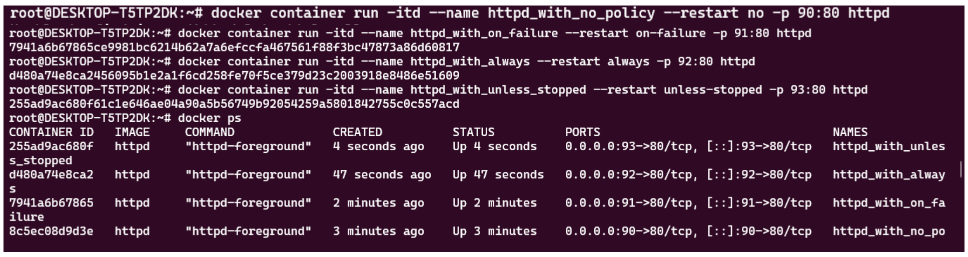
Task 19_36 On Docker

30 October 2025 16:31

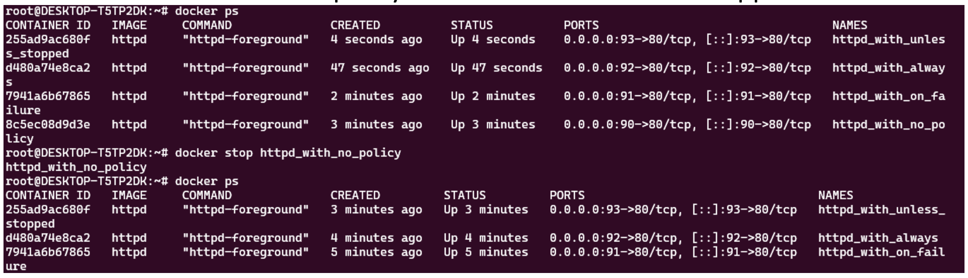
Task 19. Run four HTTPD Docker containers with distinct, meaningful names, and apply restart policies (NO, On-Failure, Always, and Unless-Stopped) to each of the four containers, respectively. Demonstrate that the restart policies function as expected.

# creation of four containers and listing them.



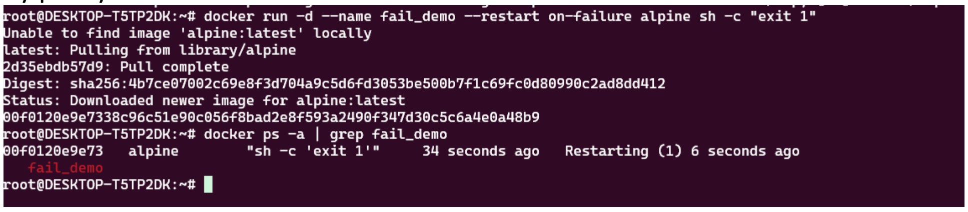
```
root@DESKTOP-T5TP2DK:~# docker container run -itd --name httpd_with_no_policy --restart no -p 90:80 httpd
root@DESKTOP-T5TP2DK:~# docker container run -itd --name httpd_with_on_failure --restart on-failure -p 91:80 httpd
7941a6b67865ce9981bc6214b62a7a6efccfa467561f88f3bc47873a86d60817
root@DESKTOP-T5TP2DK:~# docker container run -itd --name httpd_with_always --restart always -p 92:80 httpd
d480a74e8ca2456095b1e2a1f6cd258fe70f5ce379d23c2003918e8486e51609
root@DESKTOP-T5TP2DK:~# docker container run -itd --name httpd_with_unless_stopped --restart unless-stopped -p 93:80 httpd
255ad9ac680f61c1e646ae04a90a5b56749b92054259a5801842755c0c557acd
root@DESKTOP-T5TP2DK:~# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED          STATUS          PORTS                               NAMES
255ad9ac680f   httpd     "httpd-foreground"   4 seconds ago    Up 4 seconds    0.0.0.0:93->80/tcp, [::]:93->80/tcp   httpd_with_unles
s_stopped
d480a74e8ca2   httpd     "httpd-foreground"   47 seconds ago   Up 47 seconds   0.0.0.0:92->80/tcp, [::]:92->80/tcp   httpd_with_alway
s
7941a6b67865   httpd     "httpd-foreground"   2 minutes ago    Up 2 minutes    0.0.0.0:91->80/tcp, [::]:91->80/tcp   httpd_with_on_fa
ilure
8c5ec08d9d3e   httpd     "httpd-foreground"   3 minutes ago    Up 3 minutes    0.0.0.0:90->80/tcp, [::]:90->80/tcp   httpd_with_no_po
```

-- container with no restart policy as it not restarted after stopped.



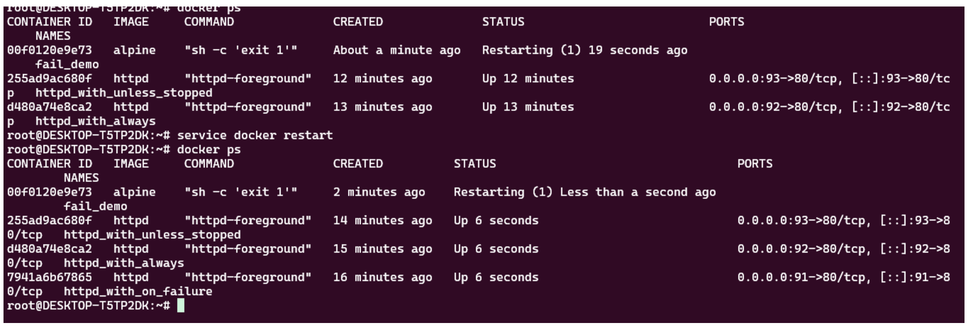
```
root@DESKTOP-T5TP2DK:~# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED          STATUS          PORTS                               NAMES
255ad9ac680f   httpd     "httpd-foreground"   4 seconds ago    Up 4 seconds    0.0.0.0:93->80/tcp, [::]:93->80/tcp   httpd_with_unles
s_stopped
d480a74e8ca2   httpd     "httpd-foreground"   47 seconds ago   Up 47 seconds   0.0.0.0:92->80/tcp, [::]:92->80/tcp   httpd_with_alway
s
7941a6b67865   httpd     "httpd-foreground"   2 minutes ago    Up 2 minutes    0.0.0.0:91->80/tcp, [::]:91->80/tcp   httpd_with_on_fa
ilure
8c5ec08d9d3e   httpd     "httpd-foreground"   3 minutes ago    Up 3 minutes    0.0.0.0:90->80/tcp, [::]:90->80/tcp   httpd_with_no_po
licy
root@DESKTOP-T5TP2DK:~# docker stop httpd_with_no_policy
httpd_with_no_policy
root@DESKTOP-T5TP2DK:~# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED         STATUS         PORTS                               NAMES
255ad9ac680f   httpd     "httpd-foreground"   3 minutes ago   Up 3 minutes   0.0.0.0:93->80/tcp, [::]:93->80/tcp   httpd_with_unless_
stopped
d480a74e8ca2   httpd     "httpd-foreground"   4 minutes ago   Up 4 minutes   0.0.0.0:92->80/tcp, [::]:92->80/tcp   httpd_with_always
7941a6b67865   httpd     "httpd-foreground"   5 minutes ago   Up 5 minutes   0.0.0.0:91->80/tcp, [::]:91->80/tcp   httpd_with_on_fail
ure
```

-- container with on-failure restart policy it will restart if crashed but not work with kill as it override by policy.



```
root@DESKTOP-T5TP2DK:~# docker run -d --name fail_demo --restart on-failure alpine sh -c "exit 1"
Unable to find image 'alpine:latest' locally
latest: Pulling from library/alpine
2d35ebdb57d9: Pull complete
Digest: sha256:4b7ce07002c69e8f3d704a9c5d6fd3053be500b7f1c69fc0d80990c2ad8dd412
Status: Downloaded newer image for alpine:latest
00f0120e9e7338c96c51e90c056f8bad2e8f593a2490f347d30c5c6a4e0a48b9
root@DESKTOP-T5TP2DK:~# docker ps -a | grep fail_demo
00f0120e9e73   alpine     "sh -c 'exit 1'"     34 seconds ago    Restarting (1) 6 seconds ago
   fail_demo
root@DESKTOP-T5TP2DK:~#
```

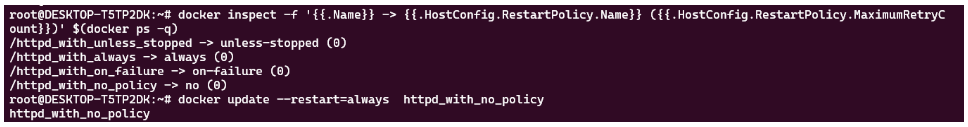
-- containers with restart policy always and unless stopped.



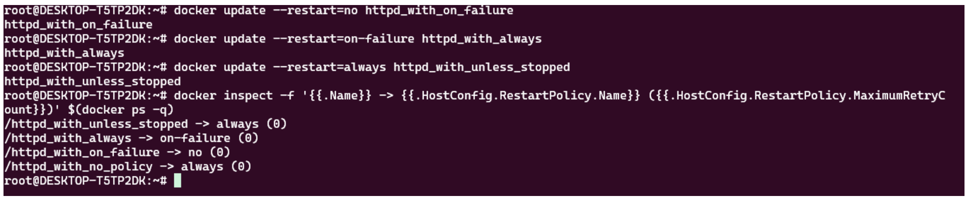
```
root@DESKTOP-T5TP2DK:~# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED              STATUS                         PORTS
    NAMES
00f0120e9e73   alpine    "sh -c 'exit 1'"     About a minute ago   Restarting (1) 19 seconds ago
    fail_demo
255ad9ac680f   httpd     "httpd-foreground"   12 minutes ago       Up 12 minutes                  0.0.0.0:93->80/tcp, [::]:93->80/tc
p   httpd_with_unless_stopped
d480a74e8ca2   httpd     "httpd-foreground"   13 minutes ago       Up 13 minutes                  0.0.0.0:92->80/tcp, [::]:92->80/tc
p   httpd_with_always
root@DESKTOP-T5TP2DK:~# service docker restart
root@DESKTOP-T5TP2DK:~# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED          STATUS                                     PORTS
        NAMES
00f0120e9e73   alpine    "sh -c 'exit 1'"     2 minutes ago    Restarting (1) Less than a second ago
        fail_demo
255ad9ac680f   httpd     "httpd-foreground"   14 minutes ago   Up 6 seconds                               0.0.0.0:93->80/tcp, [::]:93->8
0/tcp   httpd_with_unless_stopped
d480a74e8ca2   httpd     "httpd-foreground"   15 minutes ago   Up 6 seconds                               0.0.0.0:92->80/tcp, [::]:92->8
0/tcp   httpd_with_always
7941a6b67865   httpd     "httpd-foreground"   16 minutes ago   Up 6 seconds                               0.0.0.0:91->80/tcp, [::]:91->8
0/tcp   httpd_with_on_failure
root@DESKTOP-T5TP2DK:~#
```

Task 20. Change the restart policy of a above running container from the default to a custom policy using the docker update command.

--- inspected policy first, perform update on policies and verified by inspecting.



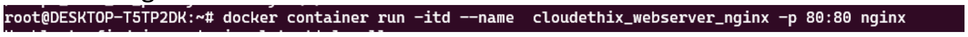
```
root@DESKTOP-T5TP2DK:~# docker inspect -f '{{.Name}} -> {{.HostConfig.RestartPolicy.Name}} ({{.HostConfig.RestartPolicy.MaximumRetryC
ount}})' $(docker ps -q)
/httpd_with_unless_stopped -> unless-stopped (0)
/httpd_with_always -> always (0)
/httpd_with_on_failure -> on-failure (0)
/httpd_with_no_policy -> no (0)
root@DESKTOP-T5TP2DK:~# docker update --restart=always  httpd_with_no_policy
httpd_with_no_policy
```



```
root@DESKTOP-T5TP2DK:~# docker update --restart=no httpd_with_on_failure
httpd_with_on_failure
root@DESKTOP-T5TP2DK:~# docker update --restart=on-failure httpd_with_always
httpd_with_always
root@DESKTOP-T5TP2DK:~# docker update --restart=always httpd_with_unless_stopped
httpd_with_unless_stopped
root@DESKTOP-T5TP2DK:~# docker inspect -f '{{.Name}} -> {{.HostConfig.RestartPolicy.Name}} ({{.HostConfig.RestartPolicy.MaximumRetryC
ount}})' $(docker ps -q)
/httpd_with_unless_stopped -> always (0)
/httpd_with_always -> on-failure (0)
/httpd_with_on_failure -> no (0)
/httpd_with_no_policy -> always (0)
root@DESKTOP-T5TP2DK:~#
```

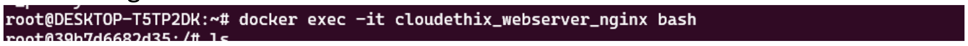
Task 21. Launch an NGINX container with a meaningful name and expose it on the host's port 80. Create an "index.html" file containing the text "Hello there, Let's be the Team CloudEthiX," and copy the file to the container's "/usr/share/nginx/html/" location. Access the container in a browser to verify that the webpage displays correctly.

-- created nginx container.



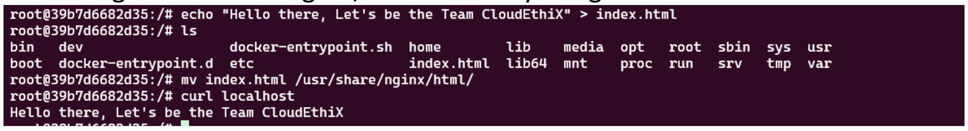
```
root@DESKTOP-T5TP2DK:~# docker container run -itd --name  cloudethix_webserver_nginx -p 80:80 nginx
```

-- accessing interactive terminal of container .



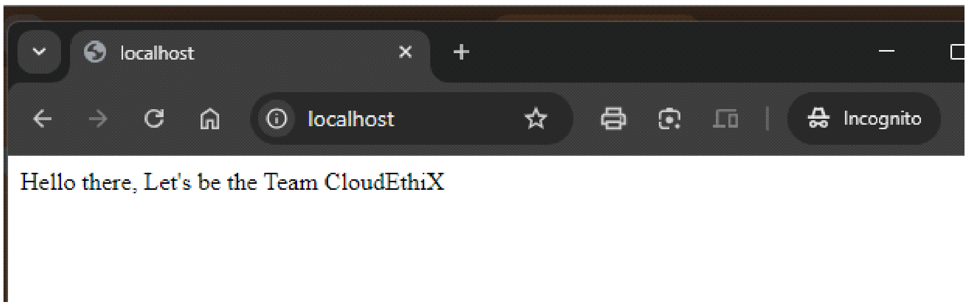
```
root@DESKTOP-T5TP2DK:~# docker exec -it cloudethix_webserver_nginx bash
```

-- creating file and moving to /html directory of nginx.



```
root@39b7d6682d35:/# echo "Hello there, Let's be the Team CloudEthiX" > index.html
root@39b7d6682d35:/# ls
bin   dev                  docker-entrypoint.sh  home        lib    media  opt   root  sbin  sys  usr
boot  docker-entrypoint.d  etc                   index.html  lib64  mnt    proc  run   srv   tmp  var
root@39b7d6682d35:/# mv index.html /usr/share/nginx/html/
root@39b7d6682d35:/# curl localhost
Hello there, Let's be the Team CloudEthiX
```

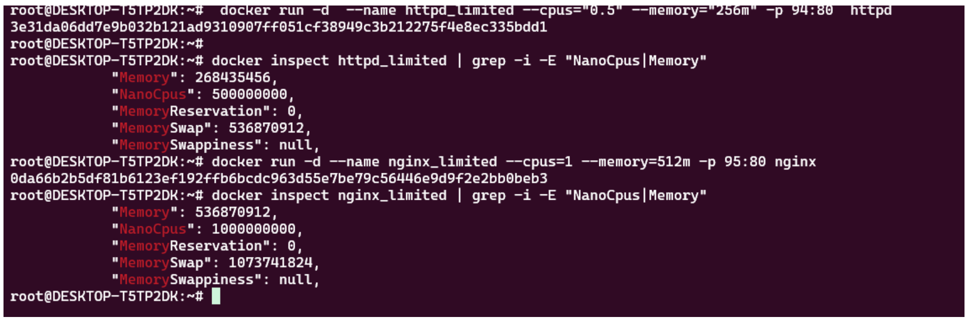
-- accused web page from Browser.



Hello there, Let's be the Team CloudEthiX

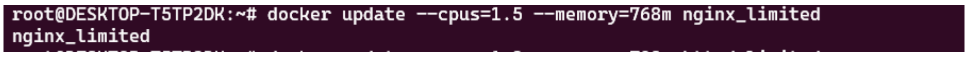
Task 22. Run a docker container with CPU and Memory limit.

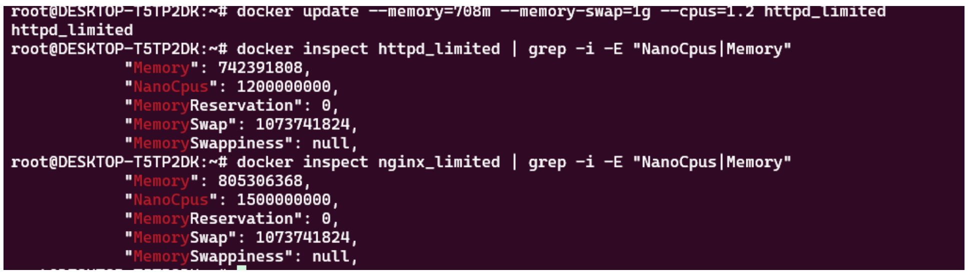
-- nginx and httpd conatiner with memory and cpu limits.


```
root@DESKTOP-T5TP2DK:~#  docker run -d  --name httpd_limited --cpus="0.5" --memory="256m" -p 94:80  httpd
3e31da06dd7e9b032b121ad9310907ff051cf38949c3b212275f4e8ec335bdd1
root@DESKTOP-T5TP2DK:~#
root@DESKTOP-T5TP2DK:~# docker inspect httpd_limited | grep -i -E "NanoCpus|Memory"
            "Memory": 268435456,
            "NanoCpus": 500000000,
            "MemoryReservation": 0,
            "MemorySwap": 536870912,
            "MemorySwappiness": null,
root@DESKTOP-T5TP2DK:~# docker run -d --name nginx_limited --cpus=1 --memory=512m -p 95:80 nginx
0da66b2b5df81b6123ef192ffb6bcdc963d55e7be79c56446e9d9f2e2bb0beb3
root@DESKTOP-T5TP2DK:~# docker inspect nginx_limited | grep -i -E "NanoCpus|Memory"
            "Memory": 536870912,
            "NanoCpus": 1000000000,
            "MemoryReservation": 0,
            "MemorySwap": 1073741824,
            "MemorySwappiness": null,
root@DESKTOP-T5TP2DK:~#
```

Task 23. Update CUP and Memory of docker container using docker update.

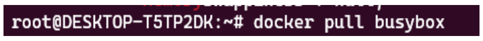
-- updated memory and cpu limits of above two containers.

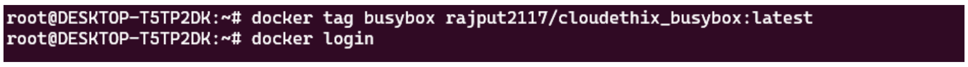

```
root@DESKTOP-T5TP2DK:~# docker update --cpus=1.5 --memory=768m nginx_limited
nginx_limited
```

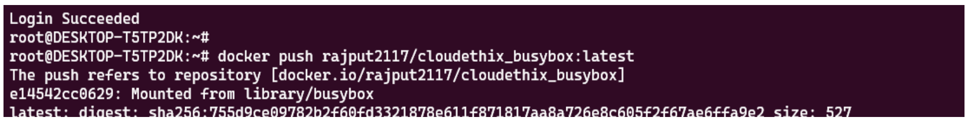

```
root@DESKTOP-T5TP2DK:~# docker update --memory=708m --memory-swap=1g --cpus=1.2 httpd_limited
httpd_limited
root@DESKTOP-T5TP2DK:~# docker inspect httpd_limited | grep -i -E "NanoCpus|Memory"
            "Memory": 742391808,
            "NanoCpus": 1200000000,
            "MemoryReservation": 0,
            "MemorySwap": 1073741824,
            "MemorySwappiness": null,
root@DESKTOP-T5TP2DK:~# docker inspect nginx_limited | grep -i -E "NanoCpus|Memory"
            "Memory": 805306368,
            "NanoCpus": 1500000000,
            "MemoryReservation": 0,
            "MemorySwap": 1073741824,
            "MemorySwappiness": null,
```

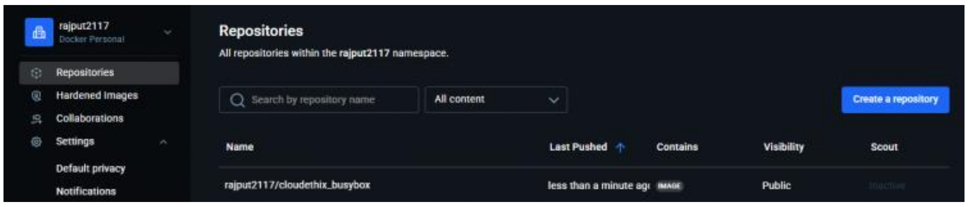
Task 24. Pull the Busy-box image to your local system, tag it, and push it to the Docker Hub repository "yourname_cloudethix_busybox."

--- pulled docker image, tagged and pushed to dockerhub.


```
root@DESKTOP-T5TP2DK:~# docker pull busybox
```


```
root@DESKTOP-T5TP2DK:~# docker tag busybox rajput2117/cloudethix_busybox:latest
root@DESKTOP-T5TP2DK:~# docker login
```


```
Login Succeeded
root@DESKTOP-T5TP2DK:~#
root@DESKTOP-T5TP2DK:~# docker push rajput2117/cloudethix_busybox:latest
The push refers to repository [docker.io/rajput2117/cloudethix_busybox]
e14542cc0629: Mounted from library/busybox
latest: digest: sha256:755d9ce09782b2f60fd3321878e611f871817aa8a726e8c605f2f67ae6ffa9e2 size: 527
```



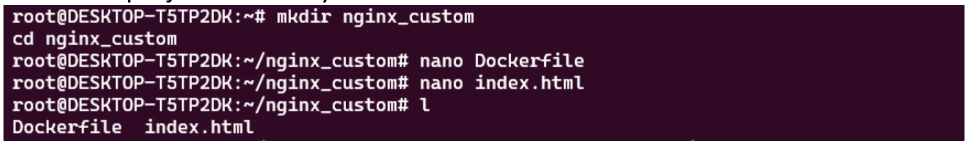
Repositories

All repositories within the rajput2117 namespace.

Name	Last Pushed	Contains	Visibility	Scout
rajput2117/cloudethix_busybox	less than a minute ag	IMAGE	Public	Inactive

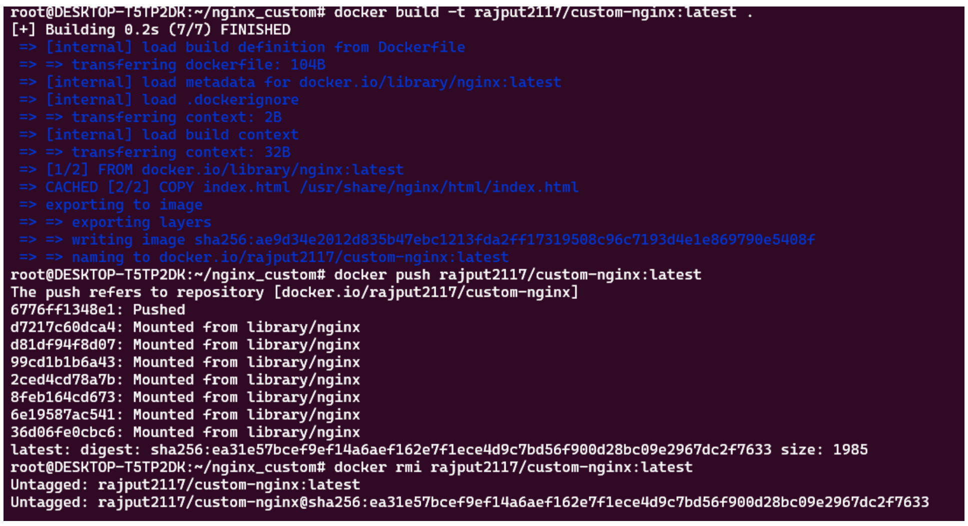
Task 25.
In your project directory, create a Dockerfile (named Dockerfile) with the following content.
FROM nginx:latest
COPY custom-index.html /usr/share/nginx/html/index.html
Ensure you also have a file named custom-index.html in the same directory. Build the Docker image using the Dockerfile you created and push it your repository. Delete the local image. Start a new container using the custom Nginx image that you just pushed. Map port 8080 on your host to port 80 in the container. Check the container page in browser.

-- created project directory and files inside it.



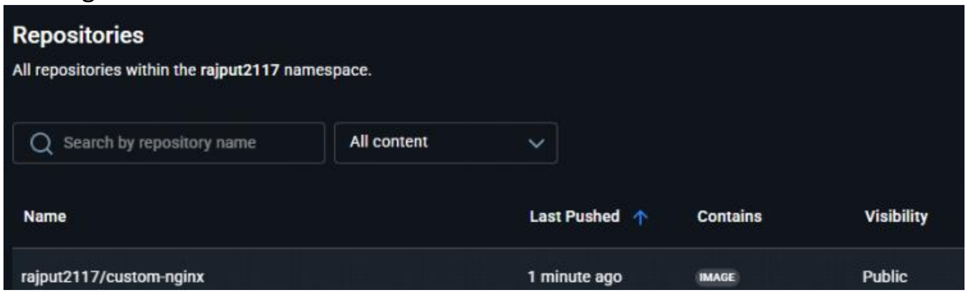
```
root@DESKTOP-T5TP2DK:~# mkdir nginx_custom
cd nginx_custom
root@DESKTOP-T5TP2DK:~/nginx_custom# nano Dockerfile
root@DESKTOP-T5TP2DK:~/nginx_custom# nano index.html
root@DESKTOP-T5TP2DK:~/nginx_custom# l
Dockerfile  index.html
```

--- build image push to Dockerhub and removed it from local .



```
root@DESKTOP-T5TP2DK:~/nginx_custom# docker build -t rajput2117/custom-nginx:latest .
[+] Building 0.2s (7/7) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 104B
 => [internal] load metadata for docker.io/library/nginx:latest
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load build context
 => => transferring context: 32B
 => [1/2] FROM docker.io/library/nginx:latest
 => CACHED [2/2] COPY index.html /usr/share/nginx/html/index.html
 => exporting to image
 => => exporting layers
 => => writing image sha256:ae9d34e2012d835b47ebc1213fda2ff17319508c96c7193d4e1e869790e5408f
 => => naming to docker.io/rajput2117/custom-nginx:latest
root@DESKTOP-T5TP2DK:~/nginx_custom# docker push rajput2117/custom-nginx:latest
The push refers to repository [docker.io/rajput2117/custom-nginx]
6776ff1348e1: Pushed
d7217c60dca4: Mounted from library/nginx
d81df94f8d07: Mounted from library/nginx
99cd1b1b6a43: Mounted from library/nginx
2ced4cd78a7b: Mounted from library/nginx
8feb164cd673: Mounted from library/nginx
6e19587ac541: Mounted from library/nginx
36d06fe0cbc6: Mounted from library/nginx
latest: digest: sha256:ea31e57bcef9ef14a6aef162e7f1ece4d9c7bd56f900d28bc09e2967dc2f7633 size: 1985
root@DESKTOP-T5TP2DK:~/nginx_custom# docker rmi rajput2117/custom-nginx:latest
Untagged: rajput2117/custom-nginx:latest
Untagged: rajput2117/custom-nginx@sha256:ea31e57bcef9ef14a6aef162e7f1ece4d9c7bd56f900d28bc09e2967dc2f7633
```

--- image on Dockerhub .

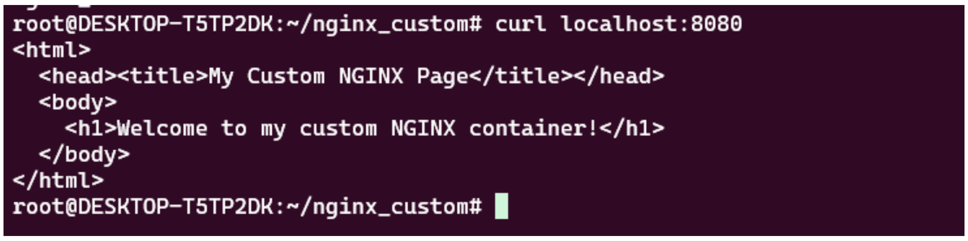


Repositories

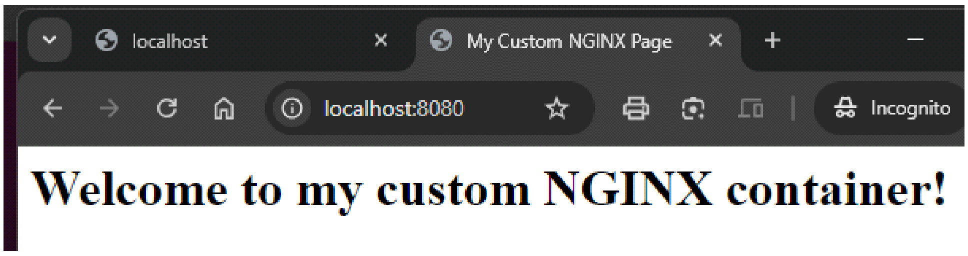
All repositories within the rajput2117 namespace.

Name	Last Pushed	Contains	Visibility
rajput2117/custom-nginx	1 minute ago	IMAGE	Public

-- accessed web page on localhost.



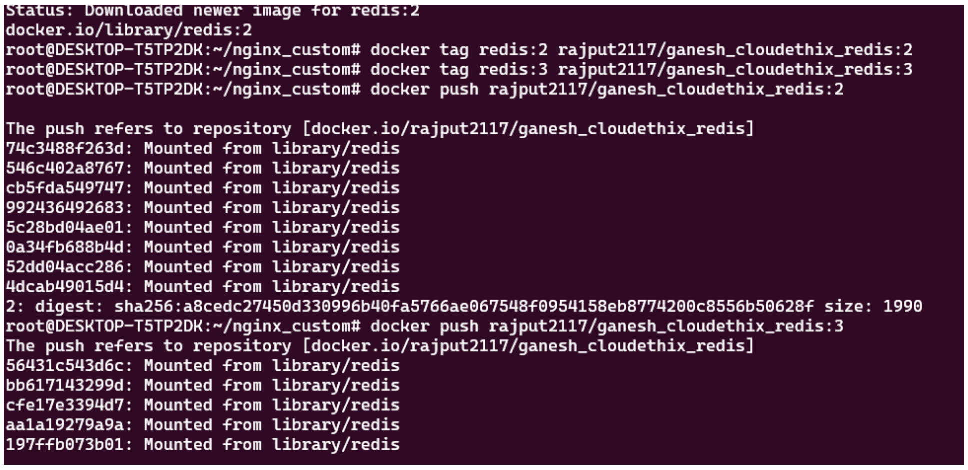
```
root@DESKTOP-T5TP2DK:~/nginx_custom# curl localhost:8080
<html>
  <head><title>My Custom NGINX Page</title></head>
  <body>
    <h1>Welcome to my custom NGINX container!</h1>
  </body>
</html>
root@DESKTOP-T5TP2DK:~/nginx_custom#
```



Welcome to my custom NGINX container!

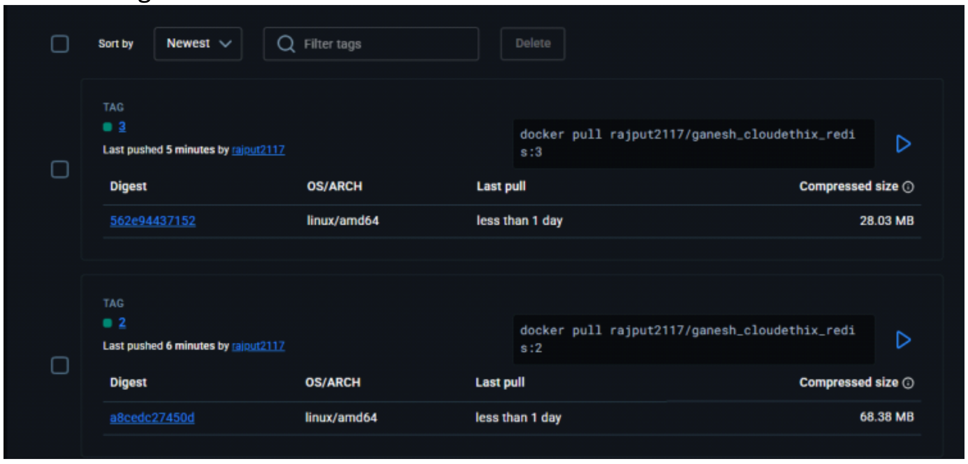
Task 26. Push the Redis images tagged as version 1 and 3 to your Docker Hub repository, named "yourname_cloudethix_redis."

-- pulled redis images of redis:2 and 3 ,Tagged them, pushed to Dockerhub .



```
Status: Downloaded newer image for redis:2
docker.io/library/redis:2
root@DESKTOP-T5TP2DK:~/nginx_custom# docker tag redis:2 rajput2117/ganesh_cloudethix_redis:2
root@DESKTOP-T5TP2DK:~/nginx_custom# docker tag redis:3 rajput2117/ganesh_cloudethix_redis:3
root@DESKTOP-T5TP2DK:~/nginx_custom# docker push rajput2117/ganesh_cloudethix_redis:2

The push refers to repository [docker.io/rajput2117/ganesh_cloudethix_redis]
74c3488f263d: Mounted from library/redis
546c402a8767: Mounted from library/redis
cb5fda549747: Mounted from library/redis
992436492683: Mounted from library/redis
5c28bd04ae01: Mounted from library/redis
0a34fb688b4d: Mounted from library/redis
52dd04acc286: Mounted from library/redis
4dcab49015d4: Mounted from library/redis
2: digest: sha256:a8cedc27450d330996b40fa5766ae067548f0954158eb8774200c8556b50628f size: 1990
root@DESKTOP-T5TP2DK:~/nginx_custom# docker push rajput2117/ganesh_cloudethix_redis:3
The push refers to repository [docker.io/rajput2117/ganesh_cloudethix_redis]
56431c543d6c: Mounted from library/redis
bb617143299d: Mounted from library/redis
cfe17e3394d7: Mounted from library/redis
aa1a19279a9a: Mounted from library/redis
197ffb073b01: Mounted from library/redis
```

-- redis image version 2 and 3 on Dockerhub.



TAG
3
Last pushed 5 minutes by rajput2117

```
docker pull rajput2117/ganesh_cloudethix_redis:3
```

Digest	OS/ARCH	Last pull	Compressed size
562e94437152	linux/amd64	less than 1 day	28.03 MB

TAG
2
Last pushed 6 minutes by rajput2117

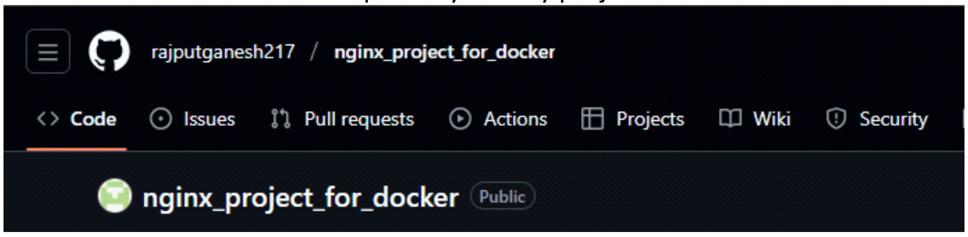
```
docker pull rajput2117/ganesh_cloudethix_redis:2
```

Digest	OS/ARCH	Last pull	Compressed size
a8cedc27450d	linux/amd64	less than 1 day	68.38 MB

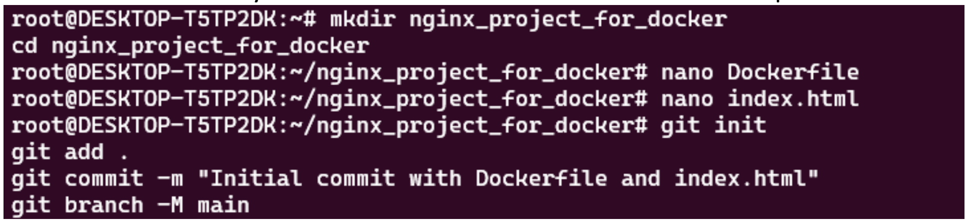
Task 27. Create a remote Git repository and add the Dockerfile and index.html files. Clone the repository locally and create a release branch. Build the Docker image from the release branch with a meaningful tag, then run a container from the image and expose it on host port 8383. Check the webpage in a browser, and upon success, push the image to your Docker Hub repository named "yourname_cloudethix_nginx." Finally, push the release branch to the remote Git repository and

merge it by creating a pull request.

--- created a remote Github repository for my project.

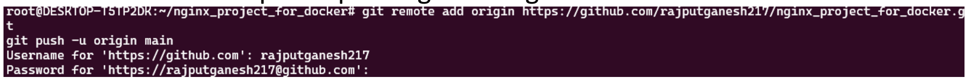


--- created a directory and files inside it and commited them for push.



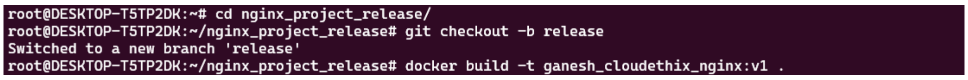
```
root@DESKTOP-T5TP2DK:~# mkdir nginx_project_for_docker
cd nginx_project_for_docker
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# nano Dockerfile
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# nano index.html
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# git init
git add .
git commit -m "Initial commit with Dockerfile and index.html"
git branch -M main
```

--- added a remote repo for pushing files to github.



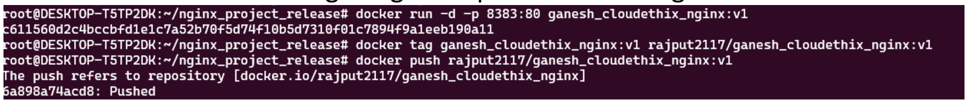
```
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# git remote add origin https://github.com/rajputganesh217/nginx_project_for_docker.g
t
git push -u origin main
Username for 'https://github.com': rajputganesh217
Password for 'https://rajputganesh217@github.com':
```

--- after pushing files cloned my github project to another directory nginx_project_release
And build a Dockerfile their.



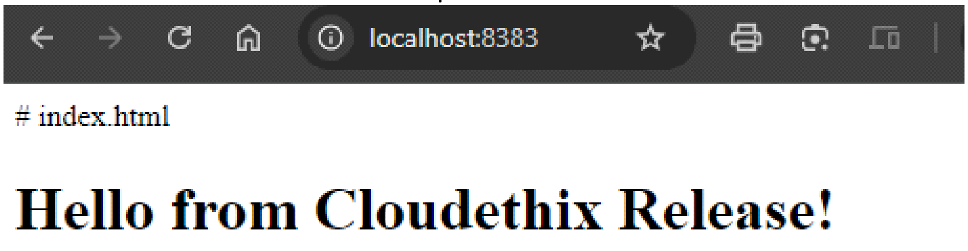
```
root@DESKTOP-T5TP2DK:~# cd nginx_project_release/
root@DESKTOP-T5TP2DK:~/nginx_project_release# git checkout -b release
Switched to a new branch 'release'
root@DESKTOP-T5TP2DK:~/nginx_project_release# docker build -t ganesh_cloudethix_nginx:v1 .
```

--- created a container using image and pushed that image to dockerhub.



```
root@DESKTOP-T5TP2DK:~/nginx_project_release# docker run -d -p 8383:80 ganesh_cloudethix_nginx:v1
c611560d2c4bccbfd1e1c7a52b70f5d74f10b5d7310f01c7894f9a1eeb190a11
root@DESKTOP-T5TP2DK:~/nginx_project_release# docker tag ganesh_cloudethix_nginx:v1 rajput2117/ganesh_cloudethix_nginx:v1
root@DESKTOP-T5TP2DK:~/nginx_project_release# docker push rajput2117/ganesh_cloudethix_nginx:v1
The push refers to repository [docker.io/rajput2117/ganesh_cloudethix_nginx]
6a898a74acd8: Pushed
```

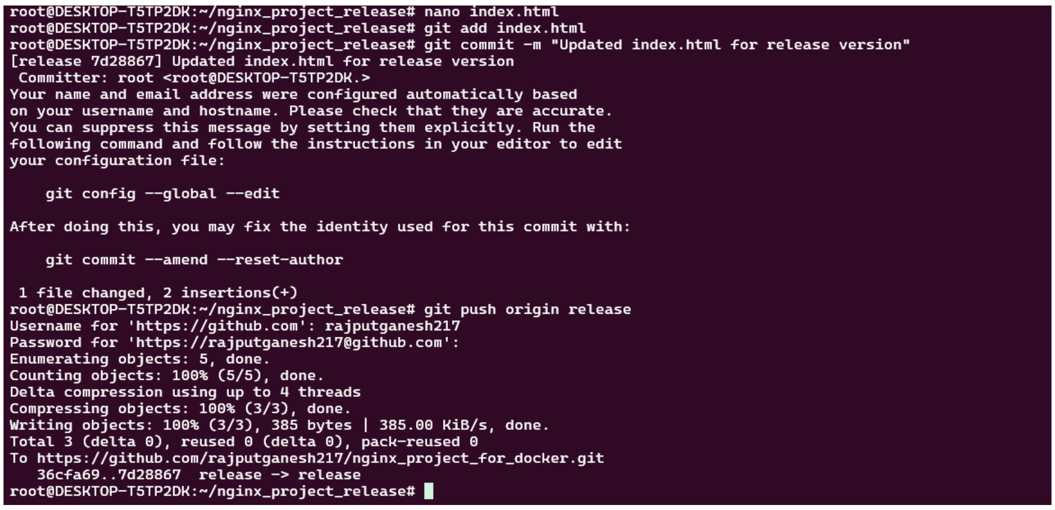
-- accessed container to Browser on port 8383



index.html

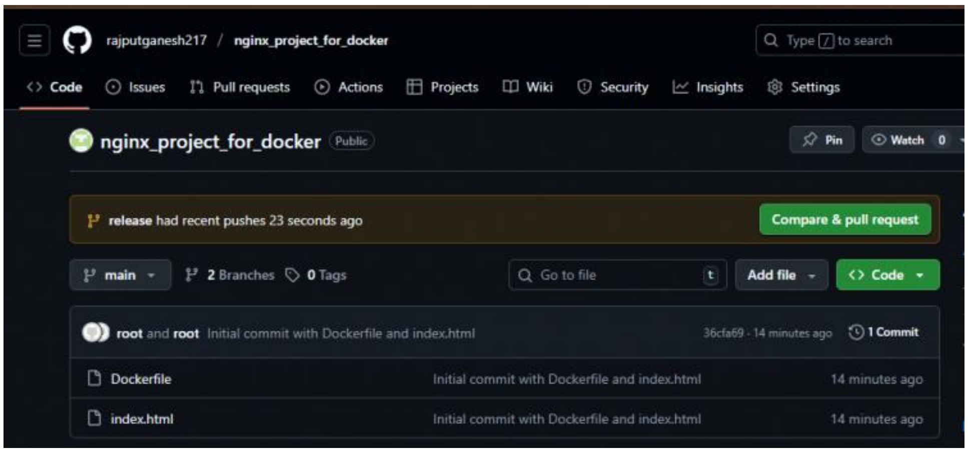
Hello from Cloudethix Release!

--- maked changes to index file present in nginx_project_release and peshed to github so pull request get generated.



```
root@DESKTOP-T5TP2DK:~/nginx_project_release# nano index.html
root@DESKTOP-T5TP2DK:~/nginx_project_release# git add index.html
root@DESKTOP-T5TP2DK:~/nginx_project_release# git commit -m "Updated index.html for release version"
[release 7d28867] Updated index.html for release version
 Committer: root <root@DESKTOP-T5TP2DK.>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 2 insertions(+)
root@DESKTOP-T5TP2DK:~/nginx_project_release# git push origin release
Username for 'https://github.com': rajputganesh217
Password for 'https://rajputganesh217@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 385 bytes | 385.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/rajputganesh217/nginx_project_for_docker.git
   36cfa69..7d28867  release -> release
root@DESKTOP-T5TP2DK:~/nginx_project_release# 
```

-- pull request generated .



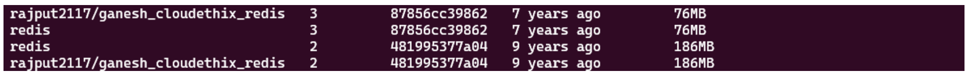
release had recent pushes 23 seconds ago

root and root Initial commit with Dockerfile and index.html 36cfa69 · 14 minutes ago

Dockerfile	Initial commit with Dockerfile and index.html	14 minutes ago
index.html	Initial commit with Dockerfile and index.html	14 minutes ago

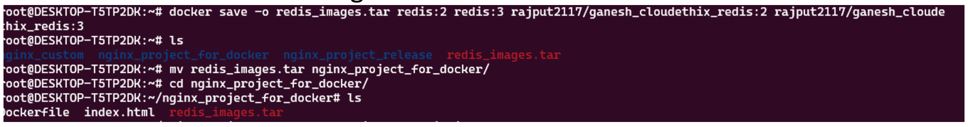
Task 28. Save all local Redis images as a .tar file in the master branch of your local repository. Delete all Redis images from your local system and push the master branch to the remote repository. Load the Redis images from the tar file to your local system, and verify that all images have been loaded correctly.

---- list the redis images



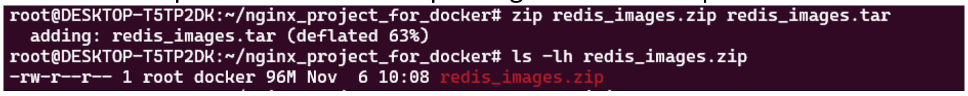
```
rajput2117/ganesh_cloudethix_redis   3          87856cc39862   7 years ago          76MB
redis                                3          87856cc39862   7 years ago          76MB
redis                                2          481995377a04   9 years ago          186MB
rajput2117/ganesh_cloudethix_redis   2          481995377a04   9 years ago          186MB
```

---- create a .tar file from images.

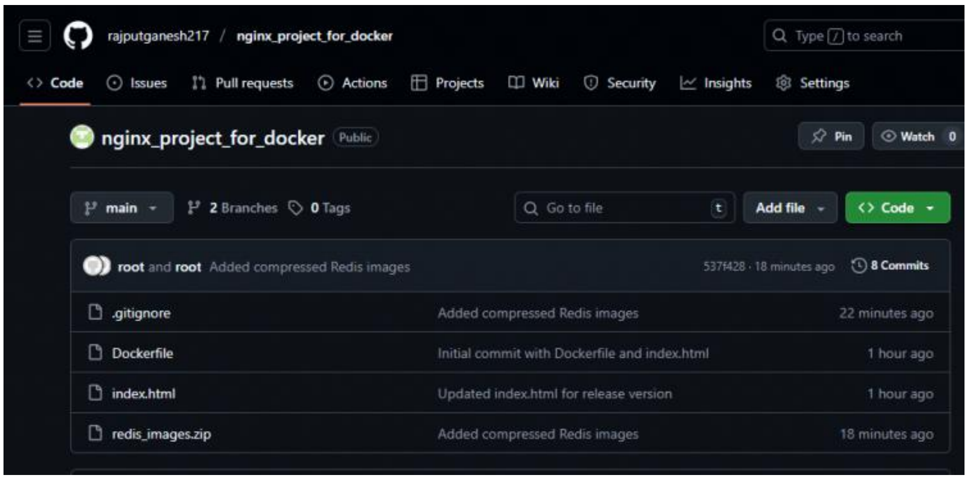


```
oot@DESKTOP-T5TP2DK:~# docker save -o redis_images.tar redis:2 redis:3 rajput2117/ganesh_cloudethix_redis:2 rajput2117/ganesh_cloude
hix_redis:3
oot@DESKTOP-T5TP2DK:~# ls
ginx_custom  nginx_project_for_docker  nginx_project_release  redis_images.tar
oot@DESKTOP-T5TP2DK:~# mv redis_images.tar nginx_project_for_docker/
oot@DESKTOP-T5TP2DK:~# cd nginx_project_for_docker/
oot@DESKTOP-T5TP2DK:~/nginx_project_for_docker# ls
ockerfile  index.html  redis_images.tar
```

--- created a zip file from tar file for pushing to Github and pushed.

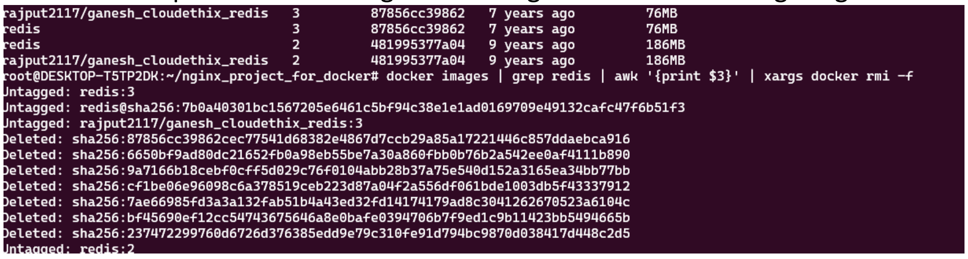


```
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# zip redis_images.zip redis_images.tar
  adding: redis_images.tar (deflated 63%)
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# ls -lh redis_images.zip
-rw-r--r-- 1 root docker 96M Nov  6 10:08 redis_images.zip
```

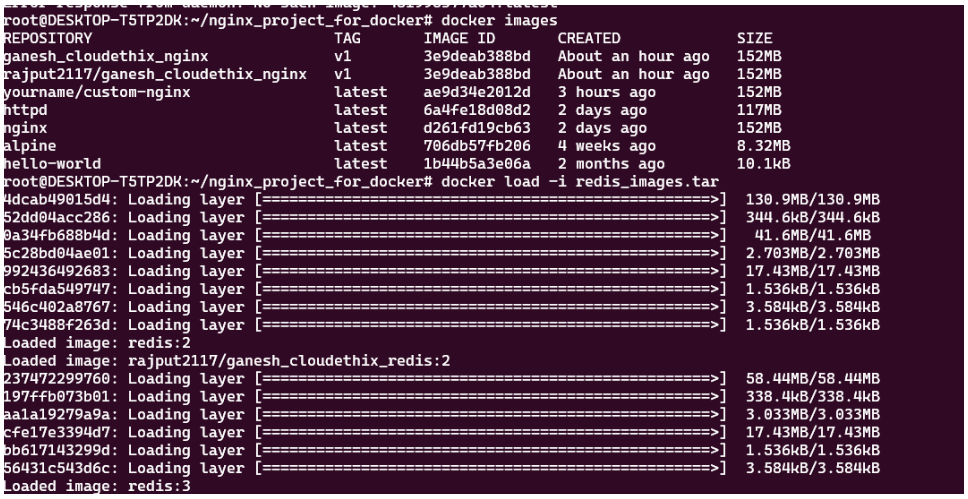


```
root and root  Added compressed Redis images              537f428 · 18 minutes ago
.gitignore        Added compressed Redis images                        22 minutes ago
Dockerfile        Initial commit with Dockerfile and index.html        1 hour ago
index.html        Updated index.html for release version               1 hour ago
redis_images.zip  Added compressed Redis images                        18 minutes ago
```

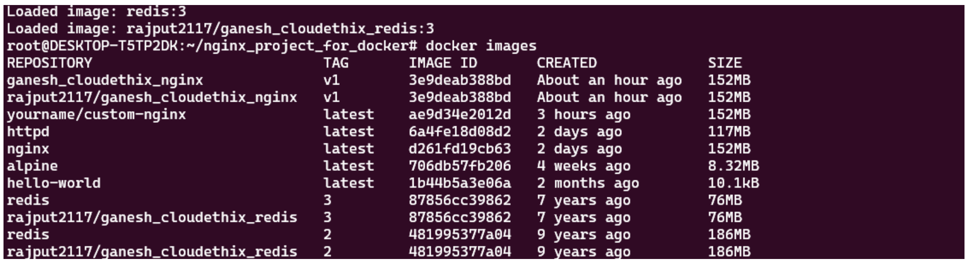
-- deleted the present redis images and using tar file created images again.



```
rajput2117/ganesh_cloudethix_redis   3         87856cc39862   7 years ago         76MB
redis                                3         87856cc39862   7 years ago         76MB
redis                                2         481995377a04   9 years ago         186MB
rajput2117/ganesh_cloudethix_redis   2         481995377a04   9 years ago         186MB
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# docker images | grep redis | awk '{print $3}' | xargs docker rmi -f
Untagged: redis:3
Untagged: redis@sha256:7b0a40301bc1567205e6461c5bf94c38e1e1ad0169709e49132cafc47f6b51f3
Untagged: rajput2117/ganesh_cloudethix_redis:3
Deleted: sha256:87856cc39862cec77541d68382e4867d7ccb29a85a17221446c857ddaebca916
Deleted: sha256:6650bf9ad80dc21652fb0a98eb55be7a30a860fbb0b76b2a542ee0af4111b890
Deleted: sha256:9a7166b18cebf0cff5d029c76f0104abb28b37a75e540d152a3165ea34bb77bb
Deleted: sha256:cf1be06e96098c6a378519ceb223d87a04f2a556df061bde1003db5f43337912
Deleted: sha256:7ae66985fd3a3a132fab51b4a43ed32fd14174179ad8c3041262670523a6104c
Deleted: sha256:bf45690ef12cc54743675646a8e0bafe0394706b7f9ed1c9b11423bb5494665b
Deleted: sha256:237472299760d6726d376385edd9e79c310fe91d794bc9870d038417d448c2d5
Untagged: redis:2
```



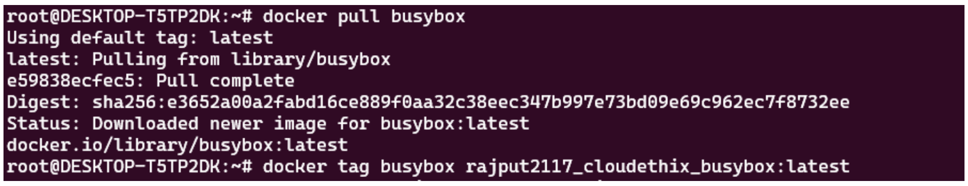
```
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# docker images
REPOSITORY                           TAG       IMAGE ID       CREATED              SIZE
ganesh_cloudethix_nginx              v1        3e9deab388bd   About an hour ago    152MB
rajput2117/ganesh_cloudethix_nginx   v1        3e9deab388bd   About an hour ago    152MB
yourname/custom-nginx                latest    ae9d34e2012d   3 hours ago          152MB
httpd                                latest    6a4fe18d08d2   2 days ago           117MB
nginx                                latest    d261fd19cb63   2 days ago           152MB
alpine                               latest    706db57fb206   4 weeks ago          8.32MB
hello-world                          latest    1b44b5a3e06a   2 months ago         10.1kB
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# docker load -i redis_images.tar
4dcab49015d4: Loading layer [==================================================>]  130.9MB/130.9MB
52dd04acc286: Loading layer [==================================================>]  344.6kB/344.6kB
0a34fb688b4d: Loading layer [==================================================>]   41.6MB/41.6MB
5c28bd04ae01: Loading layer [==================================================>]  2.703MB/2.703MB
992436492683: Loading layer [==================================================>]  17.43MB/17.43MB
cb5fda549747: Loading layer [==================================================>]  1.536kB/1.536kB
546c402a8767: Loading layer [==================================================>]  3.584kB/3.584kB
74c3488f263d: Loading layer [==================================================>]  1.536kB/1.536kB
Loaded image: redis:2
Loaded image: rajput2117/ganesh_cloudethix_redis:2
237472299760: Loading layer [==================================================>]  58.44MB/58.44MB
197ffb073b01: Loading layer [==================================================>]  338.4kB/338.4kB
aa1a19279a9a: Loading layer [==================================================>]  3.033MB/3.033MB
cfe17e3394d7: Loading layer [==================================================>]  17.43MB/17.43MB
bb617143299d: Loading layer [==================================================>]  1.536kB/1.536kB
56431c543d6c: Loading layer [==================================================>]  3.584kB/3.584kB
Loaded image: redis:3
```



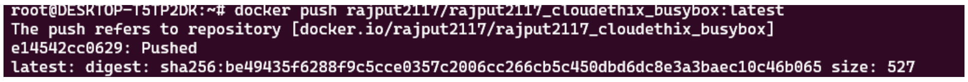
```
Loaded image: redis:3
Loaded image: rajput2117/ganesh_cloudethix_redis:3
root@DESKTOP-T5TP2DK:~/nginx_project_for_docker# docker images
REPOSITORY                           TAG       IMAGE ID       CREATED             SIZE
ganesh_cloudethix_nginx              v1        3e9deab388bd   About an hour ago   152MB
rajput2117/ganesh_cloudethix_nginx   v1        3e9deab388bd   About an hour ago   152MB
yourname/custom-nginx                latest    ae9d34e2012d   3 hours ago         152MB
httpd                                latest    6a4fe18d08d2   2 days ago          117MB
nginx                                latest    d261fd19cb63   2 days ago          152MB
alpine                               latest    706db57fb206   4 weeks ago         8.32MB
hello-world                          latest    1b44b5a3e06a   2 months ago        10.1kB
redis                                3         87856cc39862   7 years ago         76MB
rajput2117/ganesh_cloudethix_redis   3         87856cc39862   7 years ago         76MB
redis                                2         481995377a04   9 years ago         186MB
rajput2117/ganesh_cloudethix_redis   2         481995377a04   9 years ago         186MB
```

Task 29. Pull the Busy-box image to your local system, tag it, and push it to the Docker Hub repository "yourname_cloudethix_busybox." Export the Docker image from the NGINX container, create a .tar file, and import the tar file to create a Docker image with a meaningful name. After importing the image, tag it and push it to the "yourname_cloudethix_busybox" Docker Hub repository

--pulled a busybox image tagged it and pushed to dockerhub.

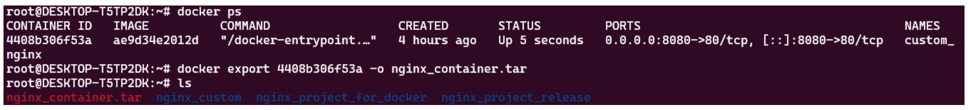


```
root@DESKTOP-T5TP2DK:~# docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
e59838ecfec5: Pull complete
Digest: sha256:e3652a00a2fabd16ce889f0aa32c38eec347b997e73bd09e69c962ec7f8732ee
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
root@DESKTOP-T5TP2DK:~# docker tag busybox rajput2117_cloudethix_busybox:latest
```

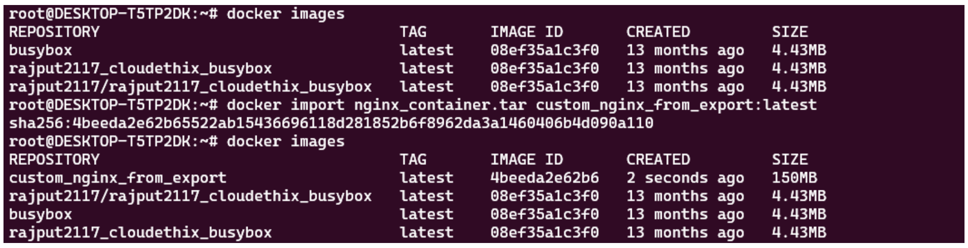


```
root@DESKTOP-T5TP2DK:~# docker push rajput2117/rajput2117_cloudethix_busybox:latest
The push refers to repository [docker.io/rajput2117/rajput2117_cloudethix_busybox]
e14542cc0629: Pushed
latest: digest: sha256:be49435f6288f9c5cce0357c2006cc266cb5c450dbd6dc8e3a3baec10c46b065 size: 527
```

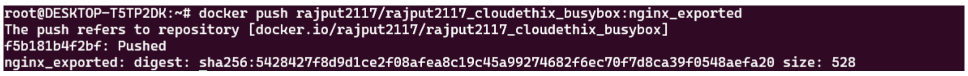
--- created a tar file from container , using tar file created an image and tagged, pushed to dockerhub.



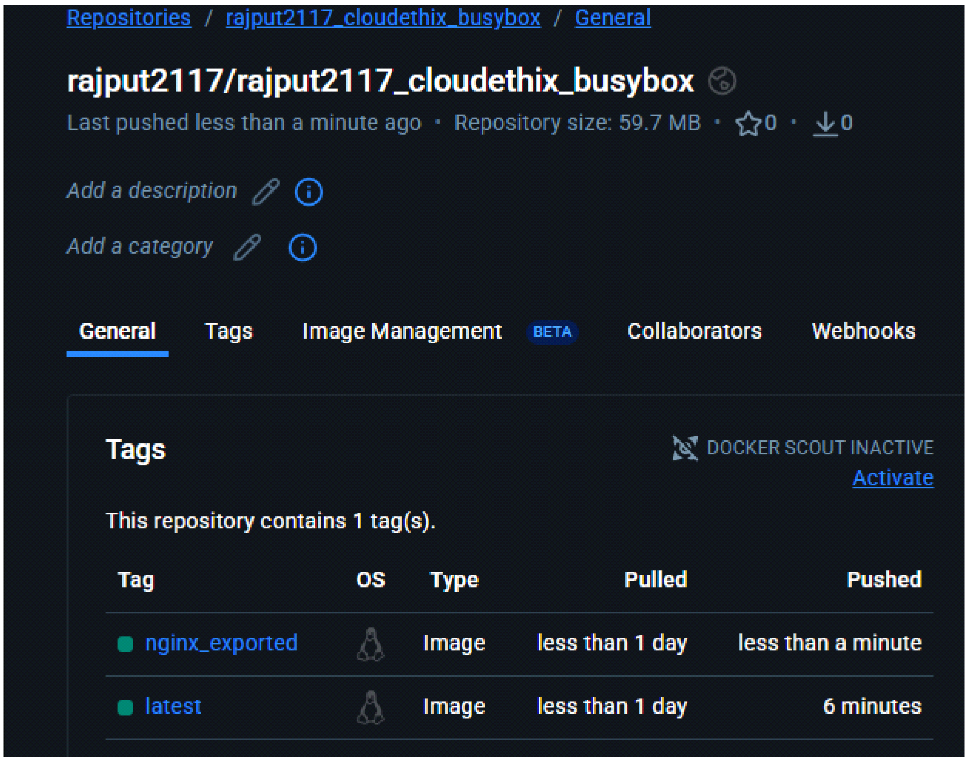
```
root@DESKTOP-T5TP2DK:~# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED       STATUS         PORTS                                   NAMES
4408b306f53a   ae9d34e2012d   "/docker-entrypoint.…"   4 hours ago   Up 5 seconds   0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   custom_
nginx
root@DESKTOP-T5TP2DK:~# docker export 4408b306f53a -o nginx_container.tar
root@DESKTOP-T5TP2DK:~# ls
nginx_container.tar  nginx_custom  nginx_project_for_docker  nginx_project_release
```



```
root@DESKTOP-T5TP2DK:~# docker images
REPOSITORY                                TAG       IMAGE ID       CREATED         SIZE
busybox                                   latest    08ef35a1c3f0   13 months ago   4.43MB
rajput2117_cloudethix_busybox             latest    08ef35a1c3f0   13 months ago   4.43MB
rajput2117/rajput2117_cloudethix_busybox  latest    08ef35a1c3f0   13 months ago   4.43MB
root@DESKTOP-T5TP2DK:~# docker import nginx_container.tar custom_nginx_from_export:latest
sha256:4beeda2e62b65522ab15436696118d281852b6f8962da3a1460406b4d090a110
root@DESKTOP-T5TP2DK:~# docker images
REPOSITORY                                TAG       IMAGE ID       CREATED         SIZE
custom_nginx_from_export                  latest    4beeda2e62b6   2 seconds ago   150MB
rajput2117/rajput2117_cloudethix_busybox  latest    08ef35a1c3f0   13 months ago   4.43MB
busybox                                   latest    08ef35a1c3f0   13 months ago   4.43MB
rajput2117_cloudethix_busybox             latest    08ef35a1c3f0   13 months ago   4.43MB
```



```
root@DESKTOP-T5TP2DK:~# docker push rajput2117/rajput2117_cloudethix_busybox:nginx_exported
The push refers to repository [docker.io/rajput2117/rajput2117_cloudethix_busybox]
f5b181b4f2bf: Pushed
nginx_exported: digest: sha256:5428427f8d9d1ce2f08afea8c19c45a99274682f6ec70f7d8ca39f0548aefa20 size: 528
```



Repositories / rajput2117_cloudethix_busybox / General

rajput2117/rajput2117_cloudethix_busybox

Last pushed less than a minute ago • Repository size: 59.7 MB • ☆0 • ↓0

Tags

DOCKER SCOUT INACTIVE Activate

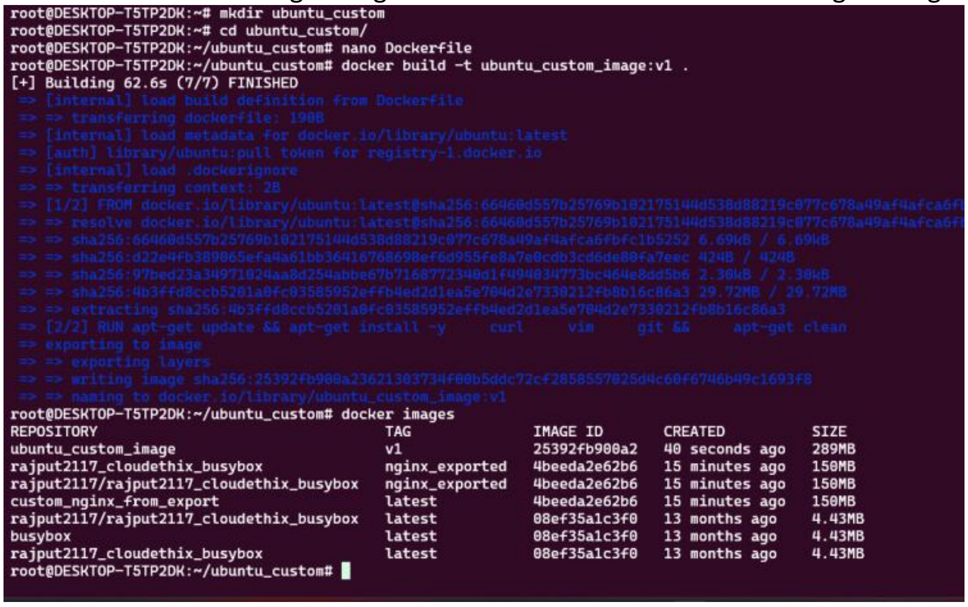
This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
nginx_exported		Image	less than 1 day	less than a minute
latest		Image	less than 1 day	6 minutes

Task 30. Dockerfile creation: Create a simple Dockerfile to build a custom image with the following

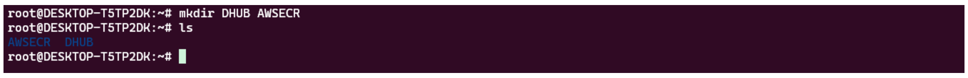
specifications:
Base image: Ubuntu
Install packages: curl, vim, and git
Set an environment variable: MY_NAME=Your_Name

--- Build the custom image using docker build and list all available images using docker images.



```
root@DESKTOP-T5TP2DK:~# mkdir ubuntu_custom
root@DESKTOP-T5TP2DK:~# cd ubuntu_custom/
root@DESKTOP-T5TP2DK:~/ubuntu_custom# nano Dockerfile
root@DESKTOP-T5TP2DK:~/ubuntu_custom# docker build -t ubuntu_custom_image:v1 .
[+] Building 62.6s (7/7) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 198B
 => [internal] load metadata for docker.io/library/ubuntu:latest
 => [auth] library/ubuntu:pull token for registry-1.docker.io
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/2] FROM docker.io/library/ubuntu:latest@sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6f
 => => resolve docker.io/library/ubuntu:latest@sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6f
 => => sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6fbfc1b5252 6.69kB / 6.69kB
 => => sha256:d22e4fb389065efa4a61bb36416768698ef6d955fe8a7e0cdb3cd6de80fa7eec 424B / 424B
 => => sha256:97bed23a34971024aa8d254abbe67b7168772340d1f494034773bc464e8dd5b6 2.30kB / 2.30kB
 => => sha256:4b3ffd8ccb5201a0fc03585952effb4ed2d1ea5e704d2e7330212fb8b16c86a3 29.72MB / 29.72MB
 => => extracting sha256:4b3ffd8ccb5201a0fc03585952effb4ed2d1ea5e704d2e7330212fb8b16c86a3
 => [2/2] RUN apt-get update && apt-get install -y     curl     vim     git &&     apt-get clean
 => exporting to image
 => => exporting layers
 => => writing image sha256:25392fb900a23621303734f00b5ddc72cf2858557025d4c60f6746b49c1693f8
 => => naming to docker.io/library/ubuntu_custom_image:v1
root@DESKTOP-T5TP2DK:~/ubuntu_custom# docker images
REPOSITORY                             TAG               IMAGE ID       CREATED          SIZE
ubuntu_custom_image                    v1                25392fb900a2   40 seconds ago   289MB
rajput2117_cloudethix_busybox          nginx_exported    4beeda2e62b6   15 minutes ago   150MB
rajput2117/rajput2117_cloudethix_busybox   nginx_exported    4beeda2e62b6   15 minutes ago   150MB
custom_nginx_from_export               latest            4beeda2e62b6   15 minutes ago   150MB
rajput2117/rajput2117_cloudethix_busybox   latest            08ef35a1c3f0   13 months ago    4.43MB
busybox                                latest            08ef35a1c3f0   13 months ago    4.43MB
rajput2117_cloudethix_busybox          latest            08ef35a1c3f0   13 months ago    4.43MB
root@DESKTOP-T5TP2DK:~/ubuntu_custom#
```

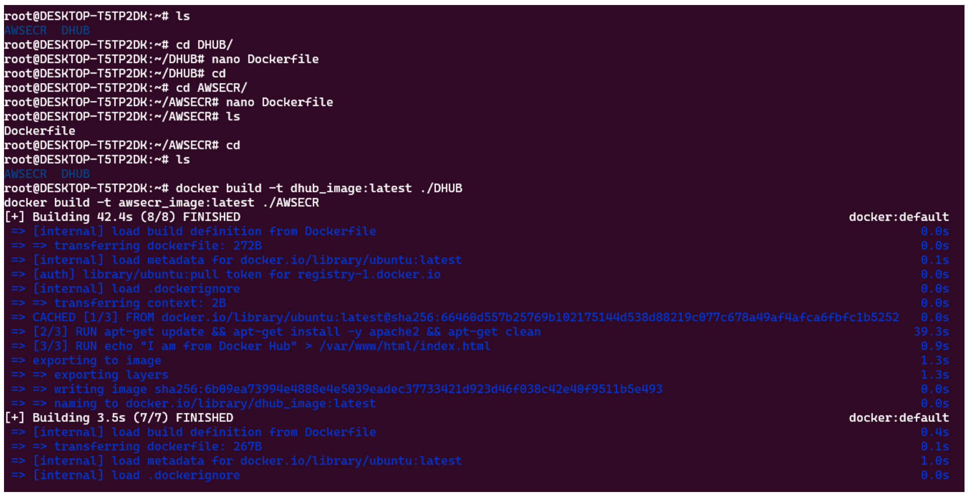
Task 31. Create Directories: Establish two directories named "DHUB" and "AWSECR."



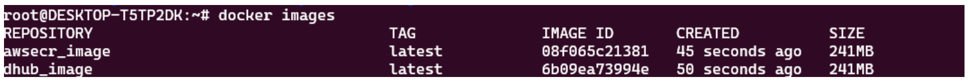
```
root@DESKTOP-T5TP2DK:~# mkdir DHUB AWSECR
root@DESKTOP-T5TP2DK:~# ls
AWSECR  DHUB
root@DESKTOP-T5TP2DK:~#
```

Task 32. Dockerfile Creation: Develop two Dockerfiles to construct custom images with the following specifications:
Base image: Ubuntu
Install packages: httpd
Add "I am from Docker Hub" to the index.html file for DHUB directory and "I am from ECR" for AWSECR directory.
Set environment variable ENV_NAME=DHUB for the DHUB directory and ENV_NAME=AWSECR for the AWSECR directory.
Start http service using ENTRYPOINT
Expose port 80.

--created Dockerfiles inside AWSECR DHUB and build images from them.

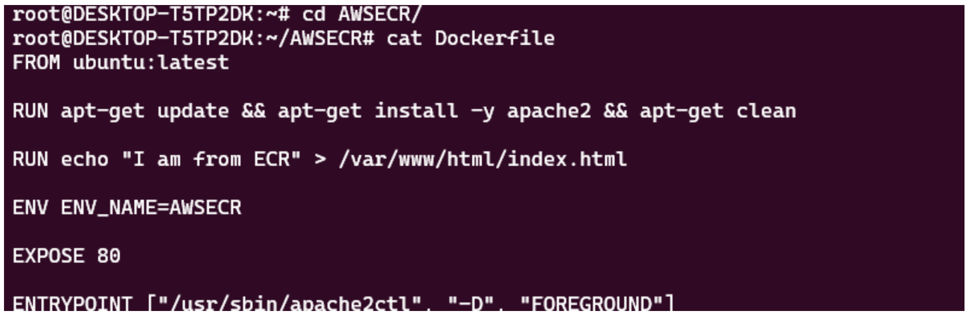


```
root@DESKTOP-T5TP2DK:~# ls
AWSECR  DHUB
root@DESKTOP-T5TP2DK:~# cd DHUB/
root@DESKTOP-T5TP2DK:~/DHUB# nano Dockerfile
root@DESKTOP-T5TP2DK:~/DHUB# cd
root@DESKTOP-T5TP2DK:~# cd AWSECR/
root@DESKTOP-T5TP2DK:~/AWSECR# nano Dockerfile
root@DESKTOP-T5TP2DK:~/AWSECR# ls
Dockerfile
root@DESKTOP-T5TP2DK:~/AWSECR# cd
root@DESKTOP-T5TP2DK:~# ls
AWSECR  DHUB
root@DESKTOP-T5TP2DK:~# docker build -t dhub_image:latest ./DHUB
docker build -t awsecr_image:latest ./AWSECR
[+] Building 42.4s (8/8) FINISHED                                                                       docker:default
 => [internal] load build definition from Dockerfile                                                              0.0s
 => => transferring dockerfile: 272B                                                                              0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest                                                  0.1s
 => [auth] library/ubuntu:pull token for registry-1.docker.io                                                     0.0s
 => [internal] load .dockerignore                                                                                 0.0s
 => => transferring context: 2B                                                                                   0.0s
 => CACHED [1/3] FROM docker.io/library/ubuntu:latest@sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6fbfc1b5252   0.0s
 => [2/3] RUN apt-get update && apt-get install -y apache2 && apt-get clean                                      39.3s
 => [3/3] RUN echo "I am from Docker Hub" > /var/www/html/index.html                                              0.9s
 => exporting to image                                                                                            1.3s
 => => exporting layers                                                                                           1.3s
 => => writing image sha256:6b09ea73994e4888e4e5039eadec37733421d923d46f038c42e40f9511b5e493                      0.0s
 => => naming to docker.io/library/dhub_image:latest                                                              0.0s
[+] Building 3.5s (7/7) FINISHED                                                                        docker:default
 => [internal] load build definition from Dockerfile                                                              0.4s
 => => transferring dockerfile: 267B                                                                              0.1s
 => [internal] load metadata for docker.io/library/ubuntu:latest                                                  1.0s
 => [internal] load .dockerignore                                                                                 0.0s
```



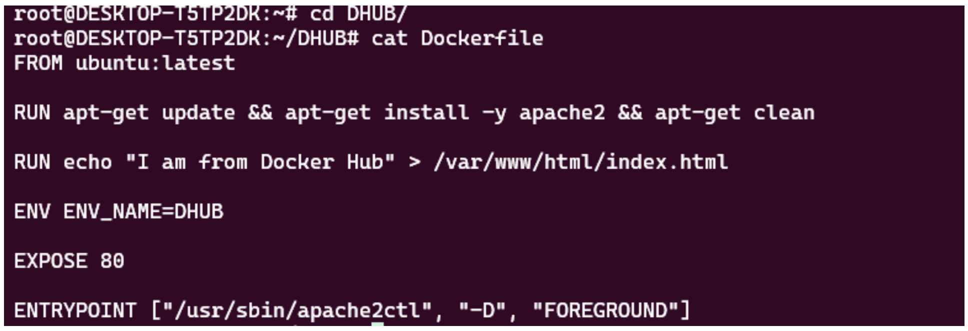
```
root@DESKTOP-T5TP2DK:~# docker images
REPOSITORY                    TAG          IMAGE ID       CREATED           SIZE
awsecr_image                  latest       08f065c21381   45 seconds ago    241MB
dhub_image                    latest       6b09ea73994e   50 seconds ago    241MB
```

--- AWSECR Dockerfile.



```
root@DESKTOP-T5TP2DK:~# cd AWSECR/
root@DESKTOP-T5TP2DK:~/AWSECR# cat Dockerfile
FROM ubuntu:latest

RUN apt-get update && apt-get install -y apache2 && apt-get clean

RUN echo "I am from ECR" > /var/www/html/index.html

ENV ENV_NAME=AWSECR

EXPOSE 80

ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

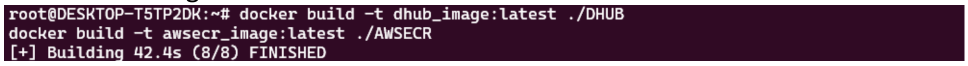
--> DHUB Dockerfile



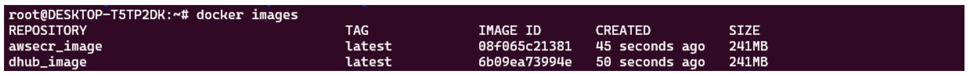
```
root@DESKTOP-T5TP2DK:~# cd DHUB/
root@DESKTOP-T5TP2DK:~/DHUB# cat Dockerfile
FROM ubuntu:latest

RUN apt-get update && apt-get install -y apache2 && apt-get clean

RUN echo "I am from Docker Hub" > /var/www/html/index.html

ENV ENV_NAME=DHUB

EXPOSE 80

ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

Task 33. Build Custom Images: Utilize the docker build command to build the custom images. List all available images using docker images.

-- build the image and list them



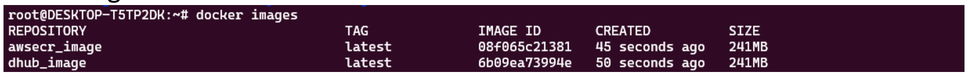
```
root@DESKTOP-T5TP2DK:~# docker build -t dhub_image:latest ./DHUB
docker build -t awsecr_image:latest ./AWSECR
[+] Building 42.4s (8/8) FINISHED
```



```
root@DESKTOP-T5TP2DK:~# docker images
REPOSITORY                    TAG          IMAGE ID       CREATED           SIZE
awsecr_image                  latest       08f065c21381   45 seconds ago    241MB
dhub_image                    latest       6b09ea73994e   50 seconds ago    241MB
```

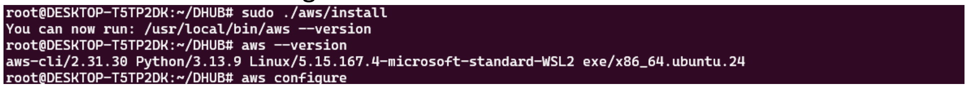
Task 34. Push Images to Repositories: Push the Docker image to Docker Hub. Push the Docker image to AWS ECR.

--list images



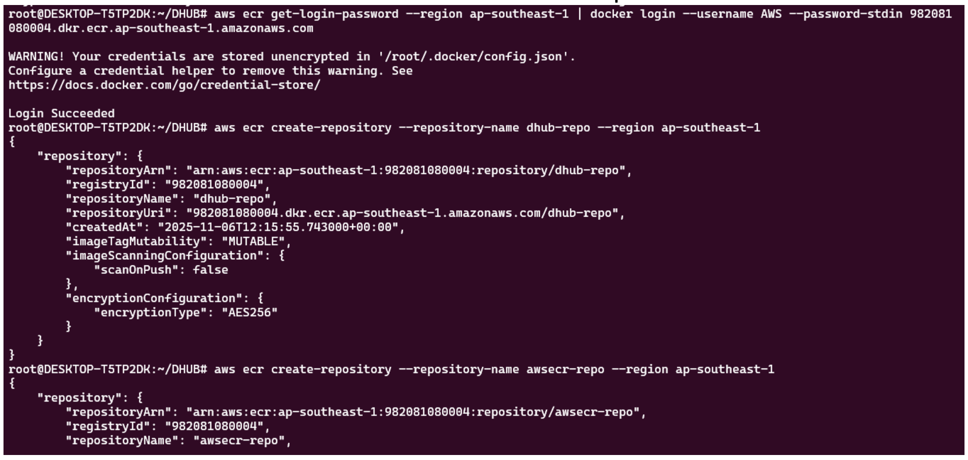
```
root@DESKTOP-T5TP2DK:~# docker images
REPOSITORY                                    TAG                 IMAGE ID       CREATED          SIZE
awsecr_image                                  latest              08f065c21381   45 seconds ago   241MB
dhub_image                                    latest              6b09ea73994e   50 seconds ago   241MB
```

-- aws cli install and configure



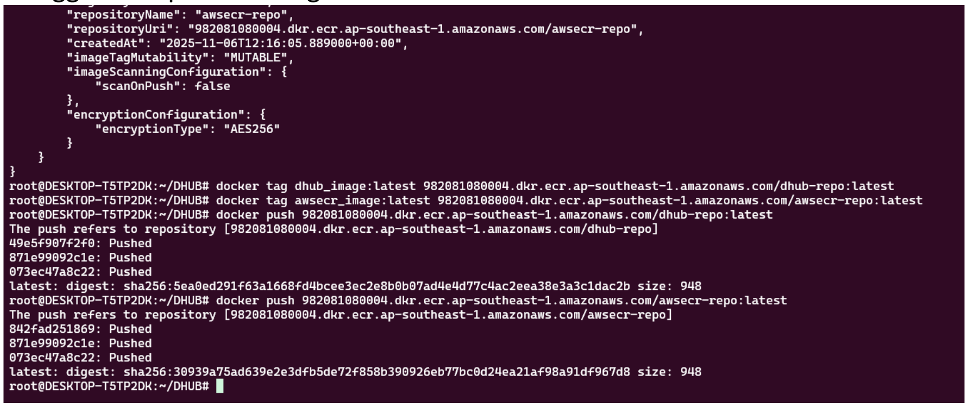
```
root@DESKTOP-T5TP2DK:~/DHUB# sudo ./aws/install
You can now run: /usr/local/bin/aws --version
root@DESKTOP-T5TP2DK:~/DHUB# aws --version
aws-cli/2.31.30 Python/3.13.9 Linux/5.15.167.4-microsoft-standard-WSL2 exe/x86_64.ubuntu.24
root@DESKTOP-T5TP2DK:~/DHUB# aws configure
```

--- Authenticate ECR with Docker and create ECR repos.



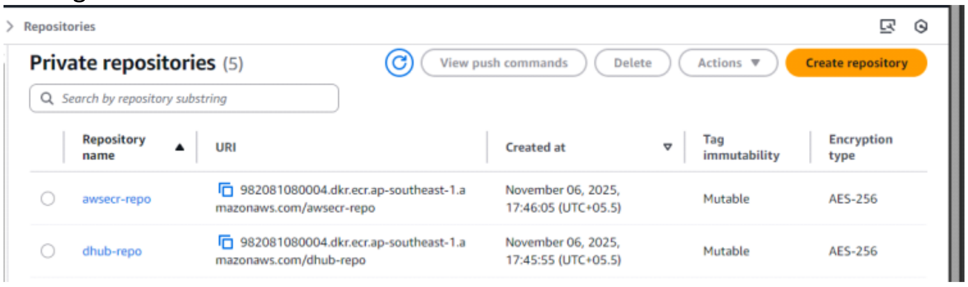
```
root@DESKTOP-T5TP2DK:~/DHUB# aws ecr get-login-password --region ap-southeast-1 | docker login --username AWS --password-stdin 982081
080004.dkr.ecr.ap-southeast-1.amazonaws.com

WARNING! Your credentials are stored unencrypted in '/root/.docker/config.json'.
Configure a credential helper to remove this warning. See
https://docs.docker.com/go/credential-store/

Login Succeeded
root@DESKTOP-T5TP2DK:~/DHUB# aws ecr create-repository --repository-name dhub-repo --region ap-southeast-1
{
    "repository": {
        "repositoryArn": "arn:aws:ecr:ap-southeast-1:982081080004:repository/dhub-repo",
        "registryId": "982081080004",
        "repositoryName": "dhub-repo",
        "repositoryUri": "982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/dhub-repo",
        "createdAt": "2025-11-06T12:15:55.743000+00:00",
        "imageTagMutability": "MUTABLE",
        "imageScanningConfiguration": {
            "scanOnPush": false
        },
        "encryptionConfiguration": {
            "encryptionType": "AES256"
        }
    }
}
root@DESKTOP-T5TP2DK:~/DHUB# aws ecr create-repository --repository-name awsecr-repo --region ap-southeast-1
{
    "repository": {
        "repositoryArn": "arn:aws:ecr:ap-southeast-1:982081080004:repository/awsecr-repo",
        "registryId": "982081080004",
        "repositoryName": "awsecr-repo",
```

-- tagged and pushed images to ECR



```
        "repositoryName": "awsecr-repo",
        "repositoryUri": "982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/awsecr-repo",
        "createdAt": "2025-11-06T12:16:05.889000+00:00",
        "imageTagMutability": "MUTABLE",
        "imageScanningConfiguration": {
            "scanOnPush": false
        },
        "encryptionConfiguration": {
            "encryptionType": "AES256"
        }
    }
}
root@DESKTOP-T5TP2DK:~/DHUB# docker tag dhub_image:latest 982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/dhub-repo:latest
root@DESKTOP-T5TP2DK:~/DHUB# docker tag awsecr_image:latest 982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/awsecr-repo:latest
root@DESKTOP-T5TP2DK:~/DHUB# docker push 982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/dhub-repo:latest
The push refers to repository [982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/dhub-repo]
49e5f907f2f0: Pushed
871e99092c1e: Pushed
073ec47a8c22: Pushed
latest: digest: sha256:5ea0ed291f63a1668fd4bcee3ec2e8b0b07ad4e4d77c4ac2eea38e3a3c1dac2b size: 948
root@DESKTOP-T5TP2DK:~/DHUB# docker push 982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/awsecr-repo:latest
The push refers to repository [982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/awsecr-repo]
842fad251869: Pushed
871e99092c1e: Pushed
073ec47a8c22: Pushed
latest: digest: sha256:30939a75ad639e2e3dfb5de72f858b390926eb77bc0d24ea21af98a91df967d8 size: 948
root@DESKTOP-T5TP2DK:~/DHUB# 
```

-- images at ECR

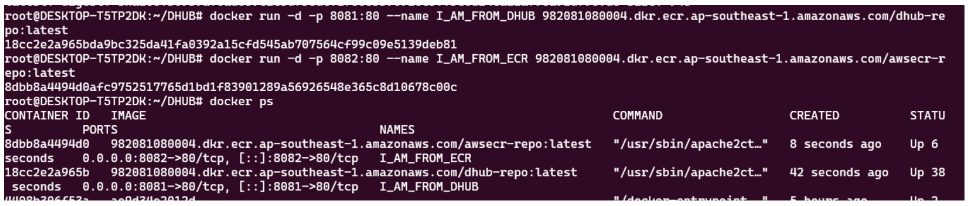


Private repositories (5)

	Repository name	URI	Created at	Tag immutability	Encryption type
○	awsecr-repo	982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/awsecr-repo	November 06, 2025, 17:46:05 (UTC+05.5)	Mutable	AES-256
○	dhub-repo	982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/dhub-repo	November 06, 2025, 17:45:55 (UTC+05.5)	Mutable	AES-256

Task 35. Run Containers:
docker run -d -p 8081:80 --name I_AM_FROM_DHUB your_docker_hub_image
docker run -d -p 8082:80 --name I_AM_FROM_ECR your_aws_ecr_image

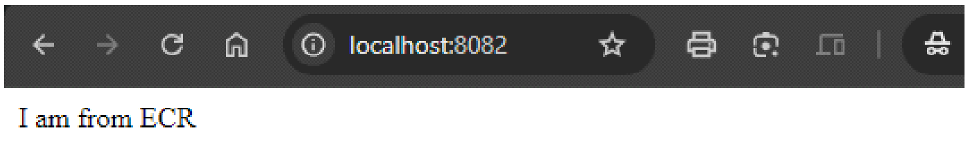
--created container using images from ECR and listed running Containers.



```
root@DESKTOP-T5TP2DK:~/DHUB# docker run -d -p 8081:80 --name I_AM_FROM_DHUB 982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/dhub-re
po:latest
18cc2e2a965bda9bc325da41fa0392a15cfd545ab707564cf99c09e5139deb81
root@DESKTOP-T5TP2DK:~/DHUB# docker run -d -p 8082:80 --name I_AM_FROM_ECR 982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/awsecr-r
epo:latest
8dbb8a4494d0afc9752517765d1bd1f83901289a56926548e365c8d10678c00c
root@DESKTOP-T5TP2DK:~/DHUB# docker ps
CONTAINER ID   IMAGE                                                            COMMAND                  CREATED          STATU
S           PORTS                                      NAMES
8dbb8a4494d0   982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/awsecr-repo:latest   "/usr/sbin/apache2ct…"   8 seconds ago    Up 6
seconds    0.0.0.0:8082->80/tcp, [::]:8082->80/tcp   I_AM_FROM_ECR
18cc2e2a965b   982081080004.dkr.ecr.ap-southeast-1.amazonaws.com/dhub-repo:latest     "/usr/sbin/apache2ct…"   42 seconds ago   Up 38
 seconds   0.0.0.0:8081->80/tcp, [::]:8081->80/tcp   I_AM_FROM_DHUB
```

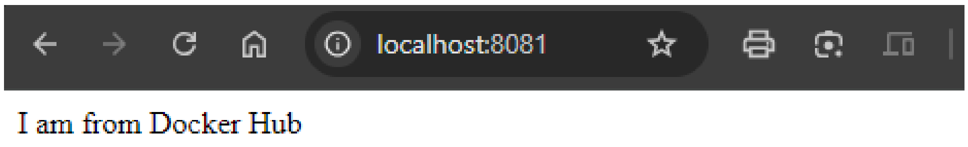
36. Access Pages from Browser:
Open a web browser and access the pages:
For Docker Hub: http://localhost:8081
For AWS ECR: http://localhost:8082

--> accessed container on localhost.



localhost:8082

I am from ECR



localhost:8081

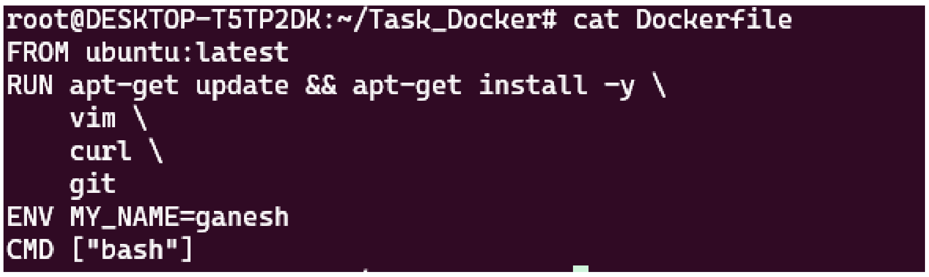
I am from Docker Hub

Tasks on volume and networking appended later.

Task 1. **Dockerfile creation**: Create a simple Dockerfile to build a custom image with the following specifications:

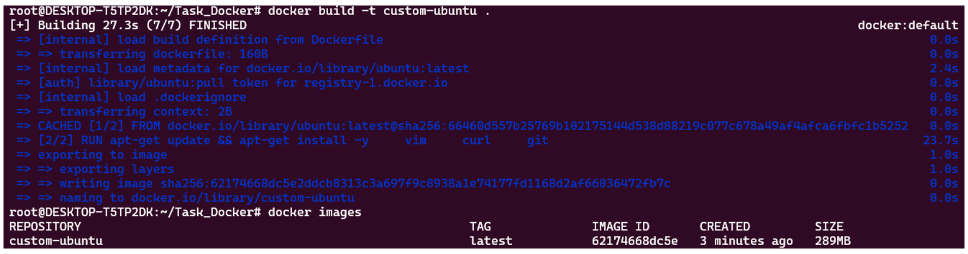
- Base image: Ubuntu
- Install packages: curl, vim, and git
- Set an environment variable: MY_NAME=Your_Name
- Build the custom image using docker build and list all available images using docker images.

--- Dockerfile for container.



```
root@DESKTOP-T5TP2DK:~/Task_Docker# cat Dockerfile
FROM ubuntu:latest
RUN apt-get update && apt-get install -y \
    vim \
    curl \
    git
ENV MY_NAME=ganesh
CMD ["bash"]
```

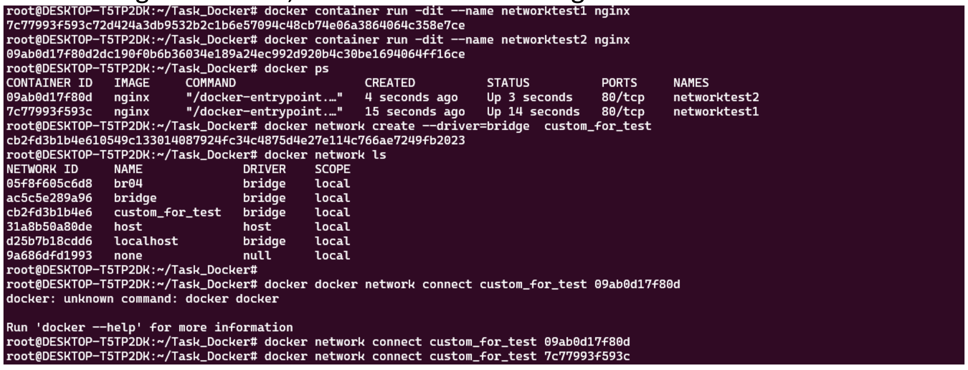
---- creating image from file



```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker build -t custom-ubuntu .
[+] Building 27.3s (7/7) FINISHED                                                                        docker:default
 => [internal] load build definition from Dockerfile                                                              0.0s
 => => transferring dockerfile: 160B                                                                              0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest                                                  2.4s
 => [auth] library/ubuntu:pull token for registry-1.docker.io                                                     0.0s
 => [internal] load .dockerignore                                                                                 0.0s
 => => transferring context: 2B                                                                                   0.0s
 => CACHED [1/2] FROM docker.io/library/ubuntu:latest@sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6fbfc1b5252  0.0s
 => [2/2] RUN apt-get update && apt-get install -y     vim     curl     git                                      23.7s
 => exporting to image                                                                                            1.0s
 => => exporting layers                                                                                           1.0s
 => => writing image sha256:62174668dc5e2ddcb8313c3a697f9c8938a1e74177fd1168d2af66036472fb7c                      0.0s
 => => naming to docker.io/library/custom-ubuntu                                                                  0.0s
root@DESKTOP-T5TP2DK:~/Task_Docker# docker images
REPOSITORY                                                       TAG               IMAGE ID       CREATED         SIZE
custom-ubuntu                                                    latest            62174668dc5e   3 minutes ago   289MB
```

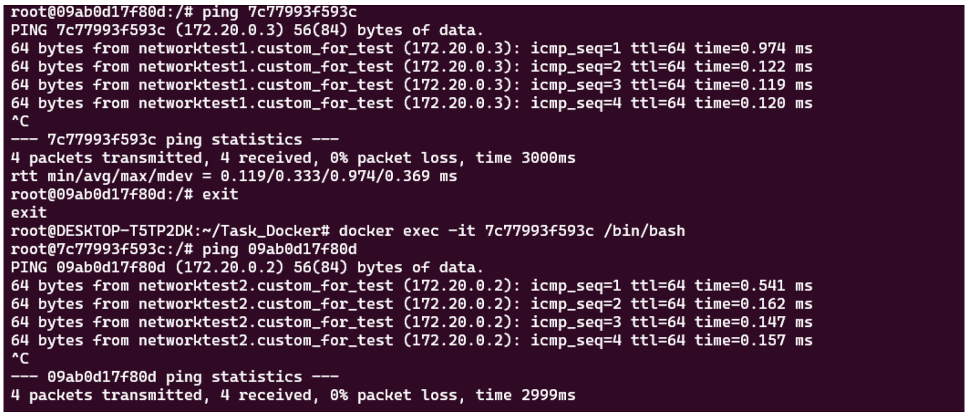
Task 2. **Docker networking**: Create two Docker containers with different names, and add them to the same custom network. Verify that the containers can communicate with each other using their names as hostnames.

---> creating containers ,network and connecting them to network .



```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker container run -dit --name networktest1 nginx
7c77993f593c72d424a3db9532b2c1b6e57094c48cb74e06a3864064c358e7ce
root@DESKTOP-T5TP2DK:~/Task_Docker# docker container run -dit --name networktest2 nginx
09ab0d17f80d2dc190f0b6b36034e189a24ec992d920b4c30be1694064ff16ce
root@DESKTOP-T5TP2DK:~/Task_Docker# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS     NAMES
09ab0d17f80d   nginx     "/docker-entrypoint.…"   4 seconds ago    Up 3 seconds    80/tcp    networktest2
7c77993f593c   nginx     "/docker-entrypoint.…"   15 seconds ago   Up 14 seconds   80/tcp    networktest1
root@DESKTOP-T5TP2DK:~/Task_Docker# docker network create --driver=bridge  custom_for_test
cb2fd3b1b4e610549c133014087924fc34c4875d4e27e114c766ae7249fb2023
root@DESKTOP-T5TP2DK:~/Task_Docker# docker network ls
NETWORK ID     NAME              DRIVER    SCOPE
05f8f605c6d8   br04              bridge    local
ac5c5e289a96   bridge            bridge    local
cb2fd3b1b4e6   custom_for_test   bridge    local
31a8b50a80de   host              host      local
d25b7b18cdd6   localhost         bridge    local
9a686dfd1993   none              null      local
root@DESKTOP-T5TP2DK:~/Task_Docker#
root@DESKTOP-T5TP2DK:~/Task_Docker# docker docker network connect custom_for_test 09ab0d17f80d
docker: unknown command: docker docker

Run 'docker --help' for more information
root@DESKTOP-T5TP2DK:~/Task_Docker# docker network connect custom_for_test 09ab0d17f80d
root@DESKTOP-T5TP2DK:~/Task_Docker# docker network connect custom_for_test 7c77993f593c
```

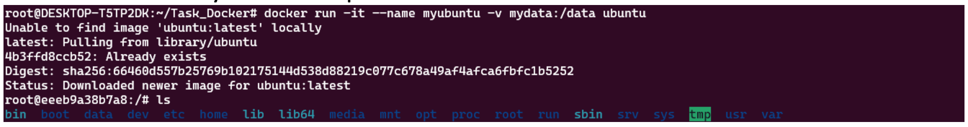
--> installed ping inside containers and then ping them each other



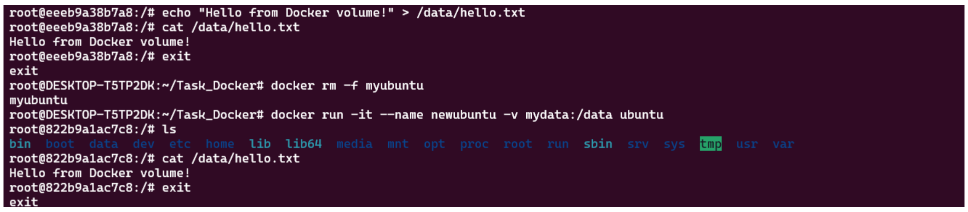
```
root@09ab0d17f80d:/# ping 7c77993f593c
PING 7c77993f593c (172.20.0.3) 56(84) bytes of data.
64 bytes from networktest1.custom_for_test (172.20.0.3): icmp_seq=1 ttl=64 time=0.974 ms
64 bytes from networktest1.custom_for_test (172.20.0.3): icmp_seq=2 ttl=64 time=0.122 ms
64 bytes from networktest1.custom_for_test (172.20.0.3): icmp_seq=3 ttl=64 time=0.119 ms
64 bytes from networktest1.custom_for_test (172.20.0.3): icmp_seq=4 ttl=64 time=0.120 ms
^C
--- 7c77993f593c ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.119/0.333/0.974/0.369 ms
root@09ab0d17f80d:/# exit
exit
root@DESKTOP-T5TP2DK:~/Task_Docker# docker exec -it 7c77993f593c /bin/bash
root@7c77993f593c:/# ping 09ab0d17f80d
PING 09ab0d17f80d (172.20.0.2) 56(84) bytes of data.
64 bytes from networktest2.custom_for_test (172.20.0.2): icmp_seq=1 ttl=64 time=0.541 ms
64 bytes from networktest2.custom_for_test (172.20.0.2): icmp_seq=2 ttl=64 time=0.162 ms
64 bytes from networktest2.custom_for_test (172.20.0.2): icmp_seq=3 ttl=64 time=0.147 ms
64 bytes from networktest2.custom_for_test (172.20.0.2): icmp_seq=4 ttl=64 time=0.157 ms
^C
--- 09ab0d17f80d ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
```

Task 3. **Data persistence with volumes**: Run a Docker container with an attached volume, create a file inside the volume, and verify that the file persists after the container is stopped and removed.

--> created a container with volume and created file inside it removed that container and attach that volume to another
container to verify that data persist.



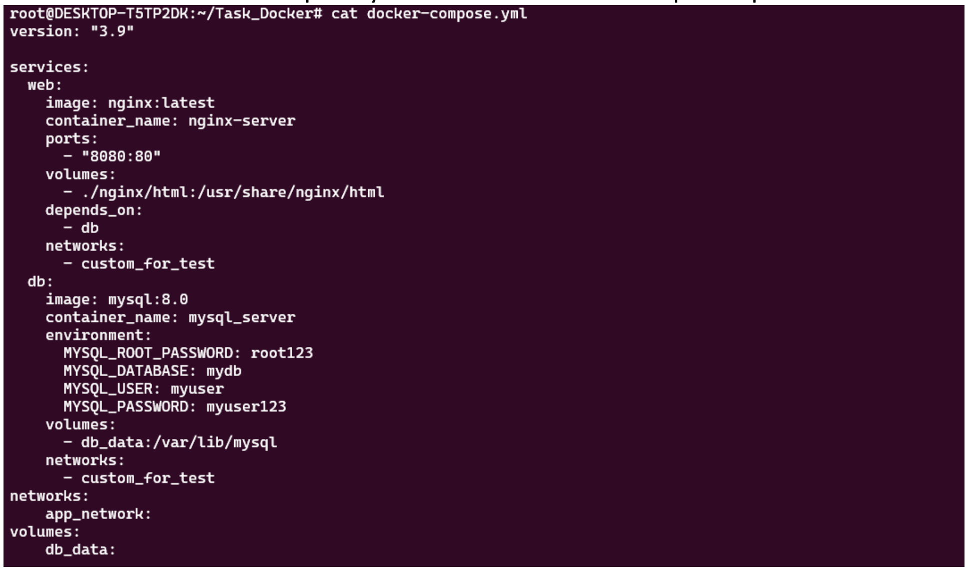
```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker run -it --name myubuntu -v mydata:/data ubuntu
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
4b3ffd8ccb52: Already exists
Digest: sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6fbfc1b5252
Status: Downloaded newer image for ubuntu:latest
root@eeeb9a38b7a8:/# ls
bin  boot  data  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
```



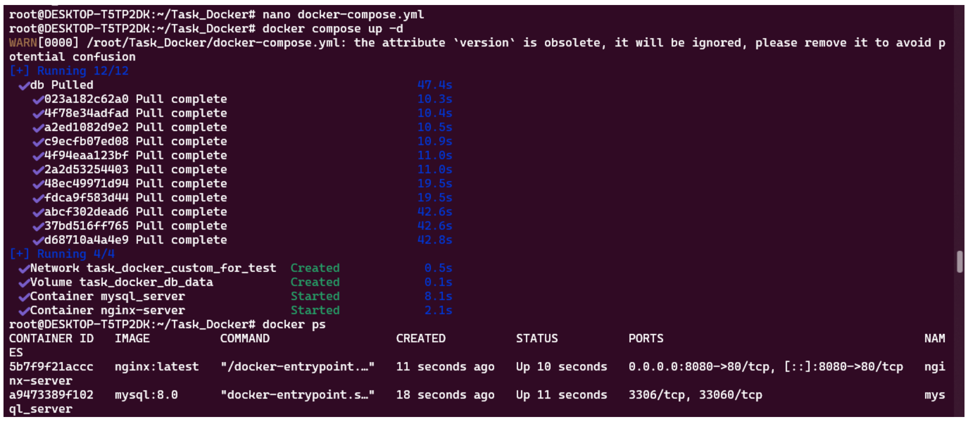
```
root@eeeb9a38b7a8:/# echo "Hello from Docker volume!" > /data/hello.txt
root@eeeb9a38b7a8:/# cat /data/hello.txt
Hello from Docker volume!
root@eeeb9a38b7a8:/# exit
exit
root@DESKTOP-T5TP2DK:~/Task_Docker# docker rm -f myubuntu
myubuntu
root@DESKTOP-T5TP2DK:~/Task_Docker# docker run -it --name newubuntu -v mydata:/data ubuntu
root@822b9a1ac7c8:/# ls
bin  boot  data  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
root@822b9a1ac7c8:/# cat /data/hello.txt
Hello from Docker volume!
root@822b9a1ac7c8:/# exit
exit
```

Task 4. **Docker Compose**: Create a docker-compose.yml file to run a multi-container application consisting of a web server (e.g., Nginx) and a database server (e.g., MySQL). Ensure that both containers are connected to the same network.

--> created a docker-compose.yml and used docker compose up -d for container creation.



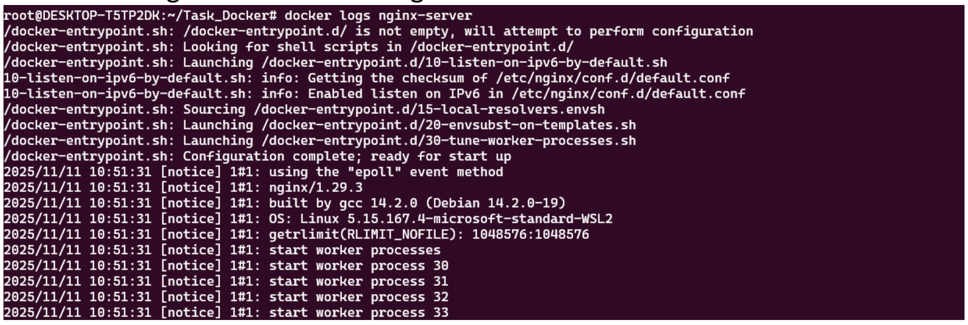
```
root@DESKTOP-T5TP2DK:~/Task_Docker# cat docker-compose.yml
version: "3.9"

services:
  web:
    image: nginx:latest
    container_name: nginx-server
    ports:
      - "8080:80"
    volumes:
      - ./nginx/html:/usr/share/nginx/html
    depends_on:
      - db
    networks:
      - custom_for_test
  db:
    image: mysql:8.0
    container_name: mysql_server
    environment:
      MYSQL_ROOT_PASSWORD: root123
      MYSQL_DATABASE: mydb
      MYSQL_USER: myuser
      MYSQL_PASSWORD: myuser123
    volumes:
      - db_data:/var/lib/mysql
    networks:
      - custom_for_test
networks:
    app_network:
volumes:
    db_data:
```



```
root@DESKTOP-T5TP2DK:~/Task_Docker# nano docker-compose.yml
root@DESKTOP-T5TP2DK:~/Task_Docker# docker compose up -d
WARN[0000] /root/Task_Docker/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid p
otential confusion
[+] Running 12/12
 ✔db Pulled                                          47.4s
   ✔023a182c62a0 Pull complete                       10.3s
   ✔4f78e34adfad Pull complete                       10.4s
   ✔a2ed1082d9e2 Pull complete                       10.5s
   ✔c9ecfb07ed08 Pull complete                       10.9s
   ✔4f94eaa123bf Pull complete                       11.0s
   ✔2a2d53254403 Pull complete                       11.0s
   ✔48ec49971d94 Pull complete                       19.5s
   ✔fdca9f583d44 Pull complete                       19.5s
   ✔abcf302dead6 Pull complete                       42.6s
   ✔37bd516ff765 Pull complete                       42.6s
   ✔d68710a4a4e9 Pull complete                       42.8s
[+] Running 4/4
 ✔Network task_docker_custom_for_test  Created          0.5s
 ✔Volume task_docker_db_data           Created          0.1s
 ✔Container mysql_server               Started          8.1s
 ✔Container nginx-server               Started          2.1s
root@DESKTOP-T5TP2DK:~/Task_Docker# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                                   NAM
ES
5b7f9f21accc   nginx:latest   "/docker-entrypoint.…"   11 seconds ago   Up 10 seconds   0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   ngi
nx-server
a9473389f102   mysql:8.0      "docker-entrypoint.s…"   18 seconds ago   Up 11 seconds   3306/tcp, 33060/tcp                       mys
ql_server
```

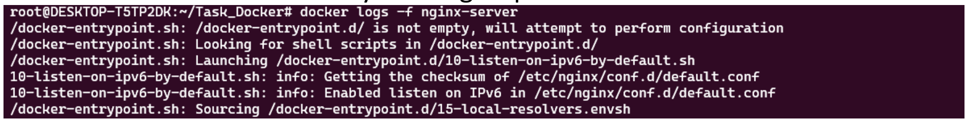
Task 5. **Container logging**: Run a Docker container that generates log output, and practice using docker logs to view and analyze the logs.

---> viewed log of above created nginx-server named container.

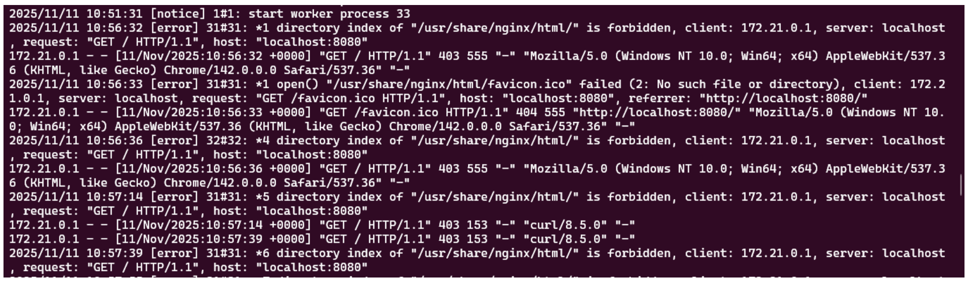


```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker logs nginx-server
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/11/11 10:51:31 [notice] 1#1: using the "epoll" event method
2025/11/11 10:51:31 [notice] 1#1: nginx/1.29.3
2025/11/11 10:51:31 [notice] 1#1: built by gcc 14.2.0 (Debian 14.2.0-19)
2025/11/11 10:51:31 [notice] 1#1: OS: Linux 5.15.167.4-microsoft-standard-WSL2
2025/11/11 10:51:31 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/11/11 10:51:31 [notice] 1#1: start worker processes
2025/11/11 10:51:31 [notice] 1#1: start worker process 30
2025/11/11 10:51:31 [notice] 1#1: start worker process 31
2025/11/11 10:51:31 [notice] 1#1: start worker process 32
2025/11/11 10:51:31 [notice] 1#1: start worker process 33
```

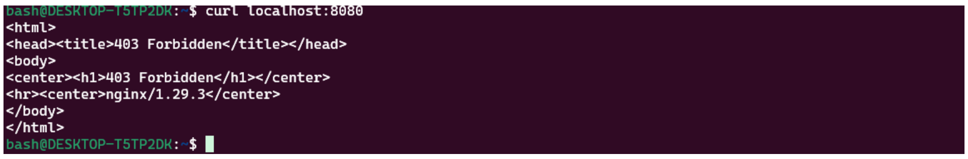
---> in live time and verified by sending requests from another terminal.



```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker logs -f nginx-server
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
```



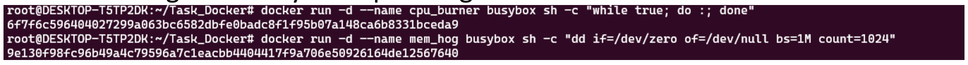
```
2025/11/11 10:51:31 [notice] 1#1: start worker process 33
2025/11/11 10:56:32 [error] 31#31: *1 directory index of "/usr/share/nginx/html/" is forbidden, client: 172.21.0.1, server: localhost
, request: "GET / HTTP/1.1", host: "localhost:8080"
172.21.0.1 - - [11/Nov/2025:10:56:32 +0000] "GET / HTTP/1.1" 403 555 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.3
6 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36" "-"
2025/11/11 10:56:33 [error] 31#31: *1 open() "/usr/share/nginx/html/favicon.ico" failed (2: No such file or directory), client: 172.2
1.0.1, server: localhost, request: "GET /favicon.ico HTTP/1.1", host: "localhost:8080", referrer: "http://localhost:8080/"
172.21.0.1 - - [11/Nov/2025:10:56:33 +0000] "GET /favicon.ico HTTP/1.1" 404 555 "http://localhost:8080/" "Mozilla/5.0 (Windows NT 10.
0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36" "-"
2025/11/11 10:56:36 [error] 32#32: *4 directory index of "/usr/share/nginx/html/" is forbidden, client: 172.21.0.1, server: localhost
, request: "GET / HTTP/1.1", host: "localhost:8080"
172.21.0.1 - - [11/Nov/2025:10:56:36 +0000] "GET / HTTP/1.1" 403 555 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.3
6 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36" "-"
2025/11/11 10:57:14 [error] 31#31: *5 directory index of "/usr/share/nginx/html/" is forbidden, client: 172.21.0.1, server: localhost
, request: "GET / HTTP/1.1", host: "localhost:8080"
172.21.0.1 - - [11/Nov/2025:10:57:14 +0000] "GET / HTTP/1.1" 403 153 "-" "curl/8.5.0" "-"
172.21.0.1 - - [11/Nov/2025:10:57:39 +0000] "GET / HTTP/1.1" 403 153 "-" "curl/8.5.0" "-"
2025/11/11 10:57:39 [error] 31#31: *6 directory index of "/usr/share/nginx/html/" is forbidden, client: 172.21.0.1, server: localhost
, request: "GET / HTTP/1.1", host: "localhost:8080"
```



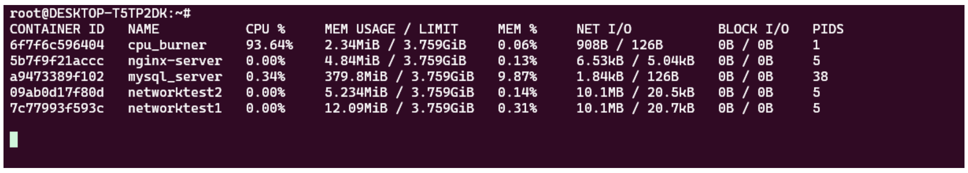
```
bash@DESKTOP-T5TP2DK:~$ curl localhost:8080
<html>
<head><title>403 Forbidden</title></head>
<body>
<center><h1>403 Forbidden</h1></center>
<hr><center>nginx/1.29.3</center>
</body>
</html>
bash@DESKTOP-T5TP2DK:~$
```

Task 6.Container resource monitoring: Run a Docker container that consumes system resources (e.g., CPU or memory) and practice using docker stats and docker top to monitor the container's resource usage.

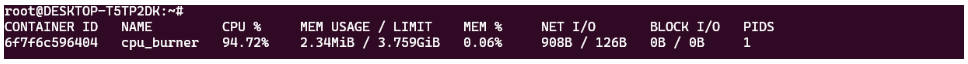
---- created high memory and cpu using containers and moniter stats.



```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker run -d --name cpu_burner busybox sh -c "while true; do :; done"
6f7f6c596404027299a063bc6582dbfe0badc8f1f95b07a148ca6b8331bceda9
root@DESKTOP-T5TP2DK:~/Task_Docker# docker run -d --name mem_hog busybox sh -c "dd if=/dev/zero of=/dev/null bs=1M count=1024"
9e130f98fc96b49a4c79596a7c1eacbb4404417f9a706e50926164de12567640
```



```
root@DESKTOP-T5TP2DK:~#
CONTAINER ID   NAME           CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O   PIDS
6f7f6c596404   cpu_burner     93.64%    2.34MiB / 3.759GiB    0.06%     908B / 126B       0B / 0B     1
5b7f9f21accc   nginx-server   0.00%     4.84MiB / 3.759GiB    0.13%     6.53kB / 5.04kB   0B / 0B     5
a9473389f102   mysql_server   0.34%     379.8MiB / 3.759GiB   9.87%     1.84kB / 126B     0B / 0B     38
09ab0d17f80d   networktest2   0.00%     5.234MiB / 3.759GiB   0.14%     10.1MB / 20.5kB   0B / 0B     5
7c77993f593c   networktest1   0.00%     12.09MiB / 3.759GiB   0.31%     10.1MB / 20.7kB   0B / 0B     5
```

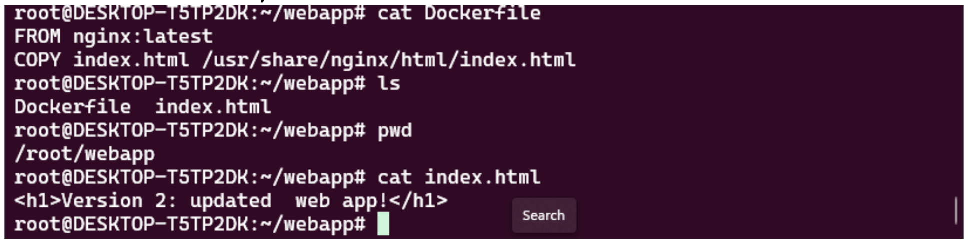


```
root@DESKTOP-T5TP2DK:~#
CONTAINER ID   NAME         CPU %     MEM USAGE / LIMIT    MEM %     NET I/O       BLOCK I/O   PIDS
6f7f6c596404   cpu_burner   94.72%    2.34MiB / 3.759GiB   0.06%     908B / 126B   0B / 0B     1
```

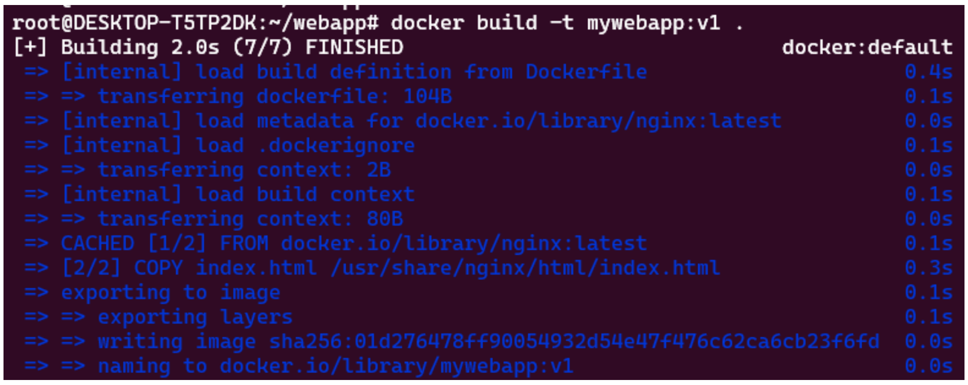
Removed container for cpu consuption stop.

Task 7. **Updating a containerized application**: Update the source code of a simple containerized application, rebuild the Docker image, and deploy the updated version while minimizing downtime.

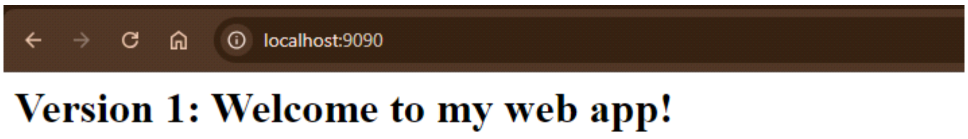
--> created a directory and dockerfile in it with source code.



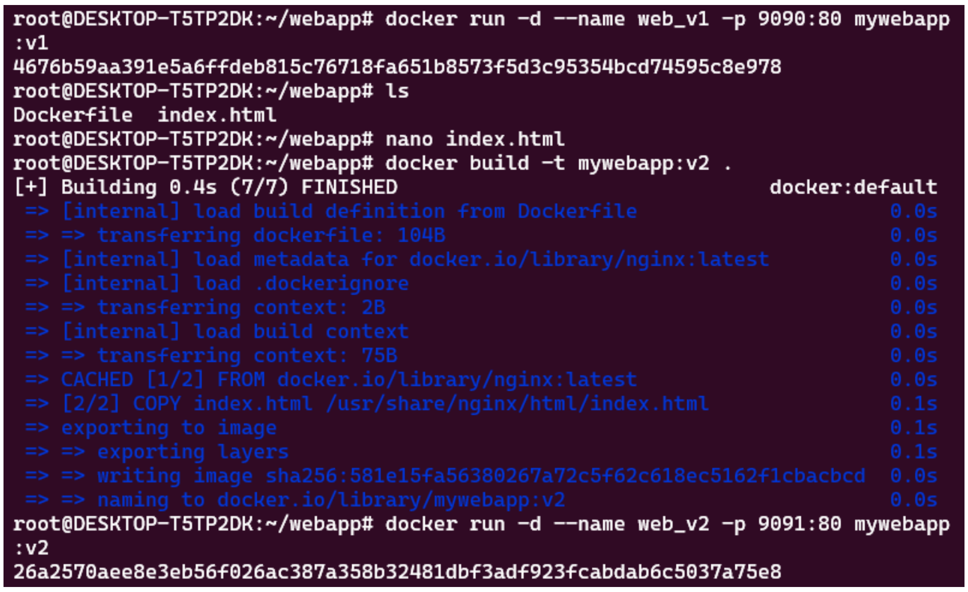
```
root@DESKTOP-T5TP2DK:~/webapp# cat Dockerfile
FROM nginx:latest
COPY index.html /usr/share/nginx/html/index.html
root@DESKTOP-T5TP2DK:~/webapp# ls
Dockerfile  index.html
root@DESKTOP-T5TP2DK:~/webapp# pwd
/root/webapp
root@DESKTOP-T5TP2DK:~/webapp# cat index.html
<h1>Version 2: updated  web app!</h1>
root@DESKTOP-T5TP2DK:~/webapp#
```

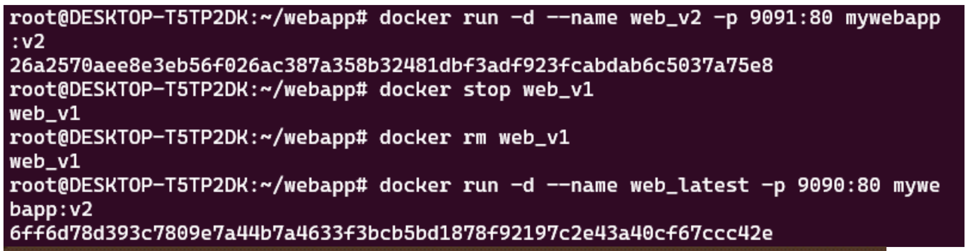


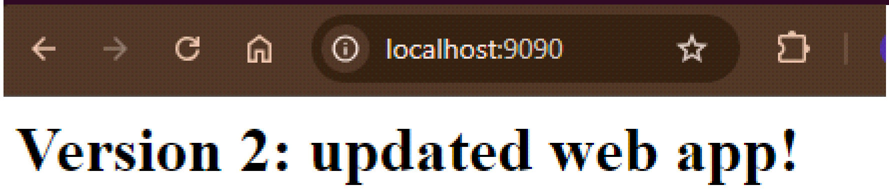
```
root@DESKTOP-T5TP2DK:~/webapp# docker build -t mywebapp:v1 .
[+] Building 2.0s (7/7) FINISHED                                docker:default
 => [internal] load build definition from Dockerfile                      0.4s
 => => transferring dockerfile: 104B                                      0.1s
 => [internal] load metadata for docker.io/library/nginx:latest           0.0s
 => [internal] load .dockerignore                                         0.1s
 => => transferring context: 2B                                           0.0s
 => [internal] load build context                                         0.1s
 => => transferring context: 80B                                          0.0s
 => CACHED [1/2] FROM docker.io/library/nginx:latest                      0.1s
 => [2/2] COPY index.html /usr/share/nginx/html/index.html                0.3s
 => exporting to image                                                    0.1s
 => => exporting layers                                                   0.1s
 => => writing image sha256:01d276478ff90054932d54e47f476c62ca6cb23f6fd   0.0s
 => => naming to docker.io/library/mywebapp:v1                            0.0s
```


localhost:9090

Version 1: Welcome to my web app!


```
root@DESKTOP-T5TP2DK:~/webapp# docker run -d --name web_v1 -p 9090:80 mywebapp
:v1
4676b59aa391e5a6ffdeb815c76718fa651b8573f5d3c95354bcd74595c8e978
root@DESKTOP-T5TP2DK:~/webapp# ls
Dockerfile  index.html
root@DESKTOP-T5TP2DK:~/webapp# nano index.html
root@DESKTOP-T5TP2DK:~/webapp# docker build -t mywebapp:v2 .
[+] Building 0.4s (7/7) FINISHED                                docker:default
 => [internal] load build definition from Dockerfile                      0.0s
 => => transferring dockerfile: 104B                                      0.0s
 => [internal] load metadata for docker.io/library/nginx:latest           0.0s
 => [internal] load .dockerignore                                         0.0s
 => => transferring context: 2B                                           0.0s
 => [internal] load build context                                         0.0s
 => => transferring context: 75B                                          0.0s
 => CACHED [1/2] FROM docker.io/library/nginx:latest                      0.0s
 => [2/2] COPY index.html /usr/share/nginx/html/index.html                0.1s
 => exporting to image                                                    0.1s
 => => exporting layers                                                   0.1s
 => => writing image sha256:581e15fa56380267a72c5f62c618ec5162f1cbacbcd  0.0s
 => => naming to docker.io/library/mywebapp:v2                            0.0s
root@DESKTOP-T5TP2DK:~/webapp# docker run -d --name web_v2 -p 9091:80 mywebapp
:v2
26a2570aee8e3eb56f026ac387a358b32481dbf3adf923fcabdab6c5037a75e8
```

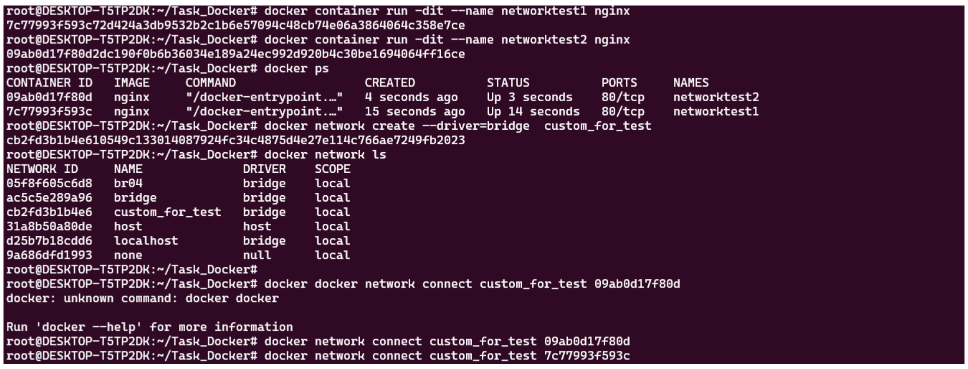

```
root@DESKTOP-T5TP2DK:~/webapp# docker run -d --name web_v2 -p 9091:80 mywebapp
:v2
26a2570aee8e3eb56f026ac387a358b32481dbf3adf923fcabdab6c5037a75e8
root@DESKTOP-T5TP2DK:~/webapp# docker stop web_v1
web_v1
root@DESKTOP-T5TP2DK:~/webapp# docker rm web_v1
web_v1
root@DESKTOP-T5TP2DK:~/webapp# docker run -d --name web_latest -p 9090:80 mywe
bapp:v2
6ff6d78d393c7809e7a44b7a4633f3bcb5bd1878f92197c2e43a40cf67ccc42e
```


localhost:9090

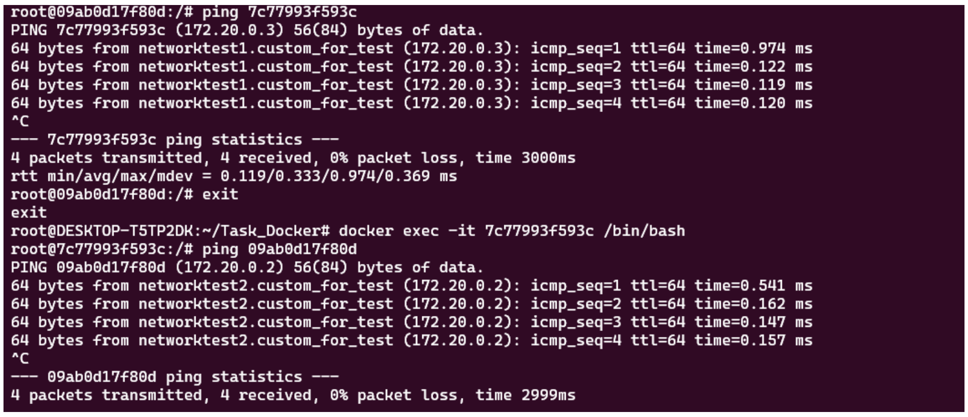
Version 2: updated web app!

Task 8. **Bridge network**: Create two Docker containers using different images, such as NGINX and MySQL. Connect them using a user-defined bridge network and ensure they can communicate with each other.

--> creating containers ,network and connecting them to network .



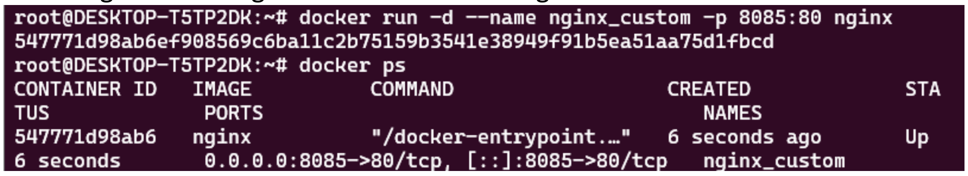
```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker container run -dit --name networktest1 nginx
7c77993f593c72d424a3db9532b2c1b6e57094c48cb74e06a3864064c358e7ce
root@DESKTOP-T5TP2DK:~/Task_Docker# docker container run -dit --name networktest2 nginx
09ab0d17f80d2dc190f0b6b36034e189a24ec992d920b4c30be1694064ff16ce
root@DESKTOP-T5TP2DK:~/Task_Docker# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS     NAMES
09ab0d17f80d   nginx     "/docker-entrypoint.…"   4 seconds ago    Up 3 seconds    80/tcp    networktest2
7c77993f593c   nginx     "/docker-entrypoint.…"   15 seconds ago   Up 14 seconds   80/tcp    networktest1
root@DESKTOP-T5TP2DK:~/Task_Docker# docker network create --driver=bridge  custom_for_test
cb2fd3b1b4e610549c133014087924fc34c4875d4e27e114c766ae7249fb2023
root@DESKTOP-T5TP2DK:~/Task_Docker# docker network ls
NETWORK ID     NAME              DRIVER    SCOPE
05f8f605c6d8   br04              bridge    local
ac5c5e289a96   bridge            bridge    local
cb2fd3b1b4e6   custom_for_test   bridge    local
31a8b50a80de   host              host      local
d25b7b18cdd6   localhost         bridge    local
9a686dfd1993   none              null      local
root@DESKTOP-T5TP2DK:~/Task_Docker#
root@DESKTOP-T5TP2DK:~/Task_Docker# docker docker network connect custom_for_test 09ab0d17f80d
docker: unknown command: docker docker

Run 'docker --help' for more information
root@DESKTOP-T5TP2DK:~/Task_Docker# docker network connect custom_for_test 09ab0d17f80d
root@DESKTOP-T5TP2DK:~/Task_Docker# docker network connect custom_for_test 7c77993f593c
```



```
root@09ab0d17f80d:/# ping 7c77993f593c
PING 7c77993f593c (172.20.0.3) 56(84) bytes of data.
64 bytes from networktest1.custom_for_test (172.20.0.3): icmp_seq=1 ttl=64 time=0.974 ms
64 bytes from networktest1.custom_for_test (172.20.0.3): icmp_seq=2 ttl=64 time=0.122 ms
64 bytes from networktest1.custom_for_test (172.20.0.3): icmp_seq=3 ttl=64 time=0.119 ms
64 bytes from networktest1.custom_for_test (172.20.0.3): icmp_seq=4 ttl=64 time=0.120 ms
^C
--- 7c77993f593c ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.119/0.333/0.974/0.369 ms
root@09ab0d17f80d:/# exit
exit
root@DESKTOP-T5TP2DK:~/Task_Docker# docker exec -it 7c77993f593c /bin/bash
root@7c77993f593c:/# ping 09ab0d17f80d
PING 09ab0d17f80d (172.20.0.2) 56(84) bytes of data.
64 bytes from networktest2.custom_for_test (172.20.0.2): icmp_seq=1 ttl=64 time=0.541 ms
64 bytes from networktest2.custom_for_test (172.20.0.2): icmp_seq=2 ttl=64 time=0.162 ms
64 bytes from networktest2.custom_for_test (172.20.0.2): icmp_seq=3 ttl=64 time=0.147 ms
64 bytes from networktest2.custom_for_test (172.20.0.2): icmp_seq=4 ttl=64 time=0.157 ms
^C
--- 09ab0d17f80d ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
```

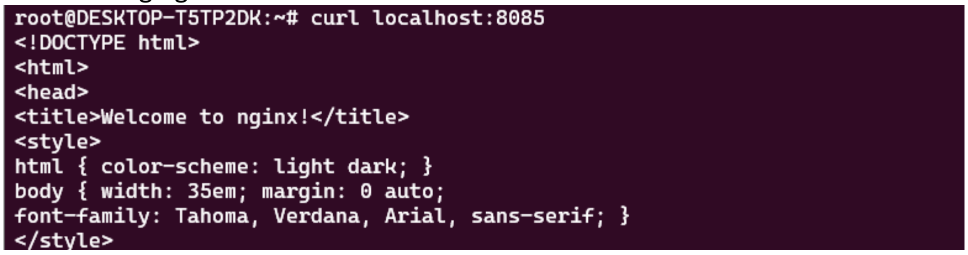
Task 9.**Port mapping**: Run a containerized web server (e.g., NGINX or Apache) and map its default port (80 or 443) to a custom port on the host machine. Verify that the web server is accessible through the custom port on the host.

--running a custom nginx container and listing it



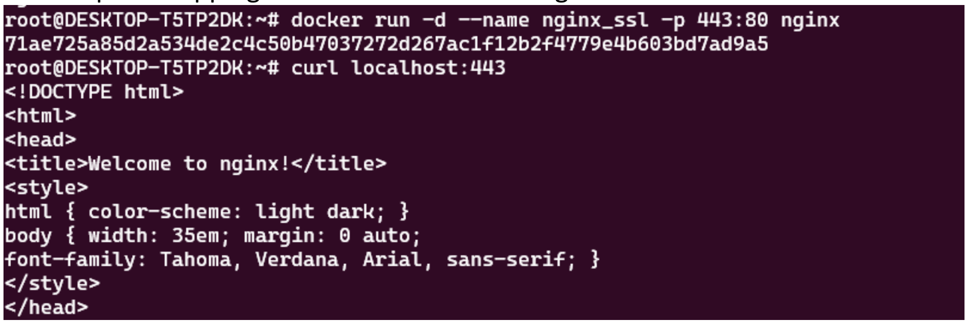
```
root@DESKTOP-T5TP2DK:~# docker run -d --name nginx_custom -p 8085:80 nginx
547771d98ab6ef908569c6ba11c2b75159b3541e38949f91b5ea51aa75d1fbcd
root@DESKTOP-T5TP2DK:~# docker ps
CONTAINER ID   IMAGE         COMMAND                   CREATED           STA
TUS             PORTS                                     NAMES
547771d98ab6   nginx         "/docker-entrypoint.…"   6 seconds ago     Up
6 seconds        0.0.0.0:8085->80/tcp, [::]:8085->80/tcp   nginx_custom
```

--- Accessing nginx



```
root@DESKTOP-T5TP2DK:~# curl localhost:8085
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
```

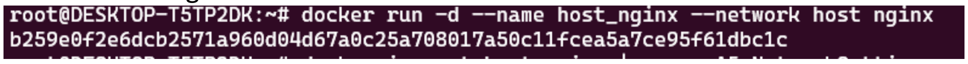
-- done port-mapping and then accessed the nginx.



```
root@DESKTOP-T5TP2DK:~# docker run -d --name nginx_ssl -p 443:80 nginx
71ae725a85d2a534de2c4c50b47037272d267ac1f12b2f4779e4b603bd7ad9a5
root@DESKTOP-T5TP2DK:~# curl localhost:443
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
```

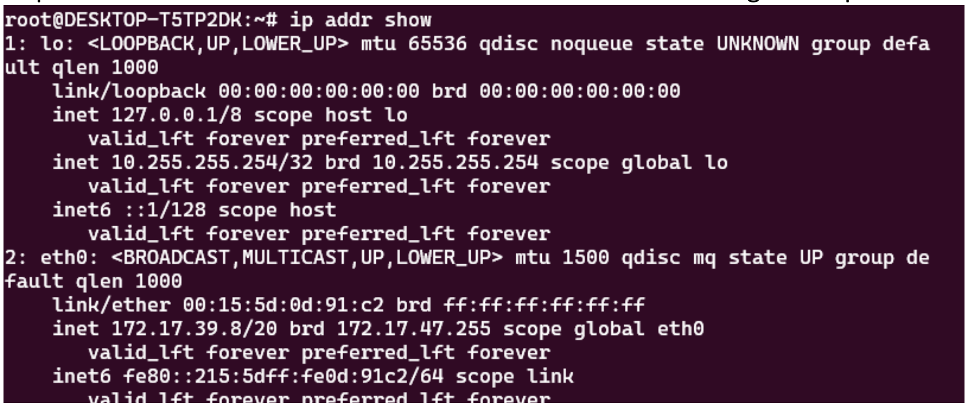
Task 10. **Host networking**: Run a container with the host networking mode and compare its network configuration with that of the host machine. Explain the advantages and disadvantages of using host networking for containers

---> created a nginx cont with host network

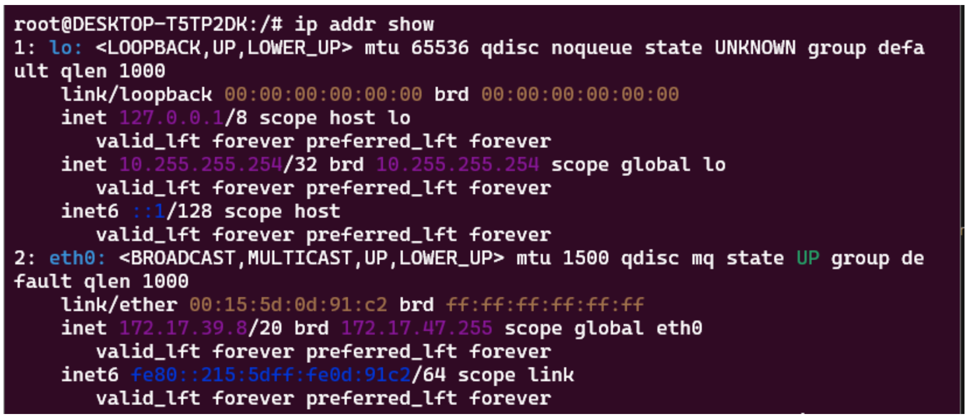


```
root@DESKTOP-T5TP2DK:~# docker run -d --name host_nginx --network host nginx
b259e0f2e6dcb2571a960d04d67a0c25a708017a50c11fcea5a7ce95f61dbc1c
```

-- Ip adress check outside container and inside container having same ip adress.



```
root@DESKTOP-T5TP2DK:~# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defa
ult qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet 10.255.255.254/32 brd 10.255.255.254 scope global lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group de
fault qlen 1000
    link/ether 00:15:5d:0d:91:c2 brd ff:ff:ff:ff:ff:ff
    inet 172.17.39.8/20 brd 172.17.47.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::215:5dff:fe0d:91c2/64 scope link
       valid_lft forever preferred_lft forever
```



```
root@DESKTOP-T5TP2DK:/# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defa
ult qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet 10.255.255.254/32 brd 10.255.255.254 scope global lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group de
fault qlen 1000
    link/ether 00:15:5d:0d:91:c2 brd ff:ff:ff:ff:ff:ff
    inet 172.17.39.8/20 brd 172.17.47.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::215:5dff:fe0d:91c2/64 scope link
       valid_lft forever preferred_lft forever
```

Advantages of Host Networking

1. **Better Performance (Low Latency):**
   Containers share the host's network stack directly—no NAT or bridge overhead—so network packets move faster.
   Ideal for high-performance or real-time apps.
2. **Simpler Networking Setup:**
   You don't need to expose or map ports (e.g., -p 8080:80), because the container uses the same IP and ports as the host.
   Easier for services that must bind to a specific port or IP.
3. **Full Access to Host Network Interfaces:**
   The container can use all interfaces and listen on any network device the host has.
   Useful for network monitoring tools, firewalls, or packet sniffers (e.g., tcpdump, prometheus-node-exporter).

Disadvantages of Host Networking

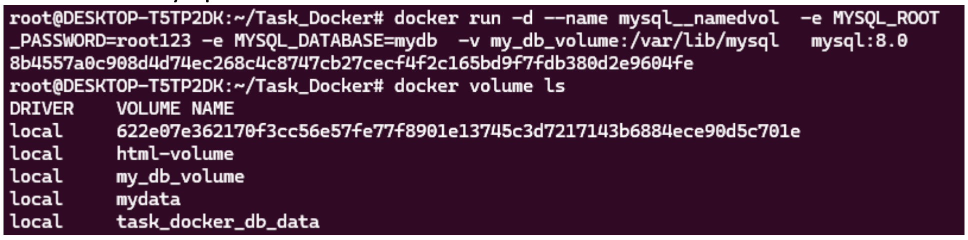
1. **Port Conflicts:**
   Since containers share the host's ports, you can't run multiple containers on the same port (e.g., two Nginx containers on port 80).
   - Leads to binding errors.
2. **Reduced Isolation:**
   The container can directly interact with the host's network stack—less secure than the default bridge network.
   - Increases risk if one container is compromised.
3. **Harder to Manage Large Deployments:**
   You lose the flexibility of Docker's virtual networks, service discovery, and network-level isolation.

- Not suitable for large microservice architectures.

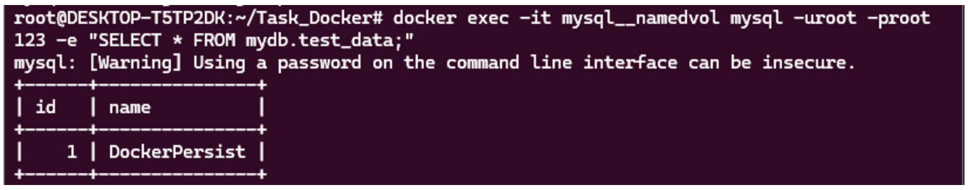
4. **Not Supported Everywhere:**
   On some platforms (like Docker Desktop for Mac/Windows), --network host doesn't behave the same way as on Linux.

Task 11. **Named volume**: Launch a containerized database (e.g., MySQL or PostgreSQL) using a named volume to store its data. Stop and remove the container, then recreate it using the same named volume. Verify that the data persists across container recreations.

-- created a mysql container with the named volume.

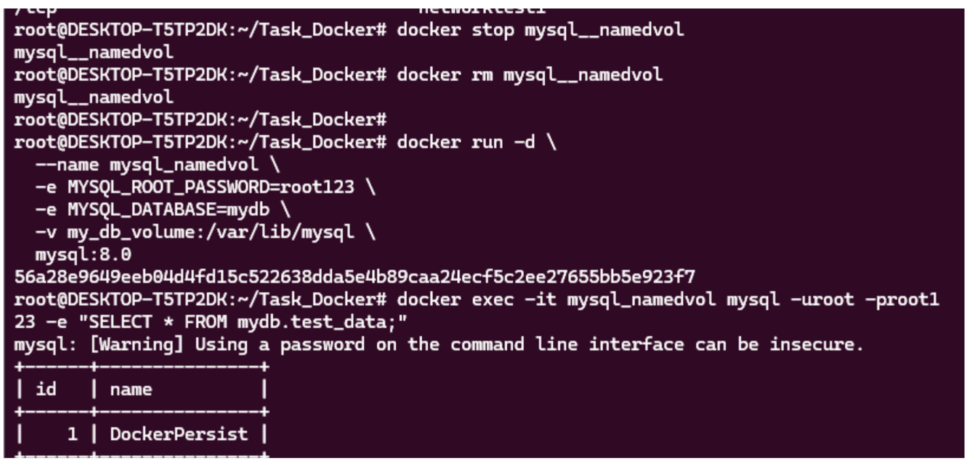


```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker run -d --name mysql__namedvol  -e MYSQL_ROOT
_PASSWORD=root123 -e MYSQL_DATABASE=mydb  -v my_db_volume:/var/lib/mysql   mysql:8.0
8b4557a0c908d4d74ec268c4c8747cb27cecf4f2c165bd9f7fdb380d2e9604fe
root@DESKTOP-T5TP2DK:~/Task_Docker# docker volume ls
DRIVER    VOLUME NAME
local     622e07e362170f3cc56e57fe77f8901e13745c3d7217143b6884ece90d5c701e
local     html-volume
local     my_db_volume
local     mydata
local     task_docker_db_data
```



```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker exec -it mysql__namedvol mysql -uroot -proot
123 -e "SELECT * FROM mydb.test_data;"
mysql: [Warning] Using a password on the command line interface can be insecure.
+------+---------------+
| id   | name          |
+------+---------------+
|    1 | DockerPersist |
+------+---------------+
```

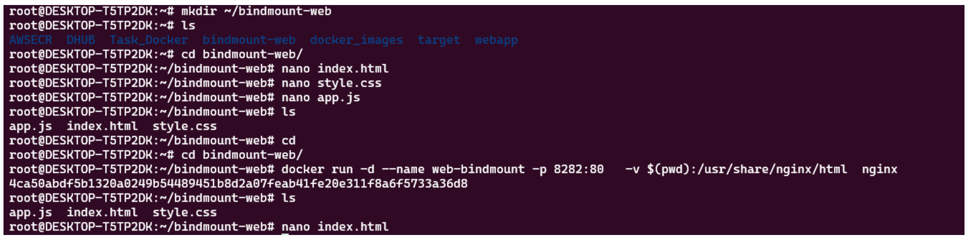
-- removed the container and verify volume persist the dat



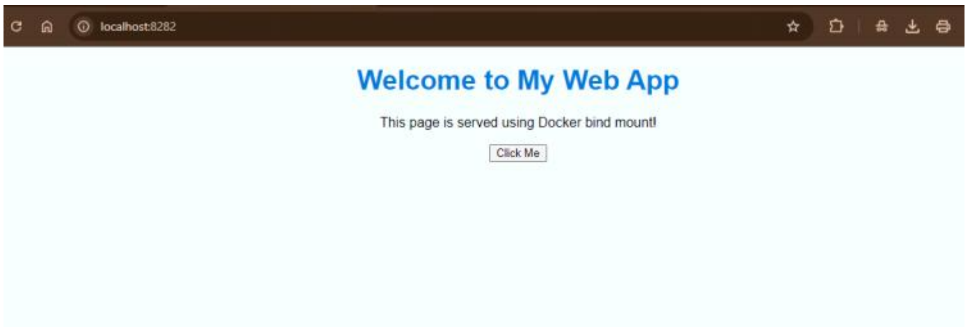
```
root@DESKTOP-T5TP2DK:~/Task_Docker# docker stop mysql__namedvol
mysql__namedvol
root@DESKTOP-T5TP2DK:~/Task_Docker# docker rm mysql__namedvol
mysql__namedvol
root@DESKTOP-T5TP2DK:~/Task_Docker#
root@DESKTOP-T5TP2DK:~/Task_Docker# docker run -d \
  --name mysql_namedvol \
  -e MYSQL_ROOT_PASSWORD=root123 \
  -e MYSQL_DATABASE=mydb \
  -v my_db_volume:/var/lib/mysql \
  mysql:8.0
56a28e9649eeb04d4fd15c522638dda5e4b89caa24ecf5c2ee27655bb5e923f7
root@DESKTOP-T5TP2DK:~/Task_Docker# docker exec -it mysql_namedvol mysql -uroot -proot1
23 -e "SELECT * FROM mydb.test_data;"
mysql: [Warning] Using a password on the command line interface can be insecure.
+------+---------------+
| id   | name          |
+------+---------------+
|    1 | DockerPersist |
```

Task 12. **Bind mount**: Create a simple web application with a local directory containing HTML, CSS, and JavaScript files. Run a web server container (e.g., NGINX or Apache) and bind mount the local directory to the container's webroot. Verify that the web application is accessible and update the local files to see if changes are reflected in the container.

--> created a source code directory for host volume to container. Runned the container using local volume and accessed on web.



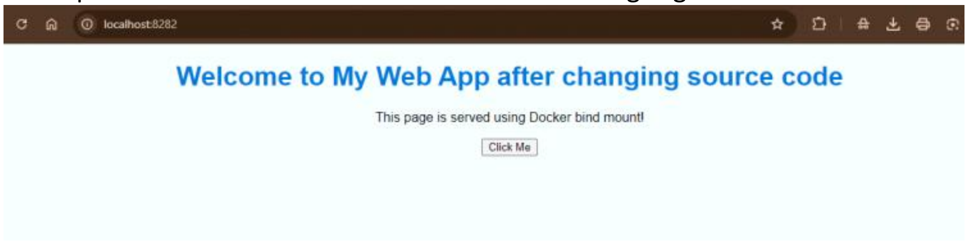
```
root@DESKTOP-T5TP2DK:~# mkdir ~/bindmount-web
root@DESKTOP-T5TP2DK:~# ls
AWSECR  DHUB  Task_Docker  bindmount-web  docker_images  target  webapp
root@DESKTOP-T5TP2DK:~# cd bindmount-web/
root@DESKTOP-T5TP2DK:~/bindmount-web# nano index.html
root@DESKTOP-T5TP2DK:~/bindmount-web# nano style.css
root@DESKTOP-T5TP2DK:~/bindmount-web# nano app.js
root@DESKTOP-T5TP2DK:~/bindmount-web# ls
app.js  index.html  style.css
root@DESKTOP-T5TP2DK:~/bindmount-web# cd
root@DESKTOP-T5TP2DK:~# cd bindmount-web/
root@DESKTOP-T5TP2DK:~/bindmount-web# docker run -d --name web-bindmount -p 8282:80   -v $(pwd):/usr/share/nginx/html  nginx
4ca50abdf5b1320a0249b54489451b8d2a07feab41fe20e311f8a6f5733a36d8
root@DESKTOP-T5TP2DK:~/bindmount-web# ls
app.js  index.html  style.css
root@DESKTOP-T5TP2DK:~/bindmount-web# nano index.html
```


localhost:8282

Welcome to My Web App

This page is served using Docker bind mount!

----> updated the source code and verified that changes get reflected.

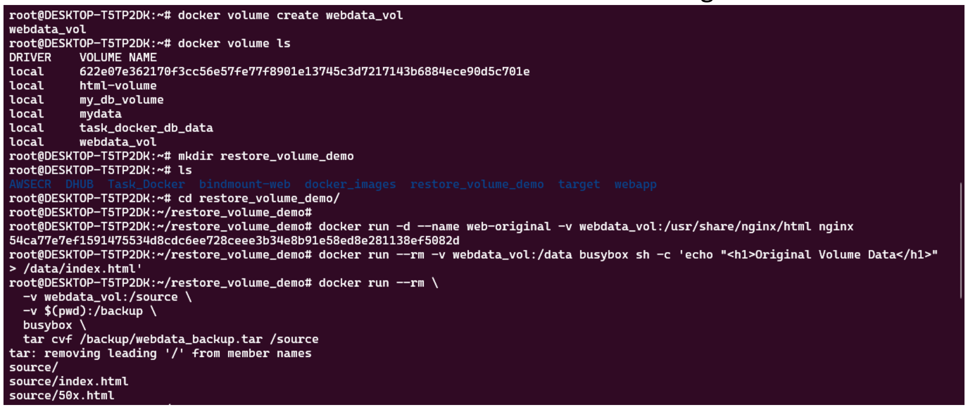

localhost:8282

Welcome to My Web App after changing source code

This page is served using Docker bind mount!

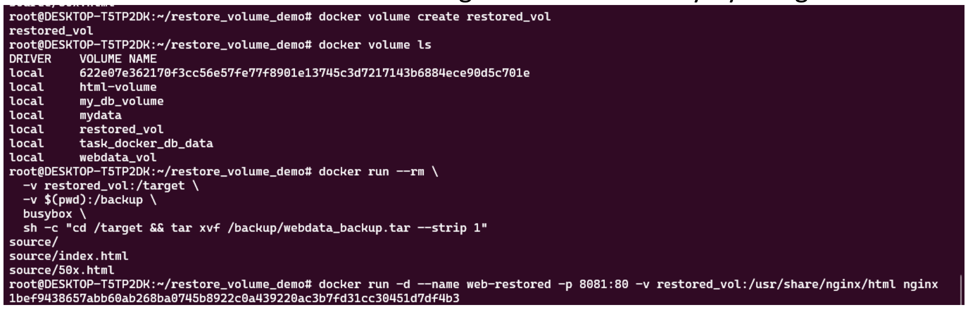
Task 13. **Volume backups**: Using a container with a named volume, create a backup of the volume's data by either exporting it as a tarball or copying the data to a host directory. Restore the data to a new container and verify that the restoration was successful.

---> created a volume listed volume created container using that volume and added file to it



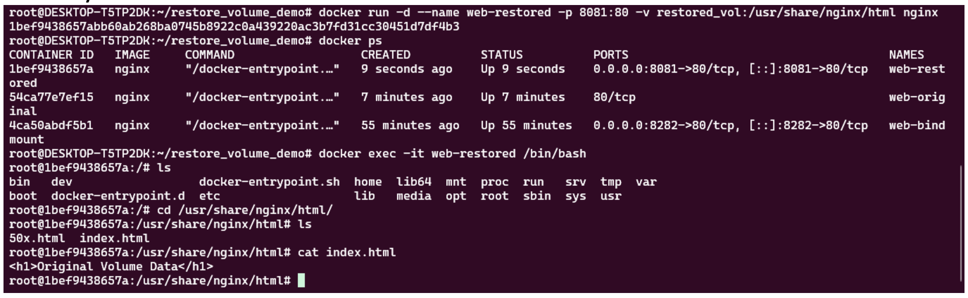
```
root@DESKTOP-T5TP2DK:~# docker volume create webdata_vol
webdata_vol
root@DESKTOP-T5TP2DK:~# docker volume ls
DRIVER    VOLUME NAME
local     622e07e362170f3cc56e57fe77f8901e13745c3d7217143b6884ece90d5c701e
local     html-volume
local     my_db_volume
local     mydata
local     task_docker_db_data
local     webdata_vol
root@DESKTOP-T5TP2DK:~# mkdir restore_volume_demo
root@DESKTOP-T5TP2DK:~# ls
AWSECR  DHUB  Task_Docker  bindmount-web  docker_images  restore_volume_demo  target  webapp
root@DESKTOP-T5TP2DK:~# cd restore_volume_demo/
root@DESKTOP-T5TP2DK:~/restore_volume_demo#
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker run -d --name web-original -v webdata_vol:/usr/share/nginx/html nginx
54ca77e7ef1591475534d8cdc6ee728ceee3b34e8b91e58ed8e281138ef5082d
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker run --rm -v webdata_vol:/data busybox sh -c 'echo "<h1>Original Volume Data</h1>"
> /data/index.html'
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker run --rm \
  -v webdata_vol:/source \
  -v $(pwd):/backup \
  busybox \
  tar cvf /backup/webdata_backup.tar /source
tar: removing leading '/' from member names
source/
source/index.html
source/50x.html
```

-- created another volume for restoring the data and verify by using container.



```
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker volume create restored_vol
restored_vol
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker volume ls
DRIVER    VOLUME NAME
local     622e07e362170f3cc56e57fe77f8901e13745c3d7217143b6884ece90d5c701e
local     html-volume
local     my_db_volume
local     mydata
local     restored_vol
local     task_docker_db_data
local     webdata_vol
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker run --rm \
  -v restored_vol:/target \
  -v $(pwd):/backup \
  busybox \
  sh -c "cd /target && tar xvf /backup/webdata_backup.tar --strip 1"
source/
source/index.html
source/50x.html
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker run -d --name web-restored -p 8081:80 -v restored_vol:/usr/share/nginx/html nginx
1bef9438657abb60ab268ba0745b8922c0a439220ac3b7fd31cc30451d7df4b3
```

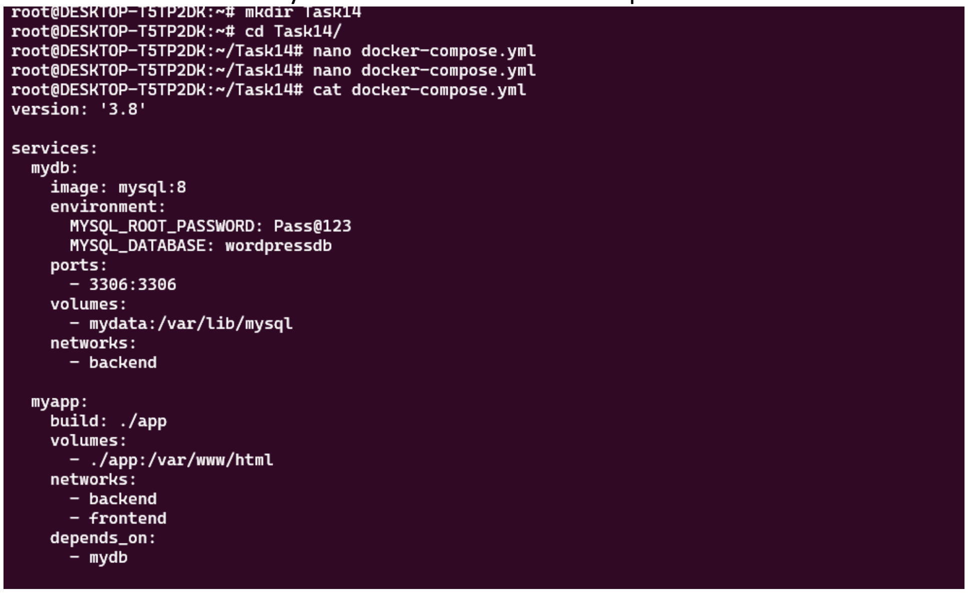
-- verify that file exists inside container.



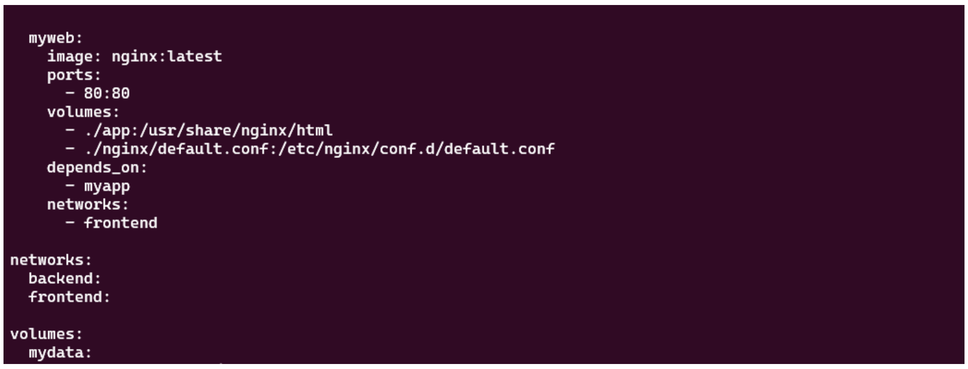
```
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker run -d --name web-restored -p 8081:80 -v restored_vol:/usr/share/nginx/html nginx
1bef9438657abb60ab268ba0745b8922c0a439220ac3b7fd31cc30451d7df4b3
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                                   NAMES
1bef9438657a   nginx     "/docker-entrypoint.…"   9 seconds ago    Up 9 seconds    0.0.0.0:8081->80/tcp, [::]:8081->80/tcp   web-rest
ored
54ca77e7ef15   nginx     "/docker-entrypoint.…"   7 minutes ago    Up 7 minutes    80/tcp                                  web-orig
inal
4ca50abdf5b1   nginx     "/docker-entrypoint.…"   55 minutes ago   Up 55 minutes   0.0.0.0:8282->80/tcp, [::]:8282->80/tcp   web-bind
mount
root@DESKTOP-T5TP2DK:~/restore_volume_demo# docker exec -it web-restored /bin/bash
root@1bef9438657a:/# 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@1bef9438657a:/# cd /usr/share/nginx/html/
root@1bef9438657a:/usr/share/nginx/html# ls
50x.html  index.html
root@1bef9438657a:/usr/share/nginx/html# cat index.html
<h1>Original Volume Data</h1>
root@1bef9438657a:/usr/share/nginx/html# 
```

Task 14. **Docker Compose networking**: Create a Docker Compose file that defines a multi-container application with a frontend, backend, and database. Set up custom networks and volumes for the services and ensure that they can communicate with each other and store data persistently

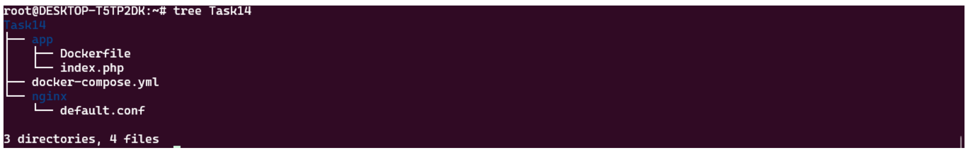
----> created a directory and Dockerfile Docker compose file for three tier architecture.



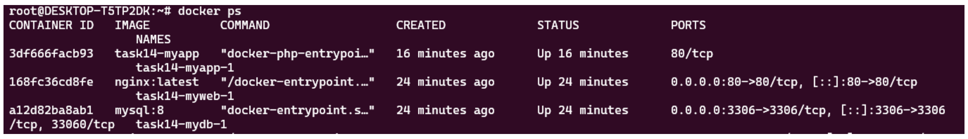
```
root@DESKTOP-T5TP2DK:~# mkdir Task14
root@DESKTOP-T5TP2DK:~# cd Task14/
root@DESKTOP-T5TP2DK:~/Task14# nano docker-compose.yml
root@DESKTOP-T5TP2DK:~/Task14# nano docker-compose.yml
root@DESKTOP-T5TP2DK:~/Task14# cat docker-compose.yml
version: '3.8'

services:
  mydb:
    image: mysql:8
    environment:
      MYSQL_ROOT_PASSWORD: Pass@123
      MYSQL_DATABASE: wordpressdb
    ports:
      - 3306:3306
    volumes:
      - mydata:/var/lib/mysql
    networks:
      - backend

  myapp:
    build: ./app
    volumes:
      - ./app:/var/www/html
    networks:
      - backend
      - frontend
    depends_on:
      - mydb
```



```
  myweb:
    image: nginx:latest
    ports:
      - 80:80
    volumes:
      - ./app:/usr/share/nginx/html
      - ./nginx/default.conf:/etc/nginx/conf.d/default.conf
    depends_on:
      - myapp
    networks:
      - frontend

networks:
  backend:
  frontend:

volumes:
  mydata:
```

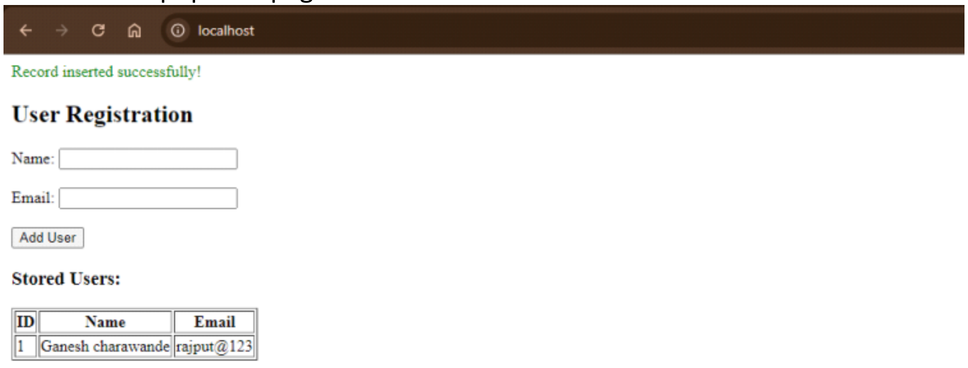


```
root@DESKTOP-T5TP2DK:~# tree Task14
Task14
├── app
│   ├── Dockerfile
│   └── index.php
├── docker-compose.yml
└── nginx
    └── default.conf

3 directories, 4 files
```



```
root@DESKTOP-T5TP2DK:~# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS
                 NAMES
3df666facb93   task14-myapp   "docker-php-entrypoi…"   16 minutes ago   Up 16 minutes   80/tcp
                 task14-myapp-1
168fc36cd8fe   nginx:latest   "/docker-entrypoint.…"   24 minutes ago   Up 24 minutes   0.0.0.0:80->80/tcp, [::]:80->80/tcp
                 task14-myweb-1
a12d82ba8ab1   mysql:8        "docker-entrypoint.s…"   24 minutes ago   Up 24 minutes   0.0.0.0:3306->3306/tcp, [::]:3306->3306
/tcp, 33060/tcp   task14-mydb-1
```

--- Accessed php web page and inserted data.



localhost

Record inserted successfully!

User Registration

Name:

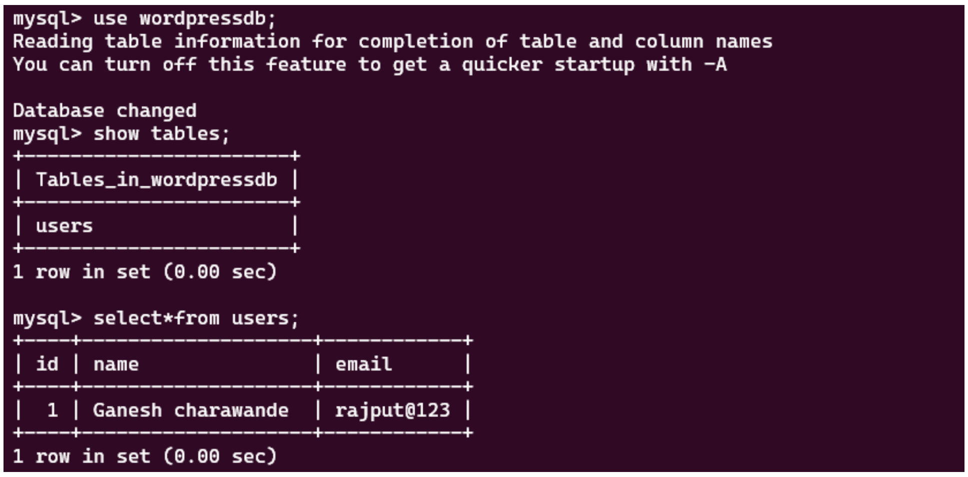
Email:

Add User

Stored Users:

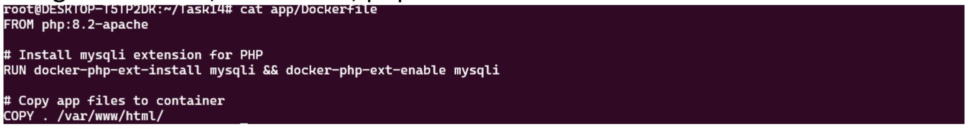
ID	Name	Email
1	Ganesh charawande	rajput@123

---> verified that data get stored in mysql .



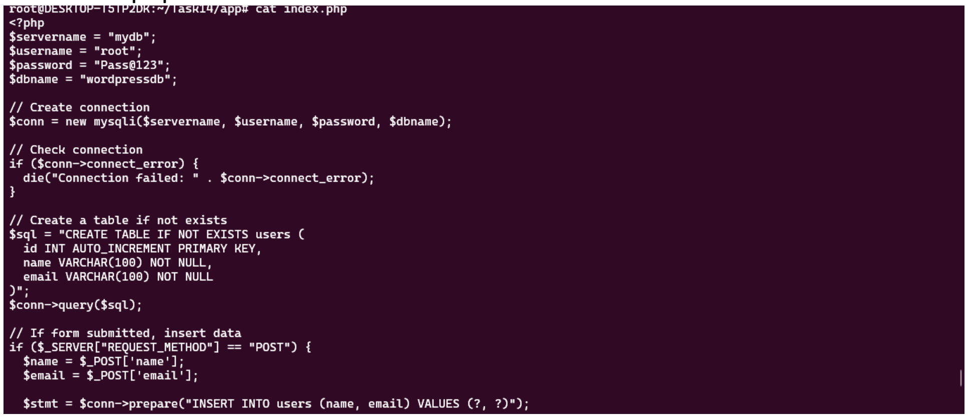
```
mysql> use wordpressdb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----------------------+
| Tables_in_wordpressdb |
+-----------------------+
| users                 |
+-----------------------+
1 row in set (0.00 sec)

mysql> select*from users;
+----+-------------------+------------+
| id | name              | email      |
+----+-------------------+------------+
|  1 | Ganesh charawande | rajput@123 |
+----+-------------------+------------+
1 row in set (0.00 sec)
```

--- nginx conf file , Docker file, php file

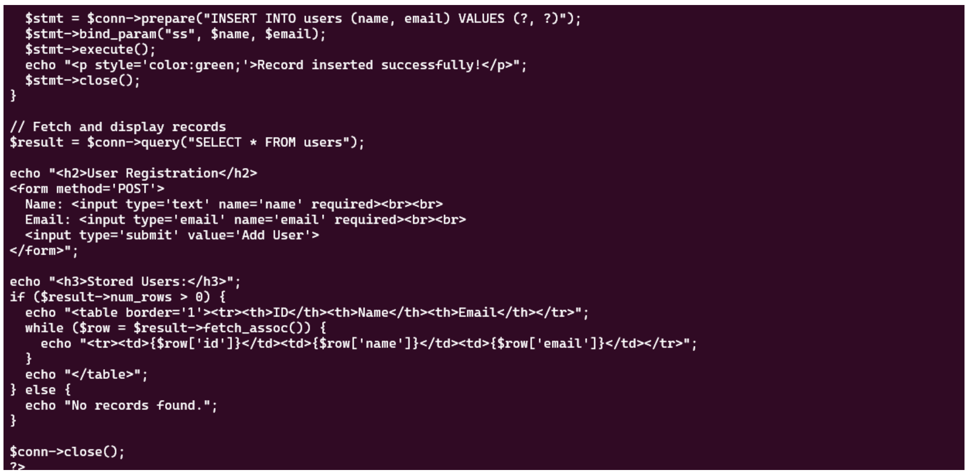


```
root@DESKTOP-T5TP2DK:~/Task14# cat app/Dockerfile
FROM php:8.2-apache

# Install mysqli extension for PHP
RUN docker-php-ext-install mysqli && docker-php-ext-enable mysqli

# Copy app files to container
COPY . /var/www/html/
```

---> index.php file

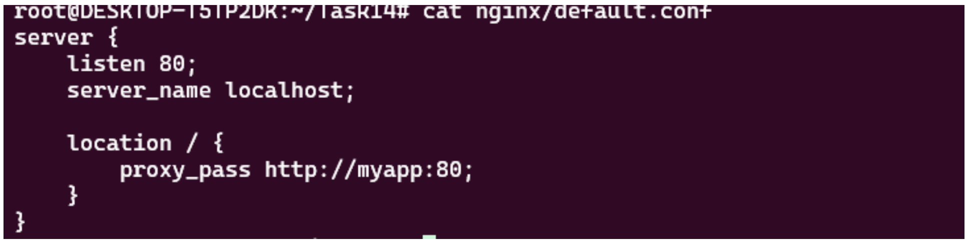


```
root@DESKTOP-T5TP2DK:~/Task14/app# cat index.php
<?php
$servername = "mydb";
$username = "root";
$password = "Pass@123";
$dbname = "wordpressdb";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

// Create a table if not exists
$sql = "CREATE TABLE IF NOT EXISTS users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  email VARCHAR(100) NOT NULL
)";
$conn->query($sql);

// If form submitted, insert data
if ($_SERVER["REQUEST_METHOD"] == "POST") {
  $name = $_POST['name'];
  $email = $_POST['email'];

  $stmt = $conn->prepare("INSERT INTO users (name, email) VALUES (?, ?)");
```



```
  $stmt = $conn->prepare("INSERT INTO users (name, email) VALUES (?, ?)");
  $stmt->bind_param("ss", $name, $email);
  $stmt->execute();
  echo "<p style='color:green;'>Record inserted successfully!</p>";
  $stmt->close();
}

// Fetch and display records
$result = $conn->query("SELECT * FROM users");

echo "<h2>User Registration</h2>
<form method='POST'>
  Name: <input type='text' name='name' required><br><br>
  Email: <input type='email' name='email' required><br><br>
  <input type='submit' value='Add User'>
</form>";

echo "<h3>Stored Users:</h3>";
if ($result->num_rows > 0) {
  echo "<table border='1'><tr><th>ID</th><th>Name</th><th>Email</th></tr>";
  while ($row = $result->fetch_assoc()) {
    echo "<tr><td>{$row['id']}</td><td>{$row['name']}</td><td>{$row['email']}</td></tr>";
  }
  echo "</table>";
} else {
  echo "No records found.";
}

$conn->close();
?>
```

--- default.conf file



```
root@DESKTOP-T5TP2DK:~/Task14# cat nginx/default.conf
server {
    listen 80;
    server_name localhost;

    location / {
        proxy_pass http://myapp:80;
    }
}
```