS

Now we have fetched the information of user. Now based on user's current skills we will give the career path & skills recommendations, just like if user have skills of Machine Learning, then it will give you the career, tools and technology recommendations.

COURSE/CERTIFICATIONS RECOMMENDATIONS

Now we have fetched the information of user. Now based on user's current skills we will give the courses & certificate recommendations, just like if user have skills of Machine Learning, then it will give you the free and paid type of courses and certificates recommendations.

DATA ANALYTICS

Now we have attached new module called data analytics, we have good number of the user's data. Now visualization is the best effective technique to understand the data's pattern. Now we are creating the visualizations (Pie Charts) for the admin side.

YOUTUBE VIDEO RECOMMENDATION

Now we have attached new module called YouTube recommendations. Now our system will recommend the two types of videos to the user. One is for resume preparation topic and another one is for the interview preparation topic. It will be directly scraped from the YouTube.

There are two modules in Intelligent Job Match, user and admin while each having its own special functions:

Resumes are uploaded by the applicants, which later undergo few pre-processing procedures. The talents are extracted from the curriculum vitae and compared to the obligatory competencies. The missing capabilities are mentioned under the recommended skills where, the access to sources that avail in development.

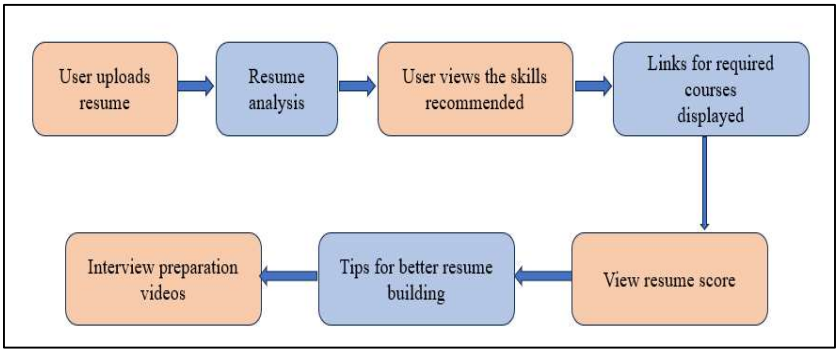


Figure 5.3 .2: *User WorkFlow*

Access to the admin module's features requires providing the necessary credentials. Admins can access a comprehensive report of all applicant resumes, including applicant name, ID, and email address. The report also displays predicted fields and experience levels determined by skills and scores. Furthermore, recommended courses, skills, and actual talents are listed. Each resume is timestamped, and details are automatically uploaded to the connected database, with the resumes saved in a designated destination folder. This streamlined process enhances efficiency and provides valuable insights for effective candidate management.

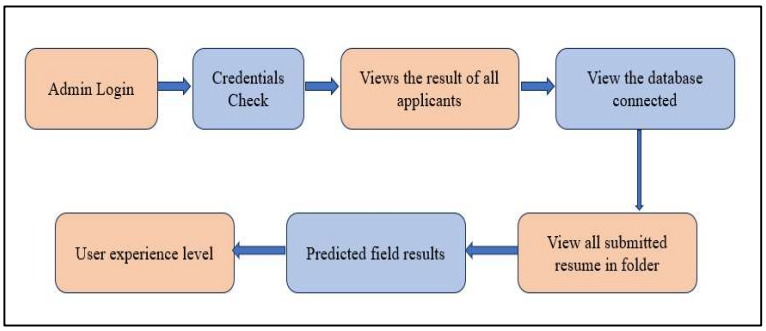


Figure 5.3 .3: *Admin WorkFlow*

5.4 SELECTION OF THE SOFTWARE

PYTHON PROGRAMMING LANGUAGE: Python is advanced, higher level & easy to learn programming language, we have chosen Python to develop Desktop GUI, We will create a desktop application using Python. In python everything is related to the Image processing is very easy to implement. The documentation & community of the python is very big, so we will get the help in development from the community.

PYCHARM IDE: PyCharm is professional IDE which is developed by JetBrains. The features which are offered by PyCharm Community version is Intelligent Python editor, Code debugger, Code inspection, VCS support and many more related to the User Interface.

STREAMLIT: Streamlit is special framework for handling the Data intensive applications, we are using it because it has had amazing UI components, we do not need to implement HTML+ CSS, everything will be handled by python code. It is very easy to use. In this deployment is also very easy.

NLTK: The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing for English written in the Python programming language. We are using NLTK to preprocess the resume.

XAMPP: It is open-source control panel which is easy to use and maintain the Apache, database, and PhpMyAdmin. It is very easy to operate. We can operate Apache, MySQL easily from it. It is easily available for windows & windows & another platform.

GITHUB: It is free & open-source backup platform for your running project. It is maintaining by GIT. It is very easy to upload your work on free. There is no storage limitation. You can do development in group & separately also. It is very easy to maintain the work with other teammates also.

6. IMPLEMENTATION

6.1 CODING OF SYSTEM

The coding is the process of transforming the design of a system into a computer language format. This coding phase of software development is concerned with software translating design specification into the source code. It is necessary to write source code & internal documentation so that conformance of the code to its specification can be easily verified. Coding is done by the coder or programmers who are independent people than the designer. The goal is not to reduce the effort and cost of the coding phase, but to cut to the cost of a later stage. The cost of testing and maintenance can be significantly reduced with efficient coding.

6.2 FUNCTIONALITY SPECIFICATION

In our system there will be two modules, admin & user. Admin will be only one. User can be multiple. Let see what user & admin can do.

User	Admin
Upload the Resume	Do Login
Check the own information	Check the all-user's report
Check the skills/career recommendations.	Can export the report in CSV.
Check the Courses/Certifications recommendations.	Admin can see the Data Analytics

Table 6.2.1: *Functionality Specification*

As specified above, the user uploads the resume and checks their own information like resume score and skills they already have. The user is provided with skills and career recommendation according to the skills in resume. Access to various courses and certifications is also given that indeed helps in increasing the resume score.

The admin logs on to the portal using their credentials and check user report of all the uploaded resume. Data visualizations are presented about predicted field of the user and user experience level.

6.3 SAMPLE CODING

The code which is mentioned below it is just main logic of giving the recommendations to the user. This code is only just functional part, not the full part with the backend.

6.3.1 PDF EXTRACTING CODE

This code is used for the PDF extraction; it will extract the Pdf of resume which is uploaded by the user.

```
def pdf_reader(file):
    resource_manager = PDFResourceManager()
    fake_file_handle = io.StringIO()
    converter = TextConverter(resource_manager, fake_file_handle, laparams=LAParams())
    page_interpreter = PDFPageInterpreter(resource_manager, converter)
    with open(file, 'rb') as fh:
        for page in PDFPage.get_pages(fh,
                                      caching=True,
                                      check_extractable=True):
            page_interpreter.process_page(page)
            print(page)
        text = fake_file_handle.getvalue()

    ## close open handles
    converter.close()
    fake_file_handle.close()
    return text
```

Figure 6.3.1 .1: *pdf extracting code*

6.3.2 RECOMMENDER CODE

This code is used for the course recommender, it will fetch the user's skills, based on that skills it will give the recommendations.

```
# course recommendations which has data already loaded from Courses.py
def course_recommender(course_list):
    st.subheader("*Courses & Certificates Recommendations 👨‍🎓*")
    c = 0
    rec_course = []
    ## slider to choose from range 1-10
    no_of_reco = st.slider('Choose Number of Course Recommendations:', 1, 10, 5)
    random.shuffle(course_list)
    for c_name, c_link in course_list:
        c += 1
        st.markdown(f"({c}) [{c_name}]({c_link})")
        rec_course.append(c_name)
        if c == no_of_reco:
            break
    return rec_course
```

Figure 6.3.2 .1: *recommender code*

6.3.3 PDF SHOWING CODE

This code is used to show the uploaded resume in the user interface, it simply allows to display the uploaded resume into the system.

```
# show uploaded file path to view pdf_display
def show_pdf(file_path):
    with open(file_path, "rb") as f:
        base64_pdf = base64.b64encode(f.read()).decode('utf-8')
    pdf_display = F'<iframe src="data:application/pdf;base64,{base64_pdf}" width="700"
                                    height="1000" type="application/pdf"></iframe>'
    st.markdown(pdf_display, unsafe_allow_html=True)
```

Figure 6.3.3 .1: *pdf showing code*

6.3.4 INSERT USER'S DATA CODE

This code is used to insert the user data into our database.

```
def insert_data(sec_token,ip_add,host_name,dev_user,os_name_ver,latlong,city,state,country,
                act_name,act_mail,act_mob,name,email,res_score,timestamp,no_of_pages,reco_field,
                cand_level,skills,recommended_skills,courses,pdf_name):
    DB_table_name = 'user_data'
    insert_sql = "insert into " + DB_table_name + """
    values (0,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"""
    rec_values = (str(sec_token),str(ip_add),host_name,dev_user,os_name_ver,
                  str(latlong),city,state,country,act_name,act_mail,act_mob,name,
                  email,str(res_score),timestamp,str(no_of_pages),reco_field,
                  cand_level,skills,recommended_skills,courses,pdf_name)
    cursor.execute(insert_sql, rec_values)
    connection.commit()
```

Figure 6.3.4 .1: *Insert user's data code*

6.3.5 RESUME RANKING CODE

This Python code is for a functionality to rank resumes based on their similarity to a given job description. Let's break down the main components:

User Interface (UI):

- It uses Streamlit (st) for building the UI.
- The user can input a job description and upload one or more resumes in PDF format.

Processing Resumes:

- It saves the uploaded resumes to a directory named "uploads".
- It processes each uploaded resume by extracting text content using extract_text_from_pdf () function and then extracting entities like names and emails using extract_entities () function.

Text Vectorization:

- It uses TF-IDF (Term Frequency-Inverse Document Frequency) vectorization to convert text data into numerical vectors.
- It creates a TF-IDF vector for the job description and each processed resume.

Similarity Calculation:

- It calculates the cosine similarity between the TF-IDF vector of the job description and each resume.
- Cosine similarity measures the similarity between two non-zero vectors of an inner product space.

Ranking and Deduplication:

- It ranks the resumes based on their similarity scores, highest to lowest.
- It deduplicates resumes based on name and email to avoid showing duplicate entries.

Displaying Results:

- It displays the ranked resumes along with their names, emails, and similarity scores in the UI.
- It provides a download button to download the results in CSV format.

CSV Download:

- It generates CSV content containing the rank, name, email, and similarity score of each ranked resume.
- It provides a download button for the user to download the CSV file.

Overall, this code provides a convenient way for users to rank and evaluate multiple resumes based on their relevance to a specific job description. It leverages text processing techniques and cosine similarity to automate the ranking process, making it efficient for recruiters or hiring managers to review job applications.

```
def run():

    elif choice == 'Resume Ranking':
        st.header("Rank Resumes")

        job_description = st.text_input("Enter job description")

        uploaded_files = st.file_uploader("Upload one or more resumes", type="pdf", accept_multiple_files=True)

        if st.button("Rank Resumes") and uploaded_files:
            results = []

            # Create a directory for uploads if it doesn't exist
            if not os.path.exists("uploads"):
                os.makedirs("uploads")

            # Process uploaded resumes
            processed_resumes = []
            for uploaded_file in uploaded_files:
                # Save the uploaded file
                save_path = os.path.join("uploads", uploaded_file.name)
                with open(save_path, "wb") as f:
                    f.write(uploaded_file.getbuffer())

                # Process the saved file
                resume_text = extract_text_from_pdf(save_path)
                emails, names = extract_entities(resume_text)
                processed_resumes.append((names, emails, resume_text))
```

Figure 6.3.5 .1: *resume ranking - 1*

```
def run():
                # Process the saved file
                resume_text = extract_text_from_pdf(save_path)
                emails, names = extract_entities(resume_text)
                processed_resumes.append((names, emails, resume_text))

            # TF-IDF vectorizer
            tfidf_vectorizer = TfidfVectorizer()
            job_desc_vector = tfidf_vectorizer.fit_transform([job_description])

            # Rank resumes based on similarity
            ranked_resumes = []
            for (names, emails, resume_text) in processed_resumes:
                resume_vector = tfidf_vectorizer.transform([resume_text])
                similarity = cosine_similarity(job_desc_vector, resume_vector)[0][0] * 100
                ranked_resumes.append((names[0] if names else "N/A", emails[0] if emails else "N/A", similarity))

            # Sort resumes by similarity score
            ranked_resumes.sort(key=lambda x: x[2], reverse=True)

            # Deduplicate based on name and email
            deduplicated_resumes = []
            seen_names_emails = set()
            for name, email, similarity in ranked_resumes:
                if (name, email) not in seen_names_emails:
                    deduplicated_resumes.append((name, email, similarity))
                    seen_names_emails.add((name, email))
```

Figure 6.3.5 .2: *resume ranking - 2*

```
def run():
        # Deduplicate based on name and email
        deduplicated_resumes = []
        seen_names_emails = set()
        for name, email, similarity in ranked_resumes:
            if (name, email) not in seen_names_emails:
                deduplicated_resumes.append((name, email, similarity))
                seen_names_emails.add((name, email))

        # Display results
        st.subheader("Ranked Resumes")
        for rank, (name, email, similarity) in enumerate(deduplicated_resumes, start=1):
            st.write(f"**Rank:** {rank}, **Name:** {name}, **Email:** {email}, **Similarity:**
                    {similarity:.2f}")

        # Download CSV
        st.subheader("Download CSV")
        csv_content = "Rank,Name,Email,Similarity\n" + "\n".join([f"{rank},{name},{email},
                                                                  {similarity}" for rank,
                                                                  (name, email, similarity) in
                                                                  enumerate(deduplicated_resumes, start=1)])
        st.download_button(label="Download CSV", data=csv_content, file_name="ranked_resumes.csv",
                           mime="text/csv")
```

Figure 6.3.5 .3: *resume ranking - 3*

7. EXPERIMENTAL RESULTS / OBSERVATIONS

There are two modules in Intelligent Job Match, user, and admin while each having its own special functions.

- Resumes are uploaded by the applicants, which later undergo few pre-processing procedures. The talents are extracted from the curriculum vitae and compared to the obligatory competencies. The missing capabilities are mentioned under the recommended skills where, the access to sources that avail in development.
- Access to the admin module's features requires providing the necessary credentials. Admins can access a comprehensive report of all applicant resumes, including applicant name, ID, and email address.

The necessary credentials must be supplied in order to access admin module's features. Admin can view report of all the applicant resumes, which includes applicant name, ID, and email address. Additionally, the predicted field and experience level are listed, which are computed by the skills and score. Along with these recommended courses, recommended skills, and actual talents are all provided. Each resume has a time stamp, and details get automatically uploaded on to the connected database, while the resumes get saved to the destination folder.

	Predicted Field	Timestamp	Predicted Name	Predicted Mail
4	Data Science	2024-01-26_16:25:18		akshaykumarmurarishetty@gmail.com
5	Data Science	2024-01-26_16:26:26		likhithakodumuri2003@gmail.com
6	Data Science	2024-01-26_16:26:58		akshaykumarmurarishetty@gmail.com
7	Data Science	2024-01-26_16:27:39	Kodumuri Likhitha	likhithakodumuri2003@gmail.com
8	Data Science	2024-01-26_16:35:16		likhithakodumuri2003@gmail.com
9	Data Science	2024-01-26_16:40:24		akshaykumarmurarishetty@gmail.com
10	Web Development	2024-01-26_17:59:11	Anireddy Sai	anireddysaikiran1010@gmail.com
11	Data Science	2024-01-26_18:18:25		likhithakodumuri2003@gmail.com
12	Web Development	2024-01-30_10:14:33		likhithakodumuri2003@gmail.com
13	Web Development	2024-01-30_10:19:20		likhithakodumuri2003@gmail.com

Figure 7.1: *Resume Report*

The users field forecasts and level of experience are statistically represented as a pie chart. The below pie chart is plotted against the applicant's aptitudes to predict the field they might prosper, this avails recruiters in understanding which domain are most users familiar with.

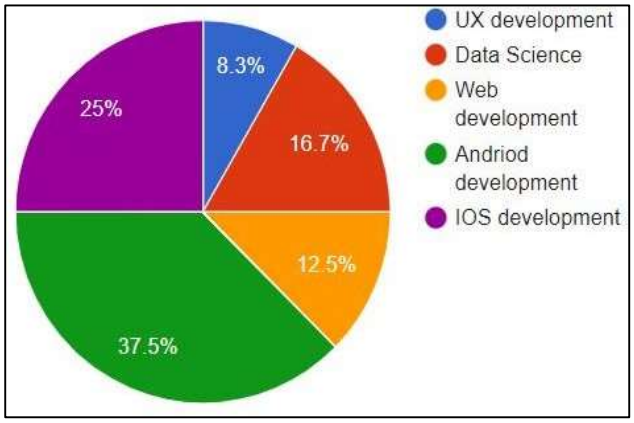


Figure 7.2: *Pie Chart for Predicted Field*

The below cited pie chart is about the user's experience level which is determined by the job description and qualifications mentioned in resume. The admin utilizes the experience level to narrow down the type of employee they are seeking for based on the job description**.**

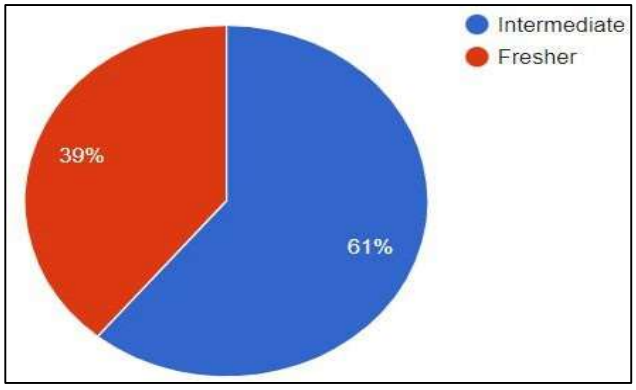


Figure 7.3: *Pie Chart for User Experience Level*

The process of bulk resume analysis and ranking involves several key steps that contribute to efficient candidate evaluation and ranking based on job relevance.

Initially, the system allows for the bulk upload of resumes, streamlining the input of candidate data. Upon upload, each resume undergoes data processing to extract vital information such as skills, experiences, and educational background.

A ranking algorithm is then applied to assess the similarity between each resume and the provided job description. This algorithm calculates a similarity score, often leveraging techniques like TF-IDF (Term Frequency-Inverse Document Frequency) and Cosine Similarity. Based on these similarity scores, resumes are ranked in descending order, with the most relevant candidates receiving higher ranks.

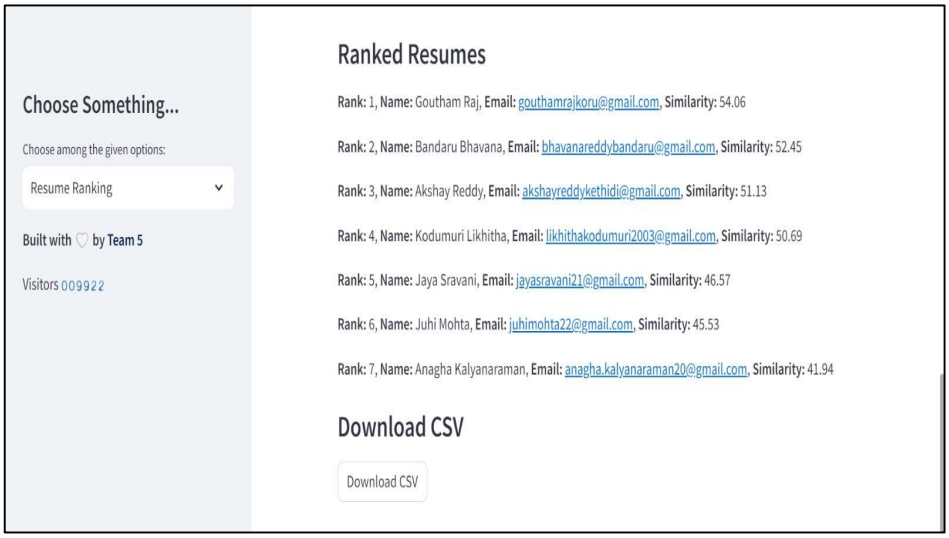
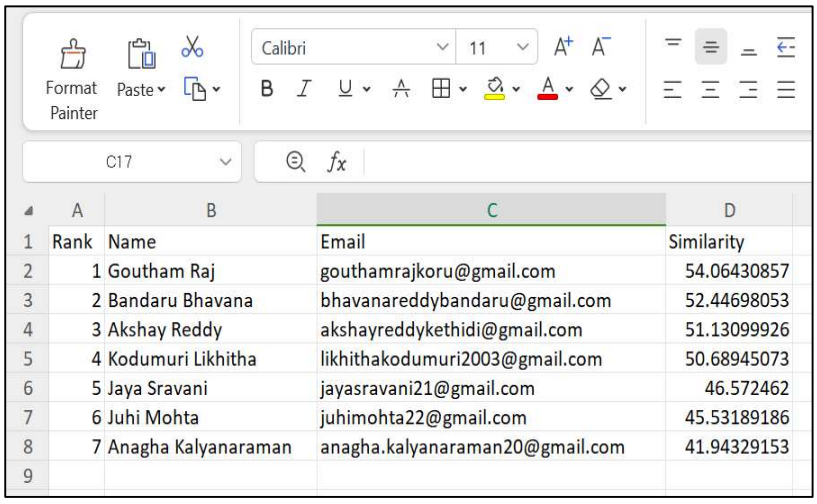


Figure 7.4: *output of resumes is ranked in descending order*

To ensure accuracy and avoid redundancy, a deduplication process may be employed, eliminating duplicate entries based on candidate names or email addresses. The ranked resumes are presented visually through tables or charts, offering recruiters a clear overview of candidate rankings and their corresponding similarities to the job description.

Moreover, the system generates downloadable reports, typically in CSV format, containing ranked lists of resumes along with pertinent details such as candidate names, email addresses, similarity scores, and ranks. These reports serve as valuable references for recruiters during the candidate selection process, facilitating informed decision-making and optimizing the recruitment workflow. Overall, this structured approach enhances recruitment efficiency and assists recruiters in identifying top candidates based on their qualifications and relevance to job requirements.



	A	B	C	D
1	Rank	Name	Email	Similarity
2	1	Goutham Raj	gouthamrajkoru@gmail.com	54.06430857
3	2	Bandaru Bhavana	bhavanareddybandaru@gmail.com	52.44698053
4	3	Akshay Reddy	akshayreddykethidi@gmail.com	51.13099926
5	4	Kodumuri Likhitha	likhithakodumuri2003@gmail.com	50.68945073
6	5	Jaya Sravani	jayasravani21@gmail.com	46.572462
7	6	Juhi Mohta	juhimohta22@gmail.com	45.53189186
8	7	Anagha Kalyanaraman	anagha.kalyanaraman20@gmail.com	41.94329153
9				

Figure 7.5: *contents of downloaded csv file*

8. CONCLUSION

The challenges faced by both recruiters and job seekers in the current job market underscore the critical necessity for innovative solutions like automated resume screening and analysis systems. Recruiters grapple with the overwhelming volume of resumes, particularly for specialized roles like Data Analysts, which strains HR teams. Manual review processes are not only time-consuming but also prone to errors, risking the oversight of vital skills and qualifications.

On the flip side, job seekers struggle to pinpoint suitable roles that match their expertise and aspirations amidst a sea of job listings with varying requirements. This leads to confusion and inefficiency in the application process, prompting applicants to scatter their efforts across multiple positions, diminishing their chances of securing an ideal job match. The proposed automated resume screening and analysis system tackles these challenges by harnessing advanced technologies such as PDF parsing, Natural Language Processing (NLP), and machine learning algorithms. By automating the extraction of crucial information from resumes and implementing a scoring mechanism based on skill relevance, the system streamlines the recruitment process for recruiters. It enables swift identification of qualified candidates, saving time and resources while ensuring a more precise screening process.

For job seekers, the system offers a targeted approach to job searching by recommending roles closely aligned with their skills and qualifications. This reduces the time and effort spent on applying to irrelevant positions, enhancing the likelihood of securing an ideal job fit. Ultimately, the proposed system aims to bridge the gap between recruiters and job seekers, fostering a more efficient and effective recruitment process that benefits both parties and contributes to a streamlined and productive hiring ecosystem.

9. FUTURE SCOPE

Future Enhancements for Intelligent Job Match:

1. Integration of Social Networking Data: Incorporate data from social networking platforms such as LinkedIn, GitHub, and professional forums to enrich candidate profiles. Analyze social behavior data to gain insights into candidates' professional networks, interests, and endorsements.
2. Collaborative Filtering Techniques: Implement collaborative filtering algorithms to match candidates with jobs based on the preferences and experiences of similar candidates. Utilize collective intelligence to improve job recommendations and increase the likelihood of successful matches.
3. Dynamic User Profiles: Develop algorithms to dynamically update user profiles based on interactions, feedback, and changes in skillsets or career preferences. Utilize machine learning models to personalize job suggestions and adapt to evolving user preferences over time.
4. Subject Matter Expert Involvement: Collaborate with HR professionals and industry experts to refine the job matching model and ensure alignment with industry standards and best practices. Incorporate feedback from subject matter experts to iteratively improve the accuracy and effectiveness of the matching algorithm.
5. Advanced Semantic Analysis: Explore the use of advanced semantic analysis techniques, such as latent semantic analysis (LSA), to improve the accuracy of job matching. Compare outcomes from term frequency-based similarity techniques with semantic analysis results to identify areas for optimization and refinement.
6. Personality Assessment: Integrate personality assessment tools to evaluate candidates' personalities and determine their alignment with job requirements and organizational culture. Analyze social media data included in resumes to gain insights into candidates' personalities and preferences.
7. Multimodal Data Fusion: Fuse data from multiple modalities, including textual resumes, social media profiles, and behavioral data, to create comprehensive candidate profiles. Develop algorithms to extract relevant information from diverse data sources and integrate it into the job matching process.

8. Continuous Learning and Improvement: Implement mechanisms for continuous learning and improvement, such as feedback loops and model retraining, to enhance the accuracy and relevance of job recommendations. Monitor system performance and user feedback to identify areas for enhancement and optimization over time.
9. Ethical Considerations and Bias Mitigation: Address ethical considerations and potential biases in the job matching process, such as algorithmic fairness and privacy concerns. Implement measures to mitigate biases and ensure fair and transparent job matching outcomes for all candidates.

By incorporating these future enhancements, the intelligent job matching system can become more sophisticated, adaptive, and effective in connecting candidates with opportunities that align with their skills, preferences, and career aspirations.

10. REFERENCES

[1] Web Application for Screening Resume-
https://ieeexplore.ieee.org/document/8945869

[2] Recruitment Prediction using Machine Learning-
https://ieeexplore.ieee.org/document/9276955

[3] Business Intelligence Assistant for Human Resource Management for IT Companies-
https://ieeexplore.ieee.org/document/9325471

[4] Screening and Ranking Resumes using Stacked Model-
https://ieeexplore.ieee.org/document/9707977

[5] Automated Resume Screener using Natural Language Processing(NLP)-
https://ieeexplore.ieee.org/document/9777194