



**Documentation Project**

**AI Resume Ranker**

Team: TensorX

Category: AI And Machine Learning

**Table of Content**

Problem definition:

* Introduction.
* Proposed solution.

Design specifications:

* Project purpose.
* Project scope.
* Constraints.
* Functional Requirements.
* Non-Functional Requirements.

Diagrams:

* Interface Diagram.
* User Flow Diagram.

Test Data Used in the Project.

Proper Steps to execute the project

Project Installation Instructions.

Links.

Problem definition:

Introduction

**In today's ever-increasing number of job applicants and the competition for jobs, hiring managers face a huge number of resumes for each vacancy. Traditional methods of manually screening resumes are time-consuming, labor-intensive, and prone to human error, resulting in significant cost and costly human and financial losses. As we enter the age of technology and artificial intelligence, the "AI Resume Ranker " system was designed to help hiring managers and human resources managers select and screen the best candidates for jobs based on specific criteria.**

Proposed solution

**The "** **AI Resume Ranker " system was designed to address the problem of sorting and selecting the best talent for jobs according to specific criteria, using artificial intelligence and machine learning algorithms. The system receives up to 500 CVs, a job description, and key information. The system analyzes, sorts, and displays the ten best CVs that match the entered description in less than five seconds. This high performance sets our system apart from its peers in the field, helping hiring managers and HR professionals select the best talent.**

Design specifications:

Project purpose

**The " AI Resume Ranker " form was designed to facilitate the process of screening resumes and selecting the best candidates. Hiring managers and human resources staff are always looking to select the best candidates for the job. This allows them to enter a description of the desired position and specify key information such as education, practical skills, experience, and certificates. The form then displays the ten best candidates for the job.**

Project scope

**This project includes a simple interface for non-technical HR staff to use, allowing them to upload CV files in (PDF, Word) formats, up to a maximum of 500 files, and also allowing them to enter the job description for the vacant position in (PDF, txt) formats. The form displays the results of ten candidates, and the results can be exported in (Excel, PDF) formats, and test whether the job has been accepted or not.**

Constraints

* **The system relies heavily on the quality of the job description entered. If the description is vague or inaccurate, it can affect the accuracy of the classification results.**
* **The system currently supports only English. This means it cannot process resumes written in other languages.**
* **The system is designed to work efficiently with well-organized PDF and DOCX files. You may encounter difficulty processing files containing non-standard formats or text images (non-copiable text).**
* **Sensitive personal data contained in resumes must be handled securely. The importance of designing the system to minimize any biases that may appear during the classification process must also be considered.**

Functional Requirements.

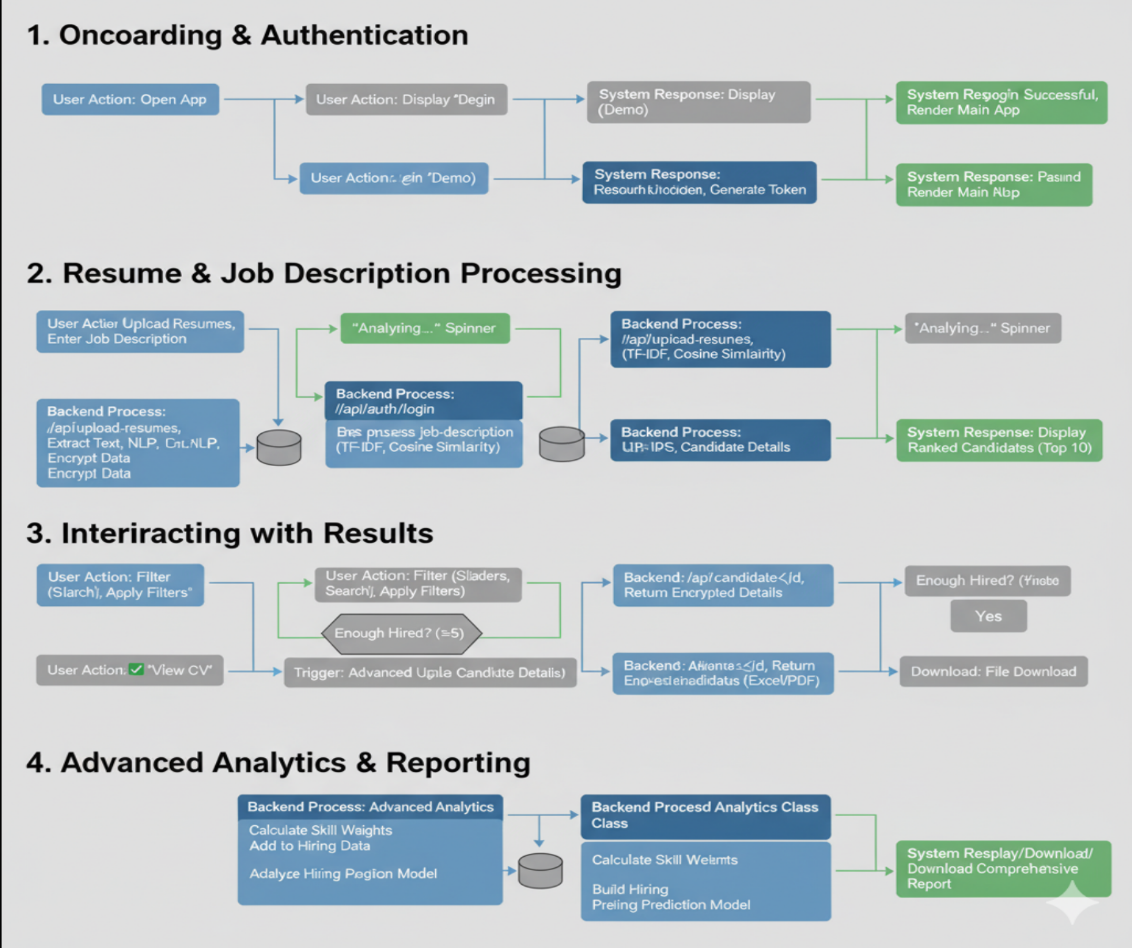
* **File Uploading: The system must allow users to upload resume files, either individually or in batches (several files at once).**
* **Job Description Entry: The system must provide a means to enter or upload the job description for the desired position.**
* **Automated Classification and Ranking: The system automatically ranks and ranks candidates based on how well their resumes match the job description.**
* **Viewing and Exporting Results: The system allows viewing selected resumes and exporting a list of results in common formats such as Excel and PDF.**
* **User Interface: The system must have a simple and easy-to-understand user interface for non-technical users.**
* **Search and Filtering: The system allows users to search and filter based on criteria such as experience, skills, and academic degrees.**
* **Model Improvement: The system must include a Feedback Loop mechanism to continuously retrain and improve the model.**

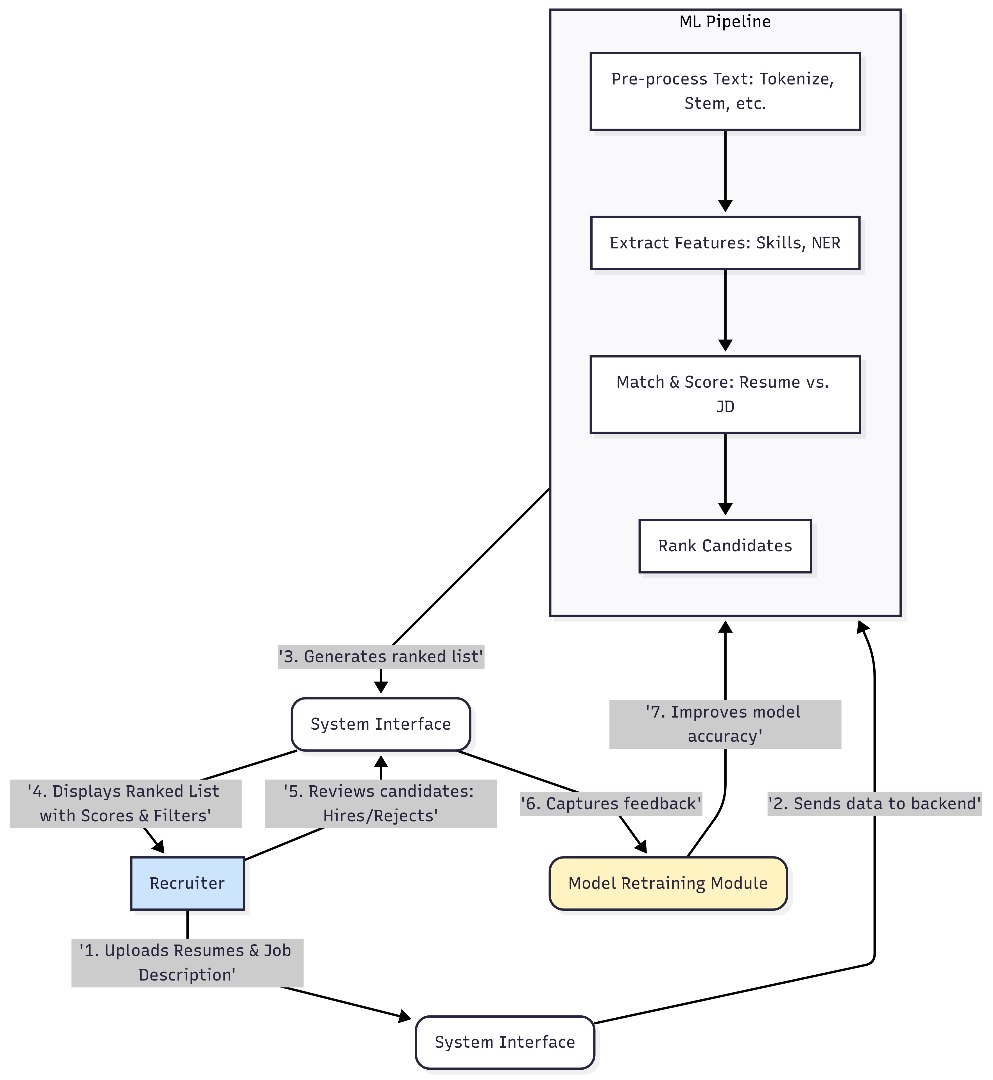
Non-Functional Requirements

* **Performance: The system should be fast, sorting resumes in record time for each job description.**
* **Scalability: The system must support processing up to 500 resumes in a batch without compromising performance.**
* **Usability: The interface must be designed to ensure ease of use for HR administrators.**
* **Security: Sensitive candidate data must be stored and processed encrypted and securely to ensure privacy.**
* **Compatibility: The system must support PDF and DOCX files and be able to run seamlessly across different browsers and devices.**

Diagrams:

Dialog Flow:

******

******

System Architecture

Test Data Used in the Project

**To ensure the effectiveness of the AI ​​Resume Ranker system, a diverse dataset was used for testing. This dataset consists of 228 resumes in .docx format, covering a range of professional roles, such as full-stack developers, data engineers, and analysts.**

**The cases are arranged in a "test case" format for each position. Test Cases:**

**1. Exact Match Test**

**Description: The system's ability to identify a candidate whose skills and experience fully match the job description was tested.**

**Input:**

**Resumes: A set of 10 resumes, one of which includes all the essential skills in the job description.**

**Job Description: Software Developer with experience in Python, Machine Learning, and SQL.**

**Result: A resume system matching the first format should perform 95% of the job searches.**

**2. Multiple Match (Partial Match) Test**

**Description: The system's ability to classify all those who possess only a portion of the core skills was tested.**

**Input:**

**Resumes: A set of 5 resumes containing some, but not all, of the core skills.**

**Job Description: Data Analyst with experience in SQL, Tableau, and R.**

**Result: The system should accurately rank the number of shared skills, with an accuracy of less than 80%.**

**3. Test for Handling Missing or Unstructured Data (Incomplete/Unstructured Data)**

**Description: The system's ability to handle resumes containing incomplete formats or missing information was tested.**

**Input:**

**Resumes: An unstructured docx resume (such as using tables or images other than text).**

**Job Description: Project Manager with experience in Agile, Scrum, and Project Management.**

**Outcome: The system should be able to extract and classify the essential information, although the matching score may be lower.**

**4. Irrelevant Data Testing**

**Description: Test the system to filter out resumes that are irrelevant to the job description.**

**Input:**

**Resumes: Resumes of candidates in a completely different field (such as a doctor or a lawyer).**

**Job Description: Network Engineer with experience in CCNA and Cisco.**

**Outcome: The system should make very minor adjustments to these resumes or perform precise sizing.**

**These cases demonstrate that the system is capable of handling a variety of resume screening scenarios, ensuring the efficiency and accuracy of the hiring process.**

Project Installation Instructions.

* **Access the project:**

**Click the GitHub link: https://github.com/YourProjectName/AI-Resume-Ranker.git.**

* **Open the project in your working environment:**

**After accessing the GitHub repository, download the project files.**

**Open a command prompt or terminal in the downloaded project folder.**

* **Additional Requirements:**

**You must have Python 3.9 or later installed on your machine.**

**Ensure you have an internet connection when running the project for the first time to install the required libraries.**

Proper Steps to execute the project

* **Accessing Project Files:**

**Click the available GitHub link to access the AI ​​Resume Ranker repository.**

* **Opening the Project in a Workspace:**

**Locate the main project launch file, flask\_backend.py .**

* **Running the Interface:**

**In the Command Prompt or Terminal, ensure you are in the correct project folder.**

**Execute the streamlit run streamlit\_frontend.py command to launch the user interface.**

* **Interacting with the App:**

**The app will automatically open in your web browser.**

**Follow the instructions in the interface to upload resume and job description files.**

**Explore various functions, such as searching and filtering, to take full advantage of the resume ranking features.**

Links:

[Blog link](https://www.tumblr.com/tensorx1/794723600065347584/ai-resume-ranker-assisting-hr-and-recruitment?source=share)

[Images link](https://gemini.google.com/)

[GitHub link](https://github.com/TensorX1/AI_Resume_Ranker/tree/main)