[HCM23_FRF.V_Devops_02] Docker_ Topic 4: Docker Compose

Task1: Create a simple web server using Docker Compose that runs a basic HTML page. Access the page from your browser

In our workdir we create 3 file: Dockerfile, compose.yaml, index.html for our web html:

Dockerfile

```
FROM nginx

COPY index.html /usr/share/nginx/html/index.html

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

File index.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Welcome to NTD151298 Website</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #d6acac;
            margin: 0;
            padding: 0;
            display: flex;
            align-items: center;
            justify-content: center;
            height: 100vh;
        }
```

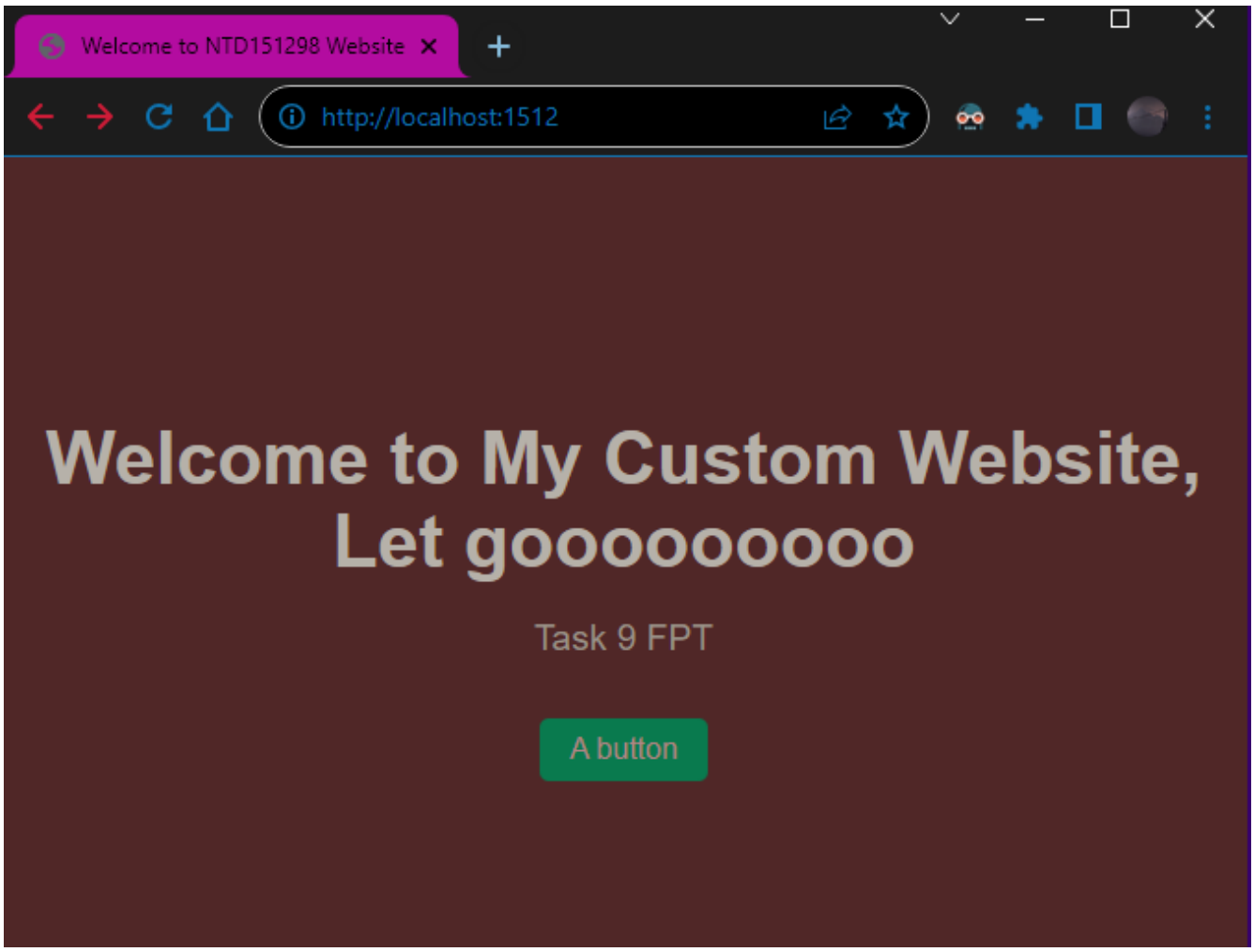
Compose.yaml (with build and name to our container)

```
version: '3'
services:
  task1:
    build:
      context: .
      dockerfile: Dockerfile
    container_name: fist_task_webapp
    ports:
      - "1512:80"
```

We using command docker compose up to build our docker file and deploy it to our container

```
PS D:\Devops_FPT_Foudations\docker\Final\Task1> docker compose up
[+] Building 4.6s (8/8) FINISHED                                                                                docker:default
 => [task1 internal] load .dockerignore                                                                                  0.1s
 => => transferring context: 2B                                                                                          0.1s
 => [task1 internal] load build definition from Dockerfile                                                               0.1s
 => => transferring dockerfile: 137B                                                                                     0.1s
 => [task1 internal] load metadata for docker.io/library/nginx:latest                                                    3.6s
 => [task1 auth] library/nginx:pull token for registry-1.docker.io                                                       0.0s
 => [task1 internal] load build context                                                                                  0.1s
 => => transferring context: 1.54kB                                                                                      0.0s
 => CACHED [task1 1/2] FROM docker.io/library/nginx@sha256:104c7c5c54f2685f0f46f3be607ce60da7085da3eaa5ad22d3d9f01594295e9c  0.0s
 => [task1 2/2] COPY index.html /usr/share/nginx/html/index.html                                                         0.2s
 => [task1] exporting to image                                                                                           0.1s
 => => exporting layers                                                                                                  0.1s
 => => writing image sha256:6c7e3c3ae1da799233bfa756f99e00aba1bc6329ef49d7d96c13faef0de80294                             0.0s
 => => naming to docker.io/library/task1-task1                                                                           0.0s
[+] Running 2/2
 ✓ Network task1_default       Created                                                                                   0.8s
 ✓ Container fist_task_webapp  Created                                                                                   0.3s
Attaching to fist_task_webapp
fist_task_webapp  | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
fist_task_webapp  | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
```

We then enter localhost:1512



Task2: Create a Docker Compose file for a multi-container deployment: 1 container with nginx:1.22 image and 1 container with httpd:latestimage. Expose the above containers at 0.0.0.0:80 Ensure they can communicate

Fist we wirte our docker compose file with the deployment of 2 coantainer nginx and httpd port 80

powershell | compose.yaml U

docker > Final > Task2 > compose.yaml

```yaml
version: '3'
services:

  nginx:
    image: ${nginx_image}
    container_name: ${nginx_container}
    ports:
      - "${nginx_port}"
    networks:
      - task_02

  httpd:
    image: ${http_image}
    container_name: ${http_conatainer}
    ports:
      - "${http_port}"
    networks:
      - task_02

networks:
  task_02:
```

.env U

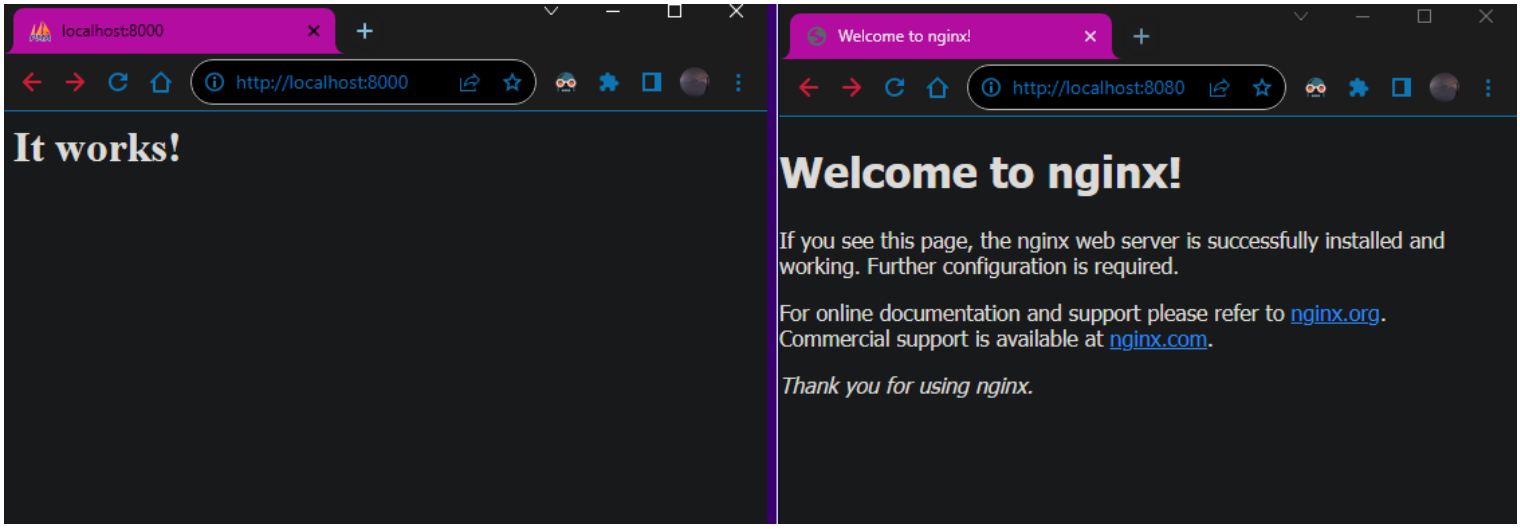
docker > Final > Task2 > .env

```env
nginx_image=nginx:1.22
nginx_container=nginx001
nginx_port=0.0.0.0:8080:80

http_image=httpd:latest
http_conatainer=http001
http_port=0.0.0.0:8000:80
```

We using command docker compose up -d to start 2 of our conatainer

```
PS D:\Devops_FPT_Foudations\docker\Final\Task2> docker compose up -d
[+] Running 7/7
 ✓ nginx 6 layers [⣿⣿⣿⣿⣿⣿]      0B/0B      Pulled                                    16.3s
   ✓ f1f26f570256 Already exists                                                        0.0s
   ✓ fd03b214f774 Already exists                                                        0.0s
   ✓ ef2fc869b944 Already exists                                                        0.0s
   ✓ ac713a9ef2cc Already exists                                                        0.0s
   ✓ fd071922d543 Already exists                                                        0.0s
   ✓ 2a9f38700bb5 Already exists                                                        0.0s
[+] Running 1/3
 ✓ Network task2_task_02  Created                                                       1.0s
 - Container nginx001     Starting                                                      1.8s
 - Container http001      Starting                                                      1.8s
```

We have success expose httpd to port 8000 and nginx to port 8080



To ensure they can communicate we enter 1 of the running container

We enter nginx container

```
PS D:\Devops_FPT_Foudations\docker\Final\Task2> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS                   NAMES
ac72ade9d154   httpd:latest   "httpd-foreground"       5 minutes ago   Up 5 minutes   0.0.0.0:8000->80/tcp    http001
ef0f0ea8e081   nginx:1.22     "/docker-entrypoint.…"   7 minutes ago   Up 7 minutes   0.0.0.0:8080->80/tcp    nginx001
PS D:\Devops_FPT_Foudations\docker\Final\Task2> docker exec -it nginx001 bash
root@ef0f0ea8e081:/# ls
bin   dev                  docker-entrypoint.sh  home  lib64  mnt  proc  run   srv  tmp  var
boot  docker-entrypoint.d  etc                   lib   media  opt  root  sbin  sys  usr
root@ef0f0ea8e081:/#
```

And then we call httpd conatainer with it domain name

```
root@ef0f0ea8e081:/# curl http001
<html><body><h1>It works!</h1></body></html>
root@ef0f0ea8e081:/#
exit
PS D:\Devops_FPT_Foudations\docker\Final\Task2>
```

We also can inspect network of 2 conatainer we just create

```
PS D:\Devops_FPT_Foudations\docker\Final\Task2> docker network ls
NETWORK ID     NAME              DRIVER    SCOPE
41f40f535680   bridge            bridge    local
a2cec830f010   docker_gwbridge   bridge    local
eeb939a64378   host              host      local
24ml2wzapgd7   ingress           overlay   swarm
fb9c40f5fca2   none              null      local
1f24f0930bf2   task2_task_02     bridge    local
82eb06763a3e   test_demo_net_1   bridge    local
PS D:\Devops_FPT_Foudations\docker\Final\Task2> docker network inspect task2_task_02
```

And find out them ip address within docker user bridge network and call them using their ip add

```
"Containers": {
    "ac72ade9d154a75df58c4cb116650ada253f096939ef91f9ca43ed4352d491ee": {
        "Name": "http001",
        "EndpointID": "13cdd5dff970fb57dda3a868a5018a39167fbf273ba146f79326736727b29cf6",
        "MacAddress": "02:42:ac:15:00:03",
        "IPv4Address": "172.21.0.3/16",
        "IPv6Address": ""
    },
    "ef0f0ea8e0815ab400558b7ca81967747c577a8b3cf10804bb05ac2b9b8ef11f": {
        "Name": "nginx001",
        "EndpointID": "5bb1d42b7380173f76253a7599149318339c54d315b0be67ace6cdab737619c5",
        "MacAddress": "02:42:ac:15:00:02",
        "IPv4Address": "172.21.0.2/16",
        "IPv6Address": ""
```

Call http using it’s ip address

```
PS D:\Devops_FPT_Foudations\docker\Final\Task2> docker exec -it nginx001 bash
root@ef0f0ea8e081:/# curl 172.21.0.3
<html><body><h1>It works!</h1></body></html>
root@ef0f0ea8e081:/#
exit
PS D:\Devops_FPT_Foudations\docker\Final\Task2>
```

Task3: Use Docker Compose to set environment variables for the 2 container above: ENV=dev for both container; APP=frontend for nginxcontainer and APP=backend for httpdcontainer. Check if those variable are available in each container

We modify our compose.yaml file we create at task 2, we add enviroment config for nginx and httpd

powershell | compose.yaml U

docker > Final > Task2 > compose.yaml

```yaml
version: '3'
services:
  # nginx container and enviroment config
  nginx:
    image: ${nginx_image}
    container_name: ${nginx_container}
    environment:
      - ENV=${nginx_env}
      - APP=${nginx_app}
    ports:
      - "${nginx_port}"
    networks:
      - task_02
  # httpd container and enviroment config
  httpd:
    image: ${http_image}
    container_name: ${http_conatainer}
    environment:
      - ENV=${http_env}
      - APP=${http_app}
    ports:
      - "${http_port}"
    networks:
      - task_02
# network use for 2 container
networks:
  task_02:
```

.env U

docker > Final > Task2 > .env

```env
# nginx image, port and container config
nginx_image=nginx:1.22
nginx_container=nginx001
nginx_port=0.0.0.0:8080:80
# nginx enviroment config
nginx_env=dev
nginx_app=frontend

# httpd image, port and container config
http_image=httpd:latest
http_conatainer=http001
http_port=0.0.0.0:8000:80
#httpd enviroment config
http_env=dev
http_app=backend
```

We then run new compose.yaml file

```
[+] Running 3/3 _Foudations\docker\Final\Task2>
 ✓ Network task2_task_02  Created                    0.7s
 ✓ Container nginx001     Started                    2.3s
 ✓ Container http001      Started                    2.4s
```

We enter nginx container to check enviroment variables we set

```
PS D:\Devops_FPT_Foudations\docker\Final\Task2> docker exec -it nginx001 bash
root@545a0d77efb2:/# echo $ENV
dev
root@545a0d77efb2:/# echo $APP
frontend
root@545a0d77efb2:/#
exit
```

We do the same with httpd container and check enviroment variables we set

```
PS D:\Devops_FPT_Foudations\docker\Final\Task2> docker exec -it http001 bash
root@098a085b62c3:/usr/local/apache2# echo $ENV
dev
root@098a085b62c3:/usr/local/apache2# echo $APP
backend
root@098a085b62c3:/usr/local/apache2#
exit
```

Task4: Set up a Docker Compose file that uses volume mounting: create a bind mount volume binding /tmpof the host filesystemand mount it to both of the container above at any location. Create a file named index.html in the /tmpof the host filesystemto see if it is available in both containerv

Fist we create tmpof folder from the host filesystemand with no file in it

```
PS D:\Devops_FPT_Foudations\docker\Final\Task4> ls


    Directory: D:\Devops_FPT_Foudations\docker\Final\Task4


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         8/21/2023   7:30 PM                before
d-----         8/21/2023   7:31 PM                tmpof
-a----         8/21/2023   7:30 PM            432 .env
-a----         8/21/2023   7:21 PM            553 compose.yaml


PS D:\Devops_FPT_Foudations\docker\Final\Task4> ls tmpof
PS D:\Devops_FPT_Foudations\docker\Final\Task4>
```

We set up a docker compose file that uses volume mounting: bind mount folder /tmpof from host filesystemand to folder that will display our web front page in 2 container nginx and httpd

```yaml
version: '3'
services:
  # nginx container and enviroment config
  nginx:
    image: ${nginx_image}
    container_name: ${nginx_container}
    volumes:
      - ${bind_dir}:${nginx_html}
    ports:
      - "${nginx_port}"
    networks:
      - task_02
  # httpd container and enviroment config
  httpd:
    image: ${http_image}
    container_name: ${http_conatainer}
    volumes:
      - ${bind_dir}:${http_html}
    ports:
      - "${http_port}"
    networks:
      - task_02
# network use for 2 container
networks:
  task_02:
```

```env
# nginx image, port and container config
nginx_image=nginx:1.22
nginx_container=nginx001
nginx_port=0.0.0.0:8080:80
# nginx display web
nginx_html=/usr/share/nginx/html

# httpd image, port and container config
http_image=httpd:latest
http_conatainer=http001
http_port=0.0.0.0:8000:80
# http display web
http_html=/usr/local/apache2/htdocs

# bindmount dir
bind_dir=D:\Devops_FPT_Foudations\docker\Final\Task4\tmpof
```

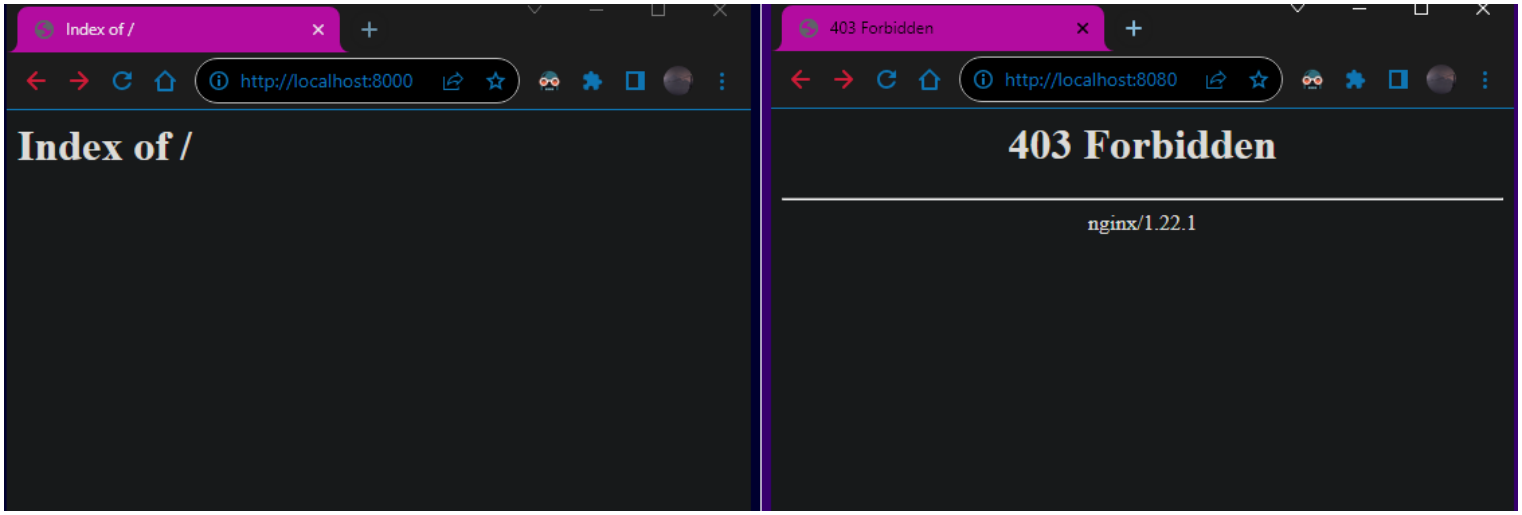
${bind_dir}=D:\Devops_FPT_Foudations\docker\Final\Task4\tmpof (host files folder)

${ nginx_html }=/usr/share/nginx/html (folder display nginx frontend web)

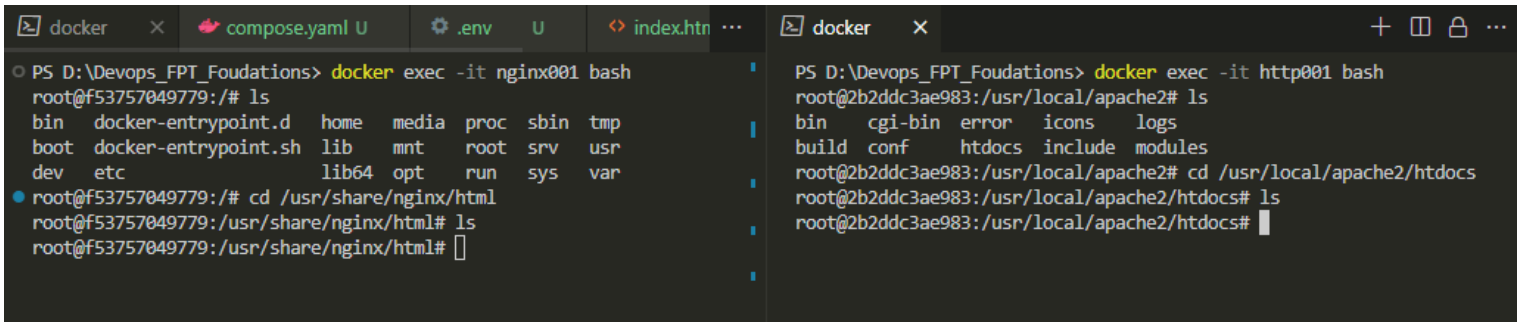
${ http_html }=/usr/local/apache2/htdocs (folder display httpd frontend web)

```
PS D:\Devops_FPT_Foudations\docker\Final\Task4> docker compose up -d
[+] Running 3/3
 ✓ Network task4_task_02  Created
 ✓ Container nginx001     Started
 ✓ Container http001      Started
PS D:\Devops_FPT_Foudations\docker\Final\Task4>
```

We fist check nginx and httpd web

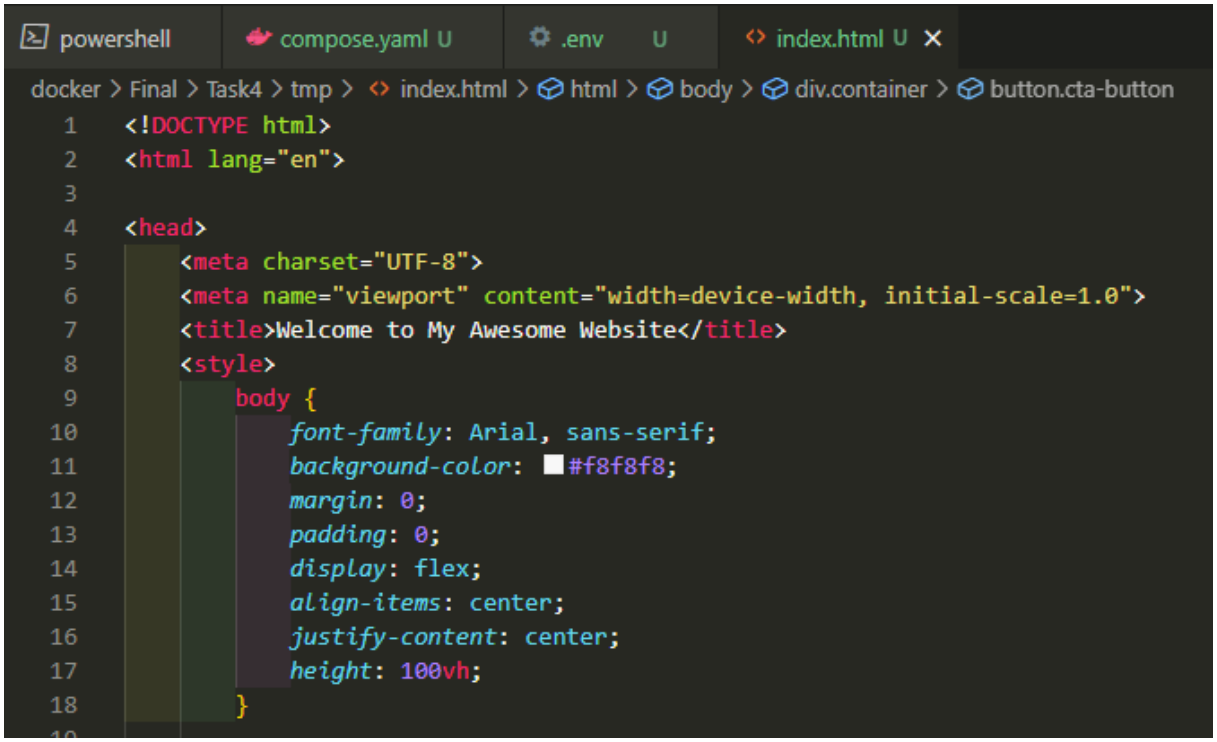


Folder that bind mount in 2 container

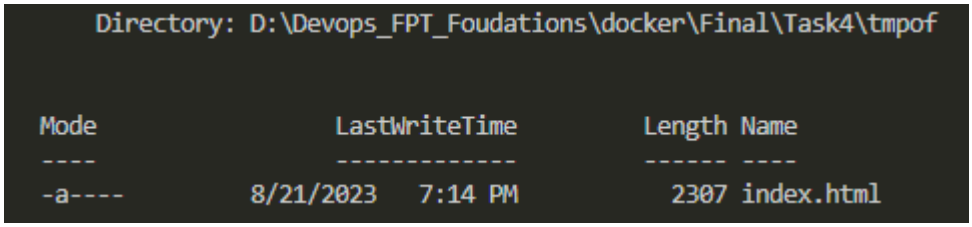


As you can see in the folder there is notthing to display

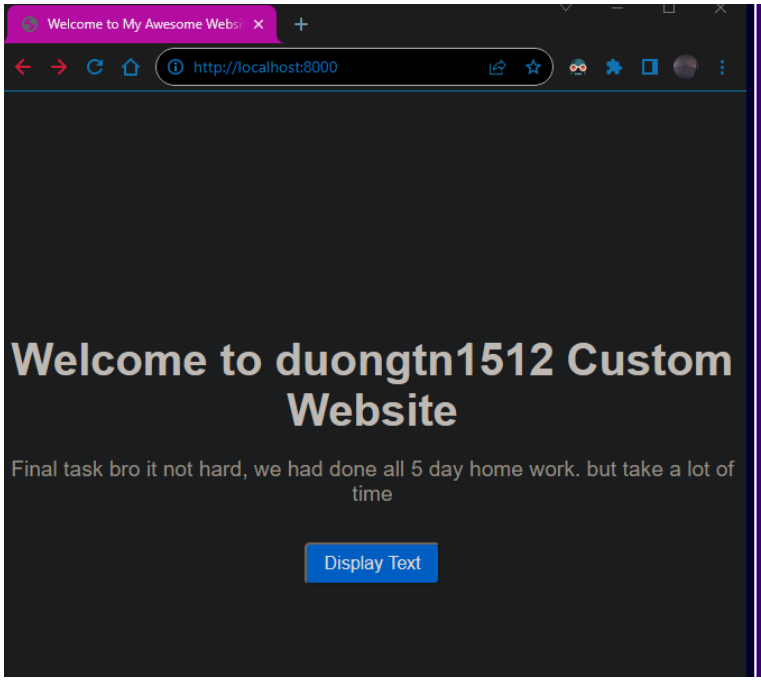
Now we create a html file

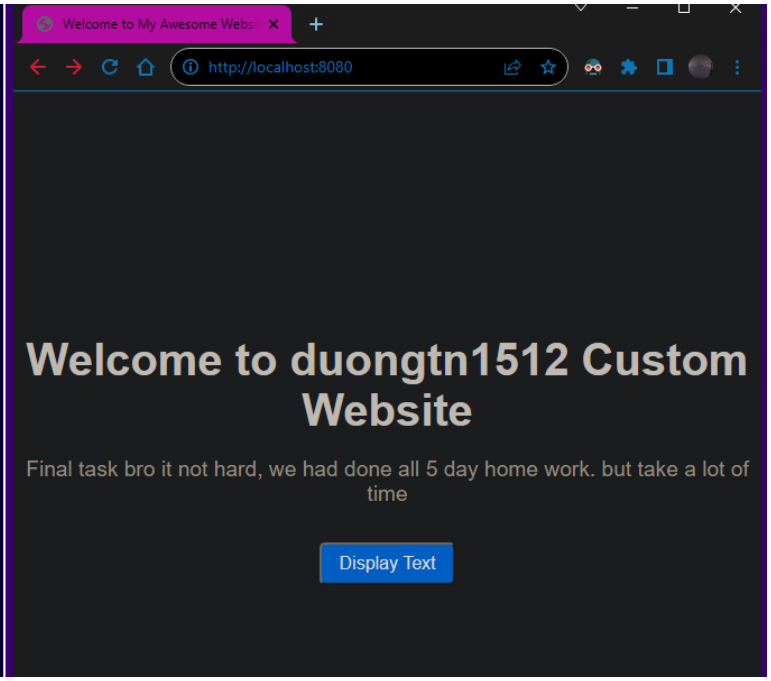


We then put the index.html file to /tmpof folder



We recheck 2 container now






```
PS D:\Devops_FPT_Foudations> docker exec -it nginx001 bash
root@f53757049779:/# ls
bin   docker-entrypoint.d   home   media  proc  sbin  tmp
boot  docker-entrypoint.sh  lib    mnt    root  srv   usr
dev   etc                   lib64  opt    run   sys   var
root@f53757049779:/# cd /usr/share/nginx/html
root@f53757049779:/usr/share/nginx/html# ls
root@f53757049779:/usr/share/nginx/html# ls
index.html
root@f53757049779:/usr/share/nginx/html# cat index.html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=
1.0">
    <title>Welcome to My Awesome Website</title>
    <style>
```

```
PS D:\Devops_FPT_Foudations> docker exec -it http001 bash
root@2b2ddc3ae983:/usr/local/apache2# ls
bin    cgi-bin  error   icons    logs
build  conf     htdocs  include  modules
root@2b2ddc3ae983:/usr/local/apache2# cd /usr/local/apache2/htdocs
root@2b2ddc3ae983:/usr/local/apache2/htdocs# ls
root@2b2ddc3ae983:/usr/local/apache2/htdocs# ls
index.html
root@2b2ddc3ae983:/usr/local/apache2/htdocs# cat index.html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=
1.0">
    <title>Welcome to My Awesome Website</title>
    <style>
        body {
```

We add text.txt in the tmpof folder for frontend display text

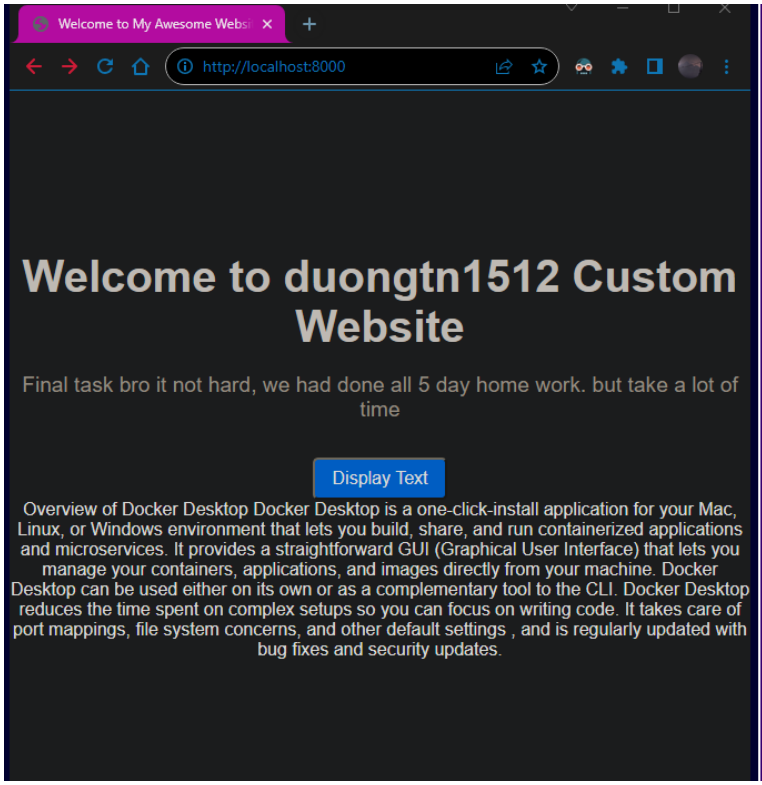
text.txt U | docker | compose.yaml U | .env U | docker | powershell

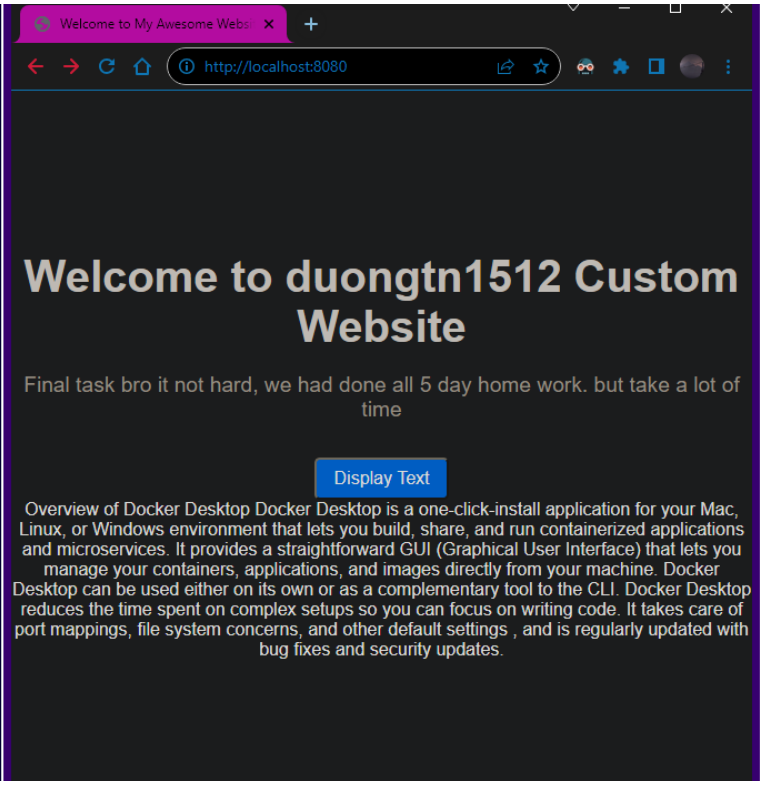
docker > Final > before > text.txt

```
Overview of Docker Desktop
Docker Desktop is a one-click-install application for your Mac, Linux, or Windows
environment that lets you build, share, and run containerized applications and microservices.

It provides a straightforward GUI (Graphical User Interface) that lets you manage your
containers, applications, and images directly from your machine. Docker Desktop can be
used either on its own or as a complementary tool to the CLI.

Docker Desktop reduces the time spent on complex setups so you can focus on writing
code. It takes care of port mappings, file system concerns, and other default settings
, and is regularly updated with bug fixes and security updates.
```

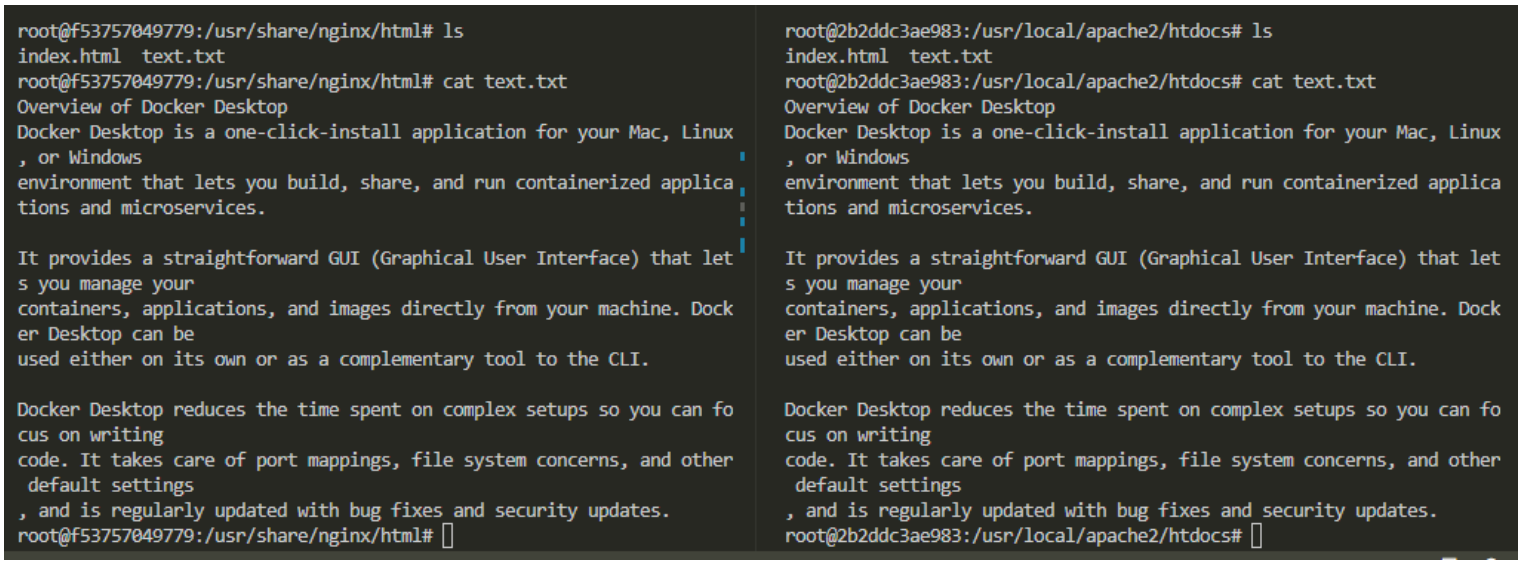
```
    Directory: D:\Devops_FPT_Foudations\docker\Final\Task4\tmpof


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        8/21/2023   7:14 PM           2307 index.html
-a----        8/19/2023   6:56 PM            691 text.txt
```

Now for our final check





In conatiner



End.