**PROCEDURE:**

**--------------------**

• Installed VMware Workstation 17 on a laptop and created a virtual machine running Ubuntu Jammy Jellyfish 22.04 following the instructions in the documentation [here](https://docs.docker.com/desktop/install/linux-install/).

• After deploying the VM, install GNOME for non-GNOME Linux environments. However, in Ubuntu 22.04, GNOME is pre-installed by default, which can be confirmed in the system details by navigating to Settings > About > GNOME version.

**CREATE REPOSITORY FOR DOCKER INSTALLATION:**

**------------------------------------------------------------------**

Before installing Docker Engine for the first time on a new host machine, it is necessary to set up the Docker repository.

* **APT (Advanced Package Tool)** is used in the command line to configure the repository or install packages on the host.
* Use the following commands to create the repository:

**# Add Docker's official GPG key:**

sudo apt-get update

sudo apt-get install ca-certificates curl

sudo install -m 0755 -d /etc/apt/keyrings

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc

**# Add the repository to APT sources:**

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \

$(. /etc/os-release && echo "$VERSION\_CODENAME") stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

**INSTALL THE DOCKER PACKAGE:**

**------------------------------------------**

If you log in to the VM using a root account, the 'sudo' command is not required. However, since I am using my own account, I delegate Super User authority to my account using sudo commands.

* **Install Docker Engine on the VM** using the following command:

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

* **Verify the Docker version** (if needed) with the command:

docker --version

or

docker --v

My environment is installed with the version: "Docker version 27.0.3, build 7d4bcd8".

* **Check if the Docker Engine installation is successful** by running:

sudo docker run hello-world

Note: This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

Reference: [Docker Engine Installation Guide for Ubuntu](https://docs.docker.com/engine/install/ubuntu/)

You can use any method mentioned in the documentation to install Docker on Ubuntu. I used the first method because the VM deployed in VMware Workstation is connected to the internet and can reach <http://download.docker.com> to pull the required packages for Docker installation.

**WEBAPP FUNCTIONALITY:**

**-----------------------------------**

Webapp functionality is to add, edit and delete student into student database. A request to the webserver can post, put, delete the from the web browser.

Created an application code Used MongoDB, Express.js, React.js, Node.js for creating simple Student Database project

**DEPLOYING WEBAPP in DOCKER:**

**--------------------------------------------**

1. Docker is deployed in the VM using the steps described initially.
2. Used MongoDB, Express.js, React.js, Node.js for creating simple Student Database project
3. The project has server, client and database, with their respective docker file
4. Folder- " C:\my\_proj\proj\_docker\docker"
5. **File- "Dockerfile"** to tell the application how it is going to run. Created under the folder created and specified

* **Client side - "Dockerfile":**

***FROM node:latest AS builder***

***WORKDIR /app***

***COPY package.json .***

***RUN npm install***

***COPY . .***

***RUN npm run build***

***FROM nginx:1.25.2-alpine-slim***

* **Server side - "Dockerfile":**

***FROM node:20-alpine3.17***

***WORKDIR /app***

***COPY package.json .***

***RUN npm install***

***COPY . .***

***EXPOSE 5000***

***CMD ["npm", "start"]***

***COPY --from=builder /app/build /usr/share/nginx/html***

***EXPOSE 80***

***CMD ["nginx", "-g", "daemon off;"]***

* **Docker-Compose.yaml:**

 version: '3.8'

# Services

services:

# Server service

server:

build:

context: ./server

dockerfile: Dockerfile

container\_name: backend

ports:

- "5000:5000"

env\_file: ./.env

environment:

- DB\_HOST=mongodb

- DB\_USER=$MONGODB\_USER

- DB\_PASSWORD=$MONGODB\_PASSWORD

- DB\_NAME=$MONGODB\_DATABASE

- DB\_PORT=$MONGODB\_DOCKER\_PORT

depends\_on:

- mongodb

# Client service

client:

build:

context: ./client

dockerfile: Dockerfile

container\_name: frontend

ports:

- "80:80"

depends\_on:

- server

# Database service

mongodb:

image: mongo:latest

container\_name: mongodb\_server

env\_file: ./.env

environment:

- MONGO\_INITDB\_ROOT\_USERNAME=$MONGODB\_USER

- MONGO\_INITDB\_ROOT\_PASSWORD=$MONGODB\_PASSWORD

ports:

- "27017:27017"

volumes:

- ./../mydata:/data/db

# Volumes define

volumes:

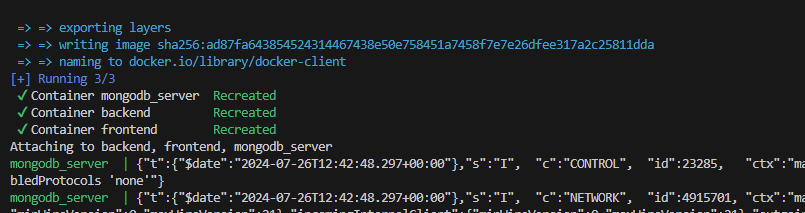
mydata:

**BUILD A DOCKER IMAGE:**

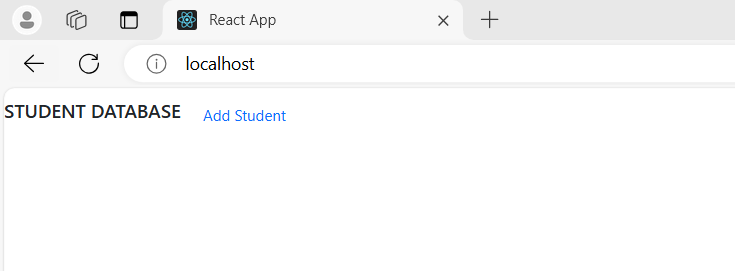
**-----------------------------------**

* Navigate in terminal to path- " C:\Users\DELL\OneDrive\Desktop\DocsProjects\docker-demo" and run the below command to build the docker image

**Sudo docker compose up --build .**



* Use the URL in local browse and confirm if it is able to fetch the image as desired.



* Add student 