Project 4 – App Tracker App

AWS S3

arn:aws:iam::769136672392:user/AppTrackerApp

ACCESS\_KEY AKIA3GFA6H2EIYO55YC7

SECRET\_ACCESS\_KEY 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

S3\_BUCKET=apptrackerapp

AWS\_REGION=us-east-2

CONFIGURATION

In the file where you interact with AWS services (such as S3), configure the AWS SDK with your credentials and region:

javascript

const AWS = require('aws-sdk');

AWS.config.update({

accessKeyId: 'YOUR\_ACCESS\_KEY\_ID',

secretAccessKey: 'YOUR\_SECRET\_ACCESS\_KEY',

region: 'YOUR\_AWS\_REGION', // For example, 'us-west-1'

});

const s3 = new AWS.S3();

Replace 'YOUR\_ACCESS\_KEY\_ID', 'YOUR\_SECRET\_ACCESS\_KEY', and 'YOUR\_AWS\_REGION' with your AWS credentials and the region where your S3 bucket resides.

Using s3 Object:

Now, you can use the s3 object to perform actions like uploading, downloading, or manipulating objects in your S3 bucket within your Node.js backend.

Frontend (Browser - React):

For the frontend (in a React component, for instance), you'll need to configure the AWS SDK with the credentials directly in your code. However, it's essential to never expose AWS credentials in your frontend code due to security concerns.

Instead, consider setting up a backend API endpoint that your frontend can communicate with to perform actions like uploading files to S3. The backend API should handle authentication and interaction with S3 using the AWS SDK configured on the server-side.

Security Note:

Avoid hardcoding AWS credentials directly into your source code. Preferably, use environment variables or a secure method (like IAM roles when deploying to AWS services like EC2, ECS, or Lambda) to manage and securely provide credentials to your server-side application. This helps prevent accidental exposure of sensitive information.

USER STORIES

* AAU I would like my index page to present a list of all my filed applications
* AAU I would like to be able to click into an application card to see a full page application detail page
* AAU I would like to be able to switch between a list view of application entries and a large calendar view of important dates like date of submission, dates to follow up, interview dates
* AAU I would like to be able to add job application entries
* AAU I would like to be able to edit job application entries
* AAU I would like to be able to delete job application entries
* AAU I would like to be able to upload documents like unique resumes and cover letters to an application entry
* AAU I would like to be able to remove documents like unique resumes and cover letters from an application entry
* AAU I would like to be able to select the particular skills that were listed as requirements for the job (selector w checkboxes)
* AAU I would like to be able to add contacts to each application entry (hiring POC, recommenders, etc)
* AAU I would like to be able to edit contacts to each application entry (hiring POC, recommenders, etc)
* AAU I would like to be able to delete contacts to each application entry (hiring POC, recommenders, etc)
* AAU I would like to be able to add skills to a selectable list
* AAU I would like to be able to edit skills in a selectable list
* AAU I would like to be able to delete skills from a selectable list
* AAU I would like the list page to have a calendar widget that shows the instances of important events (applications filed, followup dates, interviews)
* AAU I would like the main application entry list view to have a sidebar that presents the upcoming important events

MAJOR QUESTIONS

* Need help configuring AWS in app
  + Install twice (front and back), even with OUR build?
  + Is server.js the right place for the credentials?
  + Do I hardcode the KEY and SECRET KEY?
    - If not, how do I reference the .env?
* NEED help understanding HOW and WHERE front and back ends actually communicate and pass info (props?)

AAU I would like