

CMPS 350 Web Development Fundamentals - Spring 2025

Final Practical Exam – JobBoard: Internal Job Application System

Instructions

- The exam duration is **120 minutes**.
- **Push your code to GitHub** regularly to avoid any data loss.
- The exam is **open book**. The Use of AI or Plagiarism will result in a score of zero for all parties involved.
- Demo your work before leaving the exam.
- Add screenshots of the output to the testing sheet.
- Push your **code and testing sheet** to your GitHub repo under the **Final Exam** folder.

Preparation

- Open the **JobBoard base code** project and run **npm install** to get all dependencies.
- Make sure you have the **Prisma** extension in your VS Code for Prisma code highlighting.

1. Create the Data Model [10 pts]

Define two Prisma models Job and Application having one-to-many relationship as shown in the following ER diagram.

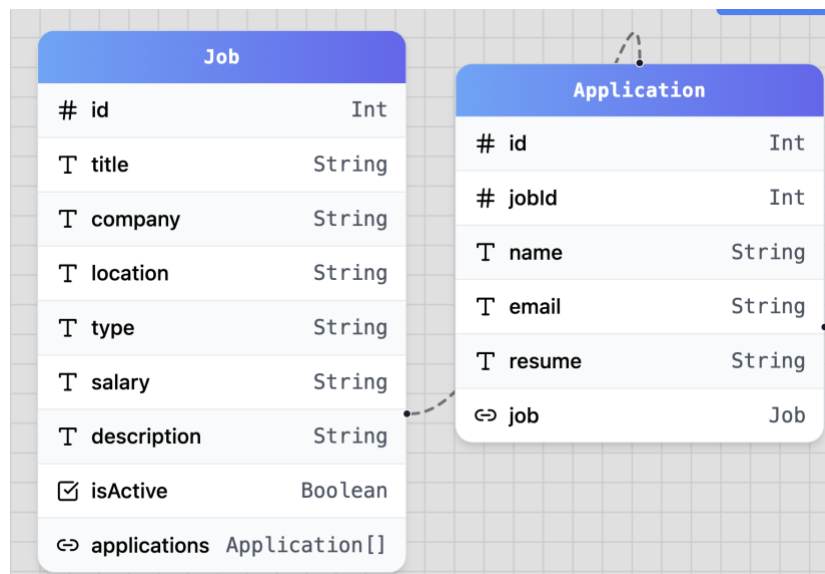


Figure 1 : JobHub ER Diagram

- `npx prisma migrate dev --name init`
- Seed the database using the provided files: `data/jobs.json` and `data/applications.json`.
- `npx prisma studio`

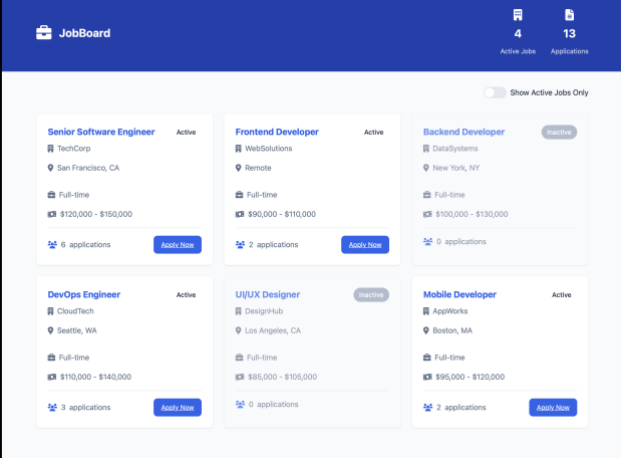

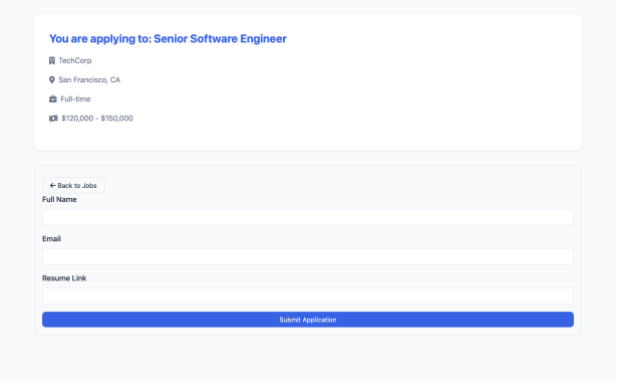

2. Implement the JobBoardRepo [15]

Implement a ApplicationRepo class to handle the database logic.

- **getJobs(status?)** // Return all jobs with their applications, with optional status filtering
- **getJob(id)** // Retrieve a specific job by ID with its applications
- **addApplication(newApplication)** // Add a new application
- **deleteApplication(id)** // Remove an application by ID
- **updateJob(id, status)** //updates the applications isActive with the given status

3. Implement JobBoard User Interface (75 pts)

Implement the following UI pages and components using React and Next.js. All backend actions should be wired using Server Actions only.

	<p> [30 pts] Job List Page</p> <ul style="list-style-type: none">- Display job postings in a responsive grid- Show job title, company, location, job type, salary, and application count- Display color-coded active/inactive status- Include a toggle to change job status (uses Server Action)- Show loading state during status update- Only show “Apply” button for active jobs
<p>Figure 2 : Jobs Page</p> 	<p>[31] [25 pts] Application Form</p> <ul style="list-style-type: none">- Clicking “Apply” opens a form for the selected job- Form fields: Full Name, Email, Resume Link- Submitting the form triggers a Server Action and adds to the application list- Show success message after submission
<p>Figure 3 : Application Page</p>	<p> [20 pts] Header & Stats</p> <ul style="list-style-type: none">- At the top of the page, show:- Total number of jobs- Total number of applications- Stats should update live after submission or status change