Task_3

- Create a reactjs simple app
- Create a dockerfile to containerize the reactapp
- Build the image and test it
- (Bonus) create a dockerfile for the same app in smaller size using multi staging

Step 1: Create a React app

First, install `create-react-app` if you don't have it yet, and then create a new app:

```
npx create-react-app my-react-app
cd my-react-app
```

Step 2: Create a Dockerfile

Create a `Dockerfile` to containerize the React app.

```
# Use the official Node.js image as the base image
FROM node:18

# Set the working directory
WORKDIR /usr/src/app

# Copy package.json and package-lock.json files
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code
COPY . .

# Expose the port
EXPOSE 3000
```

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

Step 3: Build the Docker Image

Once the `Dockerfile` is ready, you can build the image:

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

Step 4: Run the Docker Container

After building the image, you can run the container to test it:

```
docker run -p 3000:80 my-react-app
```

Open a browser and go to `http://localhost:3000` to see the running React app.

Step 5: Multi-Stage Build for Smaller Image (Bonus)

The Dockerfile above already uses a multi-stage build to minimize image size. The first stage (`build`) compiles the React app, and the second stage (`nginx:alpine`) only contains the compiled assets without the node_modules, resulting in a smaller image size.

You can further optimize the size by ensuring that only essential files are copied into the image.

```


# Stage 1: Build the React app
FROM node:18-alpine as build

WORKDIR /app

# Install dependencies only
COPY package.json package-lock.json ./
RUN npm install --production

# Copy the rest of the app and build
COPY . ./
RUN npm run build
```

```
# Stage 2: Serve the app with a lightweight NGINX server
FROM nginx:alpine

# Copy only the production build from the previous stage
COPY --from=build /app/build /usr/share/nginx/html

# Expose the port where NGINX is serving
EXPOSE 80

# Command to run NGINX
CMD ["nginx", "-g", "daemon off;"]
```