Resume Screening Application Documentation

# Use Case

In this Resume Screening Application, HR professionals can upload job descriptions along with multiple resumes. The application then analyzes the resumes based on the job description and required skills to provide a matching score. The application extracts critical information from resumes such as:  
  
- Candidate Name  
- Candidate Email  
- Candidate Mobile Number  
- Years of Experience  
- Experience Summary  
- Missing Skills (if any)  
- Candidate Skills  
  
This helps HR professionals quickly identify the most qualified candidates, making the recruitment process more efficient. The application can assist HR in shortlisting candidates faster and evaluating them based on their match with the job requirements.

# Technologies Used

## Frontend: Angular

For the frontend, Angular was used to create a dynamic, responsive, and interactive user interface. Angular's powerful features like data-binding and dependency injection helped build a scalable application.

## Backend: FastAPI, Python, Gemini AI API, PyMuPDF

### FastAPI

FastAPI is a modern, fast (high-performance) web framework for building APIs with Python 3.7+ based on standard Python type hints.  
FastAPI was chosen for its ease of use, asynchronous capabilities, and automatic generation of OpenAPI documentation. It helped us build a fast backend to handle multiple concurrent requests efficiently.

### Python

Python was used as the primary language for building the backend, which allowed us to leverage its rich ecosystem of libraries for text processing, AI model integration, and file handling.

### Gemini AI API

Gemini AI API was integrated for analyzing the job descriptions and resumes, providing candidate matching scores, missing skills, and relevant experience summary. Gemini AI offers powerful text analysis capabilities to improve resume screening efficiency.

### PyMuPDF

PyMuPDF is used to extract text from PDF resumes. It allows the application to handle different formats of resumes, providing accurate and efficient extraction of data.

# Workflow

The following is the workflow of the Resume Screening Application:  
  
1. \*\*Upload PDF\*\* - The user uploads the job description and resumes (in PDF format).  
2. \*\*Extract Text\*\* - The text is extracted from the resumes using PyMuPDF.  
3. \*\*Gemini Analysis\*\* - The extracted resume text and job description are analyzed by the Gemini AI API.  
4. \*\*Process Response\*\* - The application processes the response from the Gemini AI API.  
5. \*\*Return Results\*\* - The results are returned to the user, including candidate matching percentage, missing skills, and other details.

# Initial Tech Stack (Specy, BERT, Scoring Tools)

Initially, the application used spaCy for Named Entity Recognition (NER) and BERT for semantic analysis. While these tools provided some level of accuracy, the matching score results were not satisfactory and had limitations regarding the contextual understanding of the resumes. The limitations of spaCy and BERT include:  
  
1. \*\*Accuracy Issues\*\*: The accuracy of the analysis was low, especially about contextual matching between job descriptions and resumes.  
2. \*\*Limited Contextual Understanding\*\*: BERT could understand the text context, but the resume matching process was not optimized   
for HR-related tasks.  
  
Due to these limitations, we decided to move towards \*\*Gemini AI\*\*, which provided more accurate and specific capabilities for job-resume matching.

# Advantages and Drawbacks of Gemini AI

## Advantages

Gemini AI offers several advantages over the previous tools used:  
  
1. \*\*Higher Accuracy\*\*: Gemini AI provides a more accurate understanding of job descriptions and resumes, offering precise matching scores.  
2. \*\*Contextual Matching\*\*: It has better contextual understanding, which is essential for matching resumes with job descriptions in an HR context.  
3. \*\*Advanced Analysis\*\*: Gemini AI can extract meaningful insights, such as missing skills, which helps HR professionals evaluate candidates better.  
4. \*\*Ease of Integration\*\*: It integrates well with existing systems and frameworks, making it an ideal choice for a production-grade application.

## Drawbacks

1. \*\*Free Plan Limitations\*\*: The free tier of Gemini AI comes with restrictions on the number of requests and available features, which may limit its usage for large-scale applications.  
2. \*\*Pricing\*\*: The pricing of Gemini AI can be expensive, especially for high-volume use cases.  
3. \*\*Token Limitations\*\*: Gemini AI may have token limitations per request, which may affect performance when processing large resumes or job descriptions.  
4. \*\*Security Concerns\*\*: Being an external service, there might be concerns related to the security of the data being processed by Gemini AI.

# Overcoming Drawbacks

To overcome the drawbacks of Gemini AI:  
  
1. \*\*Upgrade Plans\*\*: The application can move to a paid plan for higher token limits and more advanced features.  
2. \*\*Data Preprocessing\*\*: Minimize token usage by processing and optimizing job descriptions and resumes before sending them to Gemini AI.  
3. \*\*Data Encryption\*\*: Implement end-to-end encryption and secure data handling to ensure privacy and security when communicating with Gemini AI.

# Future Scope

The future scope of the Resume Screening Application includes:  
  
1. \*\*Integration with Other Job Boards\*\*: We plan to integrate this system with popular job boards like LinkedIn and Indeed to automatically fetch job descriptions and resumes.  
2. \*\*Advanced Candidate Scoring\*\*: Incorporate additional parameters such as personality assessments, psychometric tests, and interviews into the candidate evaluation process.  
3. \*\*Real-time Analysis\*\*: Implement real-time processing of incoming resumes and job descriptions, allowing HR professionals to   
receive instant feedback.  
4. \*\*User Feedback Loop\*\*: Collect feedback from HR professionals to refine the algorithm and improve the accuracy of the matching system.