

NAME : GOURAV DAS
SAP : 500122586
BATCH : B2 DEVOPS

LAB EXERCISE 4- BUILDING A DOCKER IMAGE FOR AN HTML APP USING NGINX

1. SETUP

YOU WILL NEED:

- **DOCKER INSTALLED ON YOUR MACHINE.**
- **A SIMPLE HTML FILE FOR THE APP.**

2. STEP 1: CREATE THE HTML FILE

CREATE A DIRECTORY FOR YOUR HTML APP AND PLACE AN INDEX.HTML FILE IN IT.

```
mkdir nginx-html-app
```

```
cd nginx-html-app
```

INSIDE THE NGINX-HTML-APP DIRECTORY, CREATE THE HTML FILE.

```
touch index.html
```

EDIT THE INDEX.HTML FILE WITH THE FOLLOWING CONTENT (OR ANY CUSTOM HTML CONTENT YOU WANT):

```
<!DOCTYPE html>
<html>
<head>
  <title>Welcome to My Nginx HTML App</title>
</head>
<body>
```

```
<h1>Hello, Nginx Docker!</h1>
<p>This is a simple HTML app served by Nginx in a Docker container.</p>
</body>
</html>
```

```

root@nginx-html-app -- ssh - 230x88
ACDns-MacBook-Air:~$ kubectl exec ashchukashand-nginx-html-app
ACDns-MacBook-Air:~$ kubectl exec ashchukashand-16-nginx-html-app
ACDns-MacBook-Air:~$ kubectl exec ashchukashand-frontend-nginx.html
ACDns-MacBook-Air:~$ kubectl exec ashchukashand-16-index.html
ACDns-MacBook-Air:~$ kubectl exec ashchukashand cat index.html
<DOCTYPE html>
<html>
<head>
<title>Welcome to My Nginx HTML App</title>
</head>
<body>
<div>Hello, Nginx Docker!</div>
<p>This is a simple HTML app served by Nginx in a Docker container.</p>
</body>
</html>
ACDns-MacBook-Air:~$ kubectl exec ashchukashand

```

3. STEP 2: CREATE A DOCKERFILE

IN THE SAME DIRECTORY, CREATE A **DOCKERFILE**. THIS FILE WILL DEFINE HOW TO BUILD THE **DOCKER** IMAGE USING **NGINX** AS THE BASE IMAGE.

touch Dockerfile

EDIT THE DOCKERFILE AND ADD THE FOLLOWING CONTENT:

```
FROM nginx:latest
COPY index.html /usr/share/nginx/html/
EXPOSE 80
```

4. STEP 3: BUILD THE DOCKER IMAGE

NOW THAT YOU HAVE THE DOCKERFILE AND INDEX.HTML, IT'S TIME TO BUILD THE DOCKER IMAGE.

RUN THE FOLLOWING COMMAND TO BUILD THE IMAGE, GIVING IT A TAG (E.G., NGINX-HTML- APP):

```
docker build -t nginx-html-app .
```

DOCKER WILL USE THE NGINX BASE IMAGE, COPY YOUR INDEX.HTML INTO THE APPROPRIATE DIRECTORY, AND BUILD THE IMAGE.

[illegible]

5. STEP 4: RUN THE DOCKER CONTAINER

AFTER BUILDING THE IMAGE, YOU CAN RUN THE CONTAINER WITH THE FOLLOWING COMMAND:

```
docker run -d -p 8006:80 nginx-html-app
```

THIS COMMAND RUNS THE CONTAINER IN DETACHED MODE (-D) AND MAPS PORT 8006 ON YOUR HOST MACHINE TO PORT 80 INSIDE THE CONTAINER, WHERE NGINX IS SERVING YOUR HTML APP.

b840c136497597069f5c3ca0205d5fede3156efb2fd52f4cdd91bfd2b58fad03

6. STEP 5: VERIFY

OPEN A BROWSER AND GO TO `HTTP://LOCALHOST:8006`. YOU SHOULD SEE YOUR HTML PAGE

WITH THE MESSAGE “HELLO, NGINX DOCKER!”.



7. STEP 6: STOP AND REMOVE THE CONTAINER

ONCE YOU'RE DONE, YOU CAN STOP AND REMOVE THE CONTAINER:

```
docker ps # to see running containers
```

```
docker stop <container-id>
```

```
docker rm <container-id>
```

```
AKSHU-MacBook-Air:nginx-html-app akshuchauhan$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS                               NAMES
9848c1364975   nginx-html-app   /docker-entrypoint.sh   About 2 minutes ago   Up About 2 minutes   0.0.0.0:8006->80/tcp, [::]:8006->80/tcp   intelligent_williams
AKSHU-MacBook-Air:nginx-html-app akshuchauhan$ docker stop 9848c1364975
9848c1364975
AKSHU-MacBook-Air:nginx-html-app akshuchauhan$ docker rm 9848c1364975
9848c1364975
AKSHU-MacBook-Air:nginx-html-app akshuchauhan$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS                               NAMES
```