

# **DESIGN OF RESUME MATCHING SYSTEM USING NLP**

Submitted by

**SUDIP PATRA**

Roll No.-10271023038

Registration No.-231020510038 of 2023-2024

Under the guidance of

Dr. Indrajit Bhattacharya

Assistant Professor, Dept. of Computer Application, KGEC



**Kalyani Government Engineering College**

Kalyani - 741235, Nadia, WB

Affiliated to

Maulana Abul Kalam Azad University of Technology

Phone: 25826680 (PBX)

Fax : 25821309

কল্যাণী - ৭৪১ ২৩৫  
নদীয়া, পশ্চিমবঙ্গ



Kalyani 741 235  
Nadia, West Bengal, India

পত্রাঙ্ক / Ref. No.:

তারিখ / Date :

কল্যাণী গভঃ ইঞ্জিনিয়ারিং কলেজ  
Kalyani Government Engineering College  
( Govt. of West Bengal )

Certificate of Approval

This is to certify that the project report on “DESIGN OF RESUME MATCHING SYSTEM USING NLP” is a record of project work under the curriculum of Maulana Abul Kalam Azad University of Technology(MAKAUT) for the MCA 2<sup>ND</sup> year, 3<sup>RD</sup> semester Examination, 2025, for the subject “MINOR PROJECT AND VICE-VOCE (MCAN 381)” carried out by “ SUDIP PATRA (Roll: 10271023038) “ the student of Kalyani Govt. Engineering College, under the guidance of “Dr. Indrajit Bhattacharya”, Assistant Professor, CA, as a requirement for the partial fulfillment of the Degree of Master of Computer Application.

Head

Department of Computer Application  
Kalyani Govt Engg. College

Supervisor

Department of Computer Application  
Kalyani Govt Engg. College

External Examiner

Department  
Seal

## ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude and appreciation to all the individuals and organizations who have contributed to the successful completion of this project.

First and foremost, we extend our deepest thanks and gratitude to our principal, **Dr. Sourabh Kumar Das**, as well as our Head of Department, **Dr. Surya Sarathi Das**, and our project supervisor, **Dr. Indrajit Bhattacharya**, for their exceptional guidance, unwavering support, and invaluable mentorship throughout this journey. Their expertise, insightful feedback, and encouragement have been instrumental in shaping the direction and execution of our project. We are truly grateful for their dedication and guidance.

We would also like to acknowledge the contributions of our senior's, whose extensive knowledge, experience, and assistance have immensely contributed to the overall success of our project. His valuable insights and constructive criticism have greatly enriched our work, and we are thankful for his continuous support and mentorship.

We extend our sincere thanks to our institute, Kalyani Government Engineering College, for providing the resources, infrastructure, and a conducive research environment that have significantly contributed to the smooth progress of our project.

Last but not least, we are deeply grateful to our friends, family, and loved ones for their unwavering support, understanding, and encouragement throughout this endeavour. Their patience and motivation have been invaluable in keeping us inspired and focused.

Signature  
Sudip Patra  
Roll No: 10271023038

# ABSTRACT

In the era of digital transformation, traditional recruitment methods often struggle to keep pace with the ever-growing pool of talent and job opportunities. This project presents a cutting-edge AI-powered Design of Resume Matching System using NLP that revolutionizes the hiring process by automating and optimizing candidate screening. Leveraging advanced Natural Language Processing (NLP) techniques, the system preprocesses resumes and job descriptions through text normalization, tokenization, stop-word removal, and lemmatization, followed by feature extraction using TF-IDF vectorization and similarity measurement via cosine similarity.

The software offers an intuitive and user-friendly interface that caters to students and administrators alike. Students can easily register, upload resumes, and track their status, while the admin section efficiently handles resume validation, approval, and matching with job descriptions. By eliminating errors, streamlining processes, and ensuring precise matching, this system addresses the inefficiencies of manual recruitment workflows and empowers organizations to make faster, data-driven hiring decisions.

The robust testing and evaluation of the system underscore its reliability and effectiveness in aligning candidate qualifications with employer needs. Future enhancements, such as the integration of machine learning algorithms for predictive analysis, support for multiple languages, and industry-specific customizations, make this software a forward-thinking solution. This project not only reduces the time and effort required for recruitment but also ensures fair and accurate evaluation of 85% ultimately bridging the gap between talent and opportunity in the modern workforce.

# Table of Contents

I. Introduction .....	1-4
• Problem Specification.....	1
• Motivation .....	1-2
• Objectives .....	3
• Challenges .....	3-4
• Novelty .....	5-7
II. Literature Survey .....	8-11
III. Technology Used and Definitions .....	9-12
IV. Proposed Method .....	12-26
• Proposed Model Architecture .....	12-14
• Data Acquisition for the Resume Screening System .....	14-16
• Algorithm .....	17-18
• Vectorization and Similarity Calculation .....	18-20
• Work Flow or Data Flow Diagram .....	21-23
• Testing and Evaluation .....	23-26
V. Experimental Result & Analysis .....	27-32
• Input, Output Results .....	27-29
• Analysis Of System .....	30-32
VI. Future Work .....	33
VII. Conclusion .....	34
VIII. References .....	35

# I : INTRODUCTION

In today's hiring process, organizations often face the challenge of reviewing hundreds or thousands of resumes for a single job. Manual resume screening is time-consuming, inconsistent, and prone to human bias, making the recruitment process inefficient and unfair. Machine learning (ML) provides a smarter solution to streamline CV filtering.

ML-powered CV filtering software uses advanced algorithms to analyze, sort, and rank resumes automatically. By applying techniques like natural language processing (NLP) and deep learning, it goes beyond keyword matching to understand resume details, identify transferable skills, and predict candidate-job fit. This improves efficiency, saves time, and enables recruiters to focus on top candidates.

Moreover, ML minimizes bias by standardizing evaluations and learning from diverse datasets, ensuring fair and objective assessments. These systems promote inclusive hiring practices while making the recruitment process faster, accurate, and scalable. This document explores how ML-based CV filtering works, its benefits, and its role in modern hiring workflows.

## **Problem Specification :**

An automated system for efficiently filtering and ranking resumes based on employer-specific job requirements. It streamlines the recruitment process by matching candidate profiles to job descriptions, ensuring a faster and more accurate selection.

## **Motivation :**

Recruitment is critical for organizational success, but traditional methods struggle to meet modern hiring demands. Key motivations for developing machine learning-based CV filtering software include:

1. **Increasing Application Volumes:** High numbers of resumes make manual screening slow and impractical.
2. **Need for Efficiency:** Recruiters must quickly identify top candidates without wasting time on irrelevant applications.
3. **Reducing Human Bias:** ML ensures fair, unbiased evaluations based solely on qualifications and skills.

4. **Improving Candidate Experience:** Faster screening enhances applicants' perception of the company.
5. **Understanding Resume Nuances:** ML models interpret diverse formats, job titles, and transferable skills that keyword systems miss.
6. **Talent Matching:** ML identifies candidates for roles beyond those they applied for, maximizing talent utilization.
7. **Competitive Advantage:** Faster, smarter hiring helps organizations secure top talent ahead of competitors.

This software improves efficiency, fairness, and inclusivity, addressing the growing challenges of modern recruitment.

## **Objectives :**

The primary objective of the CV filtering software is to revolutionize the recruitment process by leveraging machine learning to automate and optimize resume screening. This involves addressing the inefficiencies and biases of traditional methods while enhancing the overall quality and fairness of hiring decisions. The key objectives include:

1. **Automate Screening:** Automatically extract key details like skills, qualifications, and experience from resumes.
2. **Save Time:** Quickly identify and shortlist top candidates to reduce time-to-hire.
3. **Improve Accuracy:** Analyze resumes contextually to avoid missing qualified candidates.
4. **Reduce Bias:** Ensure fair and consistent evaluations to promote inclusivity.
5. **Handle Large Volumes:** Efficiently process high numbers of applications.
6. **Rank Candidates:** Provide ranked lists of applicants based on job-specific criteria.
7. **Predict Fit:** Assess candidates for long-term success and cultural fit.
8. **Recognize Transferable Skills:** Suggest candidates for other roles they may qualify for.

**9. Seamless Integration:** Work with applicant tracking systems (ATS) and recruitment tools.

**10. Enhance Candidate Experience:** Provide timely updates and a positive application process.

These objectives aim to create a faster, fairer, and more reliable hiring process, helping organizations find the best talent efficiently.

## Challenges

The development and implementation of machine learning-based CV filtering software face challenges across technical, operational, and ethical domains that must be addressed for effective deployment. Data quality issues arise from inconsistent resume formats, biased or outdated training data, and difficulty interpreting unconventional resumes. Bias in training data risks perpetuating unfair hiring practices, while understanding nuanced language, industry-specific jargon, and transferable skills adds complexity. Overfitting to job descriptions may exclude qualified candidates with unconventional backgrounds, and privacy and security concerns necessitate robust measures to protect data and comply with laws like GDPR. Scalability, real-time performance, and integration with existing recruitment systems pose technical and operational hurdles, while user adoption requires trust and training. Ethical concerns, such as transparency and fairness, demand careful attention to avoid discrimination, and continuous learning is essential to adapt models to evolving job markets and skills. Thoughtful design, robust data strategies, and ongoing improvement are crucial to ensuring the software transforms recruitment processes while maintaining accuracy, fairness, and transparency.

## Novelty

The proposed CV filtering software stands out for its innovative application of advanced machine learning techniques to solve long-standing challenges in recruitment. Unlike traditional systems or basic keyword-based solutions, this software introduces several novel features and approaches that enhance its efficiency, accuracy, and fairness.

The proposed CV filtering software introduces three key novelties that set it apart:



### **1. Advanced NLP Preprocessing with KNN Model:**

The software integrates Natural Language Processing (NLP) with a K-Nearest Neighbors (KNN) model, enabling precise context-based filtering and matching of resumes. This approach ensures accurate evaluation of candidate skills, even from diverse and unconventional resumes.

### **2. Enhanced Privacy and Security with Local Database Storage:**

By utilizing a local database, the software ensures robust data protection and compliance with privacy regulations like GDPR. This design minimizes the risk of data breaches while maintaining control over sensitive candidate information.

### **3. Support for Multiple File Formats:**

The system seamlessly processes resumes in various formats, including Word, PDF, and plain text, ensuring compatibility and ease of use for both applicants and recruiters.

These innovations make the software a versatile, secure, and efficient tool for modern recruitment needs.

## II : LITERATURE SURVEY

### **1. Salton, G., & McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill.**

This book provides a foundational overview of information retrieval (IR) systems, introducing key concepts such as vector space models, Boolean retrieval, probabilistic models, and relevance ranking. The authors discuss various indexing techniques, query expansion strategies, and evaluation metrics that form the basis of modern search engines. Their work has significantly influenced the development of text retrieval systems and remains a cornerstone in the field of IR.

### **2. Ramos, J. (2003). *Using TF-IDF to Determine Word Relevance in Document Queries*. Proceedings of the First International Conference on Machine Learning.**

This paper explores the Term Frequency-Inverse Document Frequency (TF-IDF) weighting scheme, a statistical method used to evaluate word importance in textual data. The study highlights how TF-IDF improves document retrieval and ranking by reducing the influence of common words while emphasizing unique, contextually relevant terms. The research demonstrates TF-IDF's effectiveness in enhancing information retrieval and search performance.

### **3. Singhal, A. (2001). *Modern Information Retrieval: A Brief Overview*. IEEE Data Engineering Bulletin.**

This paper presents an overview of modern information retrieval techniques, focusing on advancements in indexing, query processing, and ranking algorithms. Singhal discusses the evolution from traditional Boolean retrieval models to machine learning-based approaches, emphasizing probabilistic models and natural language processing (NLP). The work serves as a concise introduction to contemporary IR methodologies and their real-world applications.

### **4. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space*. arXiv preprint arXiv:1301.3781.**

This paper introduces Word2Vec, a neural network-based model that efficiently learns word representations in vector space. The authors propose two architectures, Continuous Bag-of-Words (CBOW) and Skip-gram, which capture semantic relationships between words based on their context. Word2Vec has revolutionized NLP by enabling more accurate word similarity and analogy tasks, improving language understanding in AI applications.

**5. Pennington, J., Socher, R., & Manning, C. D. (2014). *GloVe: Global Vectors for Word Representation*. Proceedings of EMNLP.**

The authors present GloVe, a word representation model that leverages global statistical information from text corpora to generate meaningful word embeddings. Unlike Word2Vec, GloVe combines word co-occurrence matrices with matrix factorization techniques to enhance semantic representation. The model outperforms traditional vector representations in tasks such as word similarity, analogy detection, and sentiment analysis.

**6. Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). *Latent Dirichlet Allocation*. Journal of Machine Learning Research.**

This paper introduces Latent Dirichlet Allocation (LDA), a generative probabilistic model used for topic modeling in text data. LDA assumes that documents are mixtures of topics and that each topic is characterized by a distribution of words. The model has been widely adopted in NLP for applications such as document classification, text summarization, and recommender systems.

**7. Wang, P., Qian, L., & Kuang, H. (2018). *Recruitment Prediction Using Machine Learning Algorithms*. Journal of Human Resource Management Research.**

This study investigates the use of machine learning techniques to predict recruitment outcomes by analyzing candidate resumes, job descriptions, and hiring decisions. The authors apply various classification models, including decision trees, support vector machines (SVM), and neural networks, to automate candidate screening. The research demonstrates how AI can enhance hiring efficiency and reduce human bias in recruitment.

**8. Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). *A Fast Learning Algorithm for Deep Belief Nets*. Neural Computation.**

The authors introduce a novel algorithm for training Deep Belief Networks (DBNs), a class of generative neural networks composed of multiple layers of Restricted Boltzmann Machines (RBMs). The paper highlights the efficiency of unsupervised pretraining in improving deep learning models, leading to advancements in feature learning and representation learning across various AI applications.

**9. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. arXiv preprint arXiv:1907.11692.**

This paper introduces RoBERTa, an optimized variant of BERT that improves upon its predecessor by removing the next-sentence prediction objective and training on larger batches of data for longer durations. The study demonstrates RoBERTa's superior performance across multiple NLP benchmarks, reinforcing the effectiveness of transformer-based language models in contextual text understanding.

**10. Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media.**

This book provides a practical introduction to NLP using Python, with a focus on the Natural Language Toolkit (NLTK). It covers essential topics such as tokenization, part-of-speech tagging, parsing, and sentiment analysis, making it a valuable resource for researchers and developers working on text processing and machine learning applications.

**Key Gaps Identified:**

1. Limited context understanding in existing systems.
2. Persistent bias in candidate evaluation models.
3. Lack of dynamic transferable skill identification.
4. Scalability and real-time processing challenges.
5. Insufficient focus on explainability and transparency.

### III : TECHNOLOGY USED & DEFINITIONS

#### 1. NLP (Natural Language Processing) :

- **Definition:** Natural Language Processing (NLP) is a branch of artificial intelligence that focuses on enabling machines to understand, interpret, and respond to human language in a meaningful way. NLP combines computational linguistics with machine learning techniques to analyze and derive insights from text and speech data.

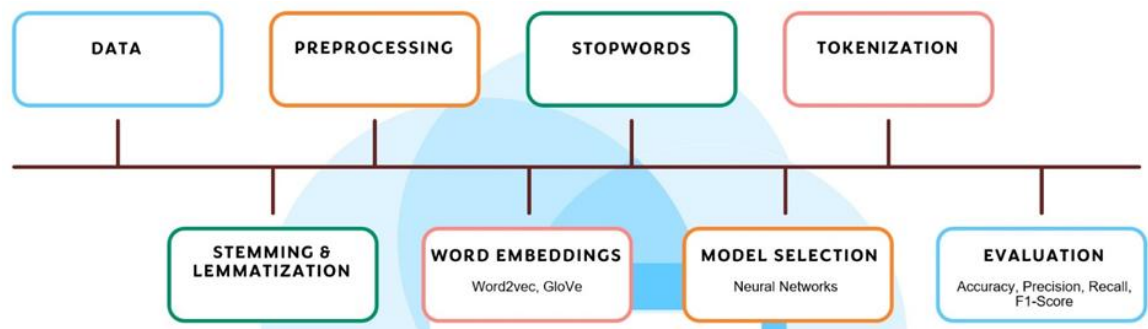


Fig 1: Basic architecture of NLP

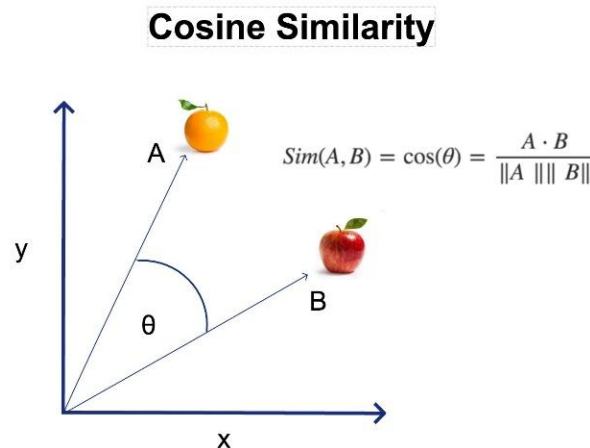
- **Elaboration:** NLP plays a pivotal role in the "Resume Matching Tool" by enabling the analysis and extraction of meaningful information from unstructured text, such as resumes and job descriptions. Through processes like tokenization, stemming, lemmatization, named entity recognition (NER), and part-of-speech tagging, the tool identifies key elements such as skills, experience, and qualifications. This ensures that the matching algorithm works with standardized and relevant data for better comparisons.

#### 2. KNN (K-Nearest Neighbors) :

- **Definition:** K-Nearest Neighbors (KNN) is a supervised machine learning algorithm used for classification and regression. It works by finding the k closest data points in a dataset to a given query point and makes predictions based on their majority class or average value.
- **Elaboration:** In the context of the "Resume Matching Tool," KNN is used to classify resumes or rank candidates based on their similarity to the job requirements. By treating resumes and job descriptions as feature vectors, the algorithm determines the top k closest matches, highlighting the most suitable candidates. This method ensures that results are interpretable and straightforward to implement.

### 3. Cosine Similarity :

- **Definition:** Cosine similarity is a metric that measures the cosine of the angle between two non-zero vectors in an n-dimensional space. It is used to assess the similarity between two documents irrespective of their size by considering the orientation rather than the magnitude of the vectors.



- **Elaboration:** Cosine similarity is central to the "Resume Matching Tool" for comparing textual data. Resumes and job descriptions are transformed into vector representations using techniques like TF-IDF or word embeddings. The cosine similarity score is then calculated to determine the degree of match between the two documents. A score close to 1 indicates a strong match, while a score closer to 0 signifies a weak correlation.

### 4. TF-IDF (Term Frequency-Inverse Document Frequency) :

- **Definition:** TF-IDF is a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents (corpus). It is calculated by multiplying the term frequency (TF) by the inverse document frequency (IDF).

$$w_{x,y} = tf_{x,y} \times \log \left( \frac{N}{df_x} \right)$$

**TF-IDF**

Term  $x$  within document  $y$

$tf_{x,y}$  = frequency of  $x$  in  $y$

$df_x$  = number of documents containing  $x$

$N$  = total number of documents

- **Elaboration:** TF-IDF is instrumental in the "Resume Matching Tool" for converting resumes and job descriptions into numerical vectors. It assigns higher weights to terms that are frequent in a single document but rare across the corpus, emphasizing their relevance. This ensures that the

similarity computation focuses on critical terms rather than common words, enhancing the accuracy of the matching process.

## 5. HTML (HyperText Markup Language) :

- **Definition:** HTML is the standard markup language for creating and structuring content on the web. It uses a system of elements such as headings, paragraphs, links, and forms to define the structure of web pages.
- **Elaboration:** In this project, HTML is utilized to design the front-end interface where users interact with the tool. This includes features like resume upload forms, buttons for submitting job descriptions, and sections for displaying results. The clear and semantic structure provided by HTML ensures that the tool is accessible and user-friendly.

## 6. CSS (Cascading Style Sheets) :

- **Definition:** CSS is a stylesheet language used to describe the presentation, layout, and formatting of a document written in HTML. It allows developers to control the visual appearance of web pages, including colors, fonts, spacing, and responsiveness.
- **Elaboration:** CSS enhances the visual appeal of the "Resume Matching Tool." It ensures that the user interface is not only functional but also aesthetically pleasing. Styles are applied to forms, buttons, tables, and other elements to make the tool intuitive and easy to navigate. CSS also supports responsive design, allowing the tool to adapt seamlessly to different devices and screen sizes.

## 7. Python Flask :

- **Definition:** Flask is a lightweight web framework for Python that enables developers to build web applications quickly and efficiently. It provides tools and libraries for routing, handling HTTP requests, and rendering templates.
- **Elaboration:** Flask is the backbone of the "Resume Matching Tool's" back-end. It handles server-side logic, processes user inputs, and integrates various components like NLP and machine learning models. Flask routes manage the submission of resumes and job descriptions, coordinate the

processing pipeline, and serve the matching results to the front end. Its flexibility allows seamless integration of APIs and other third-party services if needed.

## 8. Scikit-learn (sk-learn) :

- **Definition:** Scikit-learn is a powerful open-source machine learning library in Python that offers a wide range of algorithms and tools for data preprocessing, model building, and evaluation. It supports tasks like classification, regression, clustering, and dimensionality reduction.
- **Elaboration:** Scikit-learn is used extensively in the "Resume Matching Tool" for implementing algorithms like KNN and preprocessing textual data. Features like vectorization (e.g., CountVectorizer or TfidfVectorizer) are employed to convert textual data into numerical representations suitable for machine learning. Additionally, scikit-learn's utilities for splitting datasets, scaling data, and evaluating models ensure robust implementation of the matching algorithm.

## 9. NLTK (Natural Language Toolkit) :

- **Definition:** NLTK is a comprehensive Python library for working with human language data. It provides tools for tasks such as tokenization, stemming, lemmatization, stop-word removal, and part-of-speech tagging, making it a go-to library for text processing.
- **Elaboration:** NLTK is a critical component of the "Resume Matching Tool" for preprocessing text data. It is used to clean and normalize resumes and job descriptions by removing irrelevant words (stop words), reducing words to their root forms (stemming or lemmatization), and extracting meaningful terms. This ensures that only relevant information is passed to the similarity computation and matching stages .



## **IV : PROPOSED METHOD**

### **1. Proposed Model Architecture:**

#### **i. Frontend Layer**

- **Technologies:** HTML, CSS, JavaScript, Flask Templates
- **Functionality:**
  - Provides user-friendly interfaces for students and admins
  - Pages: login, registration, resume upload, admin dashboard
  - Interaction: Communicates with the backend via HTTP requests
- **Key Features:**
  - Form validation for inputs
  - Dynamic content for different user roles (student/admin)

#### **ii. Backend Layer**

- **Framework:** Flask (Python)
- **Functionality:**
  - Handles HTTP requests and routes
  - Processes user inputs (login, registration, resume upload)
  - Integrates with the database and ML model
  - Manages user roles (student/admin) and session control
- **Endpoints:**
  - /login: User authentication
  - /register: Register new users
  - /student: Handle resume uploads, display status
  - /admin: Admin approval/rejection of resumes

### iii. Database Layer

- **Database:** MySQL (scalable to MongoDB/PostgreSQL)
- **Entities:**
  - **Users Table:** Stores user details (id, name, contact info, degree, department, etc.)
  - **Admin Table:** Stores admin credentials (id, username, password)
- **Functionality:**
  - Store and retrieve user data, resume paths, and approval status
  - Track resume-screening decisions

### iv. Machine Learning Layer

- **Components:**
  - **NLP Model:** Pretrained model (e.g., spaCy) for resume parsing
  - **Matching Algorithm:** Measures similarity (e.g., TF-IDF + Cosine Similarity)
- **Functionality:**
  - Extracts skills, qualifications, and experience from resumes
  - Compares extracted data with job requirements
  - Ranks resumes based on relevance
- **Workflow:**
  - Admin uploads job descriptions
  - System matches resumes to the job description
  - Outputs matching scores and recommends approval/rejection

### v. File Storage Layer

- **System:** Local filesystem (scalable to AWS S3)
- **Functionality:**
  - Stores uploaded resumes
  - Organizes resumes by unique identifiers (e.g., universityRoll)

## **vi. Integration Layer**

### **○ Middleware:**

- Connects frontend to backend
- Ensures secure data transmission (HTTP methods: GET, POST)
- Implements role-based access control for students/admins

### **○ API/Endpoints:**

- REST APIs for uploading resumes, fetching status, and managing admin actions

## **2. Data Acquisition for the Resume Screening System :**

The data acquisition process is critical to developing an effective resume-screening system. It involves collecting, processing, and preparing data required for the model and application to function efficiently. Below are the key components and steps involved:

### **1. Data Sources**

#### **• a. Resumes**

- **Source:** Students upload resumes through the frontend interface (formats: PDF, DOCX, etc.).
- **Purpose:** Extract relevant data like skills, qualifications, work experience.
- **Storage:** Resumes saved in a structured folder system or cloud storage, indexed by unique identifiers (e.g., universityRoll).

#### **• b. Job Descriptions**

- **Source:** Admins upload job descriptions (formats: text files or structured forms) via the admin panel.
- **Purpose:** Define job requirements for matching against resumes.
- **Storage:** Stored in a database or file system for quick retrieval during the matching process.

## 2. Data Processing

- **a. Resume Parsing**

- **Purpose:** Extract key information such as personal details, skills, qualifications, experience, and contact information.
- **Method:** Use NLP techniques or pre-trained models (e.g., spaCy) to parse resumes.
- **Storage:** Extracted data is stored in structured form in the database, mapped to the corresponding resume.

- **b. Job Description Parsing**

- **Purpose:** Extract key requirements from job descriptions such as skills, qualifications, experience, and job title.
- **Method:** Similar to resume parsing, NLP is used to process job descriptions and identify essential criteria.
- **Storage:** Key details from job descriptions are stored in the database for matching with resumes.

## 3. Data Preparation

- **a. Resume Normalization**

- **Purpose:** Standardize resume data for consistent processing, e.g., transforming text data into a uniform format (lowercase, removing special characters).
- **Method:** Text preprocessing techniques like tokenization, stemming, and lemmatization.
- **Storage:** Cleaned data is stored in the database for easy comparison with job descriptions.

- **b. Job Description Normalization**

- **Purpose:** Standardize job description text for effective comparison with resumes.
- **Method:** Similar text preprocessing techniques to resume normalization.
- **Storage:** Pre processed job descriptions stored in the database.

## 4. Data Matching

- **a. Resume-Job Matching Algorithm**

- **Purpose:** Compare resumes to job descriptions based on extracted data (skills, qualifications, etc.).
- **Method:** Use algorithms like TF-IDF (Term Frequency-Inverse Document Frequency) and Cosine Similarity to calculate similarity scores.
- **Storage:** Matching scores and recommendations (approve/reject) are stored in the database for admin review.

## 5. Data Storage and Retrieval

- **a. Centralized Database**

- **Purpose:** Store all user data (students, admins), resumes, job descriptions, matching results, and status updates.
- **Method:** Use relational databases like SQLite, MySQL, or PostgreSQL.
- **Storage:** Structured tables for easy retrieval and querying (e.g., Users table, Resumes table, Job Descriptions table, Matching Results table).

- **b. File Storage**

- **Purpose:** Store uploaded resumes in a secure and easily accessible manner.
- **Method:** Resumes can be stored either on a local file system or using cloud storage services (e.g., AWS S3).
- **Storage:** Organized by unique identifiers (e.g., universityRoll) for easy retrieval.

### 3. Algorithm :

The algorithm for the resume matching system is designed to efficiently compare candidate resumes with job descriptions, ensuring accurate and fair matching. It processes resumes in various formats, extracts key information like skills and experience, and ranks candidates based on their suitability. The system prioritizes precision, fairness, and security, making it a reliable tool for streamlining recruitment.

The algorithm follows steps which are described below :

#### Step 1. Data Preprocessing

- **Purpose:** Clean and prepare text for analysis.
  - **Steps:**
    - Remove special characters, numbers, and extra spaces.
    - Convert text to lowercase.
    - Remove stopwords (e.g., "is," "and").
    - Apply lemmatization to get root words (e.g., "running" → "run").
  - **Output:** Cleaned text ready for vectorization.
- 

#### Step 2. Vectorization (TF-IDF)

- **Purpose:** Convert text into numerical vectors.
  - **Steps:**
    - Use TF-IDF to assign importance scores to words.
    - Transform resumes and job descriptions into vectors.
  - **Output:** Numerical vectors representing text.
- 

#### Step 3. Similarity Calculation (Cosine Similarity)

- **Purpose:** Measure how closely resumes match job descriptions.
- **Steps:**
  - Compute cosine similarity between job and resume vectors.

- Convert distances into similarity percentages (0-100).
  - **Output:** Similarity scores for ranking resumes.
- 

#### Step 4. Testing and Evaluation

- **Purpose:** Measure system performance.
  - **Metrics:**
    - Precision: Correct matches among predicted relevant resumes.
    - Recall: Correct matches among all relevant resumes.
    - F1 Score: Balances precision and recall.
    - Accuracy: Overall correct predictions.
  - **Output:** Performance metrics to validate the system.
- 

#### Step 5. Refinement

- **Purpose:** Improve accuracy and efficiency.
  - **Steps:**
    - Enhance preprocessing, fine-tune TF-IDF, and optimize similarity calculations.
  - **Output:** A robust resume screening system.
- 

### 4. Vectorization and Similarity Calculation :

To match resumes with job descriptions effectively, we convert text into numerical representations and compute their similarity. Two key techniques are used:

1. Vectorization using TF-IDF: Represents textual data numerically while emphasizing unique and meaningful words.
2. Similarity Calculation using Cosine Similarity: Measures the similarity between two text vectors based on their orientation in a vector space.

## 1. TF-IDF Vectorization :

TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical method to evaluate the importance of a word in a document relative to a collection of documents (corpus). This technique helps convert text data into numerical vectors for machine learning models.

### Steps:

- i. **Term Frequency (TF)** :- Measures how frequently a term appears in a document.

$$TF(t, d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d}$$

- ii. **Inverse Document Frequency (IDF)** :- Measures how important a term is. Terms appearing in many documents have lower IDF.

$$IDF(t) = \log \left( \frac{\text{Total number of documents}}{\text{Number of documents containing term } t} \right)$$

- iii. **TF-IDF** :- Combines TF and IDF for each term in the document.

$$TF-IDF(t, d) = TF(t, d) \times IDF(t)$$

- iv. **Vector Representation** :- Each document is represented as a vector where each element corresponds to the TF-IDF value of a term.

### Implementation Example:



- Use TfidfVectorizer from Python's scikit-learn library :

```
from sklearn.metrics.pairwise import cosine_similarity

# Example vectors
tfidf_matrix = vectorizer.fit_transform(documents)
similarity_matrix = cosine_similarity(tfidf_matrix[0:1], tfidf_matrix)
print(similarity_matrix)
```

---

## 2. Cosine Similarity :

Cosine Similarity measures the similarity between two vectors (e.g., a resume and a job description) based on the cosine of the angle between them.

### Formula:

$$\text{Cosine Similarity} = \cos(\theta) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \|\vec{B}\|}$$

Where:

- $\vec{A} \cdot \vec{B}$  = Dot product of vectors  $\vec{A}$  (resume) and  $\vec{B}$  (job description).
- $\|\vec{A}\|$  and  $\|\vec{B}\|$  = Magnitudes of  $\vec{A}$  and  $\vec{B}$ .

### Key Properties:

- Values range from -1 (opposite) to 1 (identical).
- A value of 0 indicates orthogonality (no similarity).

## 5. Workflow of the Proposed System :

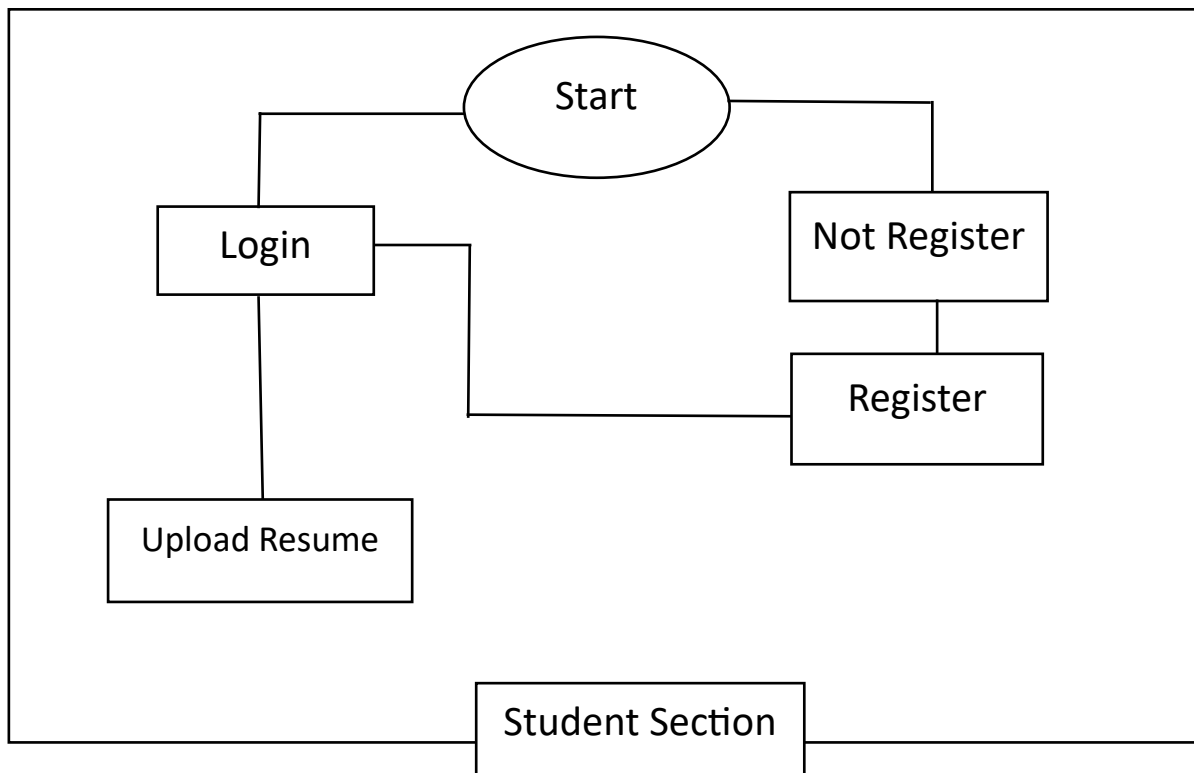


Fig 2 : Workflow of Student section

The workflow diagram shows the process for students interacting with the system:

1. Start: Users begin at the entry point.
2. Register/Not Registered: New users register; existing users proceed to log in.
3. Login: Users log in to access the system.
4. Upload Resume: Logged-in users upload their resumes.
5. Student Section: Users proceed to explore features like matching or other resources.

It's a simple flow focused on registration, login, and resume upload for students.

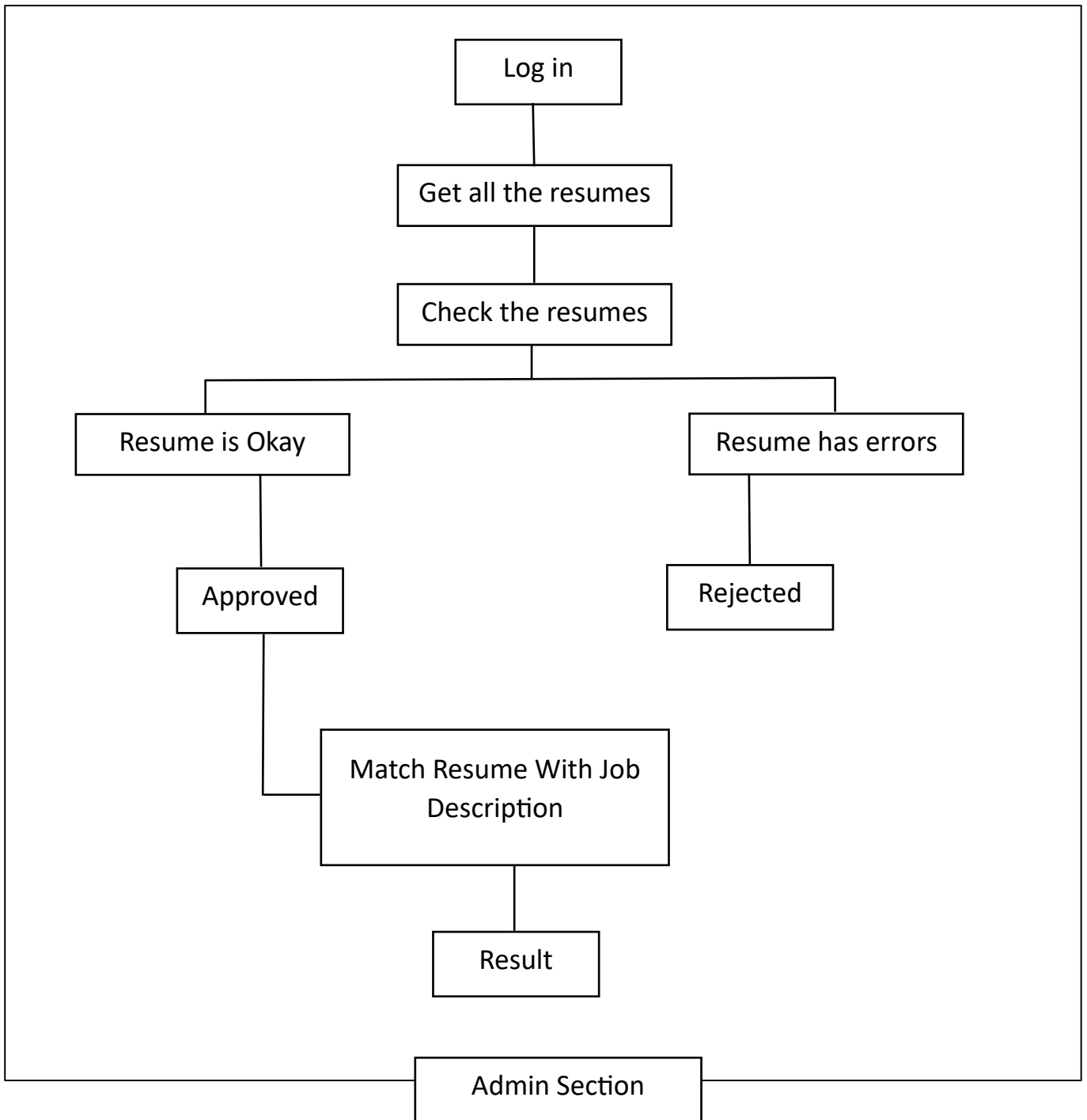


Fig 3 : Workflow of Admin Section

This workflow diagram outlines the process for resume evaluation and job matching:

1. **Log in:** Admin or authorized user logs in.
2. **Get all resumes:** System fetches submitted resumes.
3. **Check the resumes:** Resumes are evaluated for quality and errors.
  - **Resume is Okay:** Approved for further processing.

- **Resume has errors:** Rejected and flagged.
  - 4. **Match Resume with Job Description:** Approved resumes are matched with job descriptions.
  - 5. **Result:** Matching results are displayed or stored.
  - 6. **Admin Section:** Admin oversees and manages the entire process.
- It's a structured flow for resume validation and job description matching.
- 

## **6. Testing and Evaluation :**

The **testing and evaluation** phase ensures that the model meets the desired accuracy, reliability, and effectiveness in matching resumes with job descriptions. This phase involves quantitative and qualitative assessments.

### **1. Testing**

#### **Objective**

To validate the functionality of the resume screening system under various conditions.

#### **Process**

- i. **Test Dataset:**
  - Prepare a dataset of resumes and job descriptions with known relevance scores or labels (e.g., relevant, irrelevant).
  - Ensure diversity in test data to reflect real-world scenarios.
- ii. **Test Cases:**
  - **Functional Testing:** Ensure components like text preprocessing, vectorization, and similarity calculations work as intended.
  - **Integration Testing:** Verify end-to-end functionality of the system, from uploading resumes to generating matching scores.
  - **Boundary Testing:** Test edge cases such as empty resumes, extremely long text, or non-standard formats.

- **Performance Testing:** Assess the system's response time and scalability with a large number of resumes.
- iii. **Execution:**
  - Run the model with test data.
  - Log the outputs, including similarity scores and matched resumes.
- iv. **Results Validation:**
  - Compare generated outputs with expected results.

## 2. Evaluation

### Objective

To measure the system's performance using quantitative metrics that assess the accuracy, relevance, and reliability of resume matching and classification.

---

### Metrics

#### i. Precision

Measures the proportion of correctly matched resumes out of all resumes marked as relevant.

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

#### ii. Recall

Measures the proportion of correctly matched resumes out of all relevant resumes.

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

iii. **F1 Score**

The harmonic mean of precision and recall, balancing both metrics.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

iv. **Accuracy**

Measures the proportion of correctly classified resumes (both relevant and irrelevant).

$$\text{Accuracy} = \frac{\text{TP} + \text{True Negatives (TN)}}{\text{Total Samples}}$$

v. **Cosine Similarity Score**

Evaluates the semantic similarity between resumes and job descriptions.

Cosine Similarity Score = Average similarity score for correctly matched resumes.

### 3. Experimental Setup

- **Training-Testing Split:** Divide the dataset into training and testing sets (e.g., 80-20 split).
- **Baseline Comparison:**
  - Compare the model's performance with a simple baseline, such as keyword matching.
- **Cross-Validation:**
  - Perform k-fold cross-validation to ensure robust evaluation.

#### 4. Sample Results and Interpretation :

Metric	Values
Precision	<b>0.94</b>
Recall	<b>0.83</b>
F1 Score	<b>0.82</b>
Accuracy	<b>0.85</b>
Avg.cosine Similarity	<b>0.75</b>

The performance metrics of the resume matching system provide valuable insights into its efficiency and reliability in candidate selection.

**Precision (0.94)** means that when the system identifies a resume as relevant, it is highly likely to be truly relevant, effectively reducing false positives and ensuring that recruiters are not distracted by irrelevant candidates.

**Recall (0.83)** indicates that the system captures most of the relevant resumes from the entire pool, minimizing false negatives and ensuring that qualified candidates are not overlooked.

**F1 Score (0.82)** is the harmonic mean of precision and recall, offering a balanced measure that reflects the system's ability to achieve both high precision and recall, making it a reliable tool for accurate candidate matching.

**Accuracy (0.85)** reveals that the system performs with a high degree of correctness in categorizing resumes, correctly identifying candidates in the majority of cases.

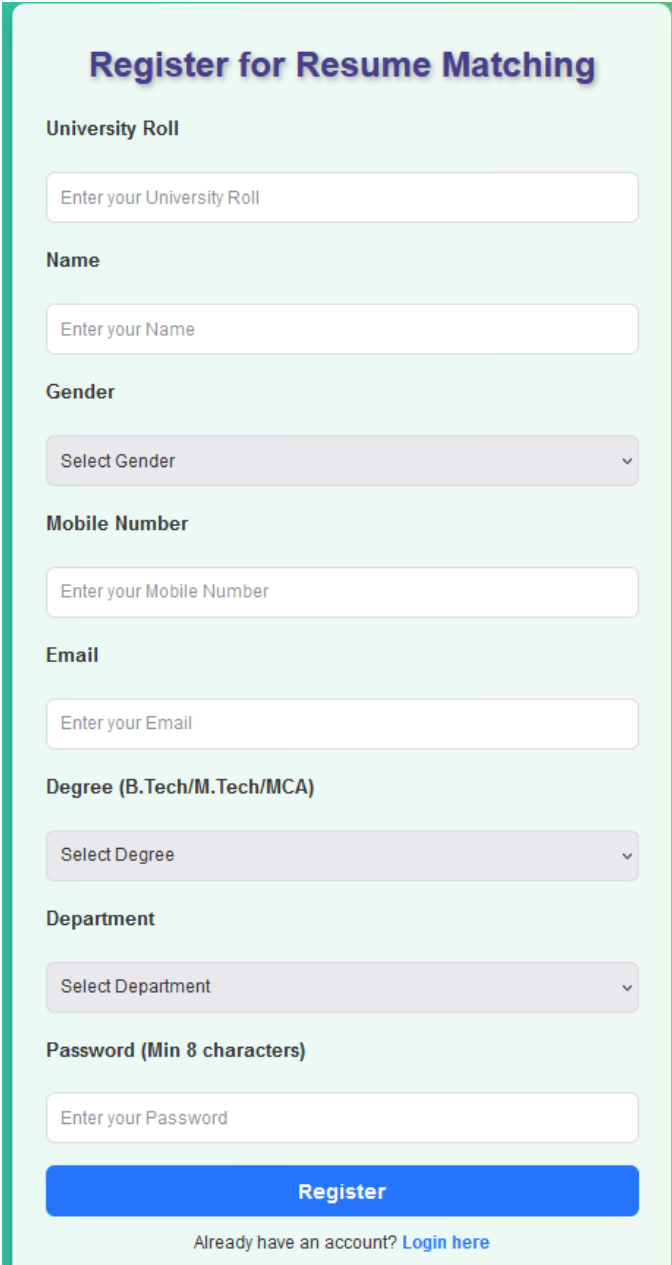
Finally, **Average Cosine Similarity (0.75)** shows that the system is able to effectively compare resumes and job descriptions based on semantic similarity, suggesting a strong alignment between the two, which is critical for accurate and context-aware resume matching.

These metrics collectively highlight the system's ability to deliver highly relevant, reliable, and efficient candidate selections while minimizing errors in the recruitment process.

## V: EXPERIMENTAL RESULT & ANALYSIS

### 1. Input ,Output Results :

- Register Page :



The image shows a registration form titled "Register for Resume Matching". The form is set against a light green background with a teal border. It contains several input fields and dropdown menus for user registration. The fields are: University Roll (text input), Name (text input), Gender (dropdown menu), Mobile Number (text input), Email (text input), Degree (B.Tech/M.Tech/MCA) (dropdown menu), Department (dropdown menu), and Password (Min 8 characters) (text input). A blue "Register" button is at the bottom, followed by a link "Already have an account? Login here".

**Register for Resume Matching**

University Roll

Enter your University Roll

Name

Enter your Name

Gender

Select Gender

Mobile Number

Enter your Mobile Number

Email

Enter your Email

Degree (B.Tech/M.Tech/MCA)

Select Degree

Department

Select Department

Password (Min 8 characters)

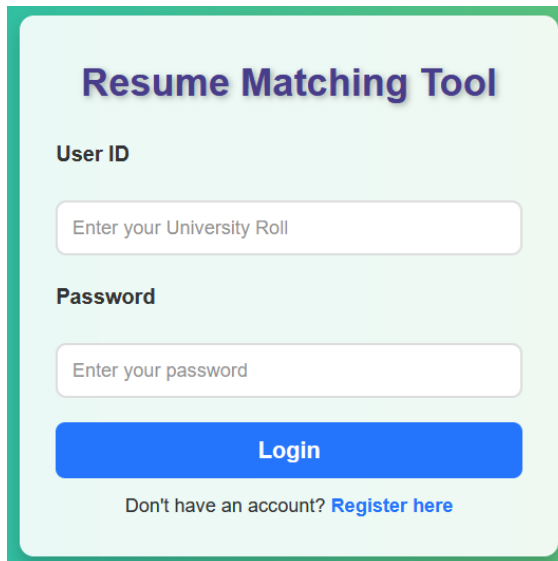
Enter your Password

**Register**

Already have an account? [Login here](#)

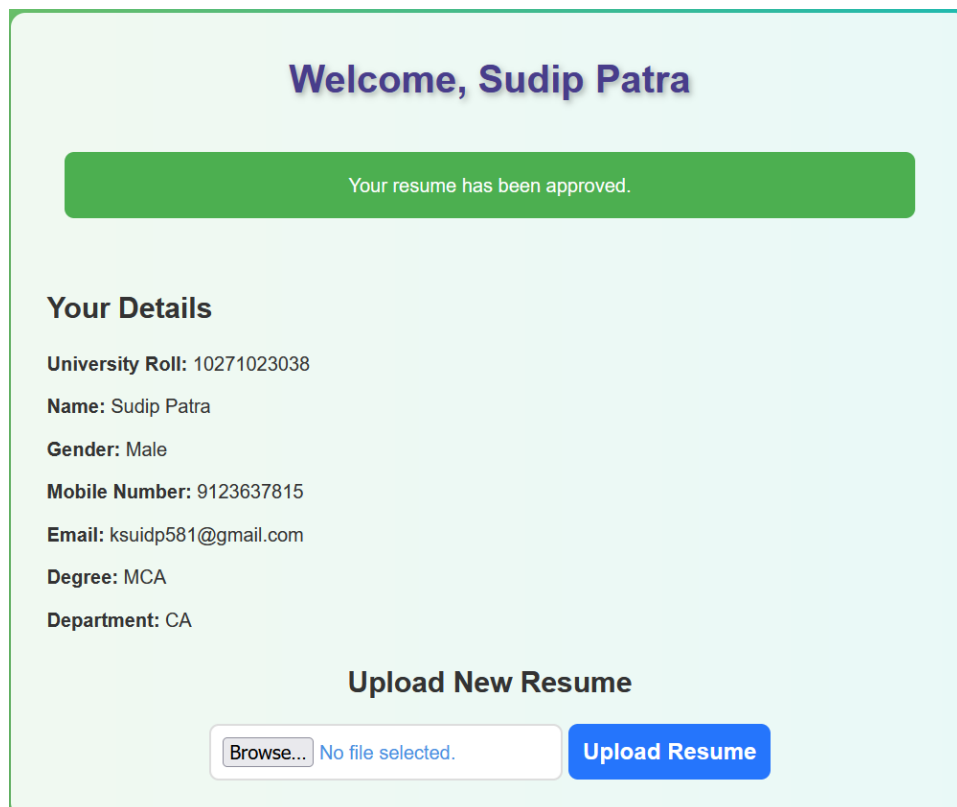


- Login Page :



The login page features a light green background with a darker green border. At the top, the title "Resume Matching Tool" is displayed in a bold, dark blue font. Below the title, the "User ID" section includes a text input field with the placeholder "Enter your University Roll". The "Password" section includes a text input field with the placeholder "Enter your password". A prominent blue "Login" button is positioned below the password field. At the bottom, a link "Don't have an account? Register here" is provided in a smaller, blue font.

- After Login Interface :



The post-login interface has a light green background with a darker green border. At the top, a personalized welcome message "Welcome, Sudip Patra" is shown in a bold, dark blue font. Below this, a green notification bar contains the text "Your resume has been approved." in white. The "Your Details" section lists the user's information: "University Roll: 10271023038", "Name: Sudip Patra", "Gender: Male", "Mobile Number: 9123637815", "Email: ksuidp581@gmail.com", "Degree: MCA", and "Department: CA". At the bottom, the "Upload New Resume" section includes a "Browse..." button, a status indicator "No file selected.", and a blue "Upload Resume" button.

• Job Description Interface :

## Resume Matching Tool

**Job Description:**

Java Python HTML CSS JavaScript

**Number of Resumes to Match:**

**Select Resumes:**

- ☒ Select All
- ☒ Aditi Mondal
- ☒ Aditya Mallick
- ☒ Aishi Biswas
- ☒ Ajay Gupta
- ☒ Akash Singh
- ☒ Arnab Kumar Ghosh
- ☒ Ashesh Karmakar
- ☒ Avirup Sett
- ☒ Ayush das
- ☒ Biswajit Malakar
- ☒ Debanjan Ghosh
- ☒ Keya Mitra
- ☒ Madhusudan Chand

• Matching Result :

Matching Results		
Student Name	View	Matching Score
Rakib Mondal	<a href="#">View Resume</a>	21.37
Souradeep Maitra	<a href="#">View Resume</a>	18.71
Keya Mitra	<a href="#">View Resume</a>	18.34
Nasim Islam	<a href="#">View Resume</a>	16.01
Souvik Dey	<a href="#">View Resume</a>	15.92
Rajendra Prasad	<a href="#">View Resume</a>	15.39
Sulagna Kundu	<a href="#">View Resume</a>	15.27
Satyanand Thapa	<a href="#">View Resume</a>	14.42
Souptik Nag	<a href="#">View Resume</a>	13.97
Sima mondai	<a href="#">View Resume</a>	13.24

## 2. Analysis Of System :

### Confusion Matrix -1

**Job Description** : Node Js , React , Mongo DB

Predicted	Actual	
	36	
	Positive	Negative
Positive	TP	FP
	10	1
Negative	FN	TN
	2	23

$$\text{Accuracy} = (10 + 23) / 36 = 0.91$$

$$\text{Precision} = (\text{TP}) / (\text{TP} + \text{FP}) = 10 / 11 = 0.90$$

$$\text{Recall} = (\text{TP}) / (\text{TP} + \text{FN}) = 10 / 12 = 0.83$$

$$\text{F1 Score} = 2 * \{(\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})\} = 86.36$$

### Confusion Matrix -2

**Job Description** : Python , Django , SQL

Predicted	Actual	
	36	
	Positive	Negative
Positive	TP	FP
	21	2
Negative	FN	TN
	5	8

$$\text{Accuracy} = (21 + 8) / 36 = 0.80$$

$$\text{Precision} = (\text{TP}) / (\text{TP} + \text{FP}) = 21 / 23 = 0.91$$

$$\text{Recall} = (\text{TP}) / (\text{TP} + \text{FN}) = 21 / 26 = 0.80$$

$$\text{F1 Score} = 2 * \{(\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})\} = 85.15$$

### Confusion Matrix -3

**Job Description :** Java , SQL , Spring Bot

		Predicted	
		36	
		Positive	Negative
Actual	Positive	TP 21	FP 2
	Negative	FN 5	TN 8

$$\text{Accuracy} = (28 + 2) / 37 = 0.81$$

$$\text{Precision} = (\text{TP}) / (\text{TP} + \text{FP}) = 28 / 28 = 1$$

$$\text{Recall} = (\text{TP}) / (\text{TP} + \text{FN}) = 28 / 35 = 0.80$$

$$\text{F1 Score} = 2 * \{(\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})\} = 88.88$$

### Confusion Matrix - 4

**Job Description :** HTML , CSS , Java Script

		Predicted	
		36	
		Positive	Negative
Actual	Positive	TP 21	FP 2
	Negative	FN 5	TN 8

$$\text{Accuracy} = (27 + 6) / 37 = 0.89$$

$$\text{Precision} = (\text{TP}) / (\text{TP} + \text{FP}) = 0.96$$

$$\text{Recall} = (\text{TP}) / (\text{TP} + \text{FN}) = 0.90$$

$$\text{F1 Score} = 2 * \{(\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})\} = 92.90$$

### Final Value:

	CM-1	CM-2	CM-3	CM-4	FINAL
Accuracy	0.91	0.80	0.81	0.89	0.85
Precision	0.90	0.91	1	0.96	0.94
Recall	0.83	0.80	0.80	0.90	0.83
F1 Score	86.36	85.15	88.88	92.90	88.32

$$\text{Final Accuracy} = 0.85$$

$$\text{Final Precision} = 0.94$$

$$\text{Final Recall} = 0.83$$

$$\text{Final F1 Score} = 88.32$$

## VI : FUTURE WORK

### 1. Enhanced Matching Algorithms

- Implement advanced machine learning models like **BERT or GPT** for better semantic understanding of resumes and job descriptions. Use **context-based similarity techniques** to improve accuracy in matching candidates with job roles.

### 2. Integration with External Platforms

- Connect with **job portals (LinkedIn, Indeed)** to automate job description retrieval and resume submissions. Integrate with **educational platforms** to verify candidate credentials and certifications.

### 3. Multi-Language Support

- Enable support for resumes and job descriptions in **multiple languages**, expanding the platform's global usability. Use **translation models** to facilitate cross-language job matching.

### 4. AI-Powered Career Guidance

- Use AI to suggest career paths, certifications, and courses based on resume content and industry trends. Provide personalized skill development recommendations to improve job readiness.

### 5. Mobile Application

- Develop a mobile-friendly version to allow users to upload resumes, check matching results, and receive notifications on the go. Enhance accessibility and engagement with a user-friendly mobile interface.

### 6. Improved Data Security and Privacy

- Implement advanced encryption techniques to protect sensitive user data. Ensure compliance with GDPR, CCPA, and other data protection regulations to build trust with users and employers.

By focusing on these key areas, the resume matching system can become more efficient, secure, and globally accessible, making job matching and career development seamless for users.

## **VII: CONCLUSION**

This software serves as an innovative and efficient platform for simplifying the process of resume evaluation and job matching. By leveraging advanced natural language processing techniques, such as TF-IDF vectorization and cosine similarity, it ensures precise and accurate alignment of candidates' resumes with job descriptions. The software not only streamlines the hiring process for recruiters but also empowers students and job seekers to refine their resumes, identify skill gaps, and enhance their employability. Its user-friendly interface and well-structured workflow make it accessible and effective for both students and administrators. Furthermore, the inclusion of features like error detection in resumes, admin oversight, and detailed feedback ensures a comprehensive solution that benefits all stakeholders involved. This system has the potential to revolutionize the traditional resume evaluation process by making it faster, more reliable, and data-driven. In conclusion, this software bridges the gap between students' skills and job requirements, acting as a powerful tool for career development and recruitment. With continuous enhancements and the integration of advanced features in the future, it can become a cornerstone in the domain of automated recruitment and career guidance systems.

## VII : REFERENCES

### i. Research Papers:

- Salton, G., & McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill.
- Ramos, J. (2003). Using TF-IDF to Determine Word Relevance in Document Queries. *Proceedings of the First International Conference on Machine Learning*.
- Singhal, A. (2001). Modern Information Retrieval: A Brief Overview. *IEEE Data Engineering Bulletin*.
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.
- Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*.
- Wang, P., Qian, L., & Kuang, H. (2018). Recruitment Prediction Using Machine Learning Algorithms. *Journal of Human Resource Management Research*.
- Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*.
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.
- Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media.

### ii. Dataset:

- College IMS for resume dataset.
- Online Job description.



## Plagiarism Report

Page No.	No. of Words	Plagiarism	Unique
1-4	876	0%	100%
5-7	791	11%	89%
8-11	976	0%	100%
12-16	797	2%	98%
16-20	619	8%	92%
21-26	639	0%	100%
27-31	150	14%	86%
32-35	876	9%	91%

Total No. of Words: 5724

Total No. of Plagiarized Words: 250 [Approx]

Total Plagiarism in Percentage: 4.36%

Plagiarism Checked From: <https://smallseotools.com/plagiarism-checker/>

