Exercise

1. How to dockerize the react.js application?

Dockerizing a React.js application involves creating a Docker image that includes the application's code, dependencies, and runtime environment. Here are the general steps for dockerizing a React.js application:
You can use your Local machine / Laptop/ use EC2 instance for practice. Please confirm, Git, Docker and NPM (Node Package Manager) installed in this machine.
For npm installation use these commands

sudo apt update
sudo apt install npm
sudo npm install --save prop-types

Download one React.js sample application by git clone method.

Example : **git clone** https://github.com/benoynsreedhar/calculator
Change the folder to **CD calculator**

1. **Create a Dockerfile**: The first step is to create a Dockerfile that describes how to build the Docker image. This file should be placed in the root directory of your React.js application.

Create Dockerfile:

```
# Use a lightweight Node.js image
FROM node:14-alpine

# Set the working directory
WORKDIR /app

# Copy package.json and package-lock.json to workdir
COPY package*.json ./

# Install dependencies
RUN npm install --production

# Install react-scripts globally
RUN npm install -g react-scripts

# Copy the rest of the application code to the workdir
COPY . .
```

```
# Build the production-ready application
RUN npm run build

# Serve the application
CMD ["npm", "start"]
```

2. **Build the Docker image**: Once you have created the Dockerfile, you can build the Docker image using the `docker build` command. This command should be run from the same directory as the Dockerfile.

```
sudo docker build -t my-react-app .
```

This command will build a Docker image with the tag `my-react-app`.

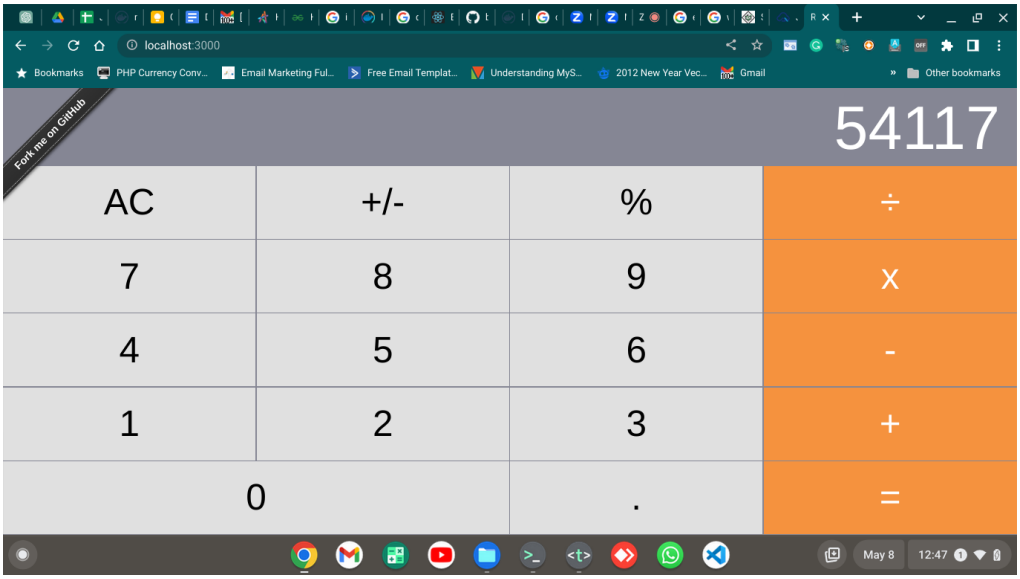
3. **Run the Docker container**: After building the Docker image, you can run it using the `docker run` command.

```
sudo docker run -d -p 3000:3000 my-react-app
```

This command will start a Docker container running the React.js application on port 3000.

Local machine: **http://localhost:3000/**

On Your AWS Instance : **http://<Instance Public IP>:3000/**



That's it! Your React.js application is now dockerized and can be run inside a Docker container. Keep in mind that there may be additional configuration steps depending on your specific use case and requirements.