Assignment:2

Use Docker Compose to manage a multi-service application.

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.

Configure the services with appropriate environment variables and networking settings.

Build and run the Docker Compose application.

Test the connectivity between the services and the functionality of the application.

Steps:

Step 1: Understanding the Existing Project:

- Understand the dependencies and interactions between the components:
- In my project the React front-end communicates with the Node.js backend, which in turn interacts with the MySQL database.

Step 2: Dockerizing Each Component

- Write Dockerfiles to build Docker images for the React front-end, Node.js backend.

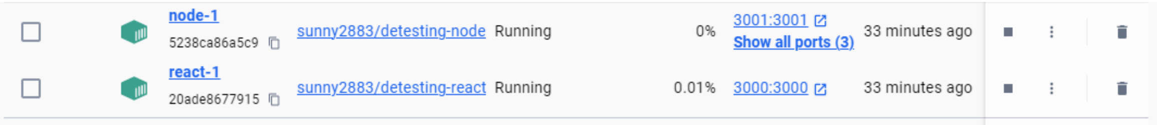
  Dockerfile for node.js

  ```
  FROM node:latest
  WORKDIR /app
  COPY package*.json ./
  RUN npm install
  COPY . .
  EXPOSE 3001 3002 3003
  CMD ["npm", "run", "startservers"]
  ```

Dockerfile for react.js

```
FROM node:latest

WORKDIR /app

COPY package*.json ./

RUN npm install

COPY . .

EXPOSE 3000

CMD ["npm", "start"]
```

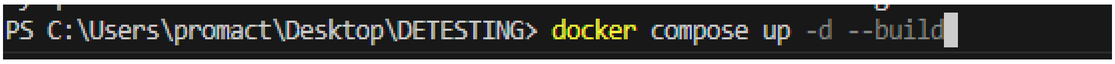
- Build Docker images: Use docker build command to build Docker images for each component based on their respective Dockerfiles.



□	node-1 5238ca86a5c9	sunny2883/detesting-node	Running	0%	3001:3001 Show all ports (3)	33 minutes ago	■ ⋮ 🗑
□	react-1 20ade8677915	sunny2883/detesting-react	Running	0.01%	3000:3000	33 minutes ago	■ ⋮ 🗑

Step 3: Define Docker Compose Configuration:

- Create a docker-compose.yml file in the root directory of the project.
- Define services for each component: Specify services for React front-end, Node.js backend, and MySQL database in the docker-compose.yml file.
- Configure environment variables: Set environment variables for each service in the **docker-compose.yml** file, providing necessary configurations.
- Run the command;
  docker compose up -d --build



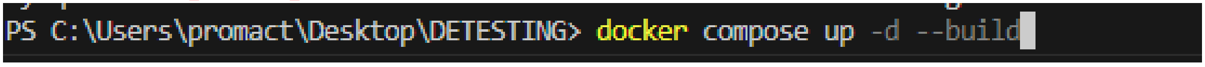
```
PS C:\Users\promact\Desktop\DETESTING> docker compose up -d --build
```

Step 4: Build and Run Docker Compose Application:

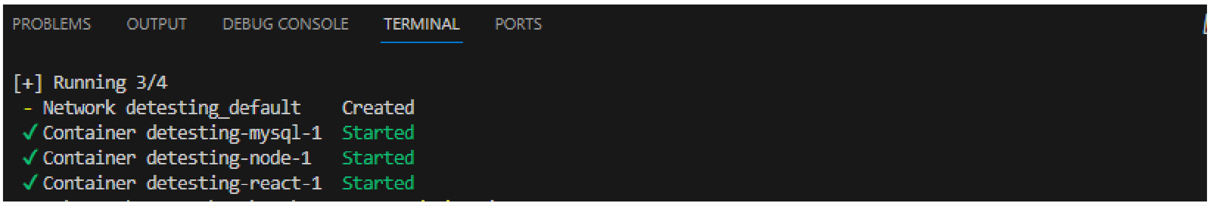
- Navigate to the root directory of the project containing the docker-compose.yml file.

- Run docker-compose build command to build Docker images for all services defined in the docker-compose.yml file.
- Once the images are built successfully, run docker-compose up command to start the Docker Compose application.

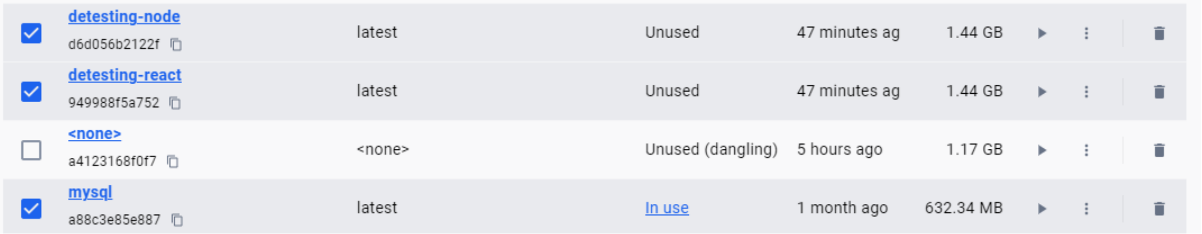
docker compose up -d --build



```
PS C:\Users\promact\Desktop\DETESTING> docker compose up -d --build
```



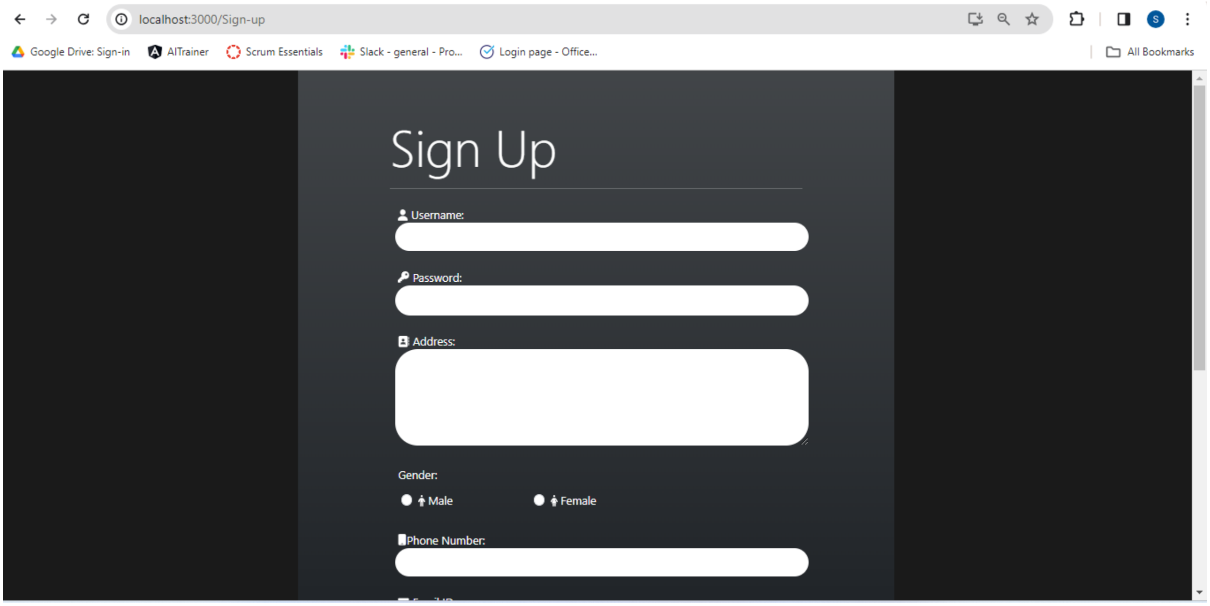
```
[+] Running 3/4
 - Network detesting_default    Created
 ✓ Container detesting-mysql-1  Started
 ✓ Container detesting-node-1   Started
 ✓ Container detesting-react-1  Started
```



☑	detesting-node d6d056b2122f	latest	Unused	47 minutes ag	1.44 GB	
☑	detesting-react 949988f5a752	latest	Unused	47 minutes ag	1.44 GB	
□	<none> a4123168f0f7	<none>	Unused (dangling)	5 hours ago	1.17 GB	
☑	mysql a88c3e85e887	latest	In use	1 month ago	632.34 MB	

Step 5: Test Connectivity and Functionality

- Access the React front-end in a web browser to ensure it's running correctly.
- Test Node.js backend to verify its functionality.
- Connect to the MySQL database from the backend service and perform database operations to ensure connectivity.
- Verify that services can communicate with each other as expected within the Docker network.



Sign Up

Username:

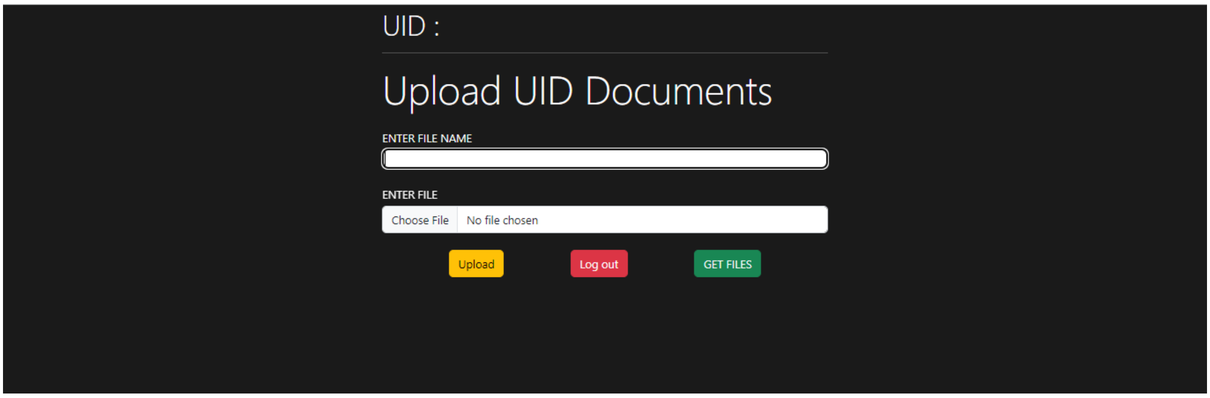
Password:

Address:

Gender:

○ Male ○ Female

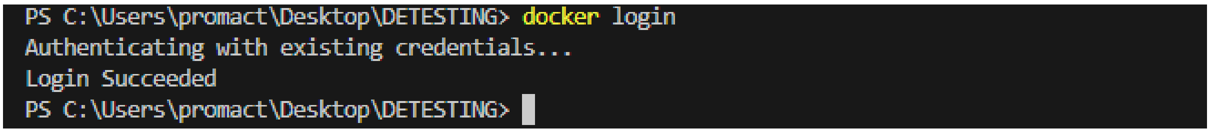
Phone Number:



Step 6: Log in to Docker Hub

Log in to Docker Hub using the docker login command, providing your Docker Hub username and password when prompted.

docker login

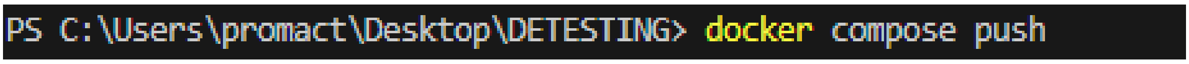


```
PS C:\Users\promact\Desktop\DETESTING> docker login
Authenticating with existing credentials...
Login Succeeded
PS C:\Users\promact\Desktop\DETESTING> 
```

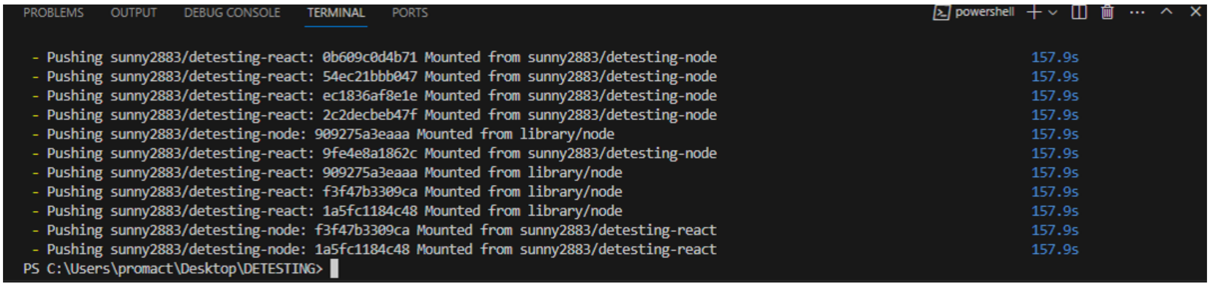
Step 7: Push Docker Images to Docker Hub using command:

docker push compose:

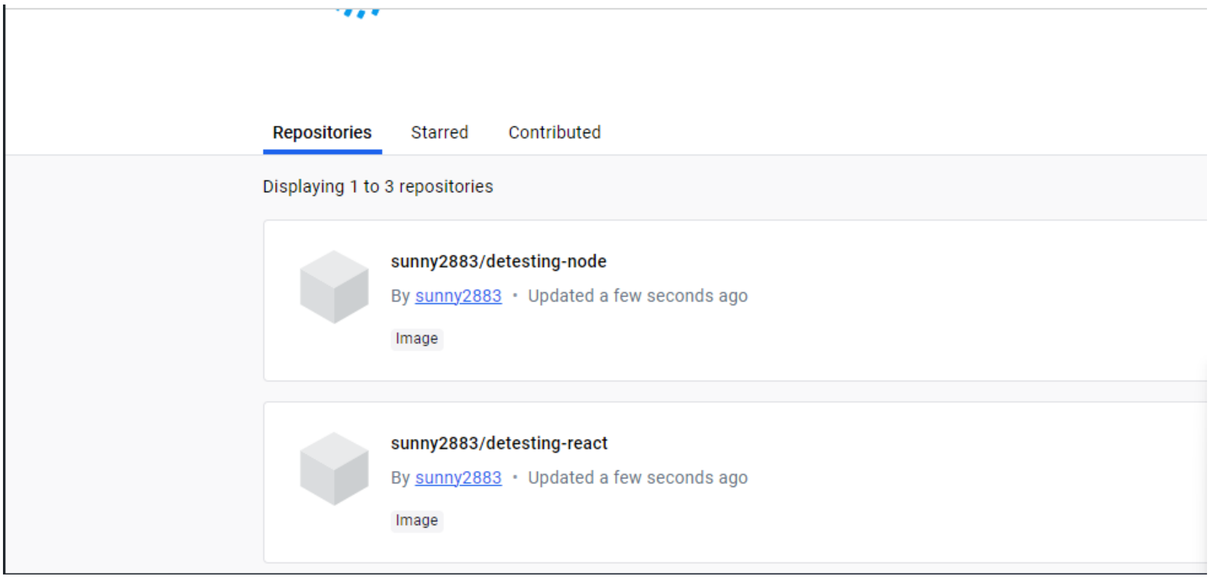
docker compose push



```
PS C:\Users\promact\Desktop\DETESTING> docker compose push
```



```
 - Pushing sunny2883/detesting-react: 0b609c0d4b71 Mounted from sunny2883/detesting-node                 157.9s
 - Pushing sunny2883/detesting-react: 54ec21bbb047 Mounted from sunny2883/detesting-node                 157.9s
 - Pushing sunny2883/detesting-react: ec1836af8e1e Mounted from sunny2883/detesting-node                 157.9s
 - Pushing sunny2883/detesting-react: 2c2decbeb47f Mounted from sunny2883/detesting-node                 157.9s
 - Pushing sunny2883/detesting-node: 909275a3eaaa Mounted from library/node                              157.9s
 - Pushing sunny2883/detesting-react: 9fe4e8a1862c Mounted from sunny2883/detesting-node                 157.9s
 - Pushing sunny2883/detesting-react: 909275a3eaaa Mounted from library/node                             157.9s
 - Pushing sunny2883/detesting-react: f3f47b3309ca Mounted from library/node                             157.9s
 - Pushing sunny2883/detesting-react: 1a5fc1184c48 Mounted from library/node                             157.9s
 - Pushing sunny2883/detesting-node: f3f47b3309ca Mounted from sunny2883/detesting-react                 157.9s
 - Pushing sunny2883/detesting-node: 1a5fc1184c48 Mounted from sunny2883/detesting-react                 157.9s
PS C:\Users\promact\Desktop\DETESTING> 
```



Displaying 1 to 3 repositories

sunny2883/detesting-node
By sunny2883 • Updated a few seconds ago
Image

sunny2883/detesting-react
By sunny2883 • Updated a few seconds ago
Image

Command to pull react image:

```
docker pull sunny2883/detesting-react:latest
```

command to pull node image;

```
docker pull sunny2883/nodejsapp:latest
```