AI Resume Intelligence System - Project Summary

1. Backend Setup

- Created backend folder and virtual environment.
- Installed Flask and PostgreSQL.
- Created database: ai_resume_dg.
- Connected Flask to PostgreSQL using psycopg2.
- Verified connection with home route.

---

2. CRUD Implementation (test_users table)

- Created test_users table in PostgreSQL.
- Implemented CREATE (POST /add-user).
- Implemented READ (GET /users).
- Implemented UPDATE (PUT /update-user/).
- Implemented DELETE (DELETE /delete-user/).
- Structured JSON responses properly.

---

3. Transition to Real System (resumes table)

- Created resumes table with fields: id, name, email, skills, score, created_at.
- Implemented POST /add-resume endpoint.
- Stored resume data in database.
- Retrieved resumes using GET /resumes.

---

4. Scoring Logic Upgrade

- Replaced fake scoring with keyword-based scoring.
- Defined REQUIRED_SKILLS list.
- Calculated match percentage based on required skills.
- Stored calculated score in database.

---

5. Recruiter Intelligence Features

- Implemented GET /resumes/top (sorted by score).
- Implemented GET /resumes/filter?skill=python.
- Used ILIKE for case-insensitive filtering.
- Added ORDER BY score DESC for ranking.

---

6. ML Preparation Phase

- Planned transition from rule-based scoring to ML.
- Discussed TF-IDF + Cosine Similarity approach.
- Planned supervised training using Logistic Regression.
- Installed scikit-learn for ML implementation.
