Experiment-2.1

Student Name:Aditi Pandey **UID:22BDO10031**
Branch: CSE(DevOps) **Section/Group:22BCD1/A**
Semester:5 Sem **Date of Performance:9sept2024**
Subject Name:Docker and Kuberenats **Subject Code: 22CSH-343**

1.Aim/Overview of the practical: To run *Node.js* application using Docker and manage the Docker volume.

2.Apparatus: VM ware, Linux , docker

3.Steps for experiment/practical:

Step 1: Create a Node.js Application:

1. Create a directory for your Node.js project.
2. In the project directory, create index.js and package. Json.

```
docker_tutorial_1 > index.js > ...
var http = require('http');

http.createServer(function (req, res) {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.end('Hello World!');
}).listen(8080);
```

Step 2: Create a Dockerfile:

3. In the root directory of your Node.js project, create a file named Dockerfile (without any file extension).
4. Add the following content to the Dockerfile

index.js | dockerfile

docker_tutorial_1 > dockerfile > ...

```
FROM node:latest
WORKDIR /usr/src/app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 8080
CMD [ "node", "index.js" ]
```

Step 3: <u>Build the Docker Image:</u>

5. Open your terminal.
6. Navigate to your project directory where the Dockerfile is located.
7. Run the following command to build the Docker image

```
Aditis-MacBook-Air:docker_tutorial_1 aditipandey$ sudo docker build -t my-node-app .
[+] Building 828.1s (10/10) FINISHED                                                                    docker:default
 => [internal] load build definition from dockerfile                                                              0.0s
 => => transferring dockerfile: 162B                                                                              0.0s
 => [internal] load metadata for docker.io/library/node:latest                                                    6.9s
 => [internal] load .dockerignore                                                                                 0.1s
 => => transferring context: 2B                                                                                   0.0s
 => [1/5] FROM docker.io/library/node:latest@sha256:cbe2d5f94110cea9817dd8c5809d05df49b4bd1aac5203f3594d88665ad37988  812.7s
 => => resolve docker.io/library/node:latest@sha256:cbe2d5f94110cea9817dd8c5809d05df49b4bd1aac5203f3594d88665ad37988    0.2s
 => => sha256:cbe2d5f94110cea9817dd8c5809d05df49b4bd1aac5203f3594d88665ad37988 6.41kB / 6.41kB                    0.0s
 => => sha256:f729d1a1b8afd4df4a78f27c27ca7d32361ea333ff0affd09daef0bf6ded2650 2.49kB / 2.49kB                    0.0s
 => => sha256:8cd46d290033f265db57fd808ac81c444ec5a5b3f189c3d6d85043b647336913 49.56MB / 49.56MB                  312.3s
 => => sha256:2ef13a9c33b09953afea4d8c6c6cfd76e04e97a24a63b109c0ea3670d3df4ccc 6.60kB / 6.60kB                    0.0s
 => => sha256:2e6afa3f266c11e8960349e7866203a9df478a50362bb5488c45fe39d99b2707 24.05MB / 24.05MB                  165.2s
 => => sha256:2e66a70da0bec13fb3d492fcdef60fd8a5ef0a1a65c4e8a4909e26742852f0f2 64.15MB / 64.15MB                  474.7s
 => => sha256:1c8ff076d818ad6b8557e03e10c83657cc716ab287c8380054ff91571c8cae81 211.27MB / 211.27MB                745.7s
 => => sha256:71a2ad2ab1a1619233b6fb9382a60ddb8fb77aab48877ed316017a6e21de71a0 3.33kB / 3.33kB                    315.1s
 => => extracting sha256:8cd46d290033f265db57fd808ac81c444ec5a5b3f189c3d6d85043b647336913                          18.1s
 => => sha256:8ed09065f0164a87a51e0ef49584beff3c51462ba2c03b19da23fcb95ef1ca40 53.86MB / 53.86MB                  510.7s
 => => extracting sha256:2e6afa3f266c11e8960349e7866203a9df478a50362bb5488c45fe39d99b2707                            4.6s
 => => sha256:8cc9946ce160852481b7c9b6bf311d32d50c76e972c072e2aad619a5dbf8ebdf 1.25MB / 1.25MB                    482.7s
 => => extracting sha256:2e66a70da0bec13fb3d492fcdef60fd8a5ef0a1a65c4e8a4909e26742852f0f2                           17.9s
 => => sha256:fa87db89e50a44939eff6087e76be9625ab46715a5944c64afbec3b71a0acc3e 451B / 451B                        484.0s
 => => extracting sha256:1c8ff076d818ad6b8557e03e10c83657cc716ab287c8380054ff91571c8cae81                           46.1s
 => => extracting sha256:71a2ad2ab1a1619233b6fb9382a60ddb8fb77aab48877ed316017a6e21de71a0                            0.0s
 => => extracting sha256:8ed09065f0164a87a51e0ef49584beff3c51462ba2c03b19da23fcb95ef1ca40                           18.2s
 => => extracting sha256:8cc9946ce160852481b7c9b6bf311d32d50c76e972c072e2aad619a5dbf8ebdf                            0.1s
 => => extracting sha256:fa87db89e50a44939eff6087e76be9625ab46715a5944c64afbec3b71a0acc3e                            0.0s
 => [internal] load build context                                                                                 0.6s
 => => transferring context: 2.26MB                                                                               0.5s
 => [2/5] WORKDIR /usr/src/app                                                                                    0.6s
 => [3/5] COPY package*.json ./                                                                                   0.1s
 => [4/5] RUN npm install                                                                                         6.2s
 => [5/5] COPY . .                                                                                                0.3s
 => exporting to image                                                                                            0.6s
 => => exporting layers                                                                                           0.6s
 => => writing image sha256:15548ee73cdc1543a6f9101a20907eb2414d592efdaf7073068a5e53f048ab25                       0.0s
 => => naming to docker.io/library/my-node-app                                                                    0.0s

View build details: docker-desktop://dashboard/build/default/default/foseusxbkwhcaw0vhj93v93t1

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
```

Verify by the docker images command.

```
Aditis-MacBook-Air:docker_tutorial_1 aditipandey$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
my-node-app   latest    15548ee73cdc   3 minutes ago    1.12GB
ubuntu        latest    b1e9cef3f297   3 weeks ago      78.1MB
nginx         latest    39286ab8a5e1   5 weeks ago      188MB
Aditis-MacBook-Air:docker_tutorial_1 aditipandey$
```

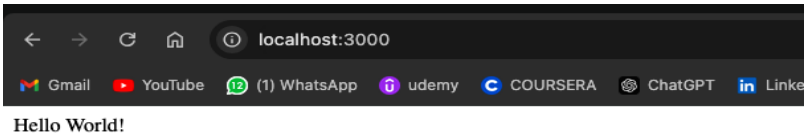
Step 4: Run the Docker Container with a Volume:

1. Use Docker to run your Node.js app and manage a volume to ensure file changes persist between container runs.
2. Run the following command to start the container.

```
sudo docker run -it --name app-container -p 8080:3000 my-node-app
```

Step 5: Access the Application:

3. Open a web browser and visit http://localhost:3000.
4. You should see the message: "Hello from Dockerized Node.js app!".



Step6:Push Docker Image:

5) Once, the docker image is built and now you can push your docker image.

6)You can tag the image first and then push it directly.

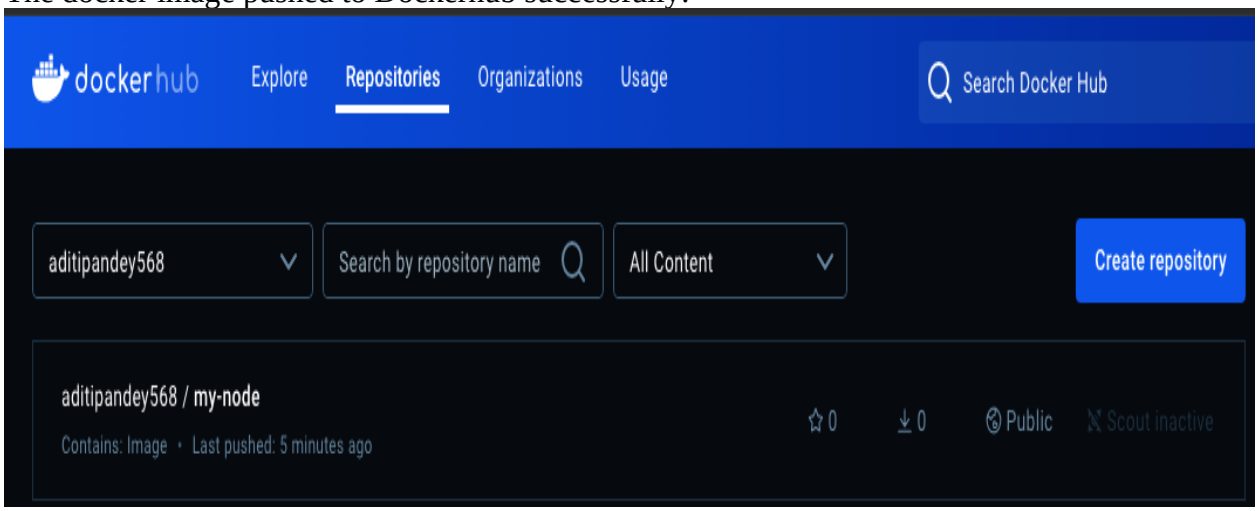
7)Sometimes, there is an access denied issue even though you are logged in. In that scenario, you can tag the image first and then push it.

COMMAND-

- docker tag my-node aditipandey568/my-node:latest
- docker push aditipandey568/my-node:latest

```
Aditis-MacBook-Air:docker_tutorial_1 aditipandey$ docker push aditipandey568/my-node:latest
The push refers to repository [docker.io/aditipandey568/my-node]
94c6be08afc6: Pushed
eb682a8bb461: Pushed
3b26596216ad: Pushed
ca6112316a3d: Pushed
820ae550c85c: Mounted from library/node
adcdaae72f66: Mounted from library/node
cee8a376a9a8: Pushing [======>                                          ]  24.48MB/177.3MB
9d53c06a7e68: Mounted from library/node
3a8081ce85fa: Mounted from library/node
045d8b74bf0d: Mounted from library/node
25879f85bbb0: Mounted from library/node
6abe10f2f601: Mounted from library/node
```

The docker image pushed to Dockerhub successfully.



To stop a specific container

```
Aditis-MacBook-Air:docker_tutorial_1 aditipandey$ docker stop 2b0c143c295c
2b0c143c295c
```

To stop all running Docker containers, you can use the following command:

```
Aditis-MacBook-Air:docker_tutorial_1 aditipandey$ sudo docker rm $(docker ps -a -q)
2b0c143c295c
ca240c45bea4
6f9c68c568bb
100d01574472
7271256e3af8
```

To remove all Docker containers, you can use the following command:

```
Aditis-MacBook-Air:docker_tutorial_1 aditipandey$ sudo docker rm $(docker ps -a -q)
2b0c143c295c
ca240c45bea4
6f9c68c568bb
100d01574472
7271256e3af8
Aditis-MacBook-Air:docker_tutorial_1 aditipandey$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
Aditis-MacBook-Air:docker_tutorial_1 aditipandey$
```

4.Result/Output/Writing Summary:

- Created a Dockerfile to define the Node.js environment.
- Built the Docker image using docker build -t my-node-app ..
- Run the application inside the container using docker run -it --name container_name -p 8080:3000 image_name.
- Tagged and pushed the Docker image to Docker Hub using docker push aditipandey568/my-node:latest.

5.Learning outcomes (What I have learnt):

1)Learned key Docker commands like docker build, docker run, and docker ps to manage containers and images.
2)Gained experience in creating a Dockerfile to build images and run Node.js applications inside containers.
3) Learned how to tag and push Docker images to Docker Hub, managing login credentials and repositories.
4) Understood how to bind ports between the host machine and containers to expose applications running inside Docker.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	**Parameters**	**Marks Obtained**	**Maximum Marks**
1.			
2.			
3.			