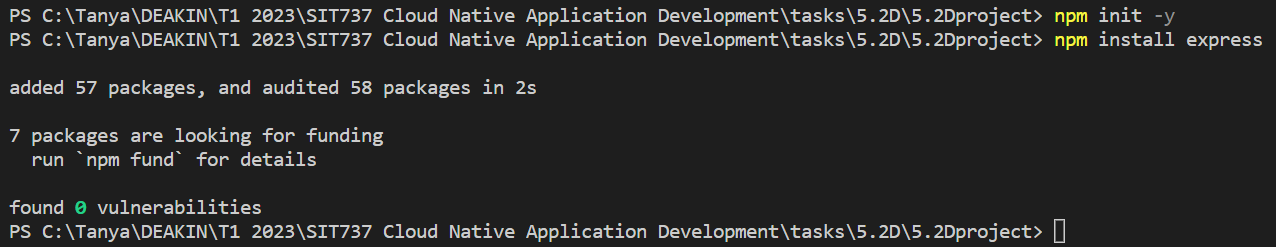
5.2D: Configuring networking between containers

1. Started with npm init -y and npm install express



1. My Dockerfile looks like –
2. #denotes base image
3. FROM node:14
4. #setting working directory
5. WORKDIR /usr/src/app
6. COPY package\*.json ./
7. #to install the package listed in package.json file
8. RUN npm install
9. COPY index.js index.js
10. #exposing port outside
11. EXPOSE 3000
12. CMD ["node", "index.js"]

3. And my docker-compose.yml looks like below. Container1 made from image1 is exposed on port 5001.

Container2 made from image2 is exposed on port 5002

A bridge network called my-network connects the two for communication

services:

  app1:

    image: image1

    build:

      context: .

      dockerfile: Dockerfile

    container\_name: container1

    restart: on-failure

    ports:

      - "5001:3000"

    networks:

      - my-network

  app2:

    image: image2

    build:

      context: .

      dockerfile: Dockerfile

    container\_name: container2

    restart: on-failure

    ports:

      - "5002:3000"

    networks:

      - my-network

networks:

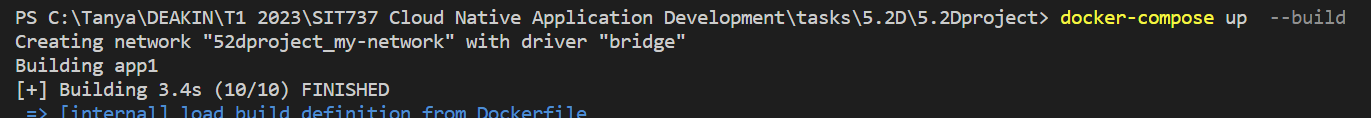
  my-network:

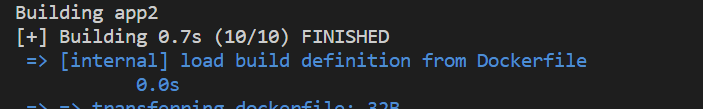
    driver: bridge

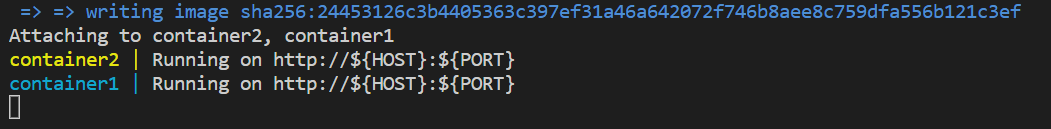
4. Executed docker compose command to make 2 images – image1 and image 2, and 2 containers- container1 and container2.

Command 🡪 docker-compose up --build

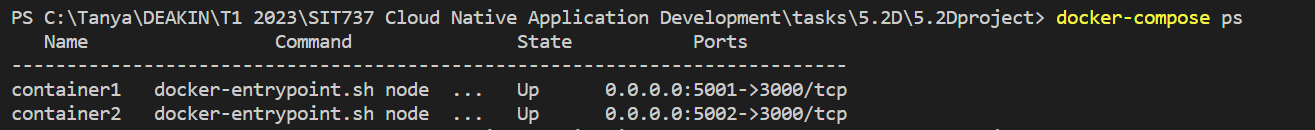
Screenshot –



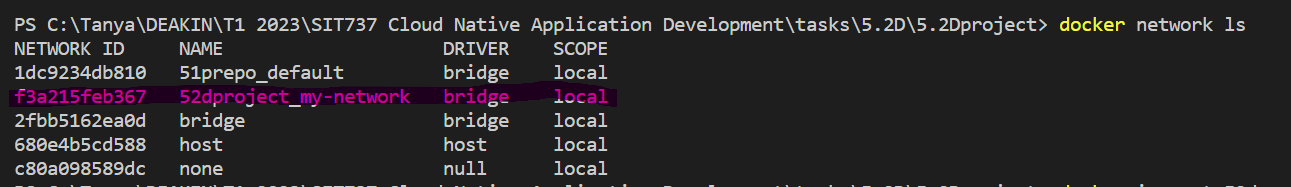




5. Docker compose ps command for the same –



6. The bridge has got created from the docker compose yaml file

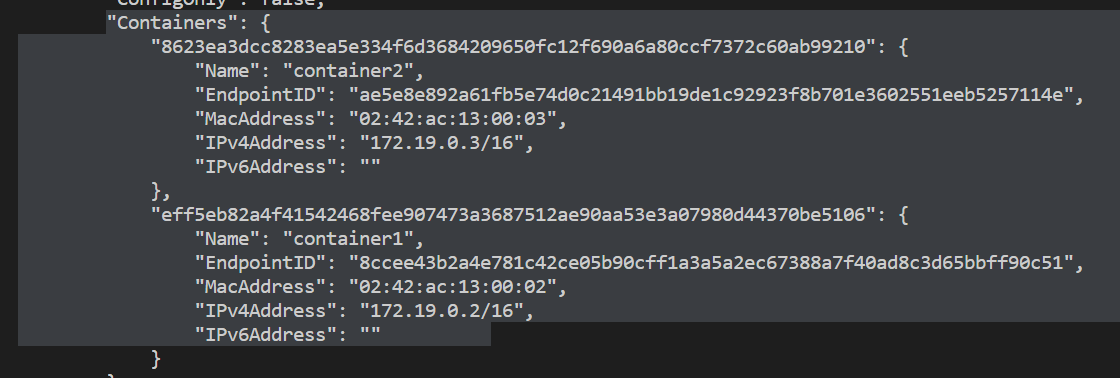


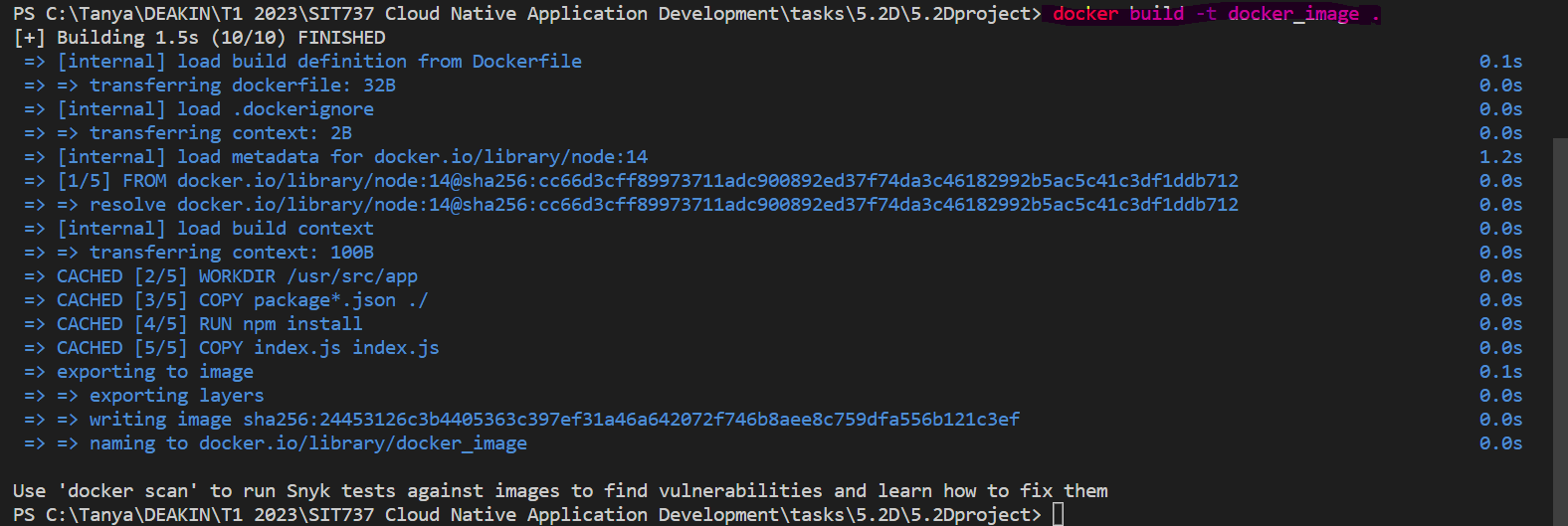
7. Ran docker inspect to check the bridge created.

**Command** used🡪 docker inspect 52dproject\_my-network

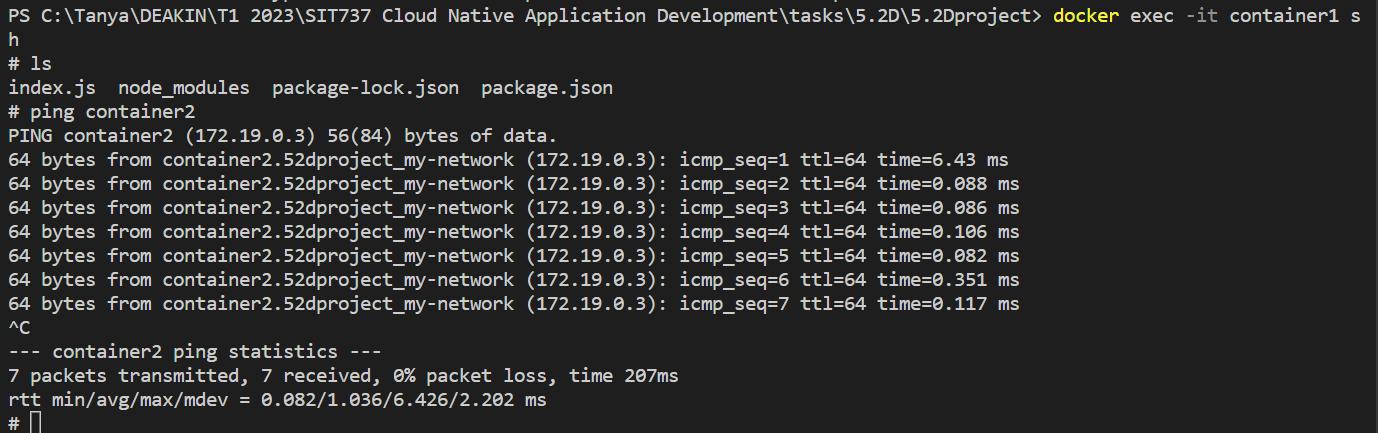
**Output**: The network shows the containers as expected. IP addresses can be seen.

**Screenshot** –

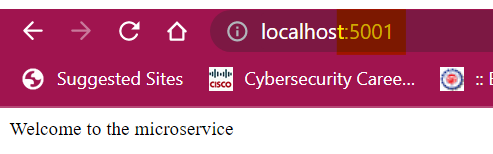


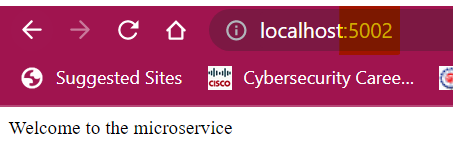


8. Messaged container2 from container1



9. Both web servers are running as below –

 and,



10.