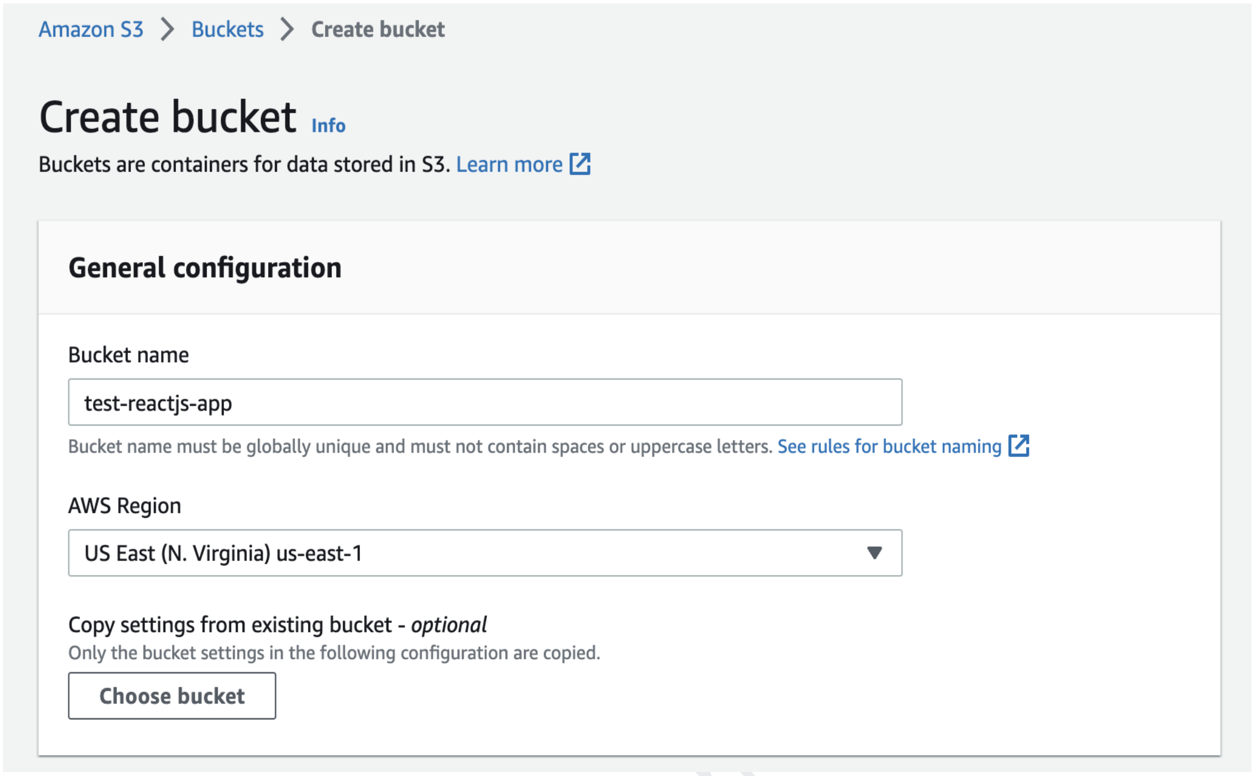
<u>#8 Task:</u> Deploy a static website on the S3 bucket to demonstrate a serverless approach.

1. Setup all the prerequisites for ReactJs App.

- Make sure you have an AWS IAM User with required S3 access to deploy the ReactJs App.
- Configure AWS CLI on the machine where you want to deploy the ReactJs App using AK, SK and Region of your choice.
- Install Node, NPM and Yarn with the LTS (Long Term Support) version.

2. Start creating the S3 bucket for hosting your ReactJs Web App.

Follow the steps given below to setup a new S3 bucket:
- Go to S3 service in AWS Console and click on **Create bucket**.


Amazon S3 > Buckets > Create bucket

Create bucket Info

Buckets are containers for data stored in S3. Learn more

General configuration

Bucket name

test-reactjs-app

Bucket name must be globally unique and must not contain spaces or uppercase letters. See rules for bucket naming

AWS Region

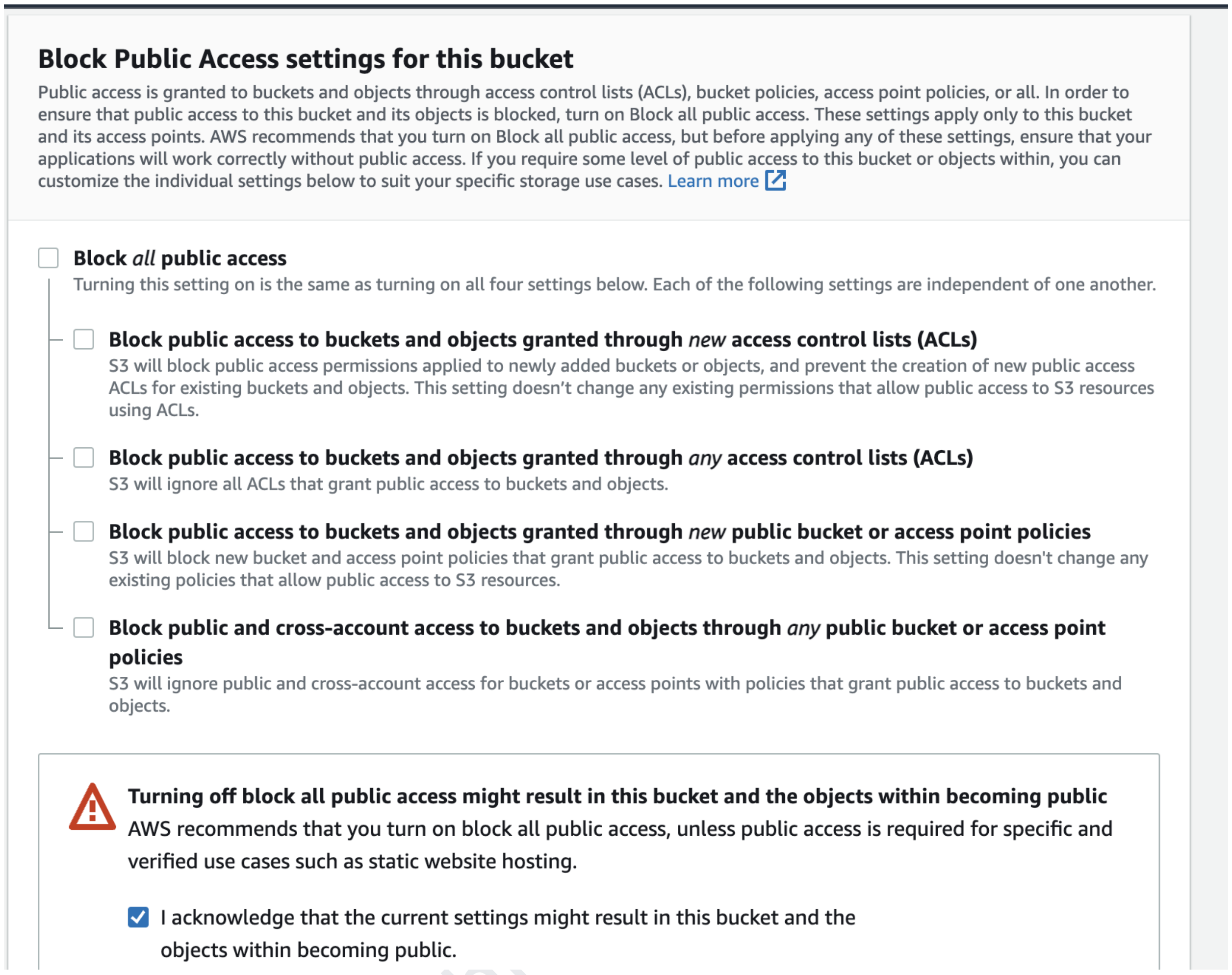
US East (N. Virginia) us-east-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Note: AWS S3 bucket names are unique as it is a global service. Hence, **make sure you choose a unique non-existing S3 bucket name** and **replace it everywhere** as given in this document where **test-reactjs-app** S3 bucket name is referred.

- Make sure you Allow Public access to let user access the ReactJs App deployed on the S3 bucket.



Block Public Access settings for this bucket

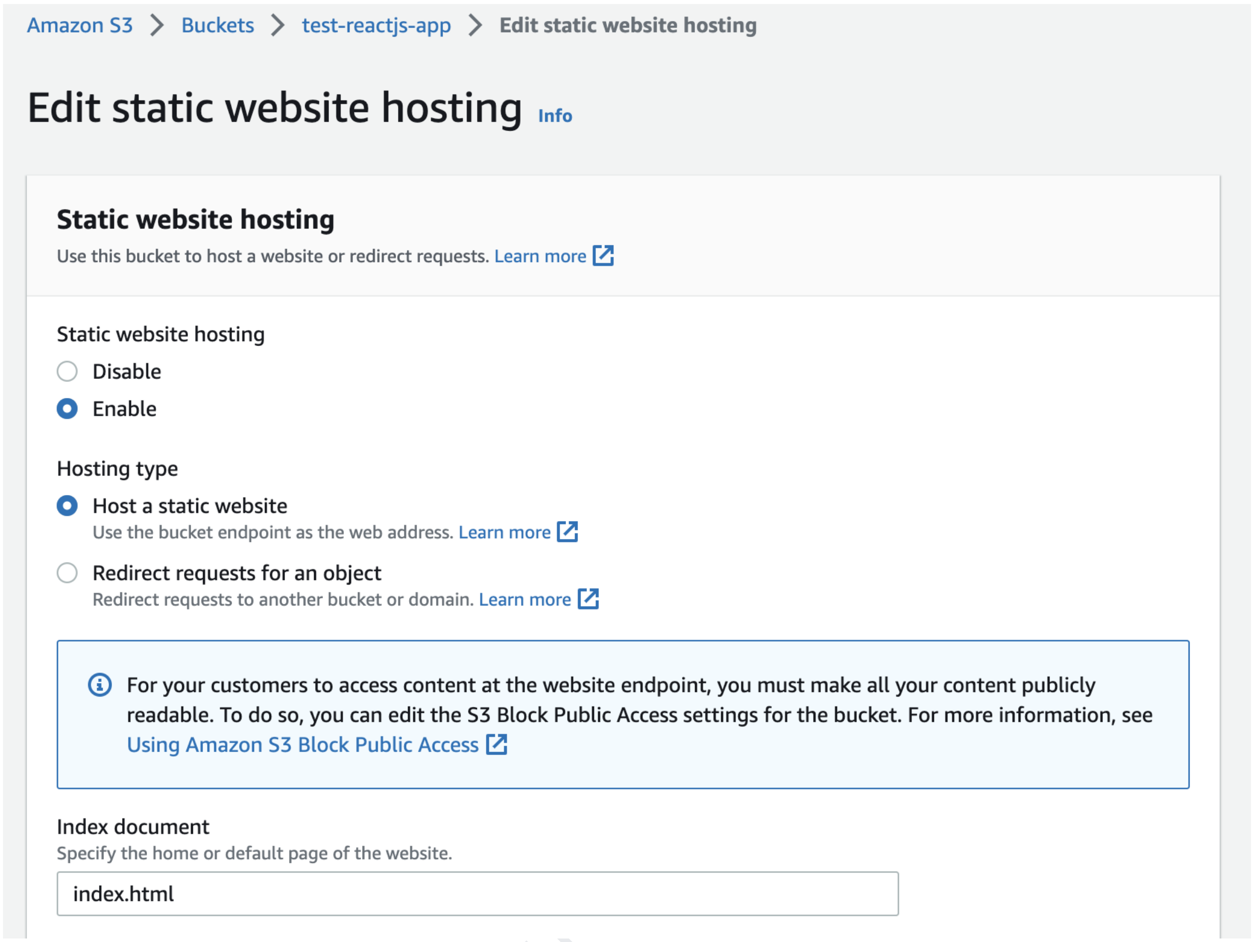
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access 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.

- [x] I acknowledge that the current settings might result in this bucket and the objects within becoming public.

- Skip all the settings as default and hit **Create bucket**.
- Go to the newly created S3 bucket and **Properties** tab.
- Click on the **Static Website Hosting** button and enter **index.html** value in the **Index document** textbox.


Amazon S3 > Buckets > test-reactjs-app > Edit static website hosting

Edit static website hosting Info

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

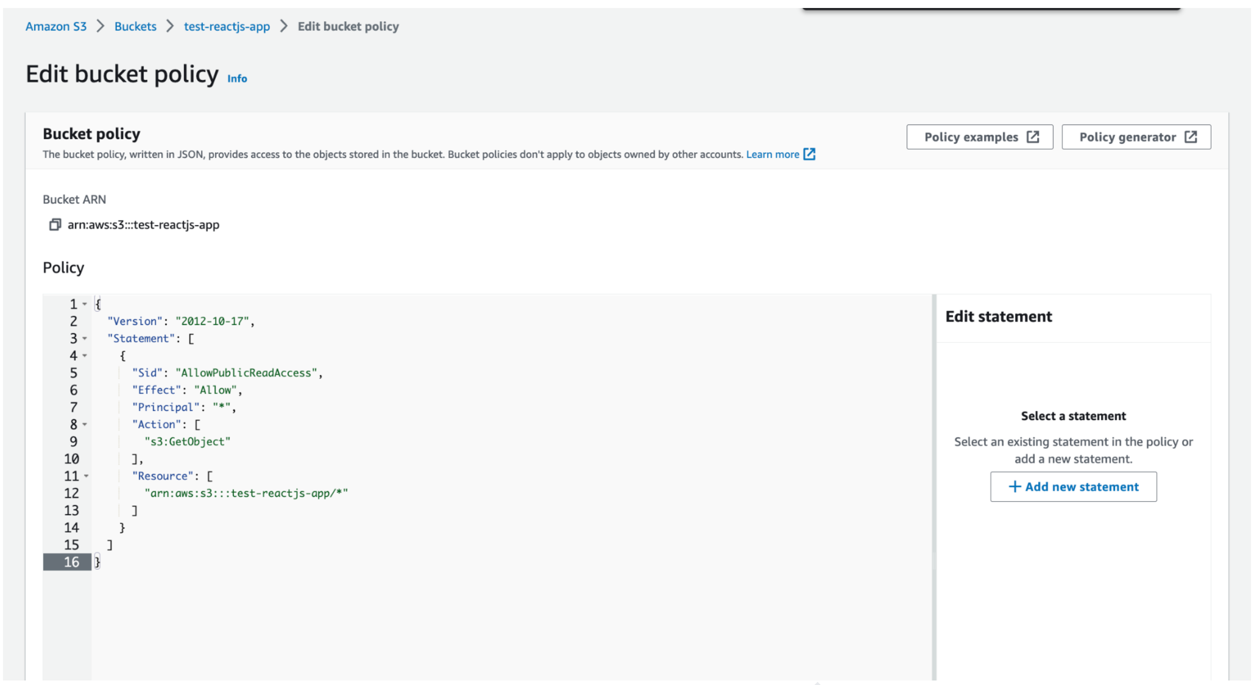
For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see Using Amazon S3 Block Public Access

Index document

Specify the home or default page of the website.

index.html

- Now, go to the **Permissions** tab under the same bucket and select the **Bucket Policy** to enter the following **JSON Access configurations**.


Amazon S3 > Buckets > test-reactjs-app > Edit bucket policy
Edit bucket policy Info
Policy examples
Policy generator
Edit statement
Select a statement
Select an existing statement in the policy or add a new statement.
+ Add new statement

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

Bucket ARN

arn:aws:s3:::test-reactjs-app

Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPublicReadAccess",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::test-reactjs-app/*"
      ]
    }
  ]
}
```

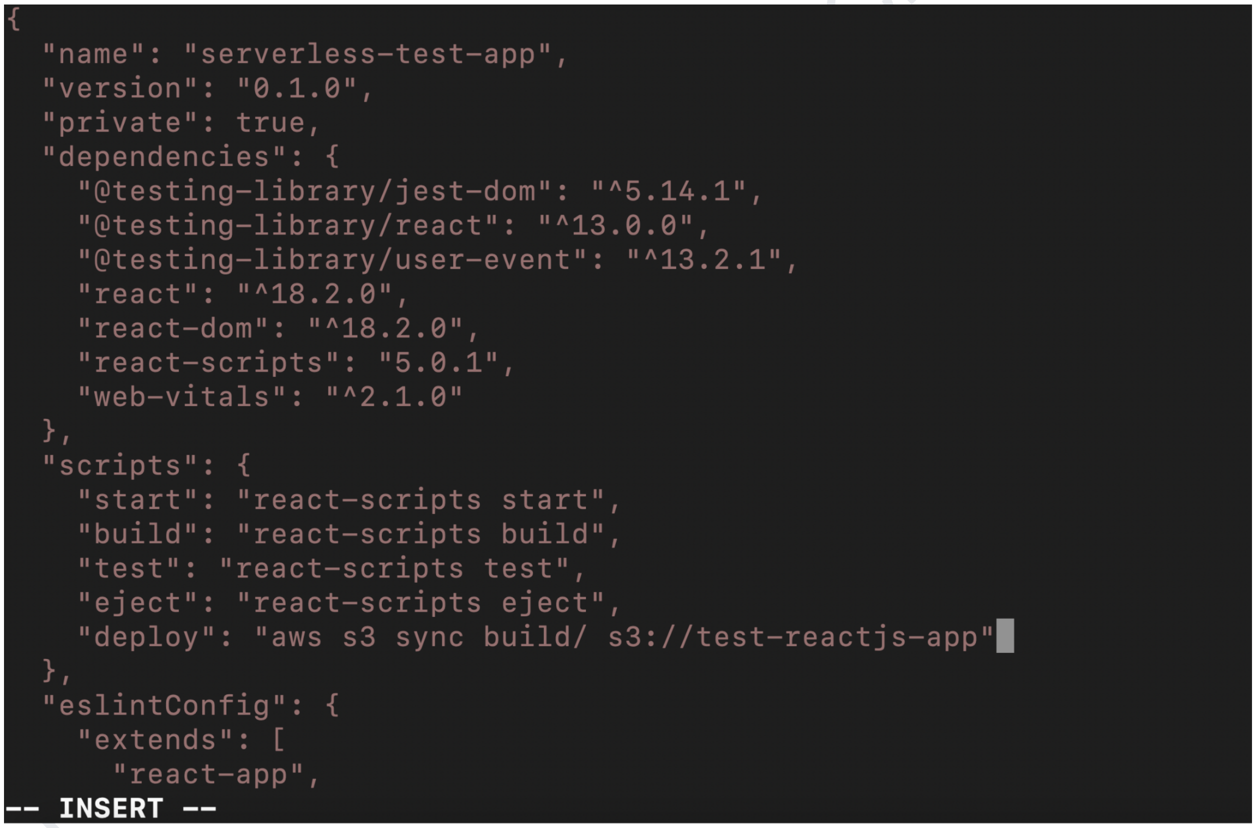
- Save the policy document and you are done with the AWS S3 setup. Below is the JSON policy given where you just need to change the **YOUR_BUCKET_NAME_HERE** value with the bucket name you created a few moments ago:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "PublicReadGetObject",
         "Effect": "Allow",
         "Principal": "*",
         "Action": "s3:GetObject",
         "Resource": "arn:aws:s3:::YOUR_BUCKET_NAME_HERE/*"
      }
   ]
}
```

3. Create a basic ReactJs Web App & Deploy it to the hosted S3 Bucket.

Since you have prepared the S3 bucket, you can start creating a very basic default provided ReactJs App by executing the below commands:

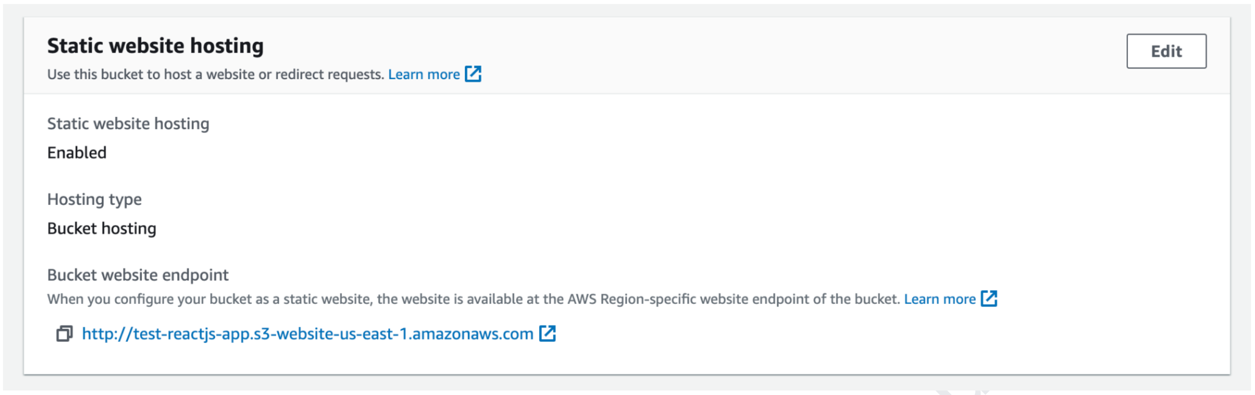
- $ node -v
- $ npm -v
- $ yarn create react-app serverless-test-app
- $ cd serverless-test-app
- $ yarn start (test the basic App on local)
- Now, to deploy the prepared & locally test ReactJs Web App to the hosted S3 Bucket, add deploy script in the package.json file.



```
{
  "name": "serverless-test-app",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.14.1",
    "@testing-library/react": "^13.0.0",
    "@testing-library/user-event": "^13.2.1",
    "react": "^18.2.0",
    "react-dom": "^18.2.0",
    "react-scripts": "5.0.1",
    "web-vitals": "^2.1.0"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject",
    "deploy": "aws s3 sync build/ s3://test-reactjs-app"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
-- INSERT --
```

- $ Yarn build && yarn deploy

4. Finally you can test your deployed ReactJs Web App from the below AWS S3 Bucket Property tab:



Static website hosting

Use this bucket to host a website or redirect requests. Learn more

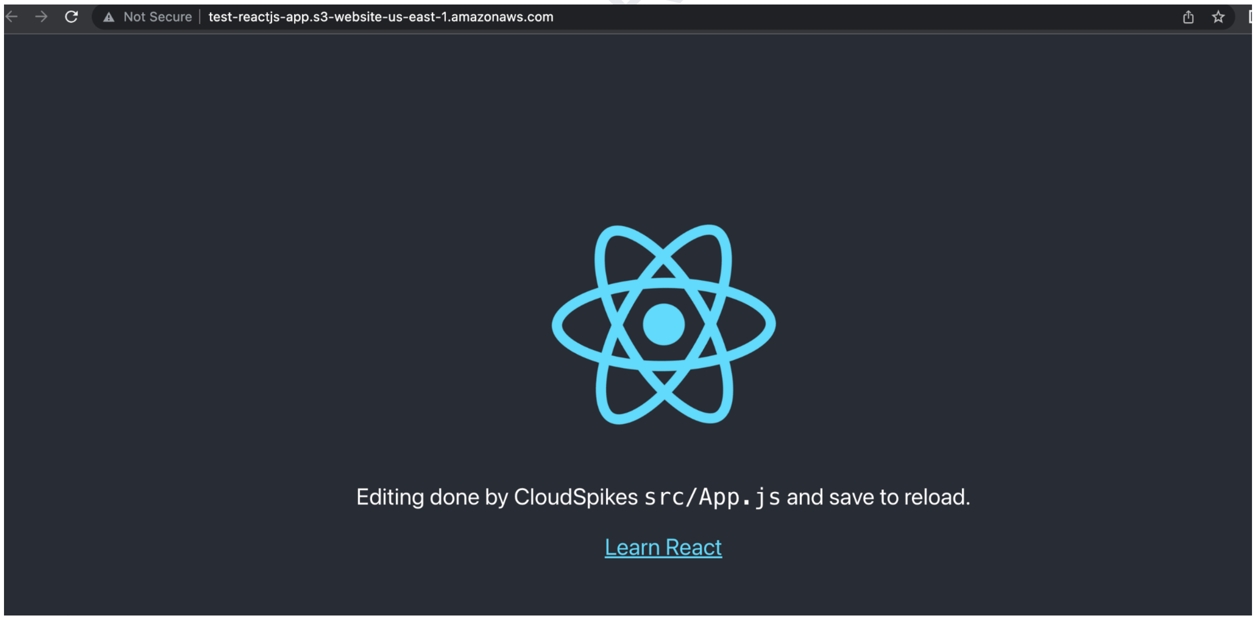
Static website hosting
Enabled

Hosting type
Bucket hosting

Bucket website endpoint
When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. Learn more

http://test-reactjs-app.s3-website-us-east-1.amazonaws.com

Hit the Bucket website endpoint link and you can see the deployed ReactJs Web App with the same view as it was found on your local environment while testing.



Not Secure | test-reactjs-app.s3-website-us-east-1.amazonaws.com

Editing done by CloudSpikes `src/App.js` and save to reload.