# **Assignment 2**

Tasks: Use Docker Compose to manage a multi-service application.

- 1. Define a multi-service application consisting of at least three services (e.g., a web app, a database, and a caching service) in a docker-compose.yml file.
- 2. Configure the services with appropriate environment variables and networking settings.
- 3. Build and run the Docker Compose application.
- 4. Test the connectivity between the services and the functionality of the application.

Steps to use docker compose for multi-service application:

- 1. I took my own react nodejs mysql and redis application.
- 2. I added Dockerfile to each api and client directory.

### /api/Dockerfile:

```
FROM node:latest
WORKDIR /app
COPY package.json ./
RUN npm install
COPY . .
EXPOSE 8800
CMD [ "npm", "start" ]
```

### /client/Dockerfile:

```
FROM node:latest
WORKDIR /app
COPY package.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD [ "npm", "start" ]
```

3. Now create a new Directory (docker-compose-app) and add docker-compose.yaml ./docker-compose-app/docker-compose.yaml:

```
version: "3.0"
services:
  #mysql service
  mysqldb:
```

```
image: mysql:8.0
    container name: mysqlcontainer
    restart: unless-stopped
    # creating volumes to keep the database data
consistent and intializing the database with initial data
    volumes:
      - ./mydb.sql:/docker-entrypoint-initdb.d/0 init.sql
      - $HOME/database:/var/lib/mysql
   # exposing differenct port because port is already in
use
    ports:
      - 3900:${MYSQL_PORT}
    expose:
      - ${MYSQL PORT}
    environment:
      MYSQL_DATABASE: ${MYSQL_DATABASE}
      MYSQL USER: ${MYSQL USER}
      MYSOL PASSWORD: ${MYSOL PASSWORD}
      MYSQL ROOT PASSWORD: ${MYSQL ROOT PASSWORD}
      SERVICE TAGS: dev
      SERVICE NAME: mysqldb
    # using custom bridge network to isolate the
containers
    networks:
      - internalnet
  nodejsapi:
    container_name: node-js-api
    build: ../api/
    image: jainilp12/nodejsapi:1.0
    ports:
      - 8800:${SERVER PORT}
    expose:
      - ${SERVER PORT}
    environment:
      DB HOST: mysqldb
      DB PORT: ${MYSQL PORT}
      DB_NAME: ${MYSQL_DATABASE}
      DB USER: ${MYSQL USER}
      DB_PASSWORD: ${MYSQL_PASSWORD}
```

```
REDIS_HOST: ${REDIS_HOST}
    REDIS_PORT: ${REDIS_PORT}
    REDIS_PASSWORD: ${REDIS_PASSWORD}
    SERVICE TAGS: dev
    SERVICE_NAME: nodejsapi
    SERVER PORT: ${SERVER PORT}
  networks:
    - internalnet
  depends on:
    - mysqldb
    - redisdb
reactwebapp:
  container_name: react-web-app-container
  build: ../client/
  image: jainilp12/reactwebapp:1.0
  ports:
    - 3000:${REACT PORT}
  expose:
    - ${REACT PORT}
  environment:
    SERVICE TAGS: dev
    SERVICE_NAME: reactwebapp
  networks:
    - internalnet
  depends_on:
    - nodejsapi
redisdb:
  image: redis:latest
  container name: rediscontainer
  restart: always
  ports:
    - 6379:${REDIS PORT}
  environment:
    SERVICE TAGS: dev
    SERVICE_NAME: redisdb
  volumes:
    - ./cache:/data
  networks:
```

#### - internalnet

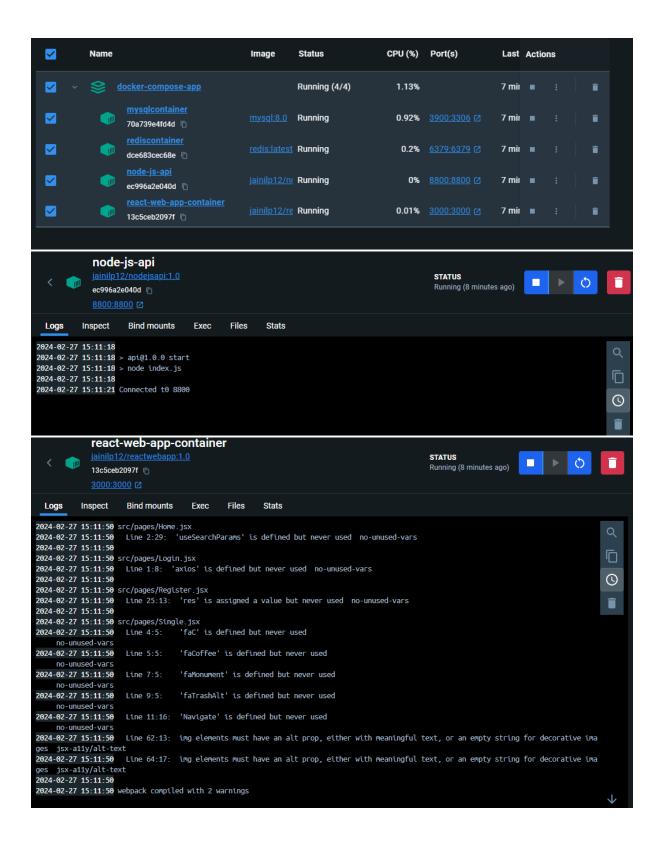
networks:

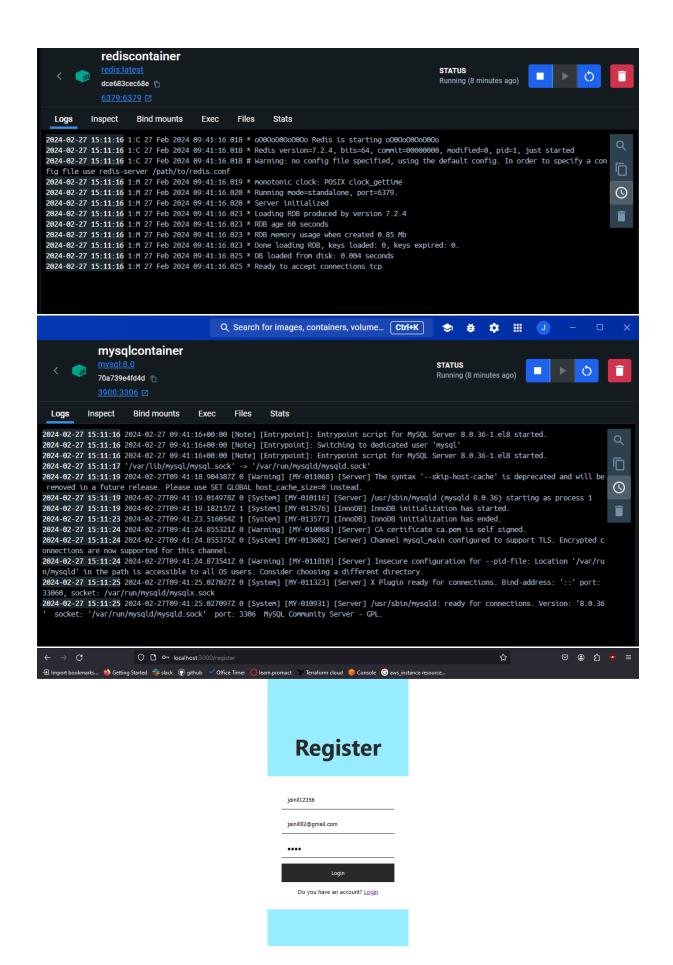
internalnet:

driver: bridge

4. Now use docker compose up to run/build/pull all the containers. docker compose up -d -build

```
PS C:\Users\promact\delthis\docker-compose-app> <mark>docker</mark> compose up -d --build
time="2024-02-27T15:10:28+05:30" level=warning msg="The \"HOME\" variable is not set. Defaulting to a blank string."
docker:default
                                                                               0.0s
0.0s
0.0s
                                                                               0.05
0.0s
0.0s
0.0s
0.0s
0.0s
0.0s
0.0s
```









# Jainil Patel First Post

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem

Read More



5. Now push the images to the docker hub (I am already logged in), use the following command:

docker compose push

