Procedure

Steps:

- Build your react app
- Choose your Cloud provider
- Get your AWS account ready
- Create your s3 bucket
- Create your build version for your app
- go to s3 bucket properties static website hosting and upload your build files(used aws cli for this)
- go to cloudfront distribution service
  - create new distribution
  - choose s3 bucket from the dropdown and leave rest as default
  - wait for it to deploy.
- go to s3 permissions and check for policies
  - check for public access in the acl and bucket policies
  - I used the following json for my bucket policy:-

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

- Time to setup a CI/CD pipeline:
  - Create a github repository
  - Create github actions workflow
  - here is mine:

```
name: Production Build
on:
  pull_request:
  push:
    branches:
      - main
```

```
jobs:
  build:
    runs-on: ubuntu-latest

    strategy:
      matrix:
        node-version: [16.16.0]

    steps:
      - uses: actions/checkout@v1
      - name: Use Node.js ${{ matrix.node-version }}
        uses: actions/setup-node@v2
        with:
          node-version: ${{ matrix.node-version }}
      - name: Yarn Install
        run: |
          yarn install
      - name: Production Build
        run: |
          yarn build
      - name: Unit Tests
        run: |
          yarn test
      - name: Deploy to S3
        uses: jakejarvis/s3-sync-action@master
        with:
          args: --acl public-read --delete
        env:
          AWS_S3_BUCKET: mytodolistdevops
          AWS_ACCESS_KEY_ID: ${{ secrets.AWS_ACCESS_KEY_ID }}
          AWS_SECRET_ACCESS_KEY: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
          AWS_REGION: us-east-1
          SOURCE_DIR: "build"
```

- now go to repo settings > secrets > add new repo secret >
  - **AWS_ACCESS_KEY_ID**
  - **AWS_SECRET_ACCESS_KEY**
- check the name for the brain that you are using in the yml file (mine was main but called master so workflow wasnt workin)
- after the the node verion wasnt suitable for my project , i needed node version 16+ but was using 12 10 and 14 so it was not working
- after that i got error in the production part of my workflow

    - had to add `yarn add --dev @babel/plugin-proposal-private-property-in-object` to my local repo and then push the changes.
- the app.test.js file was default and was searching for the learn react component , edited that and made sure the test was correct and working.
- here is the edited test.app.js:

```
import App from './App';

test('renders input field and add button', () => {
  render(<App />);
  const inputElement = screen.getByPlaceholderText('Add a new todo');
  const addButton = screen.getByText('Add');

  expect(inputElement).toBeInTheDocument();
  expect(addButton).toBeInTheDocument();
});
```

Here is my final yaml file :-
##FINAL:-

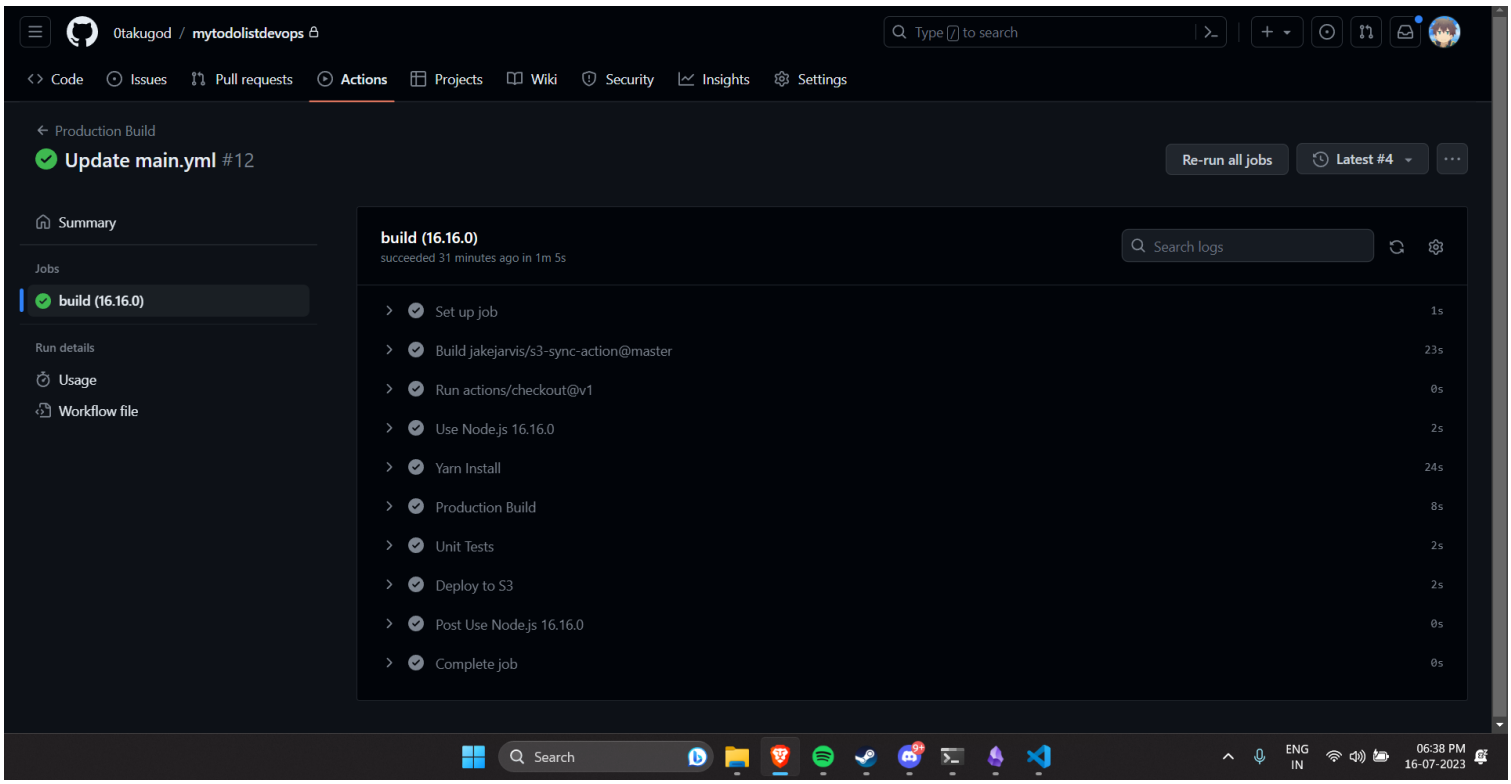
```
on:
  pull_request:
  push:
    branches:
      - main
jobs:
  build:
    runs-on: ubuntu-latest

    strategy:
      matrix:
        node-version: [16.16.0]

    steps:
      - uses: actions/checkout@v1
      - name: Use Node.js ${{ matrix.node-version }}
        uses: actions/setup-node@v2
        with:
          node-version: ${{ matrix.node-version }}
      - name: Yarn Install
        run: |
          yarn install
      - name: Production Build
        run: |
          yarn build
      - name: Unit Tests
        run: |
```

```
      yarn test
    - name: Deploy to S3
      uses: jakejarvis/s3-sync-action@master
      with:
        args: --acl public-read --delete
      env:
        AWS_S3_BUCKET: mytodolistdevops
        AWS_ACCESS_KEY_ID: ${{ secrets.AWS_ACCESS_KEY_ID }}
        AWS_SECRET_ACCESS_KEY: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
        AWS_REGION: us-east-1
        SOURCE_DIR: "build"
```

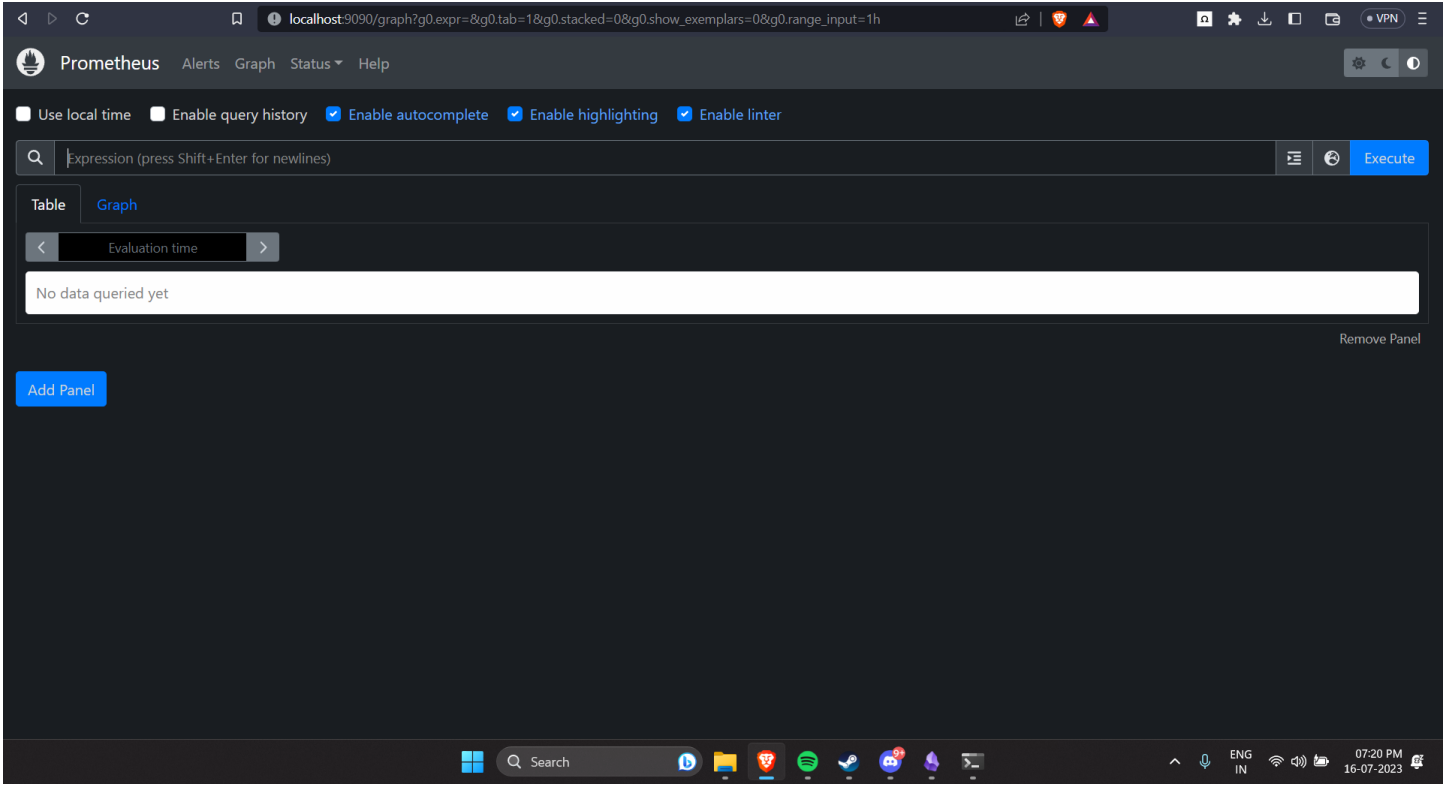
After this , went to the bucket perms and enabled the acl
refer to this -> how to enable
After that , the completed workflow should look like this :



Next steps are :

Setting up promethus and graphana for monitoring and logging and
Then write API and browser tests (eg Playwright or Selenium) for them.

- Download the prometheus from download
- extract it , add it to path variable
- use `prometheus.exe --config.file=prometheus.yml`
- now it should be serving at http://localhost:9090



- Now , you can use graphana(optional) to this and monitor your application.
- Now you can use selenium to write browser test and jest to write api test.
- Create automated alerts in DataDog or Sentry or something.

Update your IAC to include load balancers, autoscaling, caching, a CDN and whatever else you need to help your app to scale.

Then go multi-region.