Functional and Non-Functional Requirements

AI Resume Analyzer (Full-Stack + ML)

This document outlines the core requirements for building an AI-powered Resume Analyzer. The system is designed to parse resumes, match them against job descriptions using Natural Language Processing (NLP), and provide actionable feedback to candidates and recruiters.

1. Functional Requirements

What the system must do.

A. User Authentication & Profile Management

- **Registration/Login:** Support for Email/Password and OAuth (Google, LinkedIn).
- **Role-Based Access Control (RBAC):**
  - **Candidate:** Can upload resumes, view their own scores, and save reports.
  - **Recruiter:** Can create job descriptions, batch upload resumes, and view a ranked leaderboard of candidates.
  - **Admin:** Manages user bans, system configurations, and subscription plans.
- **Dashboard:** A personalized view showing recent uploads, analysis history, and saved Job Descriptions (JDs).

B. Resume Upload & Parsing Module

- **File Support:** Must accept .pdf, .docx, and .txt formats.
- **File Validation:** Enforce size limits (e.g., max 5MB) and check for corrupted files.
- **Text Extraction:** Convert binary file content into raw text while maintaining the logical order of sections.
- **Named Entity Recognition (NER):** Automatically identify and categorize:
  - **Contact Info:** Name, Email, Phone Number, LinkedIn/GitHub URLs.
  - **Education:** University names, Degrees, Graduation Years.
  - **Skills:** Technical skills (Python, SQL) and Soft skills (Leadership, Communication).
  - **Experience:** Company names, Job Titles, Duration of employment.

C. AI Analysis & Scoring Engine

- **Job Description Matching:** Allow users to paste text or upload a file for a specific Job Description.
- **Semantic Similarity Scoring:** Calculate a match percentage (0-100%) based on semantic meaning, not just exact keyword matches (using BERT/Transformers).
- **Keyword Gap Analysis:**

    - **Hard Match:** List keywords found in both documents.
    - **Missing Critical Skills:** List high-priority keywords found in the JD but missing from the resume.
- **Section-Level Feedback:**
    - Check for measurable metrics (e.g., "Did you increase sales by X%?").
    - Check for action verbs (e.g., "Led," "Developed," "Optimized").
    - Flag formatting issues (e.g., "Summary is too long").

D. Reporting & Visualization

- **Interactive Scorecard:** Display the score with a visual breakdown (Skills match, Experience match, Education match).
- **PDF Export:** Generate a downloadable PDF report of the analysis.
- **Comparison View:** Side-by-side view of the Resume text vs. JD text with highlighted keywords.

2. Non-Functional Requirements (NFRs)

System attributes and quality standards.

A. Performance & Scalability

- **Latency:** The complete parsing and analysis pipeline should take no more than **3-5 seconds** per document.
- **Concurrency:** The system must handle spikes in traffic (e.g., 50+ concurrent uploads) without service degradation.
- **Asynchronous Processing:** Heavy ML tasks should be offloaded to a background task queue (e.g., Celery/Redis) to keep the UI responsive.

B. Accuracy & Reliability

- **Parsing Precision:** The text extraction module should achieve >90% accuracy on standard single-column and double-column layouts.
- **Model Fairness:** The scoring algorithm must be bias-aware (e.g., ignoring name, gender, or address fields during scoring).
- **Uptime:** Target 99.9% availability during business hours.

C. Security & Data Privacy

- **Data Encryption:**
    - **At Rest:** Resumes stored in object storage (S3/GCS) must be encrypted (AES-256).
    - **In Transit:** All data transmission must occur over HTTPS (TLS 1.2+).
- **PII Handling:** Implementation of a "Blind Hiring" mode that temporarily masks names and contact info for recruiters.
- **Data Retention:** Automated cleanup policies (e.g., delete guest uploads after 24 hours) and GDPR compliance (Right to be Forgotten).

D. Usability & Accessibility

- **Responsive Design:** The UI must be fully functional on Desktop, Tablet, and Mobile browsers.
- **Feedback Loops:** Clear error messages for unsupported file types or unreadable fonts.
- **Accessibility:** Compliance with WCAG 2.1 AA standards (screen reader support for reports).

3. Suggested Technology Stack

Component	**Technology**	**Reasoning**
Frontend	React.js / Next.js	Fast rendering, SEO for landing pages, rich interactive charts.
Backend	Python (FastAPI or Django)	Native support for AI/ML libraries, high performance.
ML/NLP	Spacy, PyResparser, OpenAI API	Spacy for NER; Vector databases for semantic similarity.
Database	PostgreSQL	Relational data for users and structured resume data.
Storage	AWS S3 / Google Cloud Storage	Scalable storage for raw resume files.
Task Queue	Redis + Celery	To manage background ML processing jobs.