

Host a react application on an S3 Bucket and deliver it through cloudfront.

To host a react application, we first created a react app using following commands:

Sudo apt install npm

Npx create-react-app react-bijay

Cd react-bijay

Npm start

```
Compiled successfully!

You can now view react-bijay in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.1.67:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

^Cbj@batman:~/react/react-bijay$ npm run build
> react-bijay@0.1.0 build /home/bj/react/react-bijay
> react-scripts build

Creating an optimized production build...
Compiled successfully.

File sizes after gzip:

  41.34 KB  build/static/js/2.6bdd5a89.chunk.js
  1.63 KB   build/static/js/3.1d00e890.chunk.js
  1.17 KB   build/static/js/runtime-main.d2fad33b.js
  596 B     build/static/js/main.e0d3b047.chunk.js
  556 B     build/static/css/main.a617e044.chunk.css

The project was built assuming it is hosted at /.
You can control this with the homepage field in your package.json.

The build folder is ready to be deployed.
You may serve it with a static server:

  npm install -g serve
  serve -s build










Find out more about deployment here:
```

Now, for static build, we use the command: *npm run build*

This creates a build folder, with necessary static files to host the app. These files are uploaded on the s3 bucket which we created in previous assignment: intern-bijaykandel37

First, I made the s3 bucket publicly accessible by setting its permission to the public. Within the properties tab, Static Website hosting is enabled with type bucket hosting and the link of the index.html object is mentioned.

And then Upload button to upload the static files and folders from the build directory and all contents of the build directory are uploaded to the s3 bucket.

| <input type="checkbox"/> | Name ▲ | Type ▼ | Last modified ▼ | Size ▼ | Storage class ▼ |
|--------------------------|--|--------|---|---------|-----------------|
| <input type="checkbox"/> |  favicon.ico | ico | December 14, 2021, 19:05:57 (UTC+05:45) | 3.8 KB | Standard |
| <input type="checkbox"/> |  index.html | html | December 14, 2021, 19:05:58 (UTC+05:45) | 3.0 KB | Standard |
| <input type="checkbox"/> |  logo192.png | png | December 14, 2021, 19:05:59 (UTC+05:45) | 5.2 KB | Standard |
| <input type="checkbox"/> |  logo512.png | png | December 14, 2021, 19:06:00 (UTC+05:45) | 9.4 KB | Standard |
| <input type="checkbox"/> |  make_public_zabbixdb-2021-12-12.sql.gz | gz | December 12, 2021, 22:44:20 (UTC+05:45) | 3.9 MB | Standard |
| <input type="checkbox"/> |  manifest.json | json | December 14, 2021, 19:06:01 (UTC+05:45) | 492.0 B | Standard |
| <input type="checkbox"/> |  robots.txt | txt | December 14, 2021, 19:06:02 (UTC+05:45) | 67.0 B | Standard |
| <input type="checkbox"/> |  static/ | Folder | - | - | - |
| <input type="checkbox"/> |  zabbixdb-2021-12-12.sql.gz | gz | December 12, 2021, 22:44:09 (UTC+05:45) | 3.9 MB | Standard |

Now, we have to configure the cloudfront for this s3 bucket.

We create the Distribution for CloudFront service by Create Distribution button and then providing Origin domain name:

Settings

Origin domain
Choose an Amazon Web Services origin, or enter your origin's domain name.

Origin path - optional [Info](#)
Enter a URL path to append to the origin domain name for origin requests.

Name
Enter a name for this origin.

Default root object is set as index.html and other settings are left default and Create distribution.

On the error pages tab of the newly created distribution, we created a custom error response.

CloudFront > ... > Create error page response

HTTP error code

Customize the custom error response when the origin sends this error code.

404: Not Found

Error caching minimum TTL

Enter the error caching minimum time to live (TTL), in seconds.

10

Customize error response

Send a custom error response instead of the error received from the origin.

☐ No

☒ Yes

Response page path

Enter the path to the custom error response page.

/index.html

HTTP Response code

Choose the HTTP status code to return to the viewer. CloudFront can return a different response than what it received from the origin.

200: OK

And this is created and now after a few minutes we can access the index.html page which we hosted using cloudfront dns.

