**Project Title: Job Board Platform**

**🧠 Project Concept:**

Build a **Job Board** where employers can post jobs, and job seekers can browse, apply, and manage their applications. This app will include admin features, authentication, and full CRUD for jobs, profiles, and applications.

**🏗️ Project Architecture:**

| **Layer** | **Tech Stack** | **Purpose** |
| --- | --- | --- |
| Backend | Django + Django REST Framework (DRF) | REST API, Auth, Database Models |
| Frontend | Next.js + Tailwind CSS | UI/UX, Routing, SSR |
| Database | PostgreSQL | Store users, jobs, applications |
| Auth | JWT (via djangorestframework-simplejwt) | Token-based auth (access/refresh) |

**📦 Core Features:**

**🎯 User Roles:**

1. **Employer**
   * Can create and manage job postings
   * Can view applications
2. **Job Seeker**
   * Can browse jobs
   * Can apply for jobs
   * Can manage their profile and resume
3. **Admin**
   * Manage users, jobs, and applications

**🔐 Authentication:**

* Register/Login with email + password (JWT)
* Role-based access (permissions via DRF)
* Protect frontend routes via token in cookies/localStorage

**🧱 Backend Models (Django):**

**1. User (AbstractUser with Roles)**

class CustomUser(AbstractUser):

is\_employer = models.BooleanField(default=False)

is\_job\_seeker = models.BooleanField(default=False)

**2. Profile**

class Profile(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE)

bio = models.TextField(blank=True)

resume = models.FileField(upload\_to='resumes/', blank=True)

**3. Job**

class Job(models.Model):

employer = models.ForeignKey(User, on\_delete=models.CASCADE)

title = models.CharField(max\_length=255)

description = models.TextField()

location = models.CharField(max\_length=255)

salary = models.DecimalField(max\_digits=10, decimal\_places=2)

created\_at = models.DateTimeField(auto\_now\_add=True)

**4. Application**

class Application(models.Model):

job = models.ForeignKey(Job, on\_delete=models.CASCADE)

applicant = models.ForeignKey(User, on\_delete=models.CASCADE)

cover\_letter = models.TextField()

status = models.CharField(max\_length=50, default='Pending')

created\_at = models.DateTimeField(auto\_now\_add=True)

**🔄 API Endpoints (Django REST Framework):**

| **Endpoint** | **Method** | **Description** |
| --- | --- | --- |
| /api/register/ | POST | Register new user |
| /api/login/ | POST | Authenticate and get JWT |
| /api/jobs/ | GET/POST | List all jobs or create a new job |
| /api/jobs/<id>/ | GET/PUT/DELETE | Retrieve, update, or delete job |
| /api/applications/ | GET/POST | Apply to job or list applications |
| /api/profile/ | GET/PUT | Get or update user profile |
| /api/users/ | GET | (Admin) list users |

Use ViewSets, ModelSerializers, and permission classes for RBAC.

**🧭 Frontend Pages (Next.js):**

**🌐 Public Pages:**

* / — Homepage (hero section, featured jobs)
* /login — Login page
* /register — Register as employer or job seeker
* /jobs — List all jobs with filters
* /jobs/[id] — View single job details

**👤 Authenticated (Job Seeker):**

* /dashboard — Job seeker dashboard
* /profile — Update profile and upload resume
* /applications — View submitted applications

**🏢 Authenticated (Employer):**

* /employer/dashboard — Employer dashboard
* /employer/jobs — Manage job postings
* /employer/jobs/create — Post a new job
* /employer/jobs/[id] — View job + applicants

**🛠 Admin (Optional):**

* /admin — Manage users/jobs/applications

**📋 CRUD Operations Summary:**

| **Resource** | **Create** | **Read** | **Update** | **Delete** |
| --- | --- | --- | --- | --- |
| Users | ✅ (register) | ✅ (admin only) | ✅ (profile) | ❌ |
| Jobs | ✅ (employer) | ✅ | ✅ (employer) | ✅ (employer) |
| Applications | ✅ (job seeker) | ✅ (self/employer) | ✅ (status update) | ✅ (admin/employer) |
| Profile | ✅ (auto on register) | ✅ | ✅ | ❌ |

**🧠 State Management on Frontend:**

Use React Context or a library like **Zustand**/**Redux Toolkit** to store:

* Auth tokens
* User info
* Jobs and applications
* Flash messages (toasts)

**📁 Project Structure Example:**

**Backend (jobboard-backend)**

jobboard/

├── api/

│ ├── models.py

│ ├── views.py

│ ├── serializers.py

│ ├── urls.py

├── jobboard/

│ ├── settings.py

│ ├── urls.py

├── manage.py

**Frontend (jobboard-frontend)**

pages/

├── index.tsx

├── login.tsx

├── register.tsx

├── jobs/

│ ├── index.tsx

│ ├── [id].tsx

├── dashboard/

│ ├── index.tsx

│ ├── applications.tsx

├── employer/

│ ├── dashboard.tsx

│ ├── jobs/

│ ├── index.tsx

│ ├── create.tsx

│ ├── [id].tsx

**📦 Extra Features (Stretch Goals):**

* Pagination and search on job listings
* Notifications (e.g., application received)
* Dark mode (frontend)
* Email notifications (via Django)
* Image/file uploads with preview (resume)
* Server-side rendering (SSR) with Next.js for SEO

**🧪 Testing**

* Django: Use pytest or built-in TestCase
* Next.js: Use Jest + React Testing Library for components

**🚀 Deployment:**

* **Backend**: Deploy with **Render**, **Railway**, or **Heroku**
* **Frontend**: Deploy to **Vercel**
* **Database**: PostgreSQL on **Supabase** or **ElephantSQL**