DOCKER ASSIGNMENT GROUP 2

Steps

1. Downloaded docker and configured it with WSL for windows

2. Created accounts on docker hub

Creating image

- Create a .dockerignorefile

- For Backend Create a docker file and add the following lines:

```
FROM node:18-alpine

ENV NODE_ENV=production

WORKDIR /app

COPY ["package.json", "package-lock.json*", "./"]

RUN npm install --production

COPY . .

CMD ["node", "server.js"]
```

- For front end Create a docker file and add the following lines:

```
FROM node
# - Will create a node environment in the container
WORKDIR /app
# - Will create a directory app and switch to that directory
COPY package.json .
# - Copies package.json file to /app directory
RUN npm i
# - Runs npm install to create node_modules for your app
COPY . .
# - Copies the source code to /app directory
EXPOSE 5173
```

```
# - Exposes the port to access the app from outside the container i.e from the browser
CMD ["npm", "run", "dev"]
# - Executes npm run dev to start the server
```

- Run the command to build the image
  ```
  docker build --tag calculator .
  ```

- Start the container and expose port 3000 to port 3000 on the host.

  ```
  docker run --publish 3000:3000 calculator
  ```

- We then post some data to see if our backend is working

  ```
  docker run --network host curlimages/curl `

    --request POST `

    --url http://localhost:3000/api/calculations `

    --header 'content-type: application/json' `

    --data '{"value_one": "string", "value_two": "string", "operand": "string"}'
  ```

  We were able to connect to the application running inside of our container on port 3000.

Pushing image

- Next to push our work to Docker Hub we do the following steps below:

```
docker login
```

- Next we tag our image that we want to push for example

```
docker tag calculator kalibbalajohnson/calculator
```

- Next type in the command line

```
docker push
```

- And finally push our tagged image to docker hub

docker push kalibbalajohnson/calculator

Pulling image

- For pulling the image, one first access the docker hub repository online
- And navigate to the public tab to copy the link provided
- Paste the link in the command line to pull the repository

Adding Linter

- Next adding Eslint to the project. By typing the following commands

npm install eslint - -save-dev

npx eslint - -init (to initialize eslint)

npm intall eslint-plugin-react - -save-dev

- Next add Prettier to the project. By typing the following commands

npm install prettier - -save-dev

- Add a file called .eslintrc.json and add rules.

Adding tests

- We create a test file and initially make a failing test and then incrementally update our work until all the tests are running well.

Github repo

https://github.com/WarrenG123/calculator