**1. HTML (Frontend)**

The HTML page has the following elements:

* **File upload form**: Allows users to upload resumes in PDF, DOCX, or TXT format.
* **JavaScript (AJAX) handling**: Submits the form asynchronously and displays the analysis results including a skills distribution pie chart and improvement areas.
* **Results Table**: Displays uploaded resumes along with their categories and extracted skills.

**2. Flask Application (Backend)**

The Flask application provides routes and logic for handling the file uploads and analysis.

**Key Functions:**

* **allowed\_file()**: Verifies the file type to allow only PDF, DOCX, and TXT files.
* **extract\_text\_from\_pdf()**: Extracts text from uploaded PDF files using the pdfplumber library.
* **analyze\_resume()**: Analyzes the extracted text for specific skills related to categories like Programming Languages, Web Development, and Soft Skills.
* **generate\_pie\_chart()**: Creates a pie chart displaying the distribution of skills in the resume using matplotlib.
* **upload\_resume()**: Handles the resume upload, text extraction, analysis, and saving results to MongoDB.

**3. MongoDB Integration**

MongoDB is used to store the following:

* Filename
* Extracted skills and their counts
* Total score for the resume (sum of detected skills)
* Areas for improvement (skills not found in the resume)

The results are saved in the resumes collection in the resume\_screening database.

**4. Skill Categorization and Analysis**

* The **categorize\_resume()** function assigns categories (e.g., Developer, Analyst) based on keywords found in the resume text.
* The **extract\_skills()** function identifies specific skills like Python, Java, SQL, etc., and returns a list of the ones found in the resume.
* **get\_skill\_analysis()** aggregates the counts of different skills from the resumes stored in MongoDB.

