ATS backend project structure

**🔹 Week 1: Project Setup & Authentication**

**🔧 Backend**

* Set up Node.js project structure (ats-backend/ as defined)
* Configure TypeORM with MySQL (in data-source.js)
* Create User entity (with roles: ADMIN, HR, INTERVIEWER, CANDIDATE)
* Implement JWT-based Auth (login, register)
* Setup GraphQL typedefs and resolvers for auth and user

**🎨 Frontend**

* Set up React + Tailwind
* Build login & register pages
* Implement JWT handling (store in localStorage)
* Setup ApolloProvider and role-based routing

**✅ Integration Goal:** Login & Register functionality works with role-based access.

**🔹 Week 2: Job Management**

**🔧 Backend**

* Create Job entity
* Implement createJob, updateJob, archiveJob, getJobs resolvers
* Add filtering by department, location, tags

**🎨 Frontend**

* Admin/HR dashboard: create & manage jobs (form + list)
* Candidate view: public job listings with filters

**✅ Integration Goal:** Admin/HR can post jobs; Candidates can view them.

**🔹 Week 3: Candidate Application & Resume Upload**

**🔧 Backend**

* Create Candidate, Application, Resume entities
* Implement:
  + applyToJob(jobId, resume)
  + myApplications query
* Set up file upload with graphql-upload & Multer

**🎨 Frontend**

* Candidate dashboard:
  + Apply to jobs
  + Upload resume (file input + validation)
  + View status of applications

**✅ Integration Goal:** Candidates can apply with resume and view application history.

**🔹 Week 4: AI Resume Scoring Integration**

**🔧 Backend**

* Connect to Python-based AI microservice (POST /score-resume)
* On application, send resume text + job description
* Store score, matched\_keywords, missing\_keywords in Application table

**🎨 Frontend**

* HR dashboard:
  + View candidate applications with match scores
  + Filter/sort by score
  + Show matched/missing keywords

**✅ Integration Goal:** HR sees resume scores on each application.

**🔹 Week 5: Interview Scheduling & Feedback**

**🔧 Backend**

* Create Interview and Feedback entities
* Implement:
  + scheduleInterview(candidateId, jobId, interviewerId)
  + submitFeedback(interviewId, score, comments)
  + getInterviewTimeline(candidateId)
* Optional: Add email notifications (via SendGrid)

**🎨 Frontend**

* HR: Schedule interviews
* Interviewer: View assigned interviews, submit feedback
* Candidate: See interview timeline and feedback

**✅ Integration Goal:** Interview cycle with feedback is complete.

**🔹 Week 6: Analytics, Testing & Deployment**

**🔧 Backend**

* Create analytics.resolver.js for:
  + Total job posts, candidates, interviews
  + Time-to-hire, source analysis
* Write unit/integration tests using Jest
* Dockerize backend + microservice
* Deploy to Render/Railway/AWS

**🎨 Frontend**

* Admin dashboard: Use Recharts for analytics
* Polish UI, ensure mobile responsiveness
* Deploy frontend (e.g., Vercel or Netlify)

**✅ Final Deliverables:**

* Fully functional ATS with authentication, resume scoring, interview tracking, and reporting