

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.

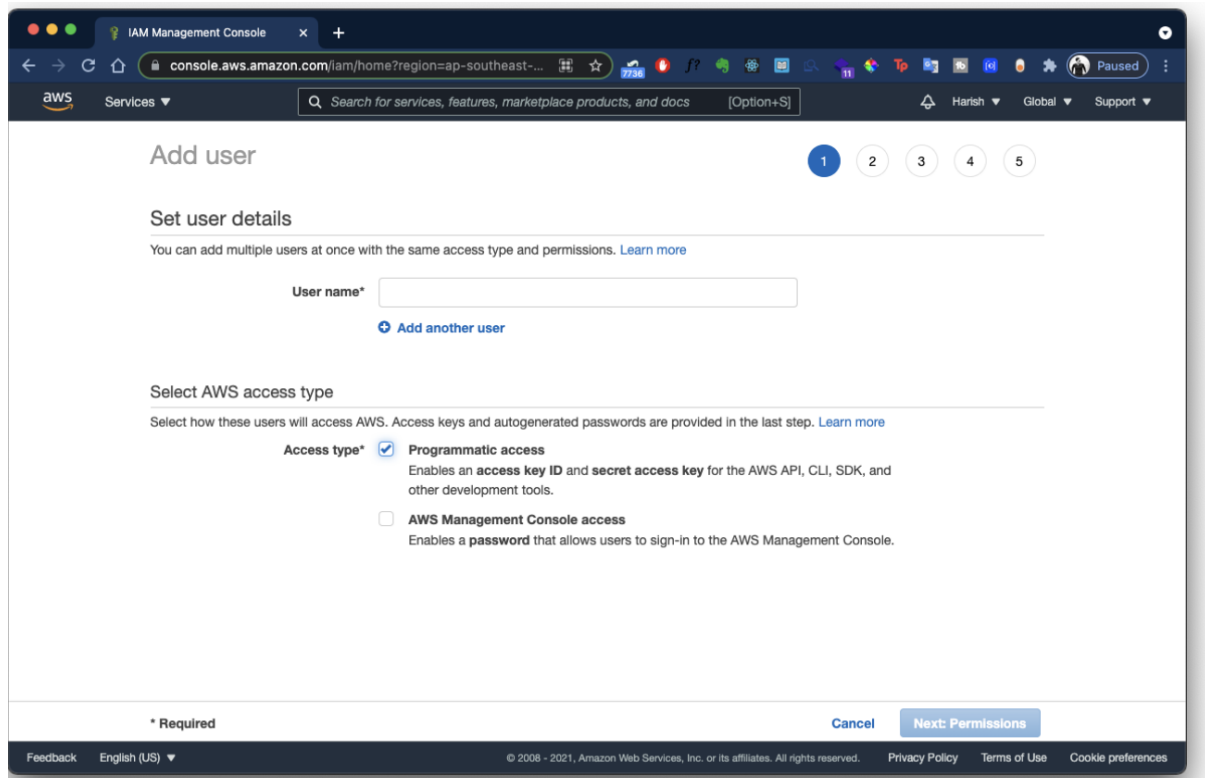
Make sure the following are completed:

You have a AWS Account: Sign up for a AWS account here:

<https://aws.amazon.com/resources/create-account/>

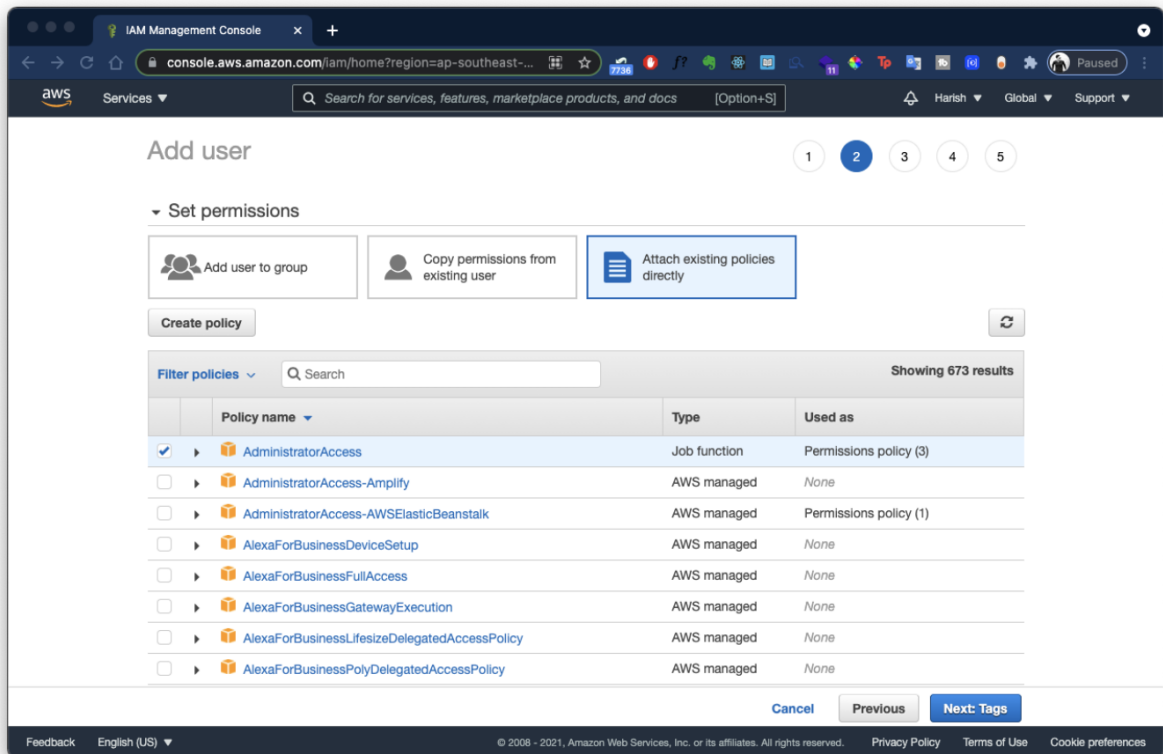
Create a AWS User: In the [AWS Console](#), go to the IAM tab, and go to the “Users” section on the sidebar.

Click on “Add User” and check “Programmatic access”.

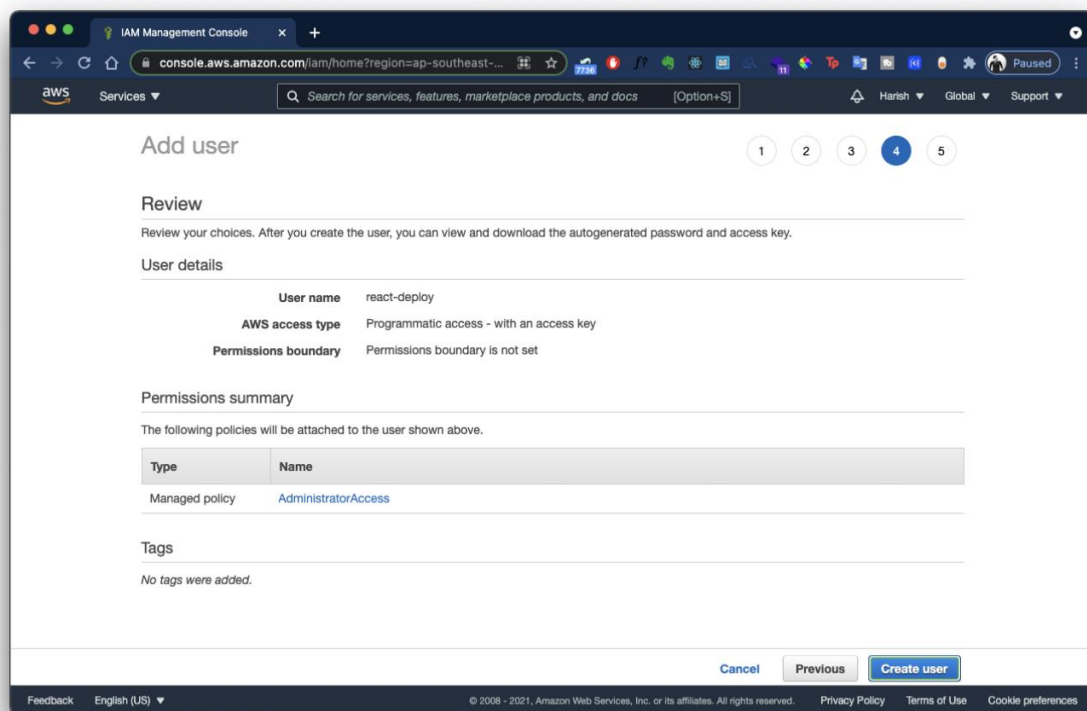
The screenshot shows the AWS IAM Management Console 'Add user' page. The browser address bar shows 'console.aws.amazon.com/iam/home?region=ap-southeast-...'. The page has a dark blue header with the AWS logo and navigation links. The main content area is titled 'Add user' and has a progress indicator with five steps, where step 1 is selected. The first step is 'Set user details', which includes a text input field for 'User name*' and a link to 'Add another user'. The second step is 'Select AWS access type', which has two options: 'Programmatic access' (selected with a checked checkbox) and 'AWS Management Console access' (unchecked). The 'Programmatic access' option includes a description: 'Enables an access key ID and secret access key for the AWS API, CLI, SDK, and other development tools.' The 'AWS Management Console access' option includes a description: 'Enables a password that allows users to sign-in to the AWS Management Console.' At the bottom of the form, there are 'Cancel' and 'Next: Permissions' buttons. The footer contains a copyright notice for 2008-2021 Amazon Web Services, Inc. and links to Privacy Policy, Terms of Use, and Cookie preferences.

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](#).

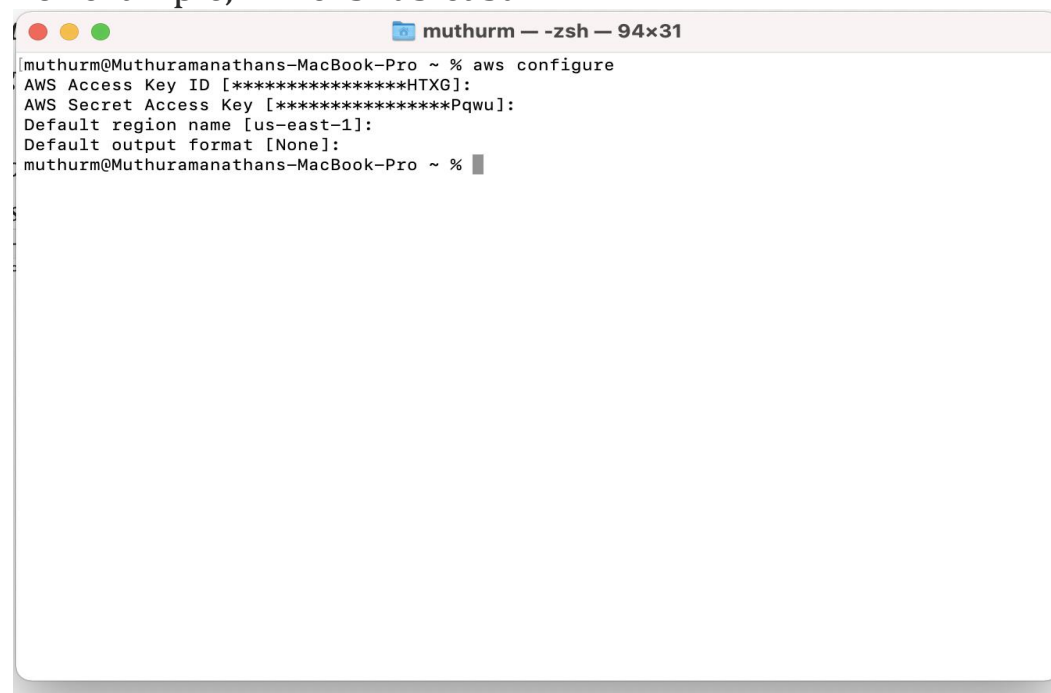
Open your terminal, and enter `brew install awscli`.

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”.

A terminal window titled 'muthurm — -zsh — 94x31' showing the execution of the 'aws configure' command. The prompt is 'muthurm@Muthuramanathans-MacBook-Pro ~ %'. The command 'aws configure' has been entered, and the terminal shows the following prompts and inputs: 'AWS Access Key ID [*****HTXG]:', 'AWS Secret Access Key [*****Pqwu]:', 'Default region name [us-east-1]:', and 'Default output format [None]:'. The prompt returns to 'muthurm@Muthuramanathans-MacBook-Pro ~ %' with a cursor at the end.

```
muthurm@Muthuramanathans-MacBook-Pro ~ % aws configure
AWS Access Key ID [*****HTXG]:
AWS Secret Access Key [*****Pqwu]:
Default region name [us-east-1]:
Default output format [None]:
muthurm@Muthuramanathans-MacBook-Pro ~ %
```

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>

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!

Create bucket [Info](#)

Buckets are containers for data stored in S3. [Learn more](#) [↗](#)

General configuration

Bucket name

myreactapp-muthu

Bucket name must be unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#) [↗](#)

AWS Region

Asia Pacific (Singapore) ap-southeast-1 ▼

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☒ **ACLs disabled (recommended)**

All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☐ **ACLs enabled**

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership

Bucket owner enforced

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

Services
Search for services, features, blogs, docs, and more
[Option+S]
Global
MyTrial

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☐ **Block all public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

☐ **Block public access to buckets and objects granted through new access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
☐ **Block public access to buckets and objects granted through any access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.
☐ **Block public access to buckets and objects granted through new public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
☐ **Block public and cross-account access to buckets and objects through any public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.
☐ I acknowledge that the current settings might result in this bucket and the objects within becoming public.

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.

Buckets (1) Info

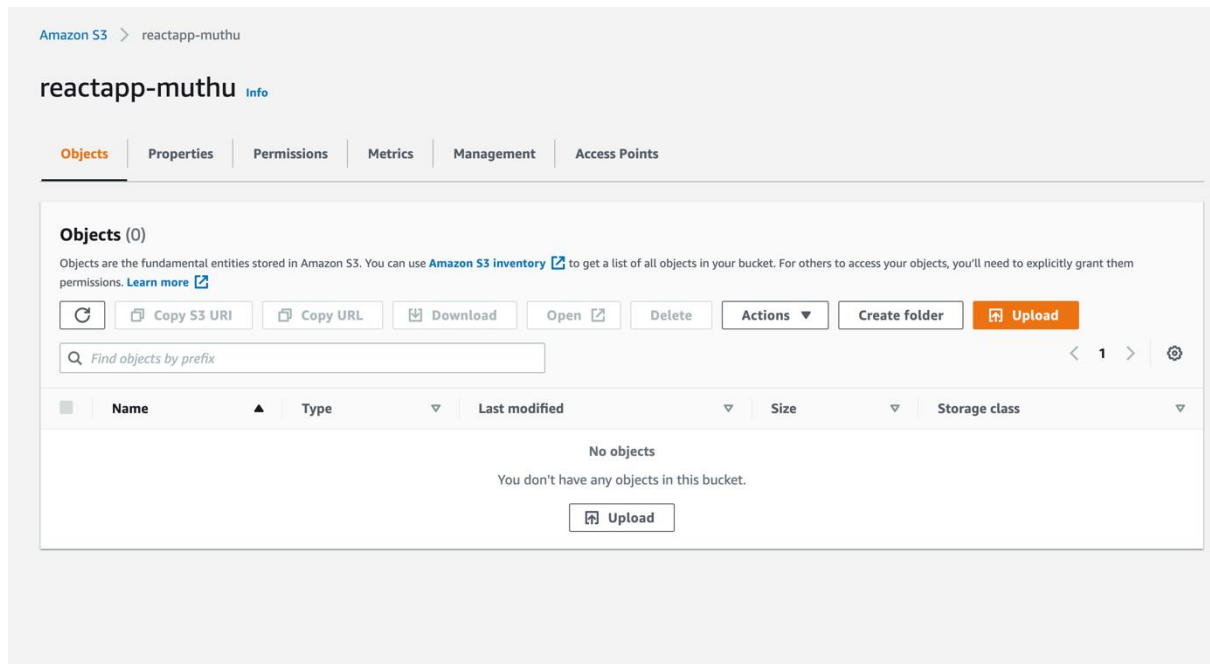
Refresh
Copy ARN
Empty
Delete
Create bucket

Buckets are containers for data stored in S3. [Learn more](#)

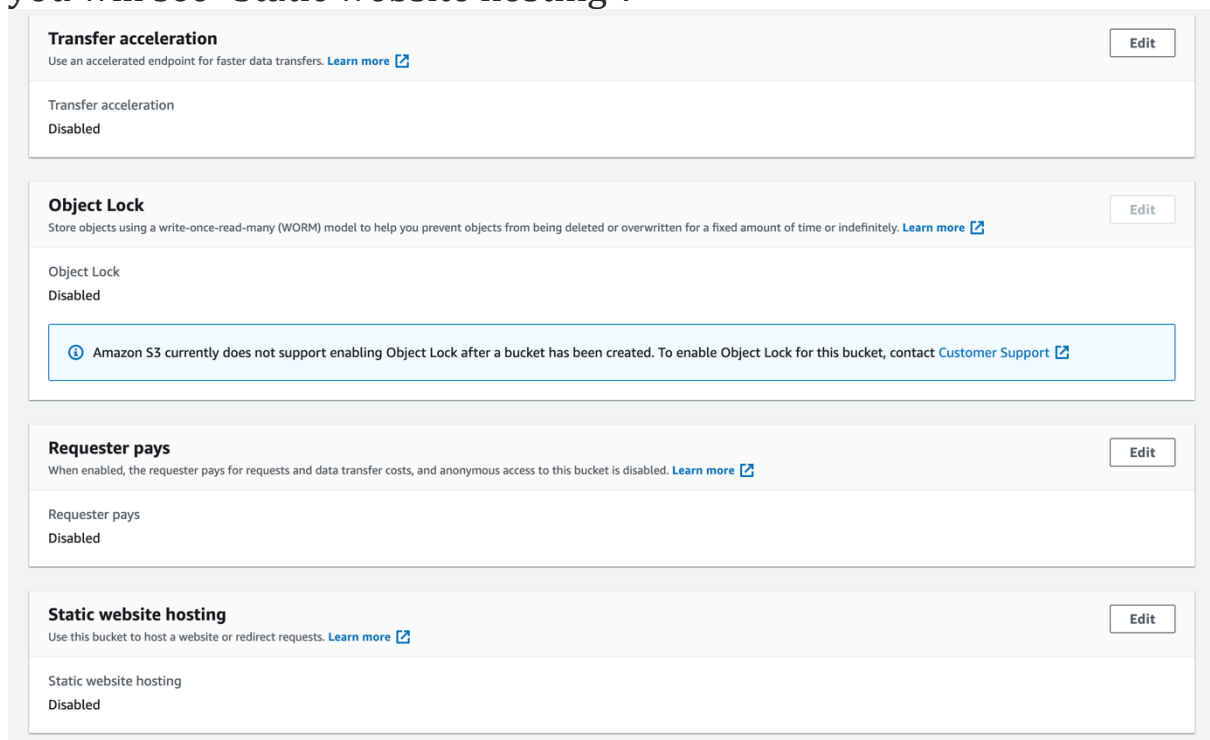
1

	Name	AWS Region	Access	Creation date
<input type="radio"/>	reactapp-muthu	Asia Pacific (Singapore) ap-southeast-1	Bucket and objects not public	March 14, 2022, 13:16:43 (UTC+08:00)

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.

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

☐ Disable

☒ Enable

Hosting type

☒ Host a static website
Use the bucket endpoint as the web address. [Learn more](#)

☐ Redirect requests for an object
Redirect requests to another bucket or domain. [Learn more](#)

i For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see [Using Amazon S3 Block Public Access](#)

Index document

Specify the home or default page of the website.

index.html

Error document - *optional*

This is returned when an error occurs.

error.html

Redirection rules – *optional*

Redirection rules, written in JSON, automatically redirect webpage requests for specific content. [Learn more](#)

1

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

Deploying to S3

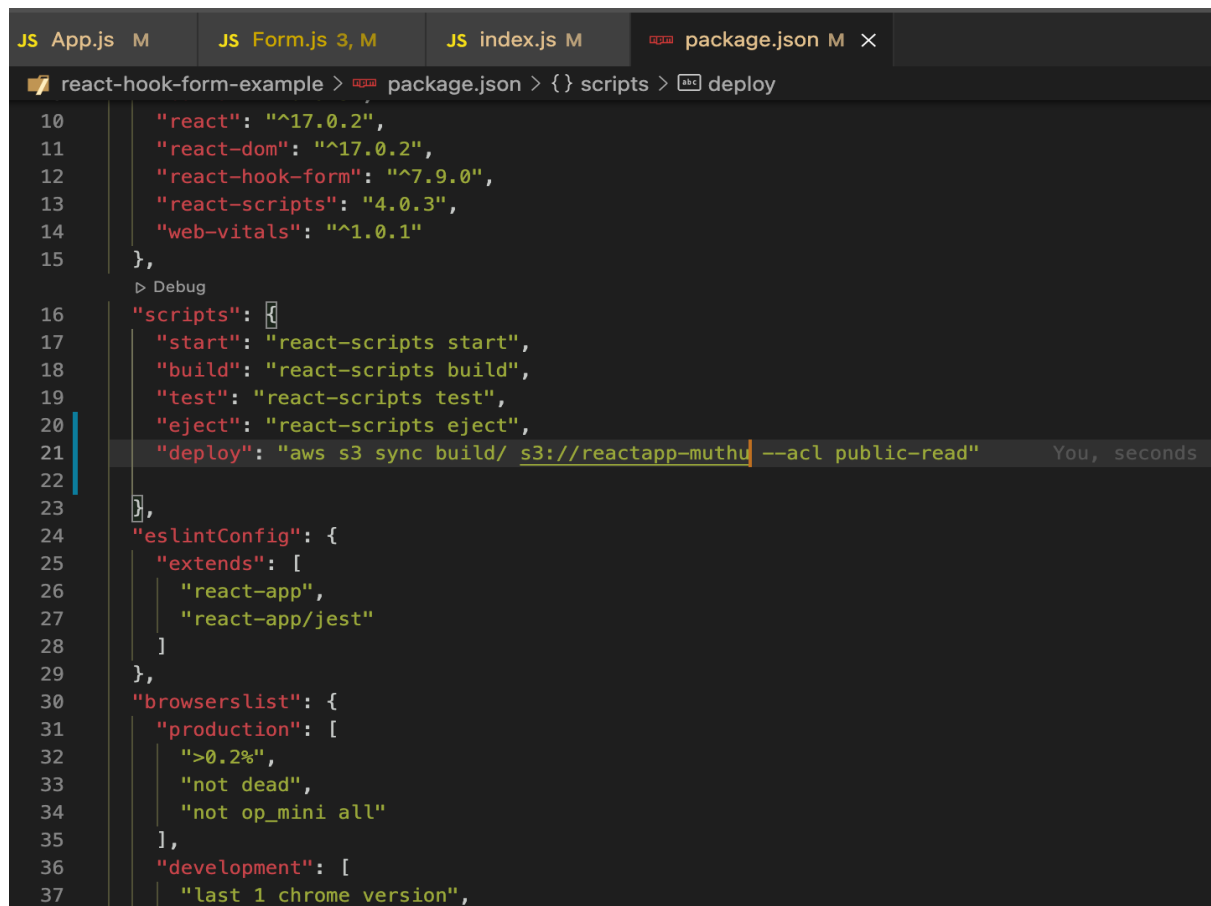
Now, we are ready to deploy our app to S3.

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.



The screenshot shows a code editor with several tabs: 'JS App.js M', 'JS Form.js 3, M', 'JS index.js M', and 'package.json M X'. The 'package.json' tab is active, showing the following content:

```
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  },
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"
22  },
23  "eslintConfig": {
24    "extends": [
25      "react-app",
26      "react-app/jest"
27    ]
28  },
29  "browserslist": {
30    "production": [
31      ">0.2%",
32      "not dead",
33      "not op_mini all"
34    ],
35    "development": [
36      "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`.

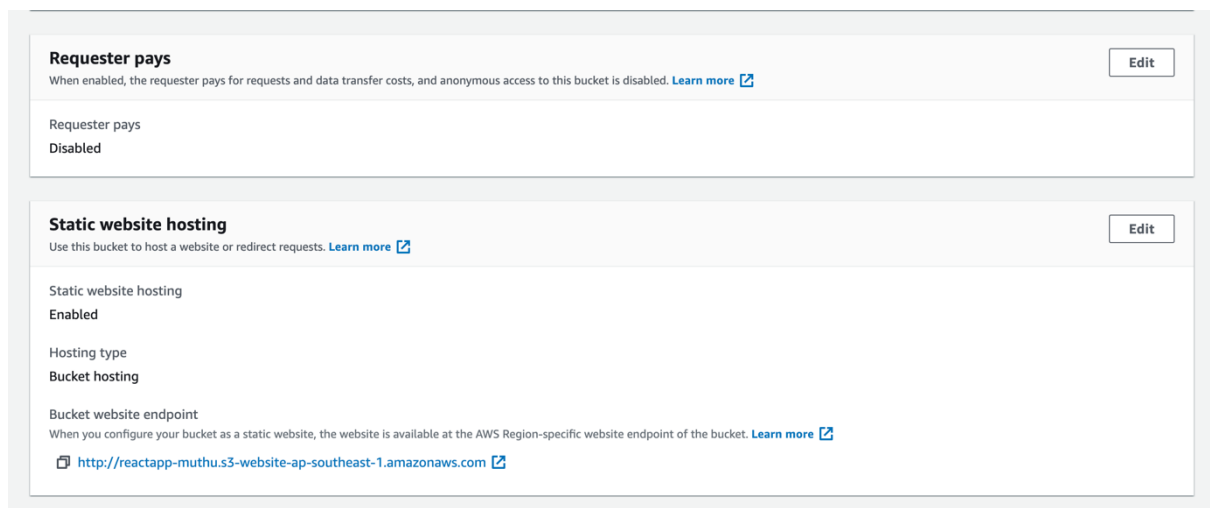
```
TERMINAL  PROBLEMS 3  OUTPUT  DEBUG CONSOLE

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!

React Forms
with React Hook Form

Name

Email
hello@mail.com

Description
e.g. Hello world

Options
Option One

☐ agree to the [terms and conditions](#)

Radio Field
☒ Yes ☐ No

Submit

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