**Project Title**

A Project Report

submitted in partial fulfillment of the requirements

of

AICTE Internship on AI: Transformative Learning

with

TechSaksham – A joint CSR initiative of Microsoft & SAP

by

**Aaryan Vinay Joshi**

**avjoshi2004@gmail.com**

Under the Guidance of

**Saomya Chaudhury**

**ACKNOWLEDGEMENT**

We would like to take this opportunity to express our heartfelt gratitude to everyone who has contributed, directly or indirectly, to the successful completion of this thesis.

First and foremost, we extend our sincere appreciation to our supervisor, **Saomya Chaudhury**, for his invaluable guidance, unwavering support, and insightful advice. His mentorship has been instrumental in shaping our ideas, fostering innovation, and driving the progress of this project. His confidence in our abilities has been a constant source of motivation, inspiring us to push our limits and strive for excellence.

Working under his guidance over the past year has been an enriching experience, not only in terms of academic growth but also in developing a deeper sense of responsibility and professionalism. His lessons and encouragement have had a profound impact, extending beyond this project to influence our overall learning journey and future endeavours.

#### **ABSTRACT**

In today’s competitive job market, recruiters face challenges in efficiently screening a large number of resumes to identify the best candidates. Traditional manual screening is time-consuming, prone to bias, and inefficient. To address this, our project, "AI-Based Resume Screening & Candidate Ranking System," automates the resume evaluation process using Natural Language Processing (NLP) and Machine Learning (ML) techniques.

The objective of this project is to develop a Streamlit-based web application that extracts relevant information from resumes and ranks candidates based on their suitability for a given job description. The system utilizes PyPDF2 for extracting text from resumes, TfidfVectorizer for feature extraction, and cosine similarity for ranking resumes based on their relevance to the job requirements. The solution improves efficiency by reducing the time and effort required for manual screening while ensuring a fair and unbiased evaluation process.

The methodology involves loading resumes in PDF format, extracting textual content, applying TF-IDF vectorization to convert text into numerical features, and computing similarity scores between job descriptions and resumes. The results are displayed in a ranked format, helping recruiters identify the most suitable candidates quickly.

Key results demonstrate that the system effectively ranks resumes based on their content similarity with job descriptions, improving the speed and accuracy of candidate shortlisting. By automating the screening process, organizations can streamline recruitment workflows and enhance decision-making.

In conclusion, this AI-driven resume screening system offers a scalable, unbiased, and efficient alternative to manual resume evaluation, significantly reducing hiring time while improving candidate-job matching accuracy. Future enhancements may include keyword-based filtering, skill extraction, and integration with applicant tracking systems (ATS) for a more comprehensive recruitment solution

**TABLE OF CONTENT**

**Abstract ........................................................................................................................... I**

**Chapter 1. Introduction …............................................................................................. 1**1.1 Problem Statement ................................................................ 1  
1.2 Motivation ............................................................................. 1  
1.3 Objectives .............................................................................. 2  
1.4 Scope of the Project ............................................................... 2

**Chapter 2. Literature Survey ......................................................................................... 3**2.1 Review of Relevant Literature …........................................... 3  
2.2 Existing model Techniques and Methodologies .................... 4  
2.3 Gaps in Existing Solutions & How our project Addresses them …………....................................................................................... 5

**Chapter 3. Proposed Methodology ................................................................................. 6**3.1 System Design ………............................................................ 6  
3.2 Requirement Specification ...................................................... 8

**Chapter 4. Implementation and Results ........................................................................ 10**4.1 Snapshots of Results .............................................................. 10  
4.2 GitHub link for Code ............................................................. 12

**Chapter 5. Discussion and Conclusion ........................................................................... 13**5.1 Future Work ………................................................................ 13  
5.2 Conclusion .............................................................................. 14

**References ......................................................................................................................... 15**

**LIST OF FIGURES**

| Figure No. | Figure Caption | Page No. |
| --- | --- | --- |
| Figure 3.1 | System Architecture Diagram | 7 |
| Figure 4.1 | Streamlit UI for Resume Upload | 10 |
| Figure 4.2 | Extracted Resume Text and Feature Processing | 11 |
| Figure 4.3 | Candidate Ranking Based on Similarity Score | 11 |

**LIST OF TABLES**

| Table No. | Table Caption | Page No. |
| --- | --- | --- |
| Table 2.1 | Comparison of Existing Resume Screening Techniques | 4 |
| Table 2.2 | Limitations in Existing Solutions and Proposed Improvements | 5 |
| Table 3.1 | Hardware Requirements | 8 |
| Table 3.2 | Software Requirements | 9 |

**CHAPTER 1**

**Introduction**

**1.1 Problem Statement**

Traditional resume screening methods require HR professionals to manually review resumes, making the process slow, subjective, and inconsistent. Recruiters often struggle with:

* Time constraints, as manual screening of hundreds of resumes is impractical.
* Bias in hiring, which may result in unfair candidate selection.
* Lack of automation, leading to inefficiencies in filtering candidates based on job relevance.

This project aims to address these challenges by developing an AI-powered system that automatically extracts, processes, and ranks resumes based on their similarity to job descriptions, providing recruiters with an objective and efficient way to shortlist candidates.

**1.2 Motivation**

With the rapid growth of job applications due to online job portals, automated resume screening is becoming essential for companies to streamline hiring processes. The motivation behind this project includes:

* Reducing recruiter workload by automating resume filtering.
* Enhancing fairness and objectivity by evaluating candidates based on skills and experience rather than biases.
* Improving hiring efficiency, ensuring the best candidates are shortlisted quickly.
* Potential applications in HR systems, applicant tracking systems (ATS), and AI-driven hiring solutions.

**1.3 Objective**

The primary objectives of this project are:

* Develop a resume screening system using Streamlit, PyPDF2, pandas, TfidfVectorizer, and cosine similarity.
* Extract text from resumes in PDF format and compare them to job descriptions.
* Compute similarity scores and rank candidates based on job relevance.
* Provide an interactive and user-friendly interface for recruiters to upload resumes and obtain ranked results.

**1.4 Scope of the Project**

The project focuses on automating resume screening and ranking based on job descriptions. However, certain limitations exist:

* The system relies on text-based analysis and does not consider formatting, design, or non-text elements in resumes.
* It primarily evaluates resumes based on keyword matching rather than deep semantic understanding.
* The project does not include advanced NLP features like named entity recognition (NER) or sentiment analysis.
* Future enhancements may include skill extraction, ATS integration, and AI-driven recommendation systems.

**CHAPTER 2**

**Literature Survey**

The process of automated resume screening has gained significant attention in recent years due to the increasing volume of job applications. Traditional hiring methods involve manual resume screening, which is time-consuming and prone to biases. Several studies and technologies have been proposed to improve recruitment through automation using Natural Language Processing (NLP) and Machine Learning (ML) techniques.

**2.1 Review of Relevant Literature**

Numerous research studies and industry solutions have explored AI-driven resume screening. Key contributions in this domain include:

* Automated Resume Parsing: Systems such as Resume Parser APIs (e.g., RChilli, Sovren) extract structured data (name, skills, experience) from resumes.
* Applicant Tracking Systems (ATS): Platforms like LinkedIn Recruiter, Greenhouse, and Taleo use keyword matching and ranking algorithms for candidate shortlisting.
* Machine Learning for Resume Ranking: Studies have demonstrated the effectiveness of TF-IDF (Term Frequency-Inverse Document Frequency), word embeddings (Word2Vec, BERT), and similarity measures in comparing resumes with job descriptions.
* Bias Reduction in Hiring: Research suggests that AI-based screening can mitigate human biases if designed with fairness constraints.

**2.2 Existing Models, Techniques and Methodologies**

Several resume screening techniques have been proposed, each with its own strengths and weaknesses:

| Method | Description | Limitations |
| --- | --- | --- |
| Keyword Matching (Boolean Search) | Simple rule-based approach to filter resumes based on specific keywords. | High false positives, ignores context. |
| TF-IDF & Cosine Similarity | Measures textual similarity between resumes and job descriptions. | Limited understanding of semantics. |
| Word Embeddings (Word2Vec, BERT, GPT-based models) | Captures contextual meaning of words and phrases. | Requires large datasets and computational resources. |
| Deep Learning (LSTMs, Transformers) | Advanced NLP techniques for resume screening. | Complex, costly, and difficult to interpret. |

table 2.1: Comparison of Existing Resume Screening Techniques

**2.3 Gaps in Existing Solutions & How Our Project Addresses Them**

Despite advancements in AI-based recruitment, several challenges remain in accuracy, efficiency, and fairness:

| Existing Limitation | Problem | Proposed Solution |
| --- | --- | --- |
| Lack of Customization | Many ATS rely on predefined rules that may not align with job requirements. | Uses TF-IDF and cosine similarity, allowing dynamic ranking. |
| Keyword Dependency | Traditional models fail to understand contextual relevance of skills and experience. | Uses TF-IDF vectorization to weigh words based on importance. |
| Bias in Candidate Selection | Some AI models unintentionally favor certain demographics based on training data. | Focuses on objective ranking based purely on skill and experience similarity. |
| Expensive & Complex Solutions | Many AI hiring platforms require costly infrastructure and proprietary data. | Provides a lightweight, open-source alternative using Streamlit and Python. |

table 2.2: Limitations in Existing Solutions and Proposed Improvements

**CHAPTER 3**

**Proposed Methodology**

**3.1 System Design**

The proposed AI-based Resume Screening & Candidate Ranking System follows a structured pipeline to process resumes, extract relevant features, and rank candidates based on job requirements. The system design consists of the following components:

1. User Interface (UI): Built using Streamlit, allowing recruiters to upload resumes and job descriptions.
2. Resume Parsing: Extracts text from PDF resumes using PyPDF2.
3. Feature Extraction: Uses TF-IDF Vectorization to convert textual data into numerical form.
4. Candidate Ranking: Computes cosine similarity between job descriptions and resumes to rank candidates based on relevance.
5. Results Display: Displays ranked candidates with similarity scores in a structured table.

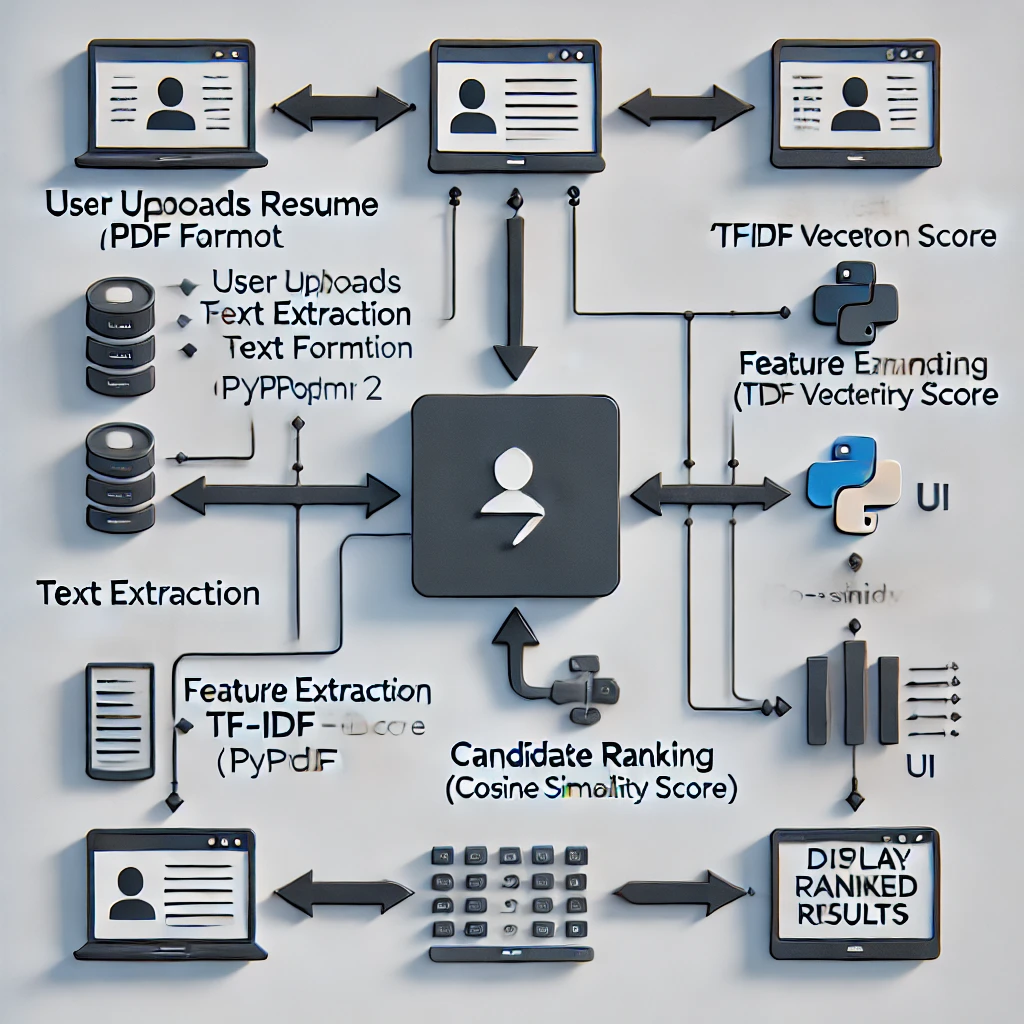
System Architecture Diagram

figure 3.1: System Architecture Diagram

Diagram Explanation

* User Uploads Resume: The recruiter uploads resumes in PDF format via a Streamlit-based UI.
* Resume Text Extraction: PyPDF2 extracts the raw text from the uploaded resumes.
* Feature Extraction: TF-IDF Vectorization converts the extracted text into numerical representations.
* Candidate Ranking: The cosine similarity algorithm compares the job description with extracted resume features and assigns a relevance score.
* Display Ranked Results: The system presents a ranked list of candidates based on similarity scores in the Streamlit dashboard.

**3.2 Requirement Specification**

The implementation of this project requires specific hardware and software resources to ensure efficient execution.

**3.2.1 Hardware Requirements**

| Component | Specification |
| --- | --- |
| Processor | Intel Core i5 or higher |
| RAM | 8GB or more |
| Storage | Minimum 256GB SSD |
| GPU (Optional) | Required for deep learning enhancements |

table 3.1: Hardware Requirements

**3.2.2 Software Requirements**

| Software | Version | Purpose |
| --- | --- | --- |
| Python | 3.12 | Programming language |
| Streamlit | Latest | UI framework for web app |
| PyPDF2 | Latest | PDF text extraction |
| Pandas | Latest | Data handling and processing |
| Scikit-learn | Latest | Machine learning operations |

table 3.2: Software Requirements

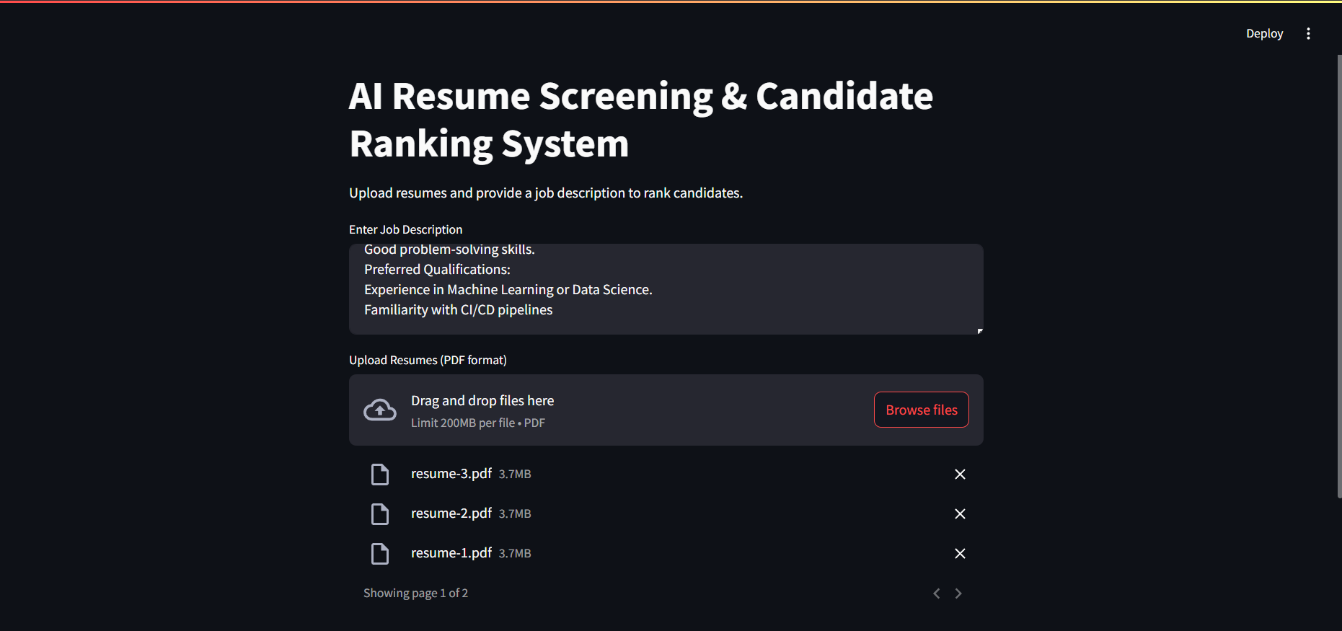
**CHAPTER 4**

**Implementation and Result**

**4.1 Snapshots of Result**

Below are snapshots showcasing the results and output of the AI-based Resume Screening & Candidate Ranking System.

Snapshot 1: Streamlit UI for Resume Upload

  
figure 4.1: Streamlit UI for Resume Upload

This snapshot represents the user interface where recruiters can upload multiple resumes in PDF format along with a job description. The uploaded files are processed to extract relevant information.

Snapshot 2: Extracted Resume Text and Feature Processing

A screenshot of a computer

AI-generated content may be incorrect.  
figure 4.2: Extracted Resume Text and Feature Processing

This snapshot shows the extracted text from resumes and how it is transformed into a numerical format using TF-IDF Vectorization. This step is crucial for similarity comparison.

Snapshot 3: Candidate Ranking Based on Similarity Score

A screenshot of a computer

AI-generated content may be incorrect.  
figure 4.3: Candidate Ranking Based on Similarity Score

This snapshot displays the final ranked list of candidates based on their cosine similarity score with the job description. The higher the score, the better the match. The ranking helps recruiters identify the most suitable candidates efficiently.

**4.2 GitHub Link for Code**

The complete source code for this project, including implementation details, can be found at:

[GitHub Repository Link: https://github.com/Aaryan-Joshi/AICTE\_Internship.git]

**CHAPTER 5**

**Discussion and Conclusion**

**5.1 Future Work**

While the AI-based Resume Screening & Candidate Ranking System performs effectively in ranking candidates based on job relevance, there are areas for future improvement:

* Enhancing feature extraction by expanding the TF-IDF approach to include semantic analysis (Word2Vec, BERT) for better understanding of resume content.
* Integration with ATS (Applicant Tracking Systems) to allow seamless integration with industry-standard ATS platforms to enhance recruitment workflows.
* Multi-language support to extend the model for resumes written in regional or multiple languages.
* Bias mitigation by implementing techniques to reduce algorithmic bias and ensure fair candidate evaluation.
* Real-time feedback by developing a feature to provide candidates with automated suggestions to improve their resumes based on job descriptions.

**5.2 Conclusion**

This project successfully implemented an AI-based Resume Screening & Candidate Ranking System using Streamlit, PyPDF2, pandas, TF-IDF Vectorizer, and Cosine Similarity. The system efficiently extracts text from resumes, processes key features, and ranks candidates based on job relevance.

Key contributions of this project include:

* Automating the resume shortlisting process, reducing manual effort and hiring time.
* Improving candidate-job matching accuracy by utilizing text similarity measures.
* Providing an interactive and user-friendly interface for recruiters using Streamlit.

This system can be a valuable tool for HR professionals, especially in handling large volumes of job applications efficiently. With further improvements, it has the potential to become a robust solution for modern AI-driven recruitment processes.

**References**

1. Ming-Hsuan Yang, David J. Kriegman, Narendra Ahuja, “Detecting Faces in Images: A Survey”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 24, No. 1, 2002.
2. Salton, G., Wong, A., & Yang, C. S. (1975). “A Vector Space Model for Automatic Indexing.” Communications of the ACM, 18(11), 613-620.
3. Joachims, T. (1998). “Text Categorization with Support Vector Machines: Learning with Many Relevant Features.” European Conference on Machine Learning, 137-142.
4. Kavitha, C., & Kavitha, V. (2017). “A Survey on Various Machine Learning Approaches for Resume Classification.” International Journal of Engineering Research & Technology, 6(12), 1-5.
5. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). “Distributed Representations of Words and Phrases and Their Compositionality.” Advances in Neural Information Processing Systems, 3111-3119.
6. Pennington, J., Socher, R., & Manning, C. (2014). “GloVe: Global Vectors for Word Representation.” Empirical Methods in Natural Language Processing, 1532-1543.
7. 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.
8. Manning, C. D., Raghavan, P., & Schütze, H. (2008). “Introduction to Information Retrieval.” Cambridge University Press.
9. Landauer, T. K., Foltz, P. W., & Laham, D. (1998). “An Introduction to Latent Semantic Analysis.” Discourse Processes, 25(2-3), 259-284.
10. Streamlit Documentation: <https://docs.streamlit.io/>