Objective:

The Main Objective of this assignment is to understanding Docker and containerization by Dockerizing a simple HTML page using Nginx as the web server.

Requirements:

Install docker in ubuntu machine

```
sudo apt install docker.io -y
```

Please find below a Sample HTMP Page which we will be using for deploying.

Basic HTML Page:

Create a plain HTML page named `index.html` with some content (e.g., "Hello, Docker!").

```
#------------------index.html---------------------------------
<!DOCTYPE html>
<html>
<head>
       <title>Welcome to Tilak's World of Devops!</title>
</head>
<body>
    <h1>Hello, Welcome to Tilak's World of Devops!!!</h1>
    <p>This is a plain HTML page for Dockerizing a Plain HTML Page with Nginx.</p>
</body>
</html>#-----------------------------------------------------------------
```

Nginx Configuration:

Since we are hosting the Webserver using nginx below is the configuration file that we need to copy to the container using docker file

Create an Nginx configuration file named `nginx.conf` that serves the `index.html` page.

Configure Nginx to listen on port 80.

```
#----------------------------nginx.conf-------------------------
events {}
http {
    server {
        listen 80;
        location / {
            root /usr/share/nginx/html;
```

```
            index index.html;
        }
    }
}
#---------------------------------------------------------------------
```

Dockerfile:

Create a `Dockerfile` to define the Docker image.

Use an official Nginx base image.

Copy the `index.html` and `nginx.conf` files into the appropriate location in the container.

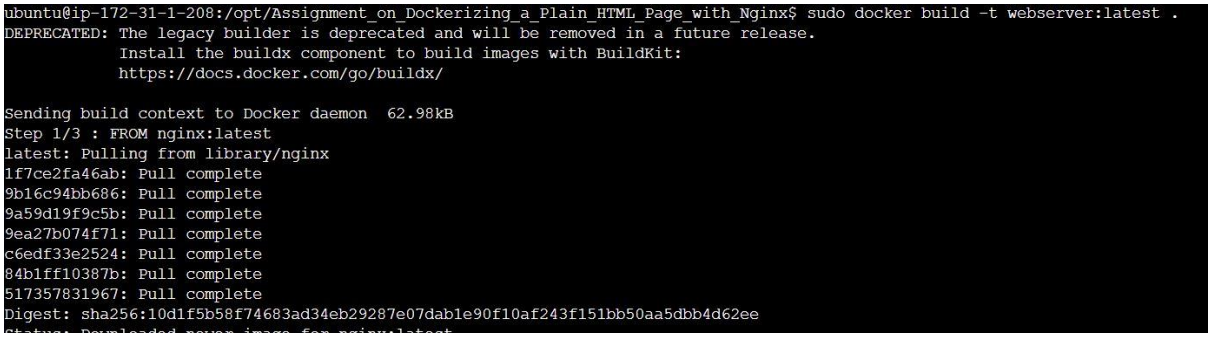
Ensure that the Nginx server is started when the container is run.

```
#--------------------------Dockerfile----------------------
FROM nginx:latest
COPY index.html /usr/share/nginx/html/index.html
COPY nginx.conf /etc/nginx/nginx.conf
#------------------------------------------------------------
```

Building the Docker Image:

Build the Docker image using the `Dockerfile`.

```
docker build -t webserver:v1 -f dockerfile .
```



Now we Test the image

```
docker run -it -d  -p 8080:80 webserver:v1
```

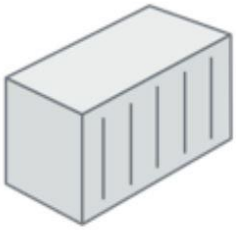
```
ubuntu@ip-172-31-1-208:/opt/Assignment_on_Dockerizing_a_Plain_HTML_Page_with_Nginx$ curl localhost:8080
<!DOCTYPE html>
<html>
<head>
        <title>Welcome to Tilak's World of Devops!</title>
</head>
<body>
    <h1>Hello, Welcome to Tilak's World of Devops!!!</h1>
    <p>This is a plain HTML page for Dockerizing a Plain HTML Page with Nginx.</p>
</body>
</html>
```

Push the image on ECR

Make the public repository and push them on the ECR

```
ubuntu@ip-172-31-1-208:/opt/Assignment_on_Dockerizing_a_Plain_HTML_Page_with_Nginx$ sudo docker push public.ecr.aws/e5q4n0t8/tilak-docker:latest
The push refers to repository [public.ecr.aws/e5q4n0t8/tilak-docker]
6b338c7312a7: Pushed
f7d8df26fc18: Pushed
0d0e9c83b6f7: Pushed
cddc309885a2: Pushed
c2d3ab485d1b: Pushed
66283570f41b: Pushed
f5525891d9e9: Pushed
8ae474e0cc8f: Pushed
92770f546e06: Pushed
latest: digest: sha256:67008d954f60b84e188d31d813835b026c7d033ae07581c0ec303b03968c6064 size: 2192
```

Amazon ECR Public Gallery > search > e5q4n0t8 > tilak-docker



e5q4n0t8/tilak-docker (0 downloads)

public.ecr.aws/e5q4n0t8/tilak-docker:latest ▼ Copy

Updated 33 minutes ago

TODO

About | Usage | Image tags

TODO

public.ecr.aws/e5q4n0t8/tilak-docker:latest