**PrepForge**

**Vision:**

Ace your next tech interview with a personalized, AI-powered preparation journey, curated questions, real-time progress tracking, and structured study paths, all in one seamless portal.

**Why is it different from other portals or apps out there?**There are portals that *partially* overlap what you’re thinking, but none that match exactly what this portal provides, especially not with the "AI + personal journey + progress tracking" combo.

**So, not even GPT?**

**No,** GPT is a great tool but it wont provide a dashboard, or checklist, or even download the progress report of a particular user. The portal uses GPT (AI), to get the required questions, based on the specific role asked by the user, but rest of the features are provided by the portal, which helps job seekers to prep in a good way.

**Tech Stack:**

Django + Django Framework

React.js

PostgreSQL

AWS

**Models.py:**  
This is used to create tables in the database.

By applying commands:

* python manage.py makemigrations
* python manage.py migrate

The first command tells Django, that these are the tables in the models.py, which need to be created in the database (blueprint).

The second command actually applies these tables to the database (actual development).

**Then we add a file serializers.py in interview folder:**

This file, actually creates the data in .json format, this helps the frontend to access the data when called from backend.  
You can think of serializers as:

**Translator between Django models and HTTP API data.**

**Views.py:**

This creates different instances such as role list, question list, options to create, update, delete the data.

|  |  |
| --- | --- |
| View | Purpose |
| RoleListView | List all roles (Frontend Developer, SDE-1, etc.) |
| QuestionListView | List all questions related to a specific role |
| UserProgressListCreateView | See / Create user's progress (bookmarked, completed, etc.) |
| UserProgressDetailView | Update / Delete user's progress |
| UserNoteListCreateView | See / Create user's notes for questions |
| UserNoteDetailView | Update / Delete user's notes |

**urls.py:**

It **maps URLs to views** like this:

|  |
| --- |

|  |  |  |
| --- | --- | --- |
| URL | View | Purpose |
| /roles/ | RoleListView | Fetch list of roles |
| /questions/<role\_id>/ | QuestionListView | Fetch questions by role |
| /progress/ | UserProgressListCreateView | Create/View progress records |
| /progress/<pk>/ | UserProgressDetailView | Edit/Delete progress |
| /notes/ | UserNoteListCreateView | Create/View notes |
| /notes/<pk>/ | UserNoteDetailView | Edit/Delete notes |

**Creating Superuser for backend:**

|  |
| --- |
| **Login to Admin Panel** |

|  |
| --- |
| <http://127.0.0.1:8000/admin/> |

**Username: Vedant**

**Email: k18ved@gmail.com**

**Pass: prepforge1806**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Quick Recap: What I've Built So Far:**

|  |
| --- |

|  |  |
| --- | --- |
| Part | Status |
| Django Backend Setup | ✅ |
| Models (Role, Question, Progress, Notes) | ✅ |
| Admin Panel | ✅ |
| APIs to fetch Roles/Questions | ✅ |
| Server Running Locally | ✅ |

**For Frontend:**

Had option of selecting Vite or React, and after researching I selected Vite, as it is the latest technology, faster server speed than normal React, and would also help by getting highlighted.

|  |  |
| --- | --- |
| Part | Status |
| Backend Django Server | Running on port 8000 |
| Frontend React (Vite) Server | Running on port 5173 |
| GitHub Repo | Clean — backend and frontend separated |
| Localhost Frontend | ✅ Live and ready |
| Localhost Backend API | ✅ Live and returning real data |

When getting the following error:

* **cd frontend**
* cd: no such file or directory: frontend

This most of the times, simply means you are in the same directory and don’t need to cd.

**Solve CORS (Cross Origin Request) Issue in Django (Backend):**

This issue arises, because backend is live on one host: 8000

And frontend is live on another host: 5173

So, the browser blocks them, so using **django-cors-headers** helps resolve this issue.

A screenshot of a computer

AI-generated content may be incorrect.

**"Back — let's continue from Questions Page!"**