Technical Report — Part 1

Resume AI Optimizer System (ATS-Based Job Match Analysis)
Group: QuantumLoops
Course: AML 3303 – Term Project

1. Introduction

Applicant Tracking Systems (ATS) now play a central role in the hiring process, filtering and ranking resumes long before they reach a human recruiter. Candidates with relevant experience often fail to progress because their resumes are not adequately aligned with the job description or structured in an ATS-friendly format.

The Resume AI Optimizer is designed to address this challenge. The system evaluates a candidate’s resume against a target job description, providing actionable feedback grounded in both quantitative scoring and qualitative critique. It supports applicants in refining their resumes iteratively, with the goal of increasing the chances of passing automated screening and securing interviews.

This document outlines the technical foundation of the system—its user flow, architectural structure, data design, and core preprocessing methodology—forming the basis for subsequent implementation phases.

2. User Flow Overview

The system is designed to be intuitive for job seekers while maintaining a structured, well-defined backend workflow.

1. **Landing Page**

   Users arrive at a simple interface featuring two primary inputs:

   - A text area for the target job description
   - A file upload section for the resume

2. **Job Description Input**

   The user copies a job posting from platforms such as LinkedIn or Indeed and pastes the content into the text area.

The system processes the text to extract essential keywords, required competencies, and experience expectations. These become the criteria used to evaluate the resume.

3. **Resume Upload**

   Users upload their resume in PDF or DOCX format. The file undergoes basic validation and is stored securely in AWS S3 for further processing.

4. **ATS-Style Analysis**

   Once both inputs are provided, the system parses the resume and compares it against the extracted job criteria.
   A local LLM conducts a structured evaluation, simulating how a recruiter or ATS would assess alignment.

5. **Results Display**

   The user receives:

   - A **Match Score**
   - A list of **Missing Keywords**
   - A concise summary of recommended improvements

6. **Iterative Refinement**

   Users may upload revised versions of their resume to track how their score evolves over time. This encourages a continuous improvement cycle and mirrors real-world job application behavior.

The flow balances technical rigor with user-friendly interaction, supporting both novice and experienced applicants.

3. System Architecture Overview

The solution is built on a modular architecture, enabling scalability, maintainability, and clear separation of concerns.

3.1 Streamlit Frontend

- Provides an accessible interface for job description input and resume uploads
- Displays match scores, keyword analysis, and feedback summaries
- Uses session state to preserve the user’s history during a session
- Designed to minimize friction and support repeated iterations

3.2 FastAPI Backend

- Acts as the processing engine behind all user actions
- Exposes RESTful endpoints for analysis, file uploads, and health checks
- Coordinates interactions between the resume parser, S3 storage, MongoDB, and the LLM
- Ensures all inputs are validated, sanitized, and securely processed

3.3 Storage Layer

- **AWS S3** stores resume files, guaranteeing durability and easy retrieval
- **MongoDB** captures structured analysis data for each session, enabling historical tracking and analytics

3.4 AI and NLP Processing Engine

- Extracts skills, entities, and contextual information from both the job description and resume
- Computes keyword overlap and semantic similarity
- Generates expert-style qualitative feedback using a local LLM, avoiding privacy concerns associated with cloud-based models

This architecture balances computational efficiency with flexibility, allowing the system to evolve as requirements grow.

4. Data Design

Each analysis session is stored as a single document within the MongoDB `resume_reviews` collection. This approach centralizes all relevant information—raw inputs, extracted fields, and system outputs—into a durable and well-structured format.

A typical record includes:

- Identifiers (UUID and timestamp)
- Source file stored in AWS S3
- Parsed job description, including extracted keywords
- Parsed resume content, including detected skills and relevant metadata
- Analysis outputs such as match score, missing keywords, and summarized feedback

This schema supports reproducibility, subsequent comparison between versions, and potential retrieval for analytics in future enhancements.

5. Methodology

The methodology underpins the system's ability to generate reliable, structured evaluations. It consists of two major phases: job description preprocessing and resume parsing.

5.1 Job Description Preprocessing

Job descriptions vary widely in format and quality. To ensure consistency, the system applies several preprocessing steps:

- **Text Normalization:** Lowercasing, removing noise, and standardizing formatting
- **Stopword Removal:** Eliminating non-informative words to emphasize meaningful content
- **Lemmatization:** Reducing words to their base forms for improved comparison
- **Keyword Extraction:** Identifying the core skills, technologies, and competencies required for the role
- **Requirement Identification:** Extracting job constraints such as required years of experience, certifications, or technical proficiencies

This creates a structured representation of the job that serves as the benchmark for evaluating resumes.

5.2 Resume Parsing

Resumes often contain complex layouts, mixed fonts, and section variations. To handle this diversity, the system employs a robust parsing pipeline:

- **Document Extraction:** Using tools such as `pdfplumber` or `PyMuPDF` to extract text reliably
- **Section Identification:** Detecting common resume sections (Skills, Experience, Education) using pattern-based heuristics
- **Keyword and Entity Extraction:** Identifying explicit skills and contextual indicators
- **Experience Estimation:** Interpreting years of experience mentioned in professional history
- **Profile Structuring:** Converting extracted information into a consistent, machine-readable format

This structured representation enables accurate comparison with the job description and feeds the scoring and LLM evaluation phases.