**Deployment Instruction Document**:

This document provides step-by-step instructions to create build artifacts and deploy them for the PostalCodeAPI and PostalCodeWeb applications.

**1. Create Build Artifacts for PostalCodeAPI:**

a. Clone the git repository available at  **https://github.com/maringanti1/PostalCodeLookupAPI.git**

b. Open the PostalCode.API.sln file in Visual Studio.

c. Clean the solution and build the projects, which will download all the dependent NuGet packages.

d. After a successful build, navigate to the BIN folder and create a zip of all files.

e. Ensure that the zip contains all the files directly, without having them in a folder inside the zip.

**2. Create Build Artifacts for PostalCodeWeb:**

a. Check the configuration file and cross-check the configurations in PostalCode.API/appsettings.json and PostalCode.API/aws-lambda-tools-defaults.json.

b. Clone the git repository available at

[**https://github.com/maringanti1/PostalCodeLookupWeb.git**](https://github.com/maringanti1/PostalCodeLookupWeb.git)

c. Open the PostcodeLookupReactApp folder in VSCode.

d. Open the terminal and run the command "npm i".

e. Start the application by running the command "npm start" and test it locally.

f. Use the "npm run build" command to create a production deployable package inside the build folder in the project root directory.

**3. Deploy PostalCodeAPI:**

a. Log in to the AWS console.

b. Manually create a bucket for the web API deployable packages.

c. Upload the zip file created in step 1(e) to the bucket created in step 3(b).

d. Upload the cloud formation script PostalCode.API/serverless.template to the same bucket.

e. Open the AWS CLI and configure it to connect to this AWS environment, if not already done.

f. Copy the following command into a text editor and update the values highlighted:

Graphical user interface, text, application, email

Description automatically generated

aws cloudformation create-stack \

--stack-name <Stack-Name> \

--template-url <S3-Bucket-URL>/serverless.template \

--capabilities CAPABILITY\_NAMED\_IAM \

--parameter-overrides \

S3Bucket=<S3-Bucket-Name> \

PostCodeApiConfigBaseURL=<API-Base-URL> \

PostCodeApiConfigLookupEndpoint=<Lookup-Endpoint> \

PostCodeApiConfigAutoCompleteEndpoint=<AutoComplete-Endpoint> \

PostCodeApiConfigPostCodeAPILimit=<API-Limit> \

PostCodeApiConfigLatitudeSouth=<Latitude-South> \

PostCodeApiConfigLatitudeMidlands=<Latitude-Midlands>

g. Run the updated final command in AWS CLI. It will start the deployment, and the deployment progress can be tracked by checking the stack "PostcodeLambdaStack."

h. Once deployed successfully, it will give an output variable "ApiURL." This URL will be consumed by the React app to get the postcode data.

i. To test the API, navigate to /swagger/index.html.

**4. Deploy PostalCodeWeb:**

a. Complete step 2(f) to create build artifacts for the PostalCodeWeb application.

b. The previous deployment of the web API has already created the S3 bucket with all the required policies.

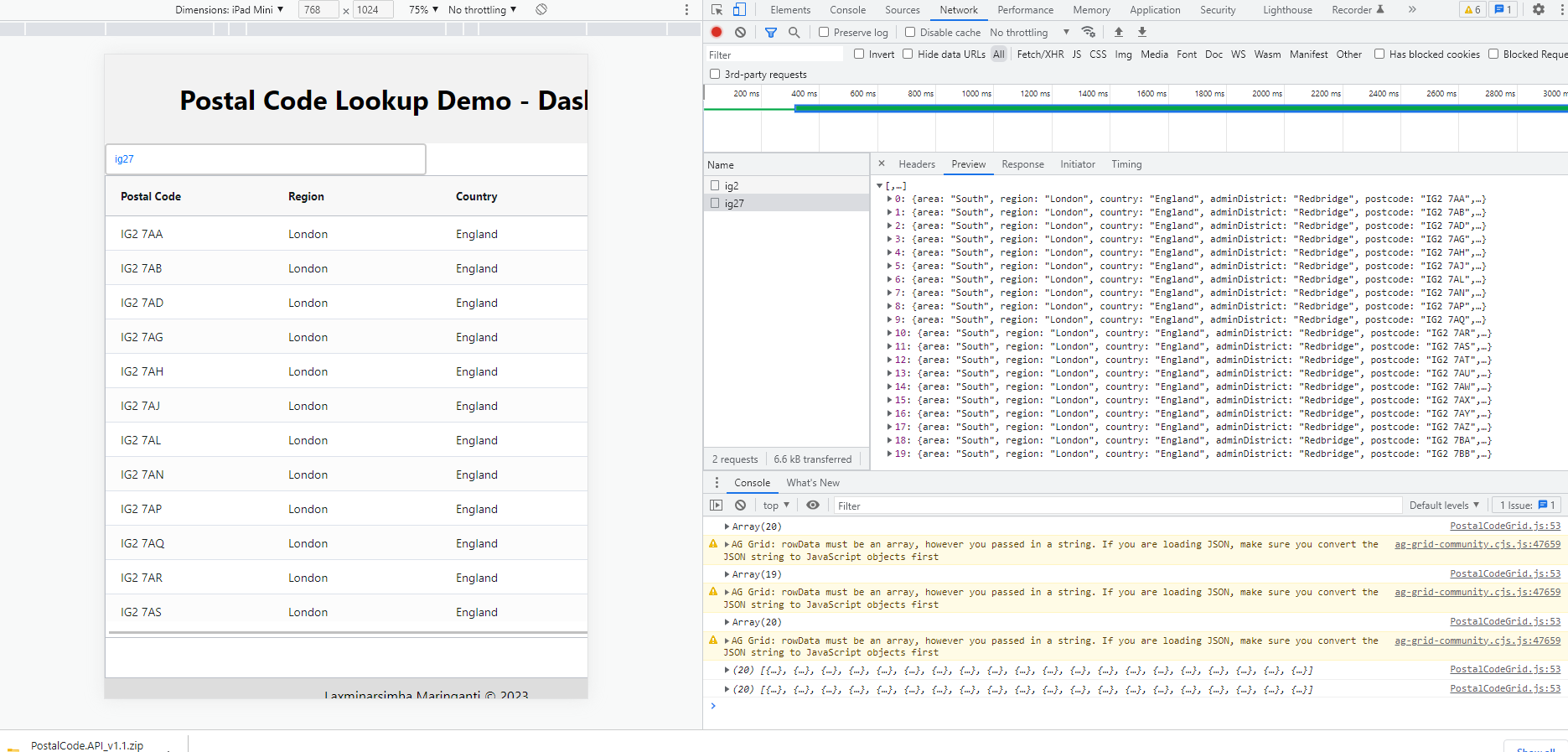
c. Open this bucket and upload all the files and folders present in the build folder directly in the bucket.

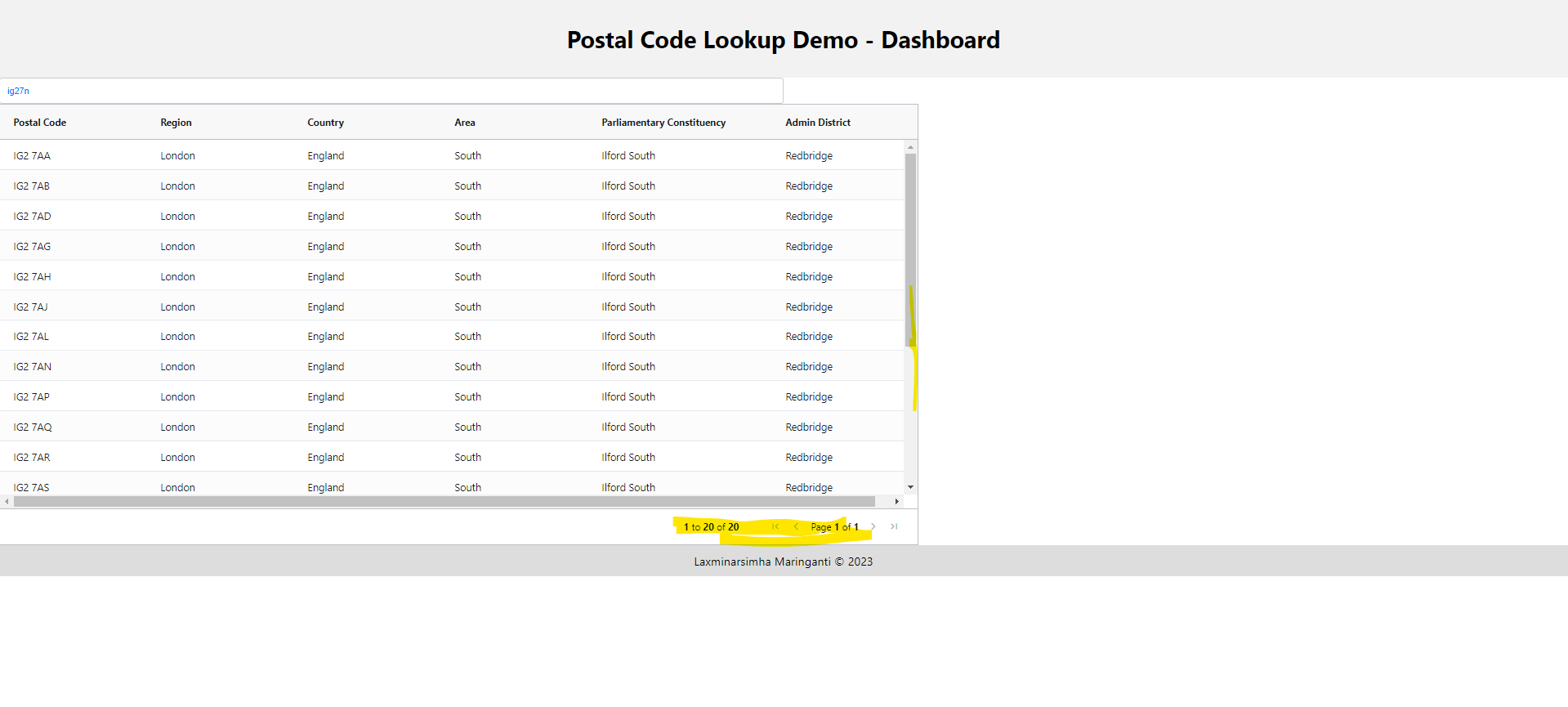
d. Go to the bucket properties and scroll down to the static website hosting section.

e. Find the bucket website endpoint, and the React app has been hosted and is available at this URL.

We have successfully created build artifacts and deployed the PostalCodeAPI and PostalCodeWeb applications.

**Demo:**





Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated