Resume Screening System: An AI-Powered Tool for Efficient Candidate Evaluation

Project Overview

The Resume Screening System is an intelligent web application designed to automate the process of evaluating job applicants' resumes against a given job description. Built using modern AI technologies, this tool extracts key information from resumes (in PDF format), compares them to job requirements, calculates a match score, identifies missing skills, and sends automated email notifications to candidates and HR personnel. It streamlines recruitment by reducing manual effort, making it ideal for HR teams, recruiters, and companies handling large volumes of applications.

The application is deployed as an interactive web interface using Streamlit, allowing users (candidates) to upload their resumes easily. Behind the scenes, it leverages AI models to perform natural language processing and similarity analysis, providing quick and objective feedback.

Problem It Solves

In traditional recruitment processes, HR professionals often spend hours manually reviewing hundreds of resumes for each job opening. This leads to several challenges:

- **Time-Consuming Manual Screening**: Reading through resumes to match skills, experience, and education with job requirements is inefficient and prone to human error or bias.
- **Scalability Issues**: For high-volume hiring (e.g., in tech companies or during job fairs), it's difficult to handle applications quickly, leading to delays in shortlisting candidates.
- **Inconsistent Evaluations**: Different reviewers might interpret resumes differently, resulting in unfair or overlooked selections.
- **Lack of Feedback for Candidates**: Rejected applicants rarely receive personalized feedback on why they weren't selected or how to improve.
- **Administrative Overhead**: Notifying candidates and internal teams (like HR) via emails adds extra workload.

This project addresses these issues by automating the screening process with AI, ensuring faster, fairer, and more data-driven decisions while providing constructive feedback.

How It Solves the Problem

The system works in a step-by-step manner to analyze resumes and make decisions. Here's a detailed breakdown of its workflow:

1. User Input and Resume Upload

- Candidates enter their name and email via a simple web form.
- They upload their resume in PDF format.
- The application validates inputs to ensure all fields are filled.

2. Resume Text Extraction

- Using a PDF parsing library, the tool extracts raw text from the uploaded resume.
- This handles multi-page PDFs and preserves formatting where possible.

3. AI-Powered Analysis

- The extracted text is fed into an AI model (via a language chain) along with a predefined job description (e.g., "Looking for a Python Developer with AI expertise").
- The AI acts as an "expert resume screener":
    - Extracts key details like skills, experience, and education from the resume.
    - Compares them to the job description.
    - Calculates a match score (0-100) based on similarity.
    - Identifies missing skills.
- Output is structured as JSON for easy processing (e.g., {"skills": [...], "experience": "...", "education": "...", "score": 85, "missing_skills": ["Machine Learning"]}).

4. Decision Making and Feedback

- Based on the score:
    - **Score > 80**: Candidate is shortlisted for an interview. An email invitation is sent, and HR is notified with the resume attached.
    - **Score 50-79**: Candidate is shortlisted for future opportunities. A notification email is sent.

    - **Score < 50**: Candidate is rejected. A polite rejection email is sent, including AI-generated feedback on skills to improve (e.g., "Focus on learning AI frameworks like TensorFlow").
- Feedback is generated dynamically using the AI model, suggesting improvements based on missing skills.

5. Email Notifications

- Automated emails are sent to the candidate with the decision and feedback.
- For shortlisted candidates, HR receives a notification with the resume attached.
- Emails are validated for format and sent securely via SMTP (e.g., Gmail server).

6. User Interface and Results Display

- Results are shown on the web page: Match score (as a progress bar), missing skills, and decision summary.
- The interface is user-friendly, with emojis and styled elements for better engagement.

This automated flow reduces screening time from hours to seconds per resume, minimizes bias through objective AI analysis, and enhances the candidate experience with timely feedback.

Technologies Used

The project is built using a combination of Python libraries and frameworks focused on AI, web development, and data processing. Here's a comprehensive list:

- **Programming Language**: Python 3.x – The core language for all logic and scripting.
- **Web Framework**: Streamlit – For creating the interactive web UI quickly without frontend expertise. It handles forms, file uploads, and dynamic displays like progress bars.
- **AI and Language Models**:
    - LangChain – A framework for building AI chains. Used to create prompts and chains for resume screening and feedback generation.

    - Google Generative AI (Gemini-1.5-flash model) – The LLM (Large Language Model) for natural language understanding, similarity calculation, and generating structured JSON responses or feedback.

- **PDF Processing**: pdfplumber – Extracts text from PDF resumes efficiently, handling layouts and multiple pages.
- **Environment Management**: dotenv – Loads sensitive data like API keys and email credentials from a .env file for security.
- **Email Integration**: smtplib and email.message (Python standard libraries) – For sending emails with attachments via SMTP.
- **Data Handling and Utilities**:
    - json – For parsing AI responses into structured data.
    - re (Regular Expressions) – For email validation.
    - os – For file and environment operations.
- **Other Dependencies**:
    - No external databases are used; everything is in-memory for simplicity.
    - The app can be deployed on platforms like Streamlit Cloud for free hosting.

All technologies are open-source or free-tier accessible, making the project cost-effective and easy to replicate.

Conclusion

This Resume Screening System demonstrates how AI can transform HR processes by making them faster, more accurate, and user-centric. It's a practical example of integrating generative AI with web applications to solve real-world problems. Future enhancements could include support for more file formats, multi-job descriptions, or integration with applicant tracking systems (ATS).

If you're interested in trying it out, check the GitHub repository [https://github.com/rubaahmedkhan/langchain-resume-screener.git] or the live demo

[https://huggingface.co/spaces/RubaKhan242/resume]. For questions or collaborations, feel free to reach out!

Developed by Ruba Ahmed Khan