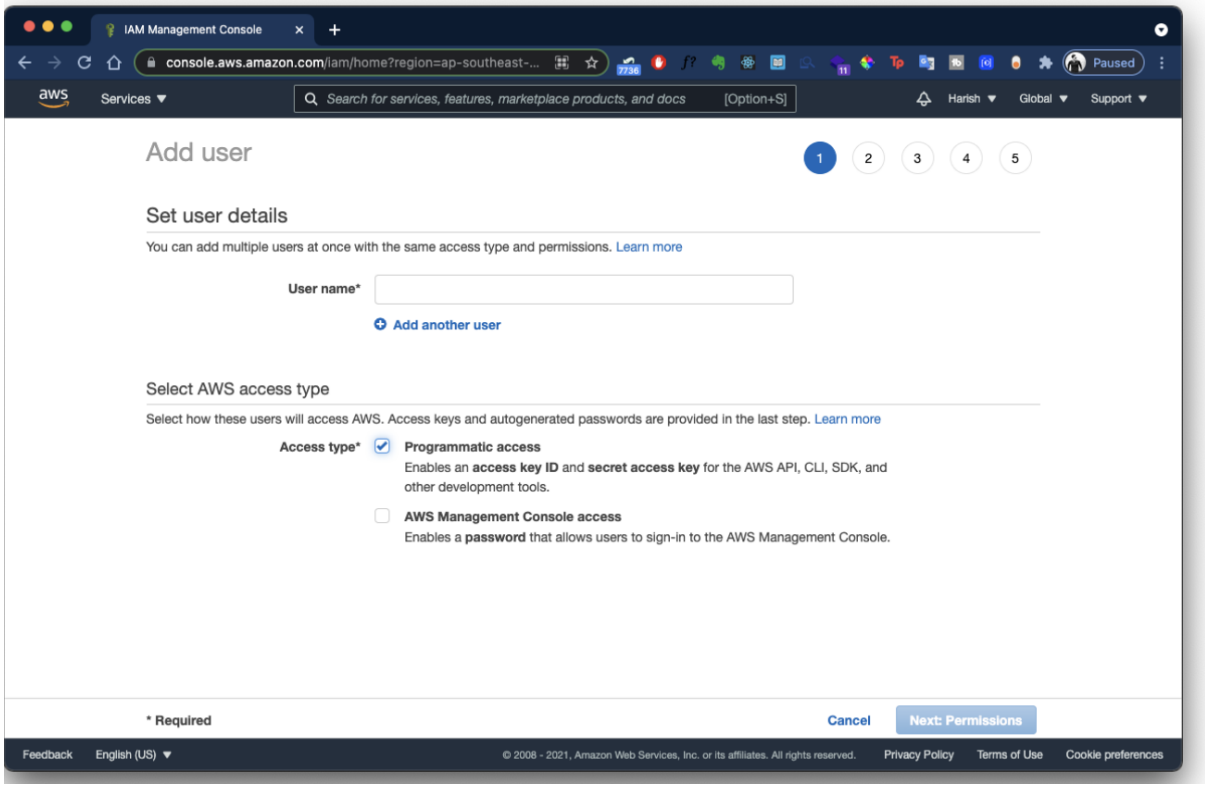
# Deploy React Apps on Amazon S3:

https://aws.plainenglish.io/deploy-react-apps-on-amazon-s3-95bb9f5870d1

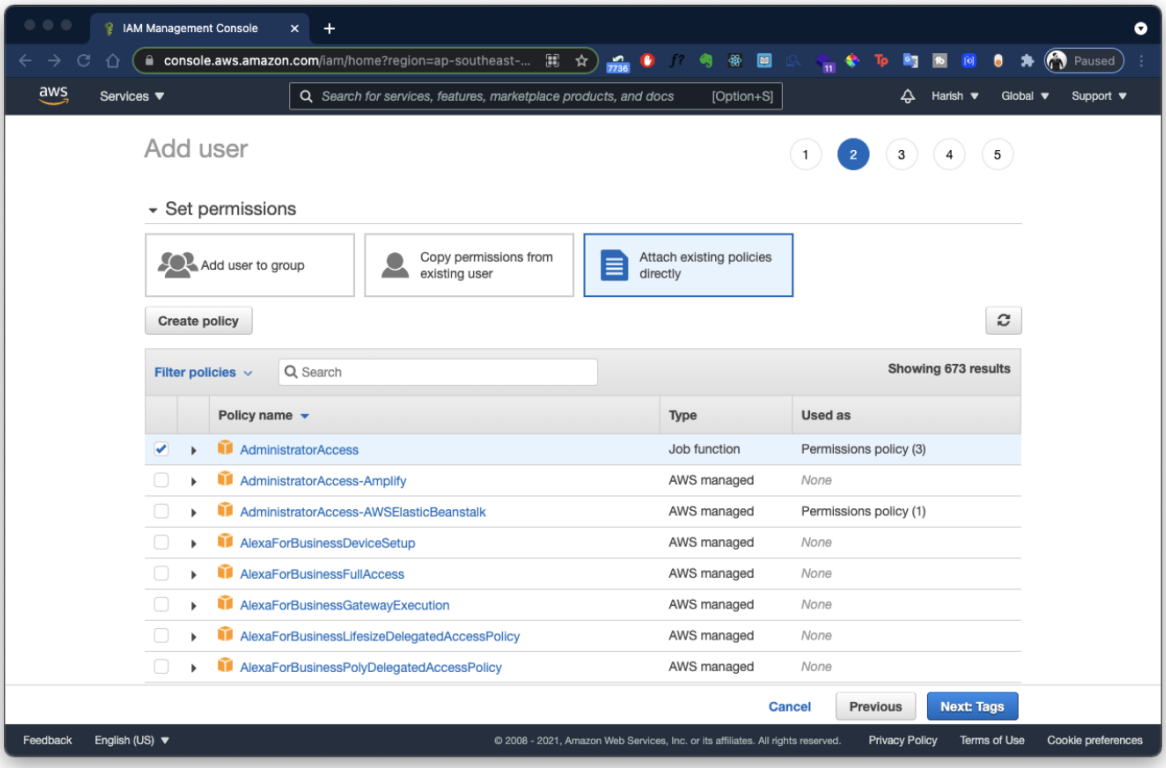
Prerequisites:
Before we go into hosting, let's get the administrative setup done.
Make sure the following are completed:
**You have a AWS Account:** Sign up for a AWS account here:
https://aws.amazon.com/resources/create-account/

**Create a AWS User:** In the AWS Console, go to the IAM tab, and go to the "Users" section on the sidebar.
Click on "Add User" and check "Programmatic access".

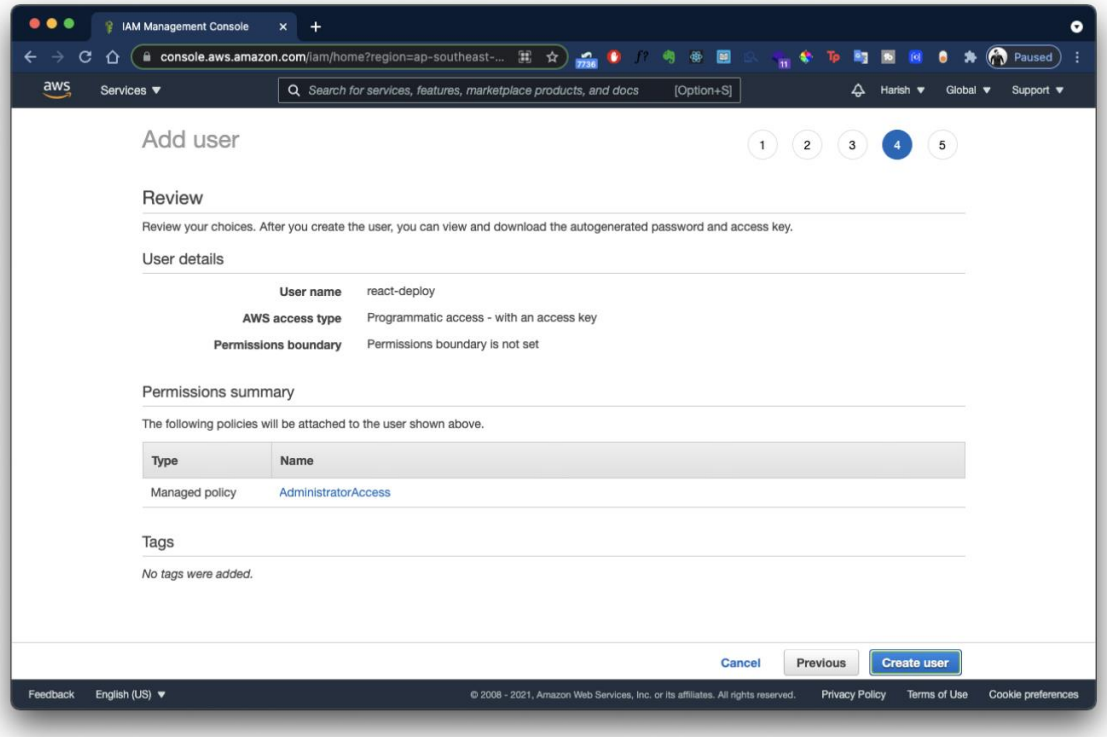



Click "Next: Permissions" and here select "Attach existing policies directly".

Check “AdministratorAccess” for the deployment purpose.




Click through the Next buttons and finally click on “Create user”.




In the final step, we receive an **Access key ID** and **Secret access key**. Download and save them to your computer as you will need this later.

Install AWS CLI

The AWS CLI is a powerful tool which can help us simplify the deployment process.

Let's install AWS CLI using [Homebrew](#) (for macOS).

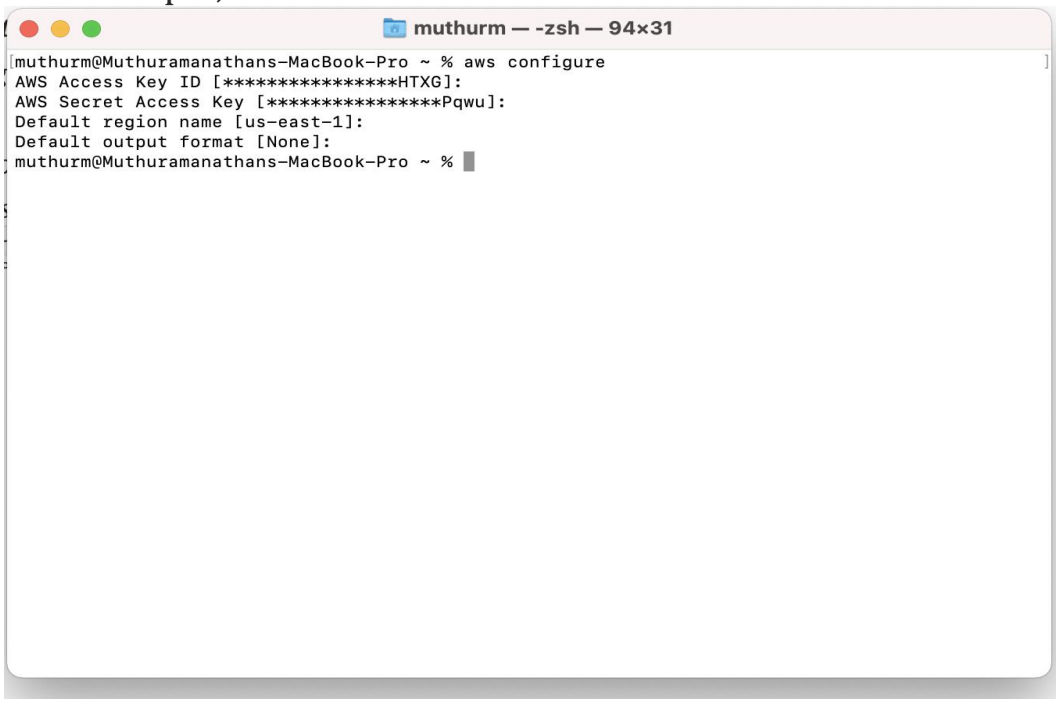
Install Homebrew if you have not already.

*Note: If you are using Windows, you can find the instructions on the official docs [here](#).*

Open your terminal, and enter `brew install awscli`.

Once the CLI tool is installed, we can configure our AWS account with `aws configure`.
This is where you will need to enter the access key information you previously downloaded. It should something like the following.

For region, you can check your URL on AWS Console and it will mention your region.
For example, mine is “us-east-1”.




## Creating a React App

You might already have your React app ready to deploy. However, if you need a sample app, you can use the following options:

1. Generate a boilerplate React app by running `npx create-react-app my-app`
2. Download my sample React app (with a form example) from [https://github.com/muthuramanathanm/reactapp-example.git](https://github.com/muthuramanathanm/reactapp-example.git)

Once done, make sure the dependencies are all installed using `yarn install` and give the app a run using `yarn start`.

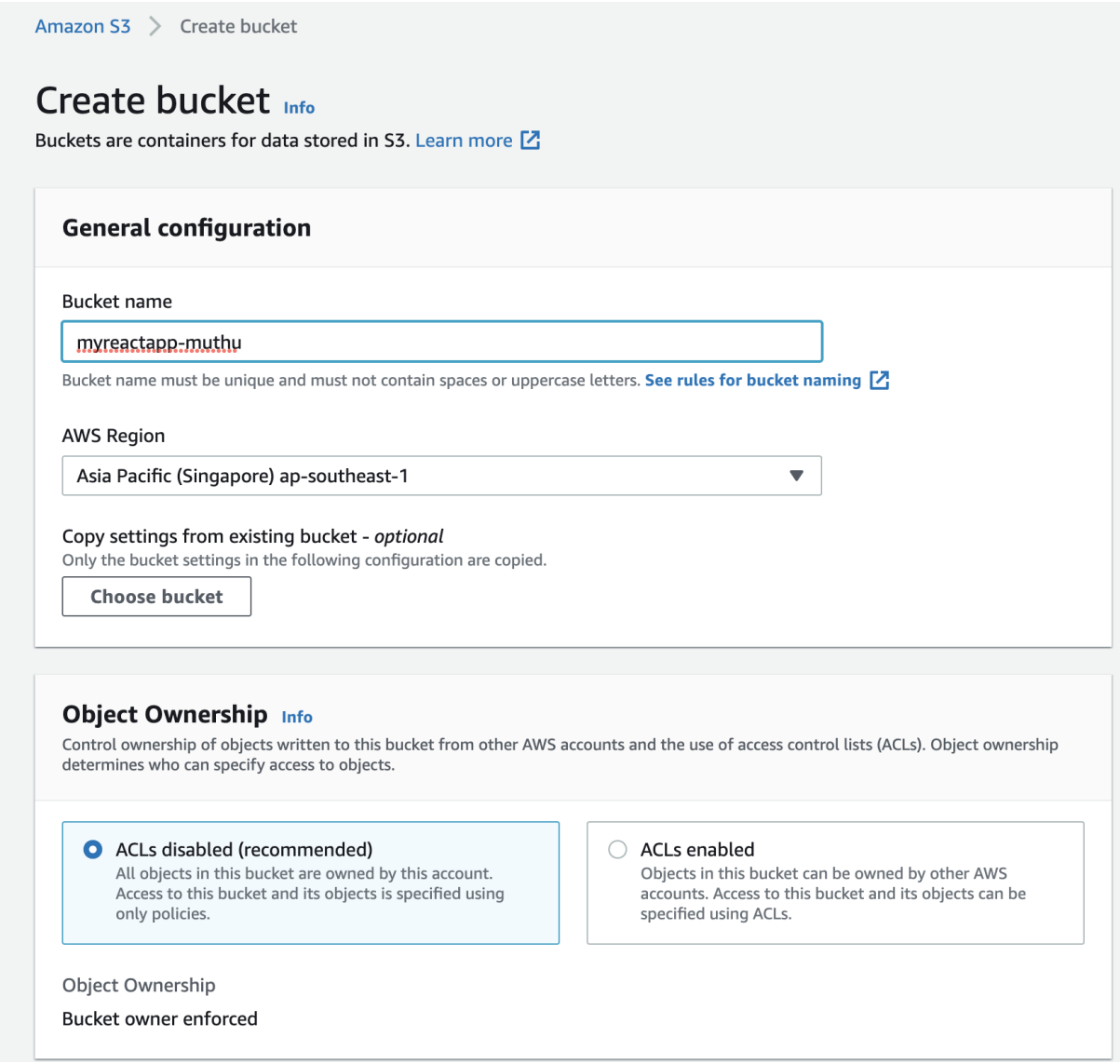
Test local on browser : http://localhost:3000/

**Setup a S3 Bucket:**

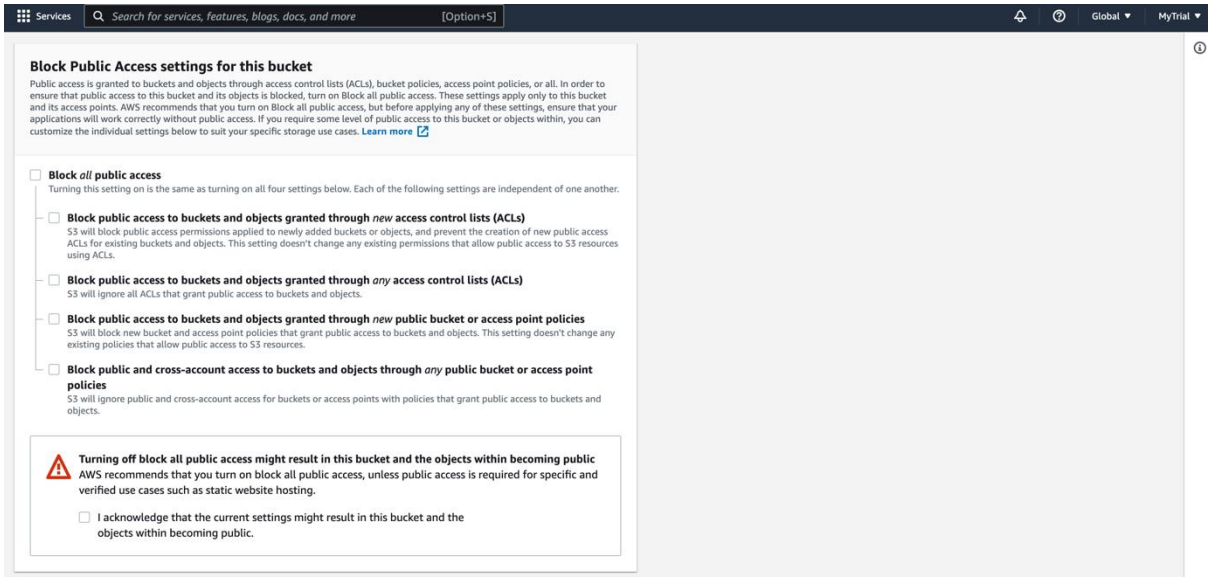
Now that we have our sample app ready, let's configure a S3 bucket to host our app.

On AWS Console, search for "S3" and go to S3 Dashboard. Click on "Create bucket" and give the bucket a name such as "my-react-app-2021" (or anything else you wish).

Note: The bucket name has to be unique across whole of S3. So choose a unique name which does not exist, else you will get an error!

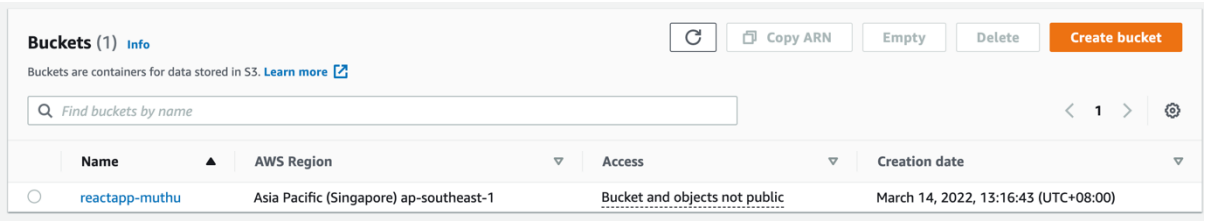



Turn on public access to the bucket since we are hosting it live.

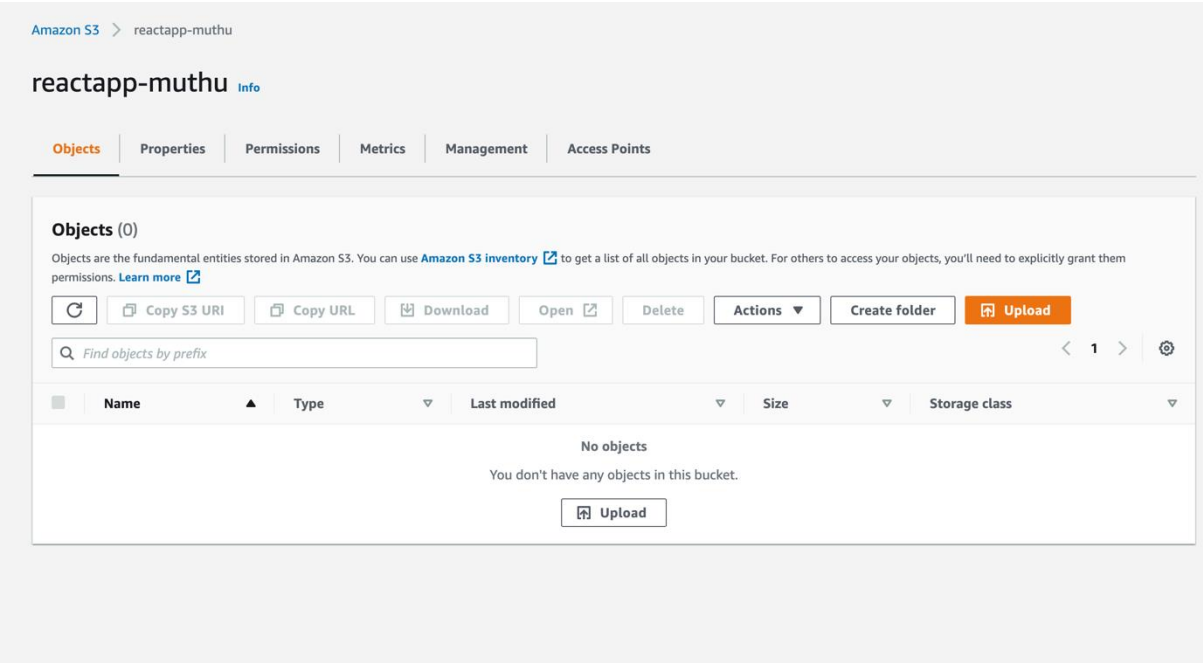



Leave the rest of the settings as what they are and click “Create bucket”.

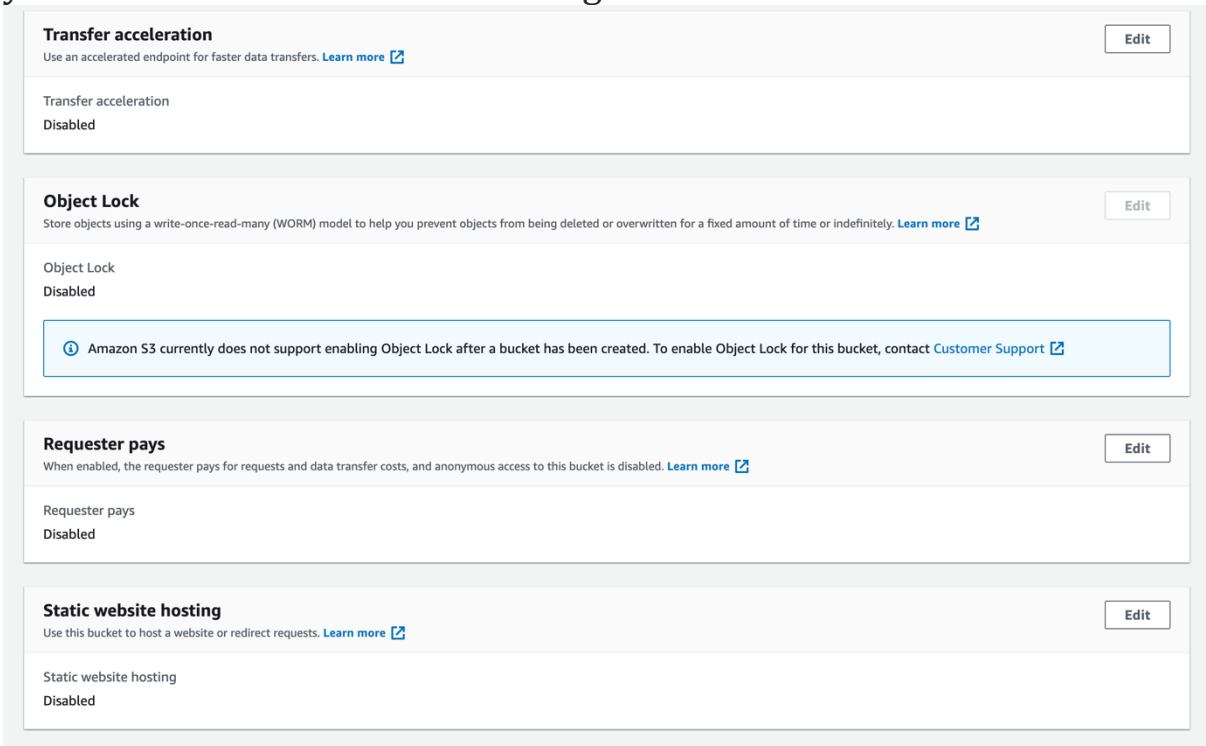
You will see an entry like the following on your S3 dashboard.



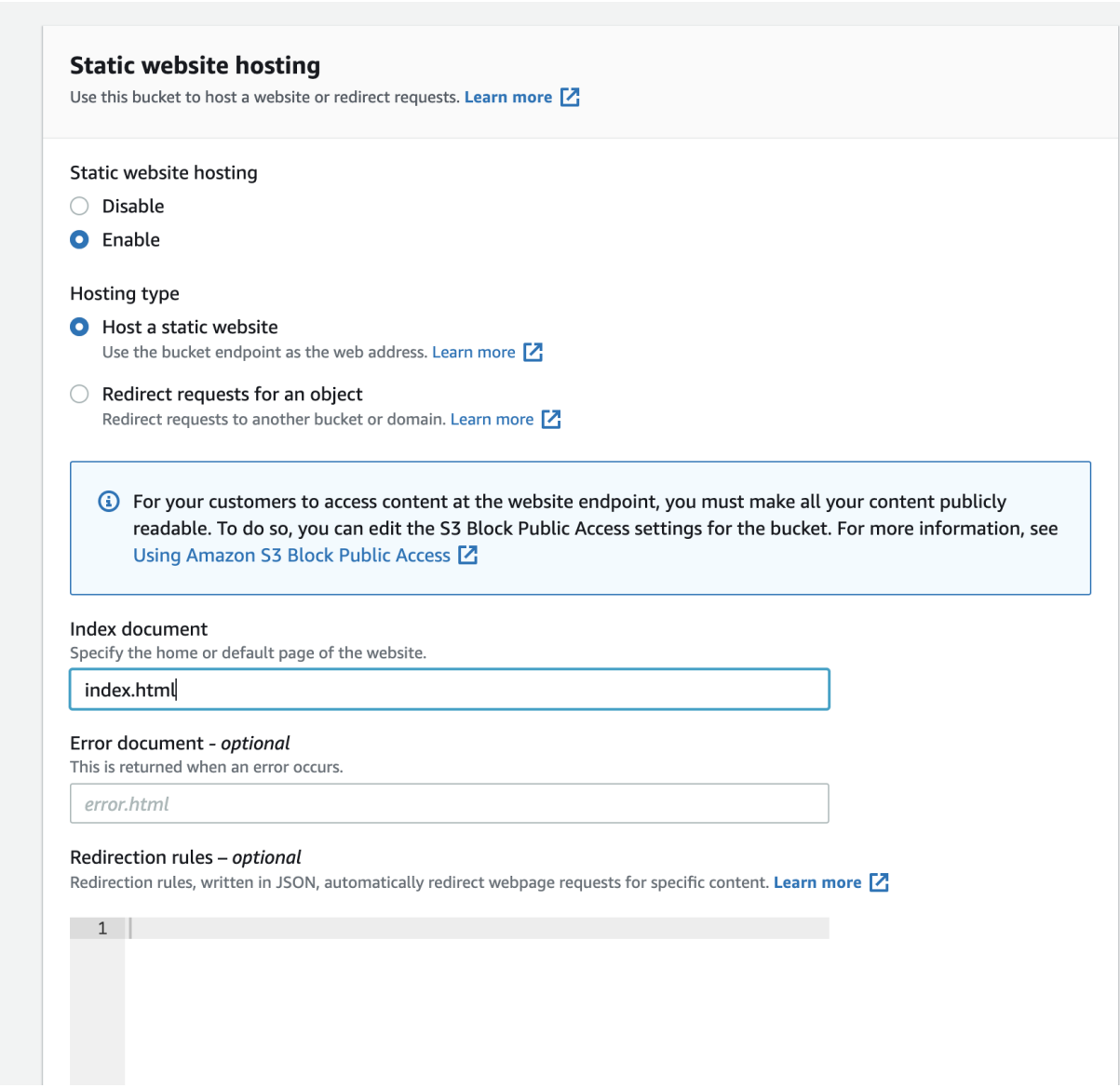

Let’s click and go into our newly created S3 bucket.




Let's click on "Properties" and scroll all the way to the bottom where you will see "Static website hosting".




Click on "Enable" and enter "index.html" under Index document.




Leave the other fields the same and click on “Save changes”.

## Deploying to S3

Now, we are ready to deploy our app to S3.

Let’s build our app using `yarn build` which helps to created an optimised production build.

The way to do that is to use the following CLI command:

```
aws s3 sync build/ s3://<your-bucket-name> --acl public-read
```

We can add this command to our package.json file too as a “deploy” script.

```
JS App.js M    JS Form.js 3, M    JS index.js M    package.json M ×
react-hook-form-example > package.json > { } scripts > deploy
10      "react": "^17.0.2",
11      "react-dom": "^17.0.2",
12      "react-hook-form": "^7.9.0",
13      "react-scripts": "4.0.3",
14      "web-vitals": "^1.0.1"
15    },
        ▷ Debug
16    "scripts": {
17      "start": "react-scripts start",
18      "build": "react-scripts build",
19      "test": "react-scripts test",
20      "eject": "react-scripts eject",
21      "deploy": "aws s3 sync build/ s3://reactapp-muthu --acl public-read"      You, seconds
22
23    },
24    "eslintConfig": {
25      "extends": [
26        "react-app",
27        "react-app/jest"
28      ]
29    },
30    "browserslist": {
31      "production": [
32        ">0.2%",
33        "not dead",
34        "not op_mini all"
35      ],
36      "development": [
37        "last 1 chrome version",
```

Next, let’s run the CLI command given above or if you have setup your publish script in package.json,
we can use `yarn deploy`.

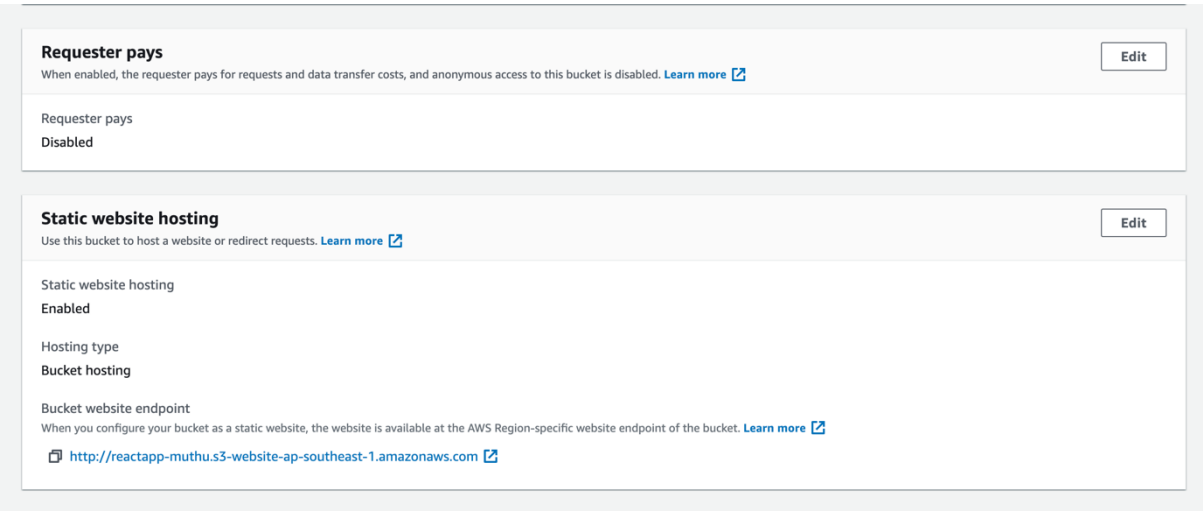
```
TERMINAL   PROBLEMS 3   OUTPUT   DEBUG CONSOLE

Muthuramanathans-MacBook-Pro:react-hook-form-example muthurm$ yarn deploy
yarn run v1.22.10
$ aws s3 sync build/ s3://reactapp-muthu --acl public-read
upload: build/asset-manifest.json to s3://reactapp-muthu/asset-manifest.json
upload: build/static/css/main.8c8b27cf.chunk.css.map to s3://reactapp-muthu/static/css/main.8c8b27cf.chunk.css.map
upload: build/index.html to s3://reactapp-muthu/index.html
upload: build/favicon.ico to s3://reactapp-muthu/favicon.ico
upload: build/static/css/main.8c8b27cf.chunk.css to s3://reactapp-muthu/static/css/main.8c8b27cf.chunk.css
upload: build/robots.txt to s3://reactapp-muthu/robots.txt
upload: build/logo192.png to s3://reactapp-muthu/logo192.png
upload: build/manifest.json to s3://reactapp-muthu/manifest.json
upload: build/static/js/2.524e871e.chunk.js.LICENSE.txt to s3://reactapp-muthu/static/js/2.524e871e.chunk.js.LICENSE.txt
upload: build/logo512.png to s3://reactapp-muthu/logo512.png
upload: build/static/js/3.d0eef414.chunk.js to s3://reactapp-muthu/static/js/3.d0eef414.chunk.js
upload: build/static/js/main.0b0d6ce0.chunk.js to s3://reactapp-muthu/static/js/main.0b0d6ce0.chunk.js
upload: build/static/js/main.0b0d6ce0.chunk.js.map to s3://reactapp-muthu/static/js/main.0b0d6ce0.chunk.js.map
upload: build/static/js/runtime-main.063f2e1d.js to s3://reactapp-muthu/static/js/runtime-main.063f2e1d.js
upload: build/static/js/3.d0eef414.chunk.js.map to s3://reactapp-muthu/static/js/3.d0eef414.chunk.js.map
upload: build/static/js/runtime-main.063f2e1d.js.map to s3://reactapp-muthu/static/js/runtime-main.063f2e1d.js.map
upload: build/static/js/2.524e871e.chunk.js to s3://reactapp-muthu/static/js/2.524e871e.chunk.js
upload: build/static/css/2.f9dc9d5b.chunk.css to s3://reactapp-muthu/static/css/2.f9dc9d5b.chunk.css
upload: build/static/css/2.f9dc9d5b.chunk.css.map to s3://reactapp-muthu/static/css/2.f9dc9d5b.chunk.css.map
upload: build/static/js/2.524e871e.chunk.js.map to s3://reactapp-muthu/static/js/2.524e871e.chunk.js.map
✨  Done in 2.02s.
Muthuramanathans-MacBook-Pro:react-hook-form-example muthurm$
```

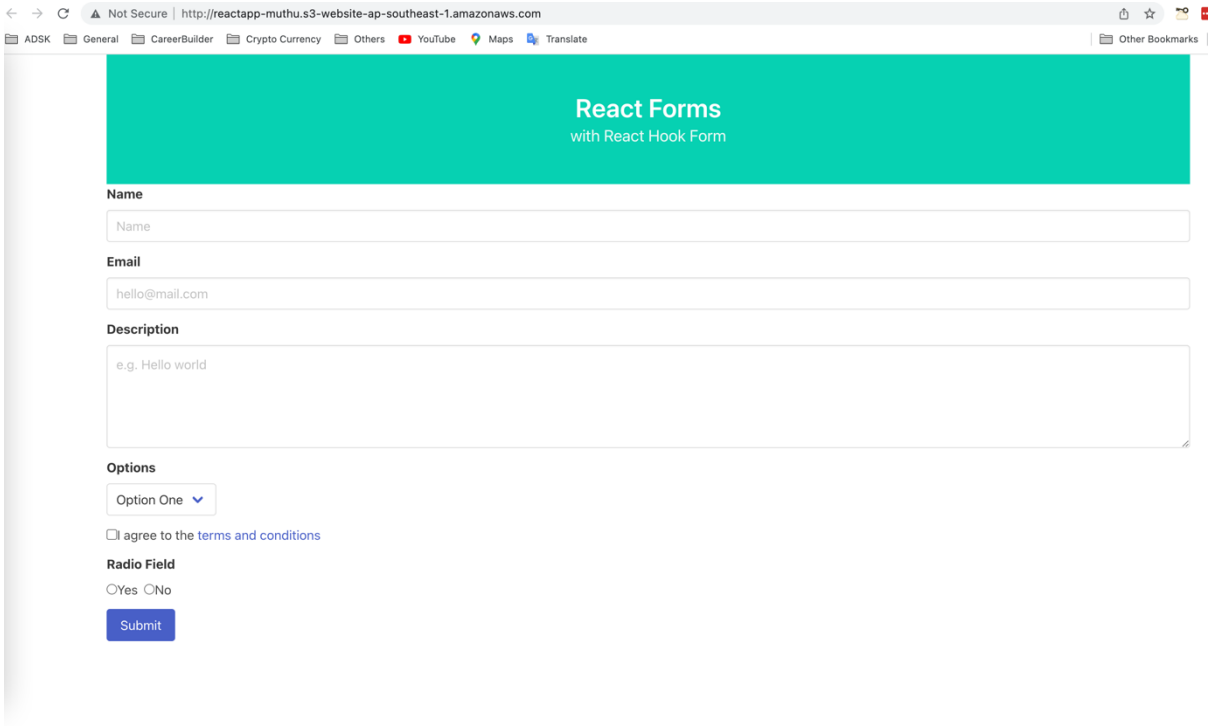
That's it, our app is now deployed!

To find where our app is hosted (the website link), go to AWS S3 console and click on the bucket you created.

Go to the "Properties" tab and scroll down all the way to the "Static website hosting" section and our URL will be there.




Click on it and you app will open up in a new tab. Here's mine!




Conclusion:

We have successfully deployed our React app to Amazon S3! This is a great choice to quickly deploy and test your React prototypes and share it with others. It's really easy and quick to deploy as well.

**Note:**

Using the IAM user credentials trying to access the S3 buckets and objects in the root user. For this we need to enable the ACL otherwise it will not allow to access them.

**Troubleshoot Error:**

If we don't enable ACL in the permission section by default then get the below error:

upload failed: build/logo512.png to s3://mansss/logo512.png **An error occurred (AccessControlListNotSupported) when calling the PutObject operation: The bucket does not allow ACLs**

Command to test:
aws configure list
aws configure list-profiles
aws s3 ls

~./aws permission denied error
cd ~./aws
chmod 777 .aws
vi credentials
remove the accesstoken from credentials