Project Documentation: AI-Powered Job Application Tracker

1. Project Overview

Project Name: AI Job Tracker

<u>Objective</u>: Help users manage and track job applications intelligently using AI features like resume matching, job suggestion, status analytics, and interview preparation.

Target Users:

- Job seekers
- Students & freshers
- Career changers

2. Core Functional Features

User Dashboard

- Add job applications (manual or via email parsing)
- Track application status (Applied, Interview, Offer, Rejected, etc.)
- Job timeline view
- Custom tags & notes for each application

AI Features

- Resume vs Job Description Matching (Similarity Score)
- Job Suggestion Engine (via API scraping or integrations)
- Cover Letter Generator
- Smart Reminders (follow-up prompts based on status & company response time)
- Interview Question Predictor (based on role & company)
- Auto-Fill Applications (browser extension)

Data & Analytics

- Visual timeline of job application journey
- Weekly application summary
- Conversion rates (Applied → Interview → Offer)

• Top industries/roles applied to

3. Architecture Overview

Frontend:

• Framework: React.js / Next.js

• Mobile App: Optional – React Native or Flutter

Backend:

• Language: Python (FastAPI / Flask) or Node.js

• AI & ML: OpenAI API, spaCy, scikit-learn, Hugging Face transformers

• Job Data Source: Indeed, LinkedIn (scraped or integrated via API)

Storage:

• **Database**: PostgreSQL / MongoDB

• File Storage: AWS S3 or Cloudinary (resumes, cover letters)

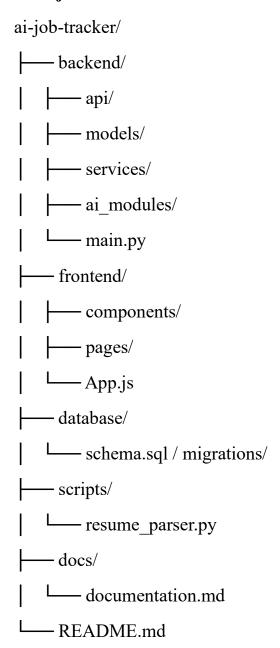
Hosting:

• Frontend: Vercel / Netlify

• Backend: Render / Heroku / AWS

• CI/CD: GitHub Actions

4.Project Structure



Database Schema (Simplified)

Tables:

- Users id, name, email, password_hash
- Applications id, user_id, company, role, status, applied_date, notes, resume_file
- AI Scores id, application id, similarity score, ai feedback
- Reminders id, application_id, due_date, type, is_sent

AI Modules

1. Resume vs JD Matching

- Uses OpenAI embeddings or TF-IDF + cosine similarity
- Output: Match score + improvement tips

2. Cover Letter Generator

- Uses GPT-4 or fine-tuned LLM
- Input: JD, resume → Output: Personalized cover letter

3. Interview Question Predictor

- Based on job title and industry
- Uses dataset of common questions + GPT for extrapolation

4. Job Suggestions

- Integrates with LinkedIn / Indeed API (or scraping if allowed)
- Ranks suggestions based on past applications + resume fit

API Endpoints

| Method | d Endpoint | Description |
|--------|-----------------------|-------------------------------------|
| POST | /api/applications | Add a job application |
| GET | /api/applications/:id | l Get application detail |
| POST | /api/resume-match | Compare resume with job description |
| POST | /api/cover-letter | Generate AI cover letter |
| GET | /api/stats | Get application analytics |
| | | |

31 Development Timeline (10–12 Weeks)

Phase Timeline

Research & Planning Week 1

UI/UX Design Week 2–3

Backend & API Week 3–5

AI Modules Integration Week 5–7

Frontend Development Week 6-8

Testing & QA Week 9–10

Deployment & Launch Week 11-12

Malytics Features

- Total applications
- Conversion funnel (Applied → Interview → Offer)
- Response rate by company
- Smart notifications

Security Considerations

- Secure file upload (validate resumes)
- Role-based access control
- OAuth for Google/LinkedIn sign-in
- Rate limiting on AI endpoints

Testing Strategy

• Backend Tests: Pytest / Jest

• Frontend Tests: React Testing Library, Cypress

• AI Tests: Validation with known examples

• Manual QA: Application flows, email reminders, etc.

Optional Features

• Browser extension to auto-save job listings

• Auto-track email confirmations (e.g., Gmail integration)

• Export application data as PDF

• Shareable public portfolio of job applications

Tech Stack Summary

| Layer | Tech | |
|--|---------------------------------|--|
| Frontend | React.js / Next.js | |
| Backend | Node.js / FastAPI | |
| AI | OpenAI API, Hugging Face, spaCy | |
| DB | PostgreSQL / MongoDB | |
| Auth | JWT / OAuth (Google) | |
| Payments (Optional) Stripe / Gumroad (for paid features) | | |

Storage AWS S3 / Cloudinary

Deployment Vercel (frontend), Render (backend)