RESUME SHORTLISTING VIA AI

Submitted By: Narendran B, Ganesh A, Karamchand suresh

1. Goal of the Project

Objective

This project involves developing a basic web application using Python's Streamlit framework to assist recruiters in evaluating and shortlisting resumes. The tool allows users to upload resume files (such as PDFs or Word documents), which are then processed to extract key information like skills, experience, education, and achievements. Leveraging generative AI (GenAI) models, the application analyzes the content to generate a scoring system based on predefined job criteria, highlighting strengths and weaknesses in a candidate's profile. This streamlines the recruitment process by providing quick insights, such as keyword matches, relevance scores, and summarized bullet points, enabling recruiters to make informed decisions faster. The interface is user-friendly, featuring upload functionality, real-time processing, and visual outputs like charts or ranked lists, making it an efficient aid for high-volume hiring scenarios.

2. Design Patterns Used

The core pattern for identifying and shortlisting resumes relies on a multi-step GenAI-driven workflow integrated within the Streamlit app. Initially, the uploaded resume is parsed using libraries like PyPDF2 or docx to extract raw text, followed by preprocessing to clean and structure the data (e.g., removing noise and tokenizing sections). A generative AI model, such as those from OpenAI or Hugging Face, is then employed to perform semantic analysis: it matches extracted elements against job-specific keywords, evaluates contextual relevance using natural language understanding, and computes a composite score based on factors like experience duration, skill proficiency, and alignment with role requirements. Advanced techniques, including embedding vectors for similarity search or prompt engineering for custom evaluations, help rank resumes in descending order of fit. Threshold-based filtering shortlists top candidates, with outputs presented as prioritized lists or heatmaps, ensuring transparency and reducing manual review time while minimizing bias through data-driven insights.

3. Technology Used

Frontend: STREAMLIT

Backend: CORE PYTHON

AI Model: Convolutional Neural Network (CNN)

Database: SQLLITE

4. Database Schema Description

Users: Stores user details such as name and email.

Predictions: Resume Summary, Quickly assess resume relevance, Reduce manual screening time, Make data-driven shortlisting decisions.

5. Git hub link

https://github.com/GANESH-A08/Resume-shortlisting-using-AI