

# 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.

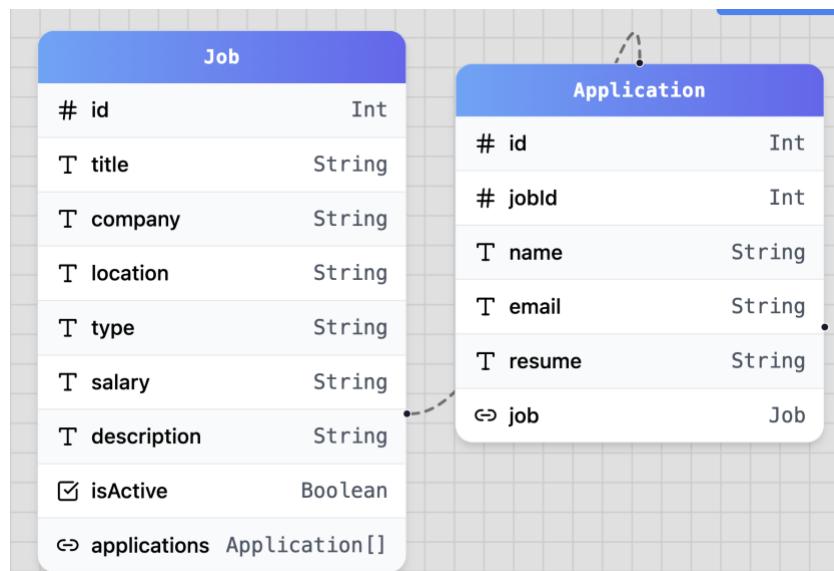


Figure 1 : JobHub ER Diagram

- `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.

The screenshot shows a dashboard titled "JobBoard". At the top right, there are buttons for "Active Jobs" (4) and "Applications" (13). A toggle switch labeled "Show Active Jobs Only" is shown. Below the header, there is a grid of job listings:

Job Title	Company	Status	Location	Type	Salary Range	Applications	Action
Senior Software Engineer	TechCorp	Active	San Francisco, CA	Full-time	\$120,000 - \$150,000	6 applications	Add Now
Frontend Developer	WebSolutions	Active	Remote	Full-time	\$90,000 - \$110,000	2 applications	Add Now
Backend Developer	DataSystems	Inactive	New York, NY	Full-time	\$100,000 - \$130,000	0 applications	
DevOps Engineer	CloudTech	Active	Seattle, WA	Full-time	\$110,000 - \$140,000	3 applications	Add Now
UI/UX Designer	DesignHub	Inactive	Los Angeles, CA	Full-time	\$85,000 - \$105,000	0 applications	
Mobile Developer	AppWorks	Active	Boston, MA	Full-time	\$95,000 - \$120,000	2 applications	Add Now

Figure 2 : Jobs Page

The screenshot shows the "Apply" form for the "Senior Software Engineer" position. The form fields are:

- You are applying to: Senior Software Engineer
- Company: TechCorp
- Location: San Francisco, CA
- Type: Full-time
- Salary Range: \$120,000 - \$150,000
- Full Name: (input field)
- Email: (input field)
- Resume Link: (input field)
- Submit Application button

Figure 3 : Application Page

### [30 pts] Job List Page

- 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

### [25 pts] Application Form

- 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

### [20 pts] Header & Stats

- At the top of the page, show:
  - Total number of jobs
  - Total number of applications
  - Stats should update live after submission or status change