Masai Job App (Full-Stack)

Instructions

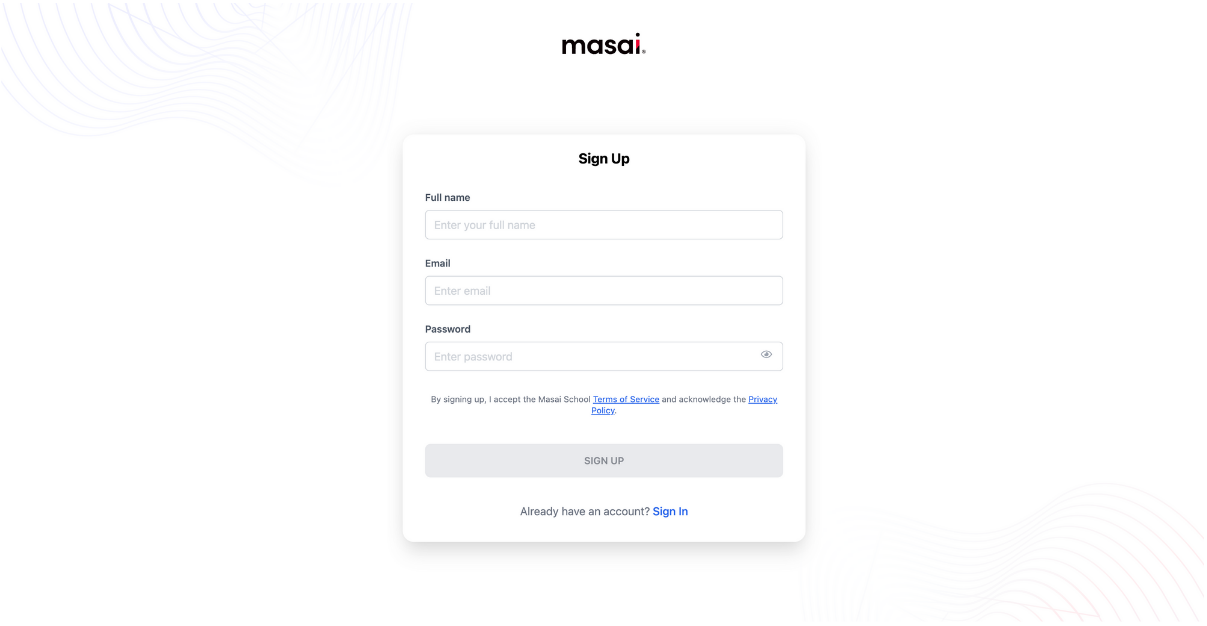
- Use react-router for routing.
- Use redux and thunk for dispatching asynchronous actions.
- Keep the code clean, commented and documented. Maintain feature based coding. (separate action and reducers - feature wise folders) example : all the action and reducer related to authentication in one folder etc.
- You are free to use any css solutions as long as it looks good. Remember this also has score
- You will have to also deploy both Frontend and the Backend (Server on Heroku and Database on Mongo Atlas)

Problem Statement:

- Your task is to build a job app for Masai, where a user can see all job openings and also as a admin, one should be able to post any job openings.
- Your App should have both user panel and admin panel.
- Both admin and user should be able to register and login

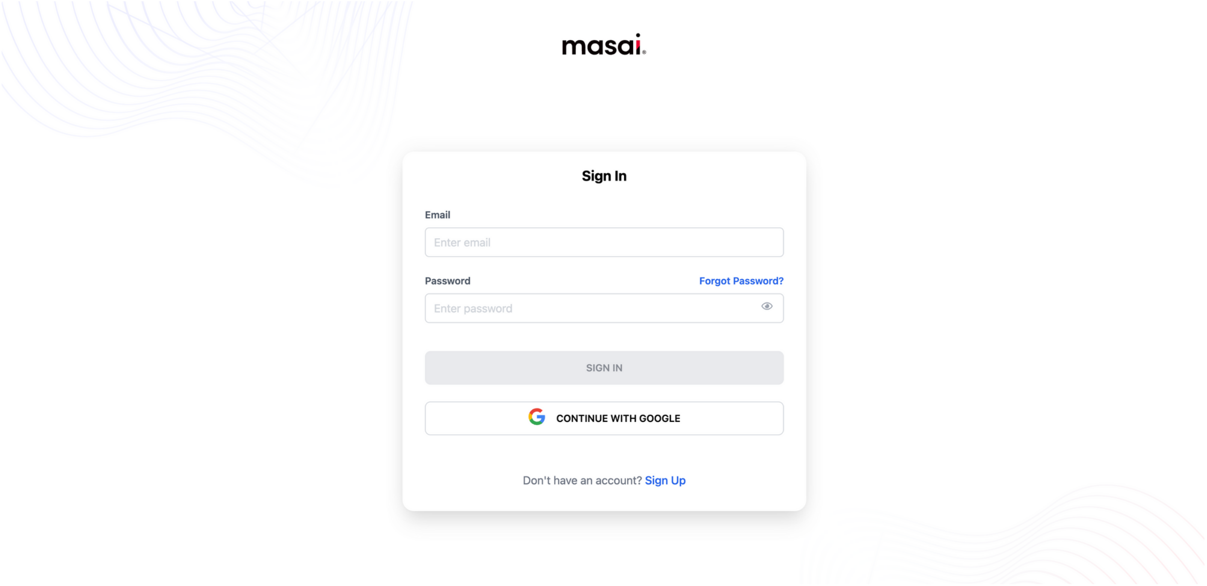
Signup and Login page

- This is the landing page of your app.
- Try to replicate exact signup and login page of masai school (You can ignore Google Oauth part and have normal Authentication with JWT etc)



By signing up, I accept the Masai School Terms of Service and acknowledge the Privacy Policy.

Already have an account? Sign In



Don't have an account? Sign Up

- User should be able to signup by filling in all above details
- Whoever signs up with domain ending with `@masaischool.com` should be considered as admin
- Whoever signs up with domain other than `@masaischool.com` should be given user access
- Store entire signup data of both user and admin in the database
- After successful signup, user and admin both should be able to Login
- If user Logs in redirect him to `Jobs page`
- If admin logs in (domain should be only `@masaischool.com` for admin) move him to `Admin page`
- Error messages should be shown for wrong credentials.

Admin page

- Admin should have two pages
  - Jobs Form page
  - Jobs Listing page

Jobs Form page:

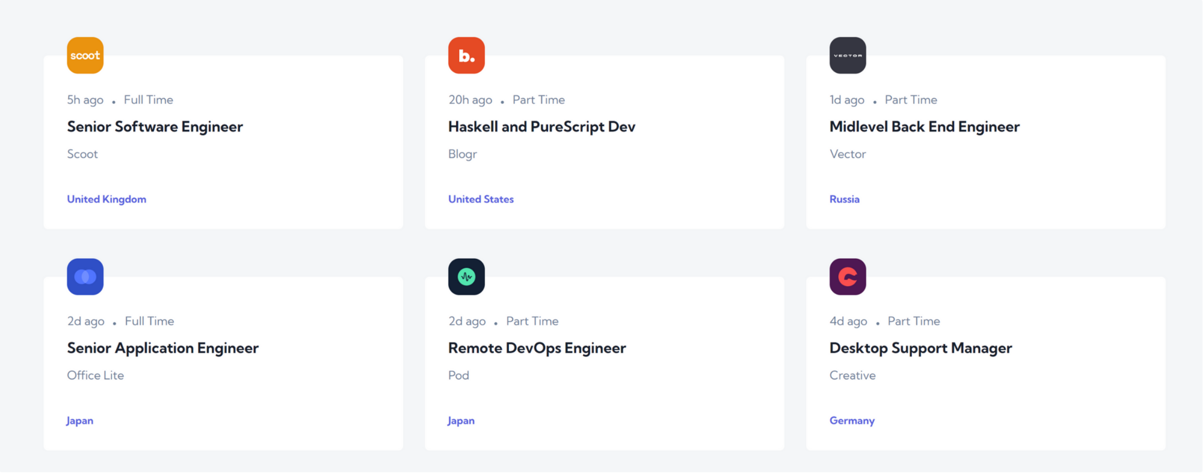
- This page should have a form where admin can post a job with following details
  - Company name
  - Position
  - Contract
  - Location
- On form submit Post data and store in the Database.
- Whatever data is posted by the admin, `admin` should be able to see it in `Jobs listing page` and `User` should be able to see it in `User Page`

Jobs Listing page:

- Whatever jobs admin posts, he should be able to see all those jobs in this page in form of table
- Admin should also be able to delete or edit particular job.

User page

- On loading of this page show some data posted by admin (getting the data from the Backend).
- UI should look exactly like below.
- Use dummy images.



5h ago • Full Time
Senior Software Engineer
Scoot
United Kingdom

20h ago • Part Time
Haskell and PureScript Dev
Blogr
United States

1d ago • Part Time
Midlevel Back End Engineer
Vector
Russia

2d ago • Full Time
Senior Application Engineer
Office Lite
Japan

2d ago • Part Time
Remote DevOps Engineer
Pod
Japan

4d ago • Part Time
Desktop Support Manager
Creative
Germany

- This page should also have following functionalities
  - Search by company name
  - Filter by location
  - Filter by contract (Part Time/ Full Time)
- Each card should also have a button “Apply”
- On clicking on button user should be able to apply for that particular job (use redux for this)
- Make another route to display all jobs which user has applied (you can use same UI component)

Note:

- Maintain flow of app as mentioned.
- Error messages should be shown, make use of any CSS library of your choice (chakra and MUI preferred)
- Use loaders.
- Good designs will fetch bonus marks.
- Submitting local host links for mock server will lead to disqualification.

Submission

- Please submit deployed link and Github link of code.

- **Push your code into `masai-repo`, don’t submit personal repo links (submitting personal repo links will lead to disqualification)**
- Please double check if deployed version works or not (run deployed version on your laptop and then submit it).
- Any issues in deployed link, will be considered null and void.
- Please verify your submissions are correct.
- Make sure you follow all instructions carefully.
- Submit before deadline.

Rubrics / Criteria to be judged upon

- HTML, CSS, React, **Redux, Backend**
- Filtering, sorting, pagination.
- Code cleanliness.
- Component structure and **Good Git Hygiene.**