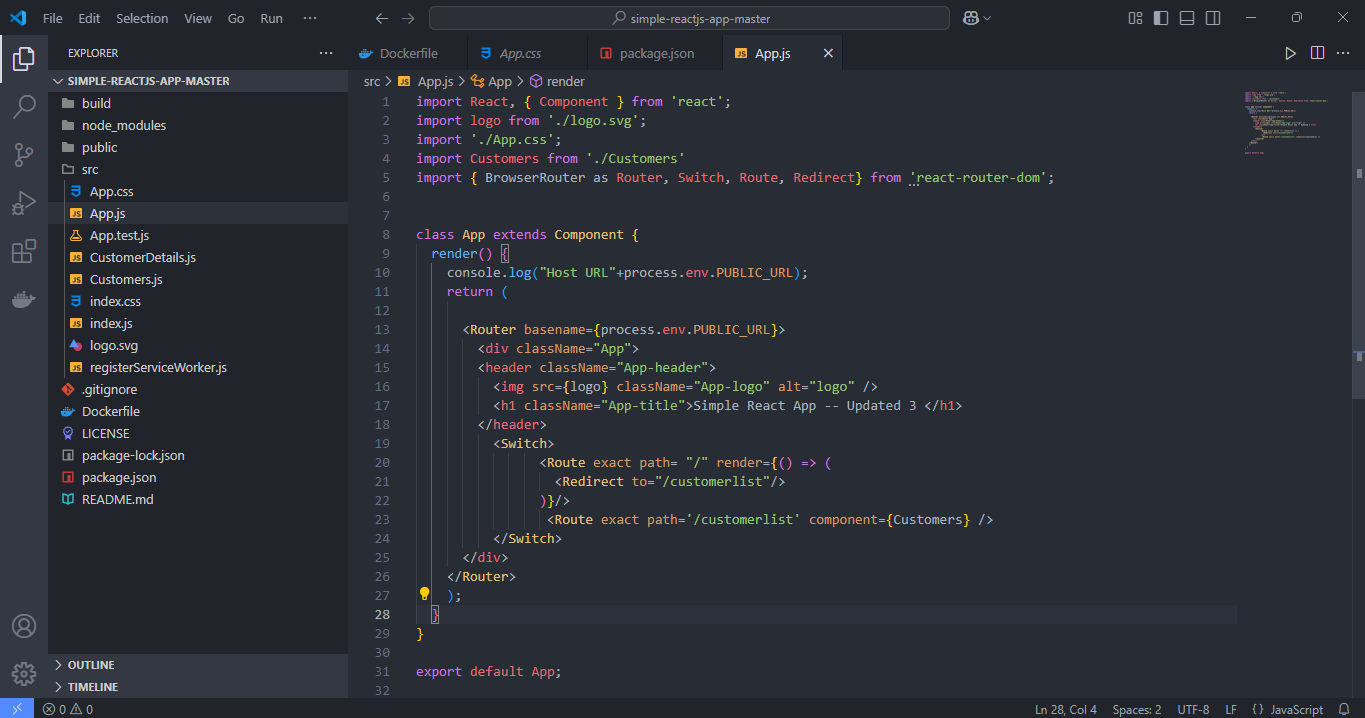
**Assignment - 1**

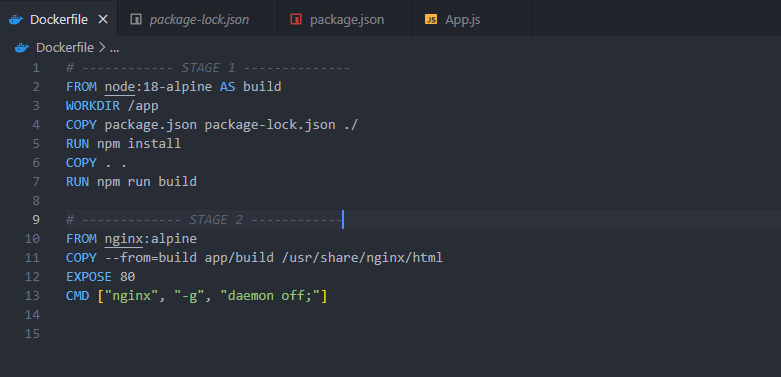
**Objective:**

Create a simple Docker container for a web application.

1. Create a web application or use an existing one (don't use the project used in labs).

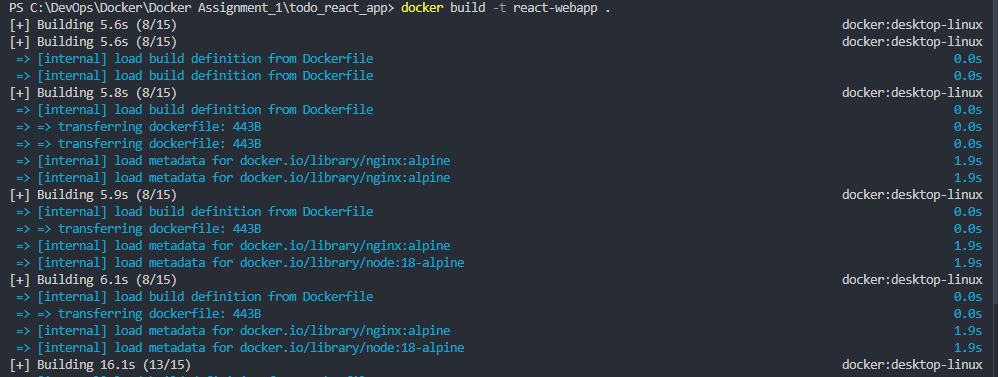


1. Write a Dockerfile (Multistage if possible) to containerize the application.

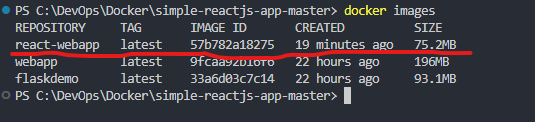


1. Build the Docker image, the smaller the size, the more marks you would get.

-> Build the docker image - react-webapp

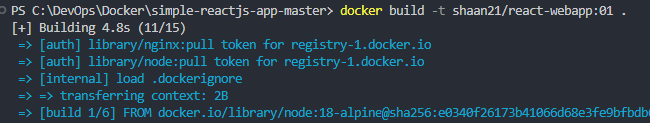


-> Check Image Size

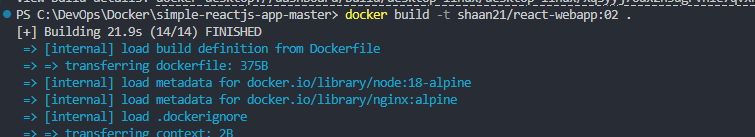


1. Push it to Docker hub, with at least 3 different tags.

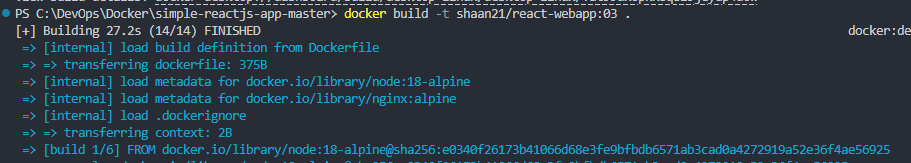
-> Build image with tag 1



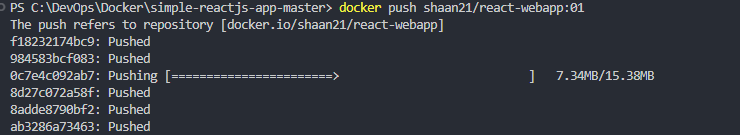
-> Build image with tag 2

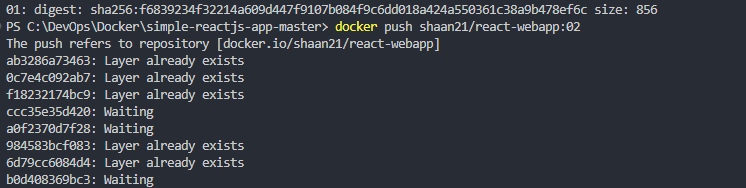


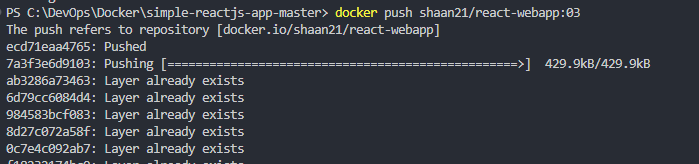
-> Build image with tag 3

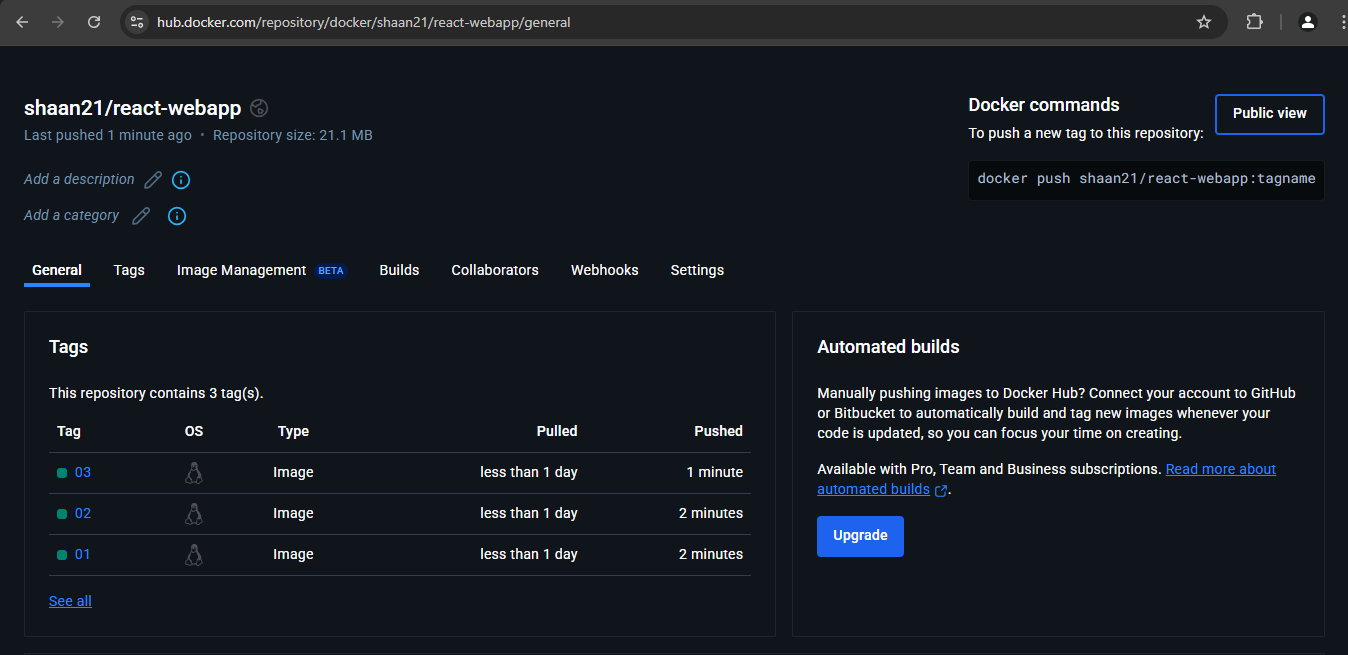


-> Push to docker hub



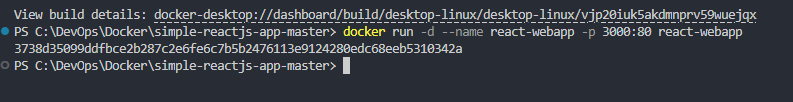




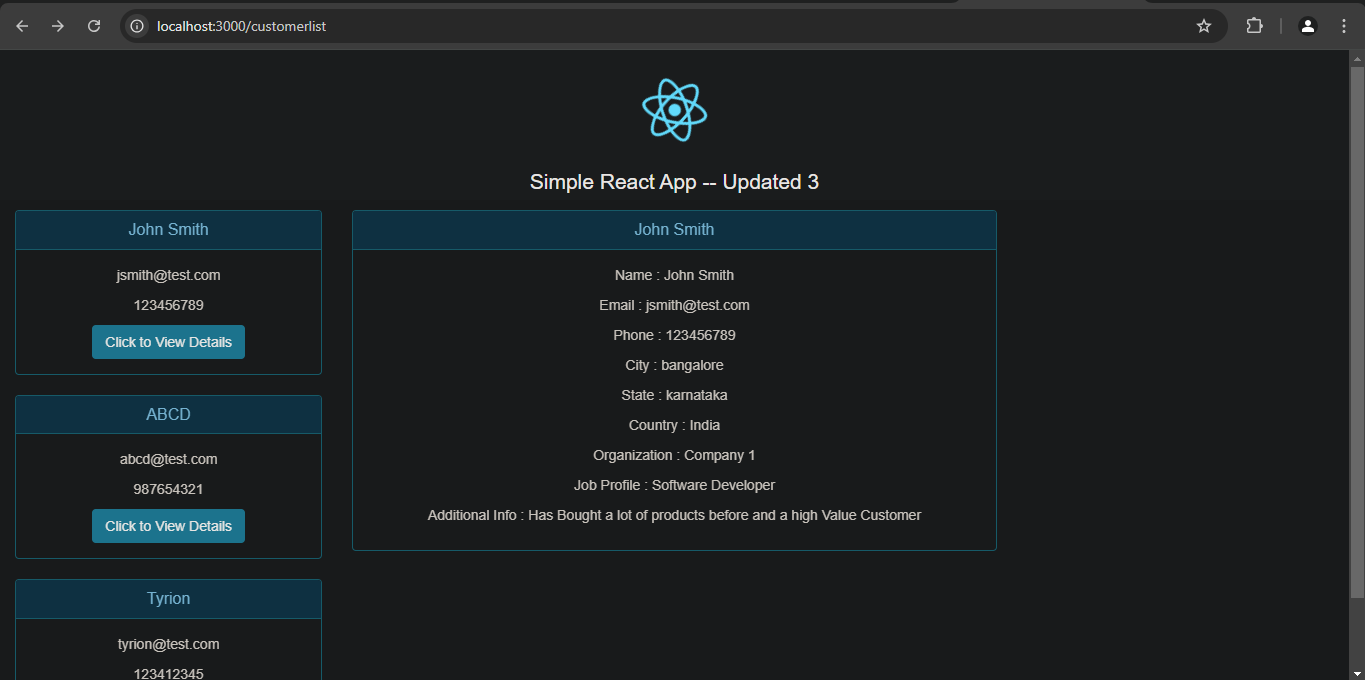


1. Run the Docker container from the image that you've built & access the web application in a web browser in local machine.

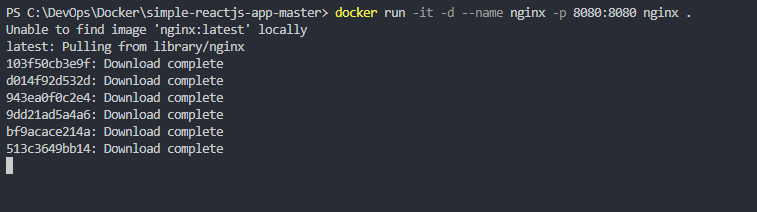
-> Run docker image to create docker container - react-webapp



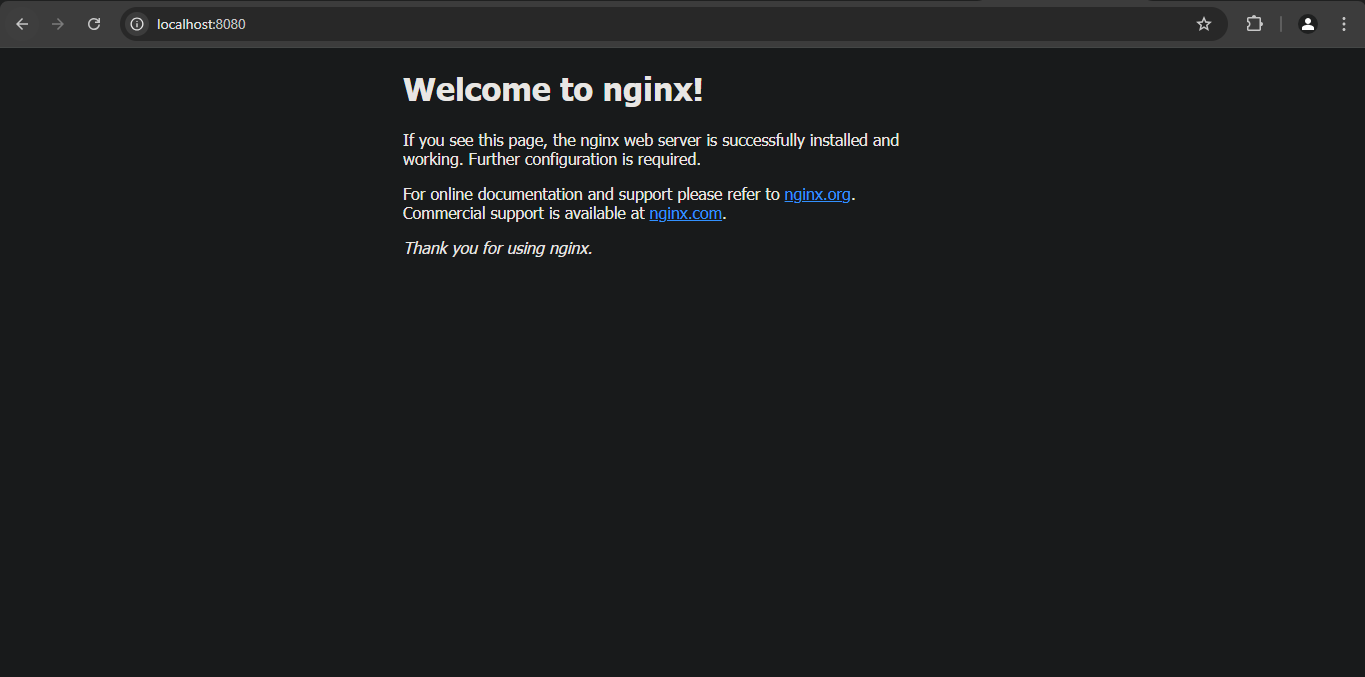
-> Open port: <http://localhost:3000/>



1. Start another docker container with latest 'nginx' image.

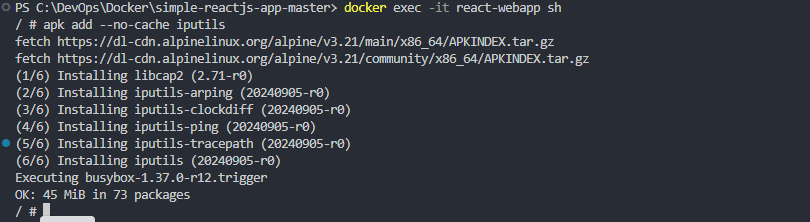


-> Open port: <http://localhost:8080/>



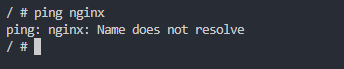
1. Go inside the application container, inside its terminal, and try to access/ping the nginx container to check the connectivity

-> Go instide the react-webapp container

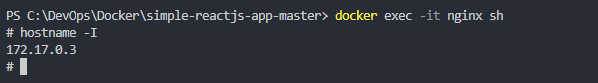


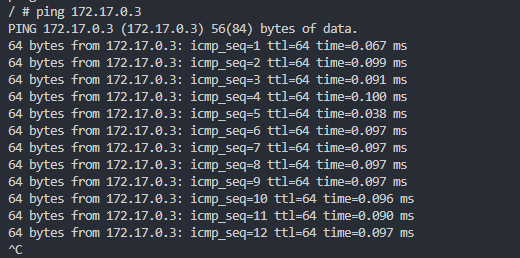
-> Try to ping nginx container.

-> This will give error as by default containers cannot communicate using names

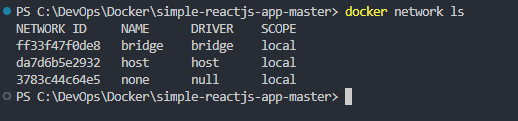


-> We can try to access using ip of the nginx container



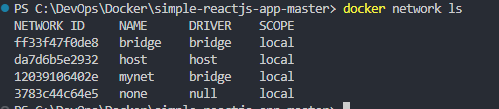


-> Or we connect both the container to a custom network



-> Create new network *mynet*

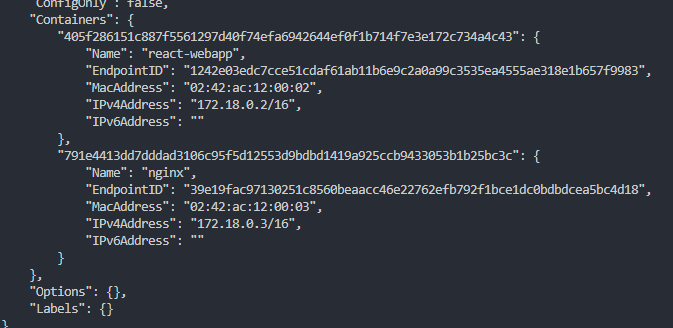




-> Connect both the containers to *mynet*







-> Now we wil be able to ping the *nginx* container by name

