Experiment -2.1

Student Name: Ayush Pandey **UID: 22BDO10038**
Branch: CSE - DevOps **Section/Group: 22BCD-1/A**
Semester : 5th **Date of Performance : 16/09/24**
Subject Name : Docker & Kubernetes **Subject code : 22CSH-343**

1. Aim/Overview of the practical:

To run a Node.js application using Docker and push it to a public repository.

2. Apparatus:

PC, Web Browser, Docker Engine, Docker Hub, Ubuntu Linux

3. Steps for experiment/practical:

1. To Fetch the Node.js repository

```
ayush@Linux:~$ curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
2024-09-23 09:07:10 - Installing pre-requisites
Hit:1 http://in.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:3 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://in.archive.ubuntu.com/ubuntu jammy-backports InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20230311ubuntu0.22.04.1).
ca-certificates set to manually installed.
curl is already the newest version (7.81.0-1ubuntu1.18).
gnupg is already the newest version (2.2.27-3ubuntu2.1).
gnupg set to manually installed.
The following NEW packages will be installed:
  apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 263 not upgraded.
Need to get 1,510 B of archives.
After this operation, 170 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 apt-transport-https all 2.4.13 [1,510 B]
Fetched 1,510 B in 3s (573 B/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 208607 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.4.13_all.deb ...
Unpacking apt-transport-https (2.4.13) ...
Setting up apt-transport-https (2.4.13) ...
Get:1 https://deb.nodesource.com/node_18.x nodistro InRelease [12.1 kB]
Hit:2 http://in.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:4 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:5 http://in.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:6 https://deb.nodesource.com/node_18.x nodistro/main amd64 Packages [10.0 kB]
Fetched 22.2 kB in 2s (9,505 B/s)
Reading package lists... Done
2024-09-23 09:07:26 - Repository configured successfully.
2024-09-23 09:07:26 - To install Node.js, run: apt-get install nodejs -y
2024-09-23 09:07:26 - You can use N|solid Runtime as a node.js alternative
2024-09-23 09:07:26 - To install N|solid Runtime, run: apt-get install nsolid -y
```

1.2 Install Node.js (Command : sudo apt install nodejs -y)

```
ayush@Linux:~$ sudo apt install nodejs -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  javascript-common libc-ares2 libjs-highlight.js libnode72
Use 'sudo apt autoremove' to remove them.
The following packages will be REMOVED:
  nodejs-doc
The following packages will be upgraded:
  nodejs
1 upgraded, 0 newly installed, 1 to remove and 263 not upgraded.
Need to get 29.6 MB of archives.
After this operation, 177 MB of additional disk space will be used.
Get:1 https://deb.nodesource.com/node_18.x nodistro/main amd64 nodejs amd64 18.20.4-1nodesource1 [29.6 MB]
Fetched 29.6 MB in 16s (1,879 kB/s)
(Reading database ... 208611 files and directories currently installed.)
Removing nodejs-doc (12.22.9~dfsg-1ubuntu3.6) ...
(Reading database ... 208432 files and directories currently installed.)
Preparing to unpack .../nodejs_18.20.4-1nodesource1_amd64.deb ...
Unpacking nodejs (18.20.4-1nodesource1) over (12.22.9~dfsg-1ubuntu3.6) ...
dpkg: error processing archive /var/cache/apt/archives/nodejs_18.20.4-1nodesource1_amd64.deb (--unpack):
 trying to overwrite '/usr/share/systemtap/tapset/node.stp', which is also in package libnode72:amd64 12.22.9~dfsg-1ubunt
u3.6
```

3. Verify Node.js installations and package managers in the host system .

```
ayush@Linux:~$ node -v
v12.22.9
ayush@Linux:~$ npm -v
8.5.1
```

4. Create and Initialize the directory with npm and it get up package.json

```
ayush@Linux:~$ mkdir nodeapp
ayush@Linux:~$ cd nodeapp
ayush@Linux:~/nodeapp$ npm init -y
Wrote to /home/ayush/nodeapp/package.json:

{
  "name": "nodeapp",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

5. Create an file < app.js > and add this basic Node.js code to app.js:

```
ayush@Linux:~/nodeapp$ vi app.js
ayush@Linux:~/nodeapp$ cat app.js
const express = require('express');
const app = express();
const PORT = 3000;

app.get('/',(req,res) => {
        res.send('Hello from Docker Node.js App!');
});

app.listen(PORT,() => {
        console.log(`Server running on http://localhost:${PORT}`);
});
```

6. Create a Dockerfile in the <nodeapp> directory and add the following contents to the Dockerfile:

```
ayush@Linux:~/nodeapp$ vi Dockerfile
ayush@Linux:~/nodeapp$ cat Dockerfile
FROM node:20.17.0
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD ["node","app.js"]
```

7. Create a Docker volume .

```
ayush@Linux:~/nodeapp$ docker volume create mynodevolume
mynodevolume
```

8. Build the Docker image to get the node.js applications make sure about the version of node installed in the host system and the port no specified .

```
ayush@Linux:~/nodeapp$ docker build -t nodeappimage .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon   16.9kB
Step 1/7 : FROM node:20.17.0
20.17.0: Pulling from library/node
8cd46d290033: Pull complete
2e6afa3f266c: Pull complete
2e66a70da0be: Pull complete
1c8ff076d818: Pull complete
d01ef74013dd: Pull complete
17076c042f96: Pull complete
35daf8b1075a: Pull complete
f6afe88969eb: Pull complete
Digest: sha256:48db4f6ea21d134be744207225753a1730c4bc1b4cdf836d44511c36bf0e34d7
Status: Downloaded newer image for node:20.17.0
 ---> dd223fd5024d
Step 2/7 : WORKDIR /app
 ---> Running in f65c7df29d1f
Removing intermediate container f65c7df29d1f
 ---> df9b03ee50b1
Step 3/7 : COPY package*.json ./
 ---> 83433c6afe50
Step 4/7 : RUN npm install
 ---> Running in 10261afb61a6

up to date, audited 1 package in 421ms

found 0 vulnerabilities
Removing intermediate container 10261afb61a6
 ---> e03b74f124a4
Step 5/7 : COPY . .
```

```
Step 5/7 : COPY . .
 ---> 1099f4ee2cbe
Step 6/7 : EXPOSE 3000
 ---> Running in 9796375f7e12
Removing intermediate container 9796375f7e12
 ---> 228d4b233b68
Step 7/7 : CMD ["node","app.js"]
 ---> Running in 2f3091e60eb8
Removing intermediate container 2f3091e60eb8
 ---> 80d077b3c6b7
Successfully built 80d077b3c6b7
Successfully tagged nodeappimage:latest
```

9. Run the container in detacted mode and by specicifying the port and Check the container status by command : (docker ps)

```
ayush@Linux:~/nodeapp$ docker run -d -p 3000:3000 --name nodeapp -v mynodevolume:/app nodeappimage
0be85a4512da14f4d00c047c3e6330429d167ef804418f7db82666326f231025
ayush@Linux:~/nodeapp$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS         PORTS                                   NAMES
5f79b2bb74c3   ubuntu    "/bin/bash"              20 minutes ago  Up 20 minutes                                          networking-demo
c17311e4b364   ubuntu    "/bin/bash"              28 minutes ago  Up 28 minutes                                          three
49fd081c1626   nginx     "/docker-entrypoint.…"   29 minutes ago  Up 29 minutes  0.0.0.0:8080->80/tcp, :::8080->80/tcp   nifty_noether
ayush@Linux:~/nodeapp$ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED            STATUS                      PORTS                                   NAMES
0be85a4512da   nodeappimage   "docker-entrypoint.s…"   About a minute ago  Exited (1) About a minute ago                                      nodeapp
5f79b2bb74c3   ubuntu         "/bin/bash"              21 minutes ago      Up 21 minutes                                                      networking-demo
c17311e4b364   ubuntu         "/bin/bash"              29 minutes ago      Up 29 minutes                                                      three
49fd081c1626   nginx          "/docker-entrypoint.…"   30 minutes ago      Up 30 minutes               0.0.0.0:8080->80/tcp, :::8080->80/tcp  nifty_noether
2faedcd77b50   ubuntu         "sleep 1000"             31 minutes ago      Exited (0) 14 minutes ago                                          objective_blackburn
b1101fa59dd1   edbfe74c41f8   "/bin/bash"              4 weeks ago         Exited (137) 4 weeks ago                                           awesome_sutherland
```

10. After verifying the container , stop the container using the command docker stop <imgID>

11. Check for docker image name and tag to be pushed to the public repository (dockerhub).

```
ayush@Linux:~/nodeapp$ docker images
REPOSITORY     TAG        IMAGE ID       CREATED          SIZE
nodeappimage   latest     80d077b3c6b7   5 minutes ago    1.1GB
ubuntu         latest     b1e9cef3f297   3 weeks ago      78.1MB
node           20.17.0    dd223fd5024d   4 weeks ago      1.1GB
nginx          latest     39286ab8a5e1   5 weeks ago      188MB
ubuntu         <none>     edbfe74c41f8   7 weeks ago      78.1MB
```

12.Push the container to the public repository ,

i) Login : Username: docker login -u <username> -p <*****>

ii) Tag : docker tag <image-id> <docker-username>/<repository-name>:<tag>

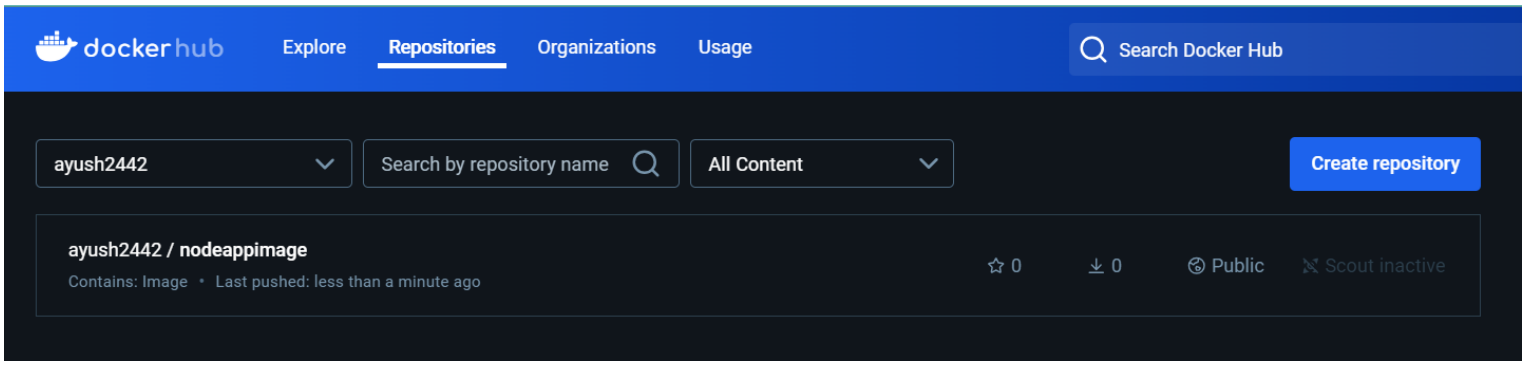
iii) Push : docker push yourusername/my-node-app:latest

```
ayush@Linux:~/nodeapp$ docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations using SSO. Learn more at https:/
/docs.docker.com/go/access-tokens/

Username: ayush2442
Password:
WARNING! Your password will be stored unencrypted in /home/ayush/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ayush@Linux:~/nodeapp$ docker push ayush2442/nodeappimage
Using default tag: latest
The push refers to repository [docker.io/ayush2442/nodeappimage]
a041a96d2fe0: Pushed
cb37bfdb0304: Pushed
a1542b6cba34: Pushed
54f209e17a6d: Pushed
021f68d24cca: Mounted from library/node
d0b827929a8c: Mounted from library/node
d5ff45ffb8ad: Mounted from library/node
7b1637822467: Mounted from library/node
3a8081ce85fa: Mounted from library/node
045d8b74bf0d: Mounted from library/node
25879f85bbb0: Mounted from library/node
6abe10f2f601: Mounted from library/node
latest: digest: sha256:6488968171382e417a65cc657bdb1870b3043b2c781b3282a2aae95bbca6cfb6 size: 2832
```

4. Result/Output/Writing Summary:



1. Docker is successfully installed allowing for containerized environments to run efficiently.
2. Docker volumes are properly set up, ensuring that application data is stored persistently outside the container, even when containers are removed.
3. The Node.js app is built, deployed, and running inside a Docker container, accessible through a mapped port on the host system.
4. The Docker image of the application is tagged and pushed to Docker Hub (or another registry), making it available for future use or sharing.
5. Through this process, debugging and troubleshooting skills for Docker-related issues like permission errors, image build failures, and registry access are improved.

Learning outcomes (What I have learnt):

1. I have learnt the concept of containerization.

2. I have learnt to configure Docker to work with different environments.

3. I have learnt how to build docker images using Dockerfile.

4. I have learnt the purpose of Docker volumes and their role in data persistence.

5. I have learnt how to use Docker Hub to push Docker images.

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.			