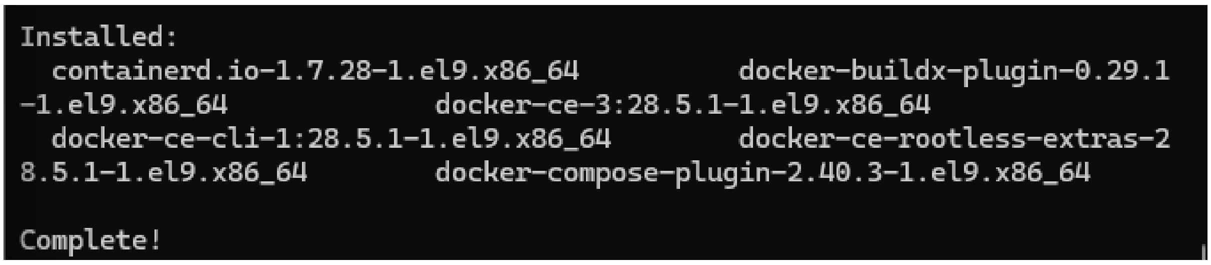
Docker Tasks

1. Docker Installation

Install Docker on your local system (Windows, macOS, or Linux).

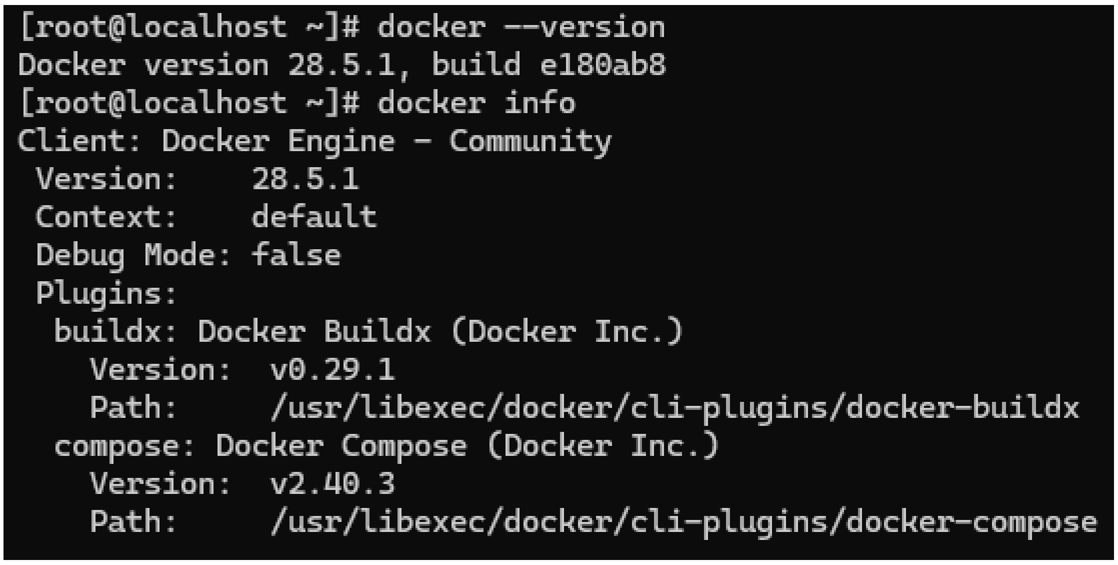


```
Installed:
  containerd.io-1.7.28-1.el9.x86_64          docker-buildx-plugin-0.29.1
-1.el9.x86_64             docker-ce-3:28.5.1-1.el9.x86_64
  docker-ce-cli-1:28.5.1-1.el9.x86_64        docker-ce-rootless-extras-2
8.5.1-1.el9.x86_64        docker-compose-plugin-2.40.3-1.el9.x86_64

Complete!
```

Verify installation by running:

docker --version

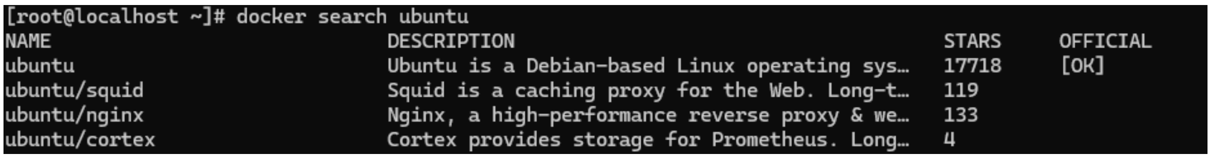
docker info



```
[root@localhost ~]# docker --version
Docker version 28.5.1, build e180ab8
[root@localhost ~]# docker info
Client: Docker Engine - Community
 Version:    28.5.1
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.29.1
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.40.3
    Path:     /usr/libexec/docker/cli-plugins/docker-compose
```

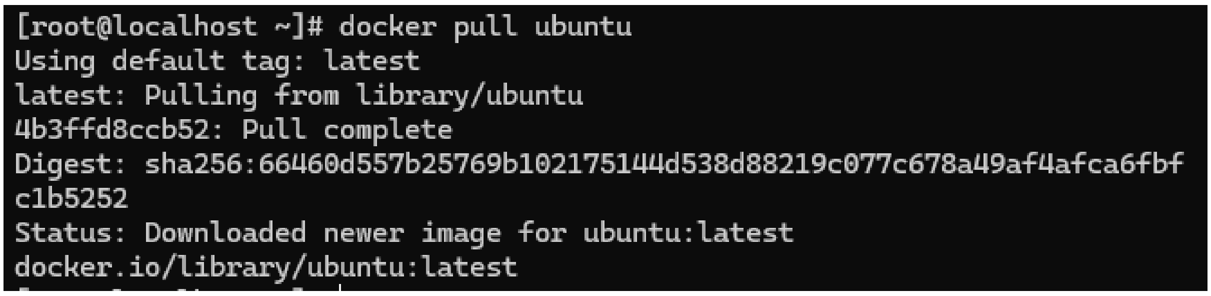
2. Working with Docker Images

Search for the official Ubuntu image on Docker Hub.



```
[root@localhost ~]# docker search ubuntu
NAME                            DESCRIPTION                                     STARS     OFFICIAL
ubuntu                          Ubuntu is a Debian-based Linux operating sys…   17718     [OK]
ubuntu/squid                    Squid is a caching proxy for the Web. Long-t…   119
ubuntu/nginx                    Nginx, a high-performance reverse proxy & we…   133
ubuntu/cortex                   Cortex provides storage for Prometheus. Long…   4
```

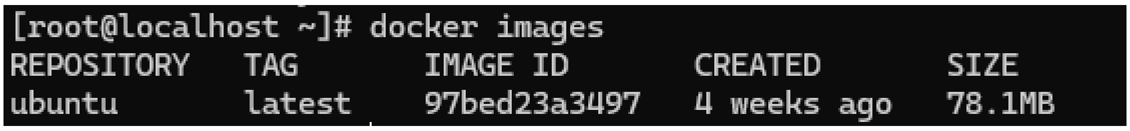
Pull it to your local system using docker pull.



```
[root@localhost ~]# docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
4b3ffd8ccb52: Pull complete
Digest: sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6fbf
c1b5252
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

List all available images using:

docker images



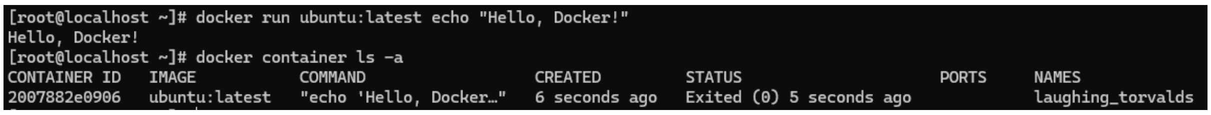
```
[root@localhost ~]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
ubuntu        latest    97bed23a3497   4 weeks ago    78.1MB
```

3. Creating a Docker Container

Run a container using the Ubuntu image.

Execute a command inside the container:

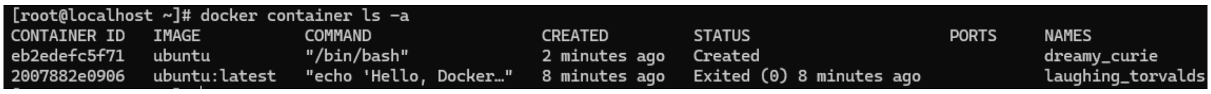
echo "Hello, Docker!"



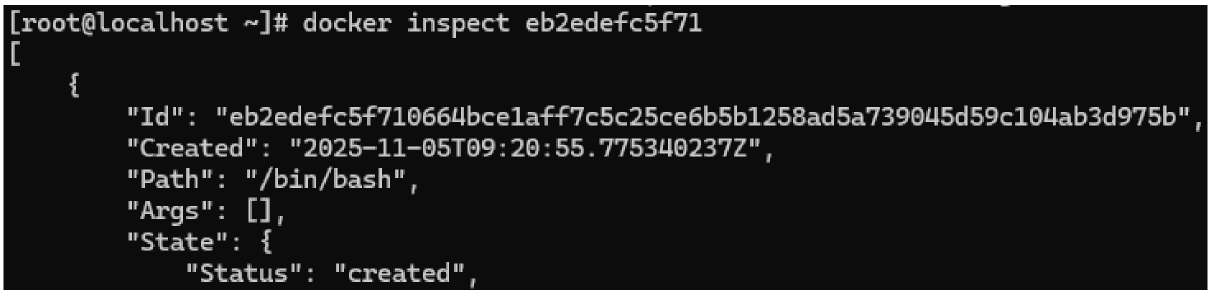
```
[root@localhost ~]# docker run ubuntu:latest echo "Hello, Docker!"
Hello, Docker!
[root@localhost ~]# docker container ls -a
CONTAINER ID   IMAGE           COMMAND                  CREATED         STATUS                     PORTS     NAMES
2007882e0906   ubuntu:latest   "echo 'Hello, Docker…"   6 seconds ago   Exited (0) 5 seconds ago             laughing_torvalds
```

4. Inspect Container (Created State)

Verify that the Ubuntu container is in the "created" state.



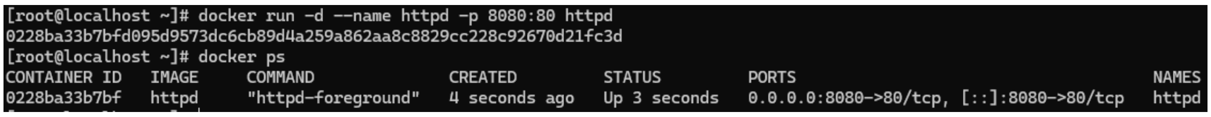
```
[root@localhost ~]# docker container ls -a
CONTAINER ID   IMAGE           COMMAND                  CREATED         STATUS                     PORTS     NAMES
eb2edefc5f71   ubuntu          "/bin/bash"              2 minutes ago   Created                              dreamy_curie
2007882e0906   ubuntu:latest   "echo 'Hello, Docker…"   8 minutes ago   Exited (0) 8 minutes ago             laughing_torvalds
```



```
[root@localhost ~]# docker inspect eb2edefc5f71
[
    {
        "Id": "eb2edefc5f710664bce1aff7c5c25ce6b5b1258ad5a739045d59c104ab3d975b",
        "Created": "2025-11-05T09:20:55.775340237Z",
        "Path": "/bin/bash",
        "Args": [],
        "State": {
            "Status": "created",
```

5. Run Container (Running State)

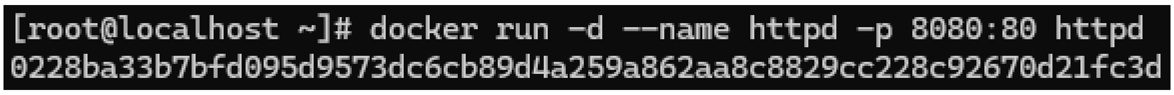
Execute the Ubuntu container and confirm it’s in the "running" state.



```
[root@localhost ~]# docker run -d --name httpd -p 8080:80 httpd
0228ba33b7bfd095d9573dc6cb89d4a259a862aa8c8829cc228c92670d21fc3d
[root@localhost ~]# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED         STATUS         PORTS                                   NAMES
0228ba33b7bf   httpd     "httpd-foreground"   4 seconds ago   Up 3 seconds   0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   httpd
```

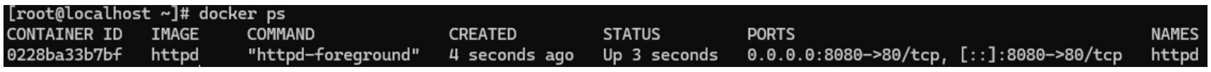
6. Launch and Remove HTTP Container

Launch an HTTP container.



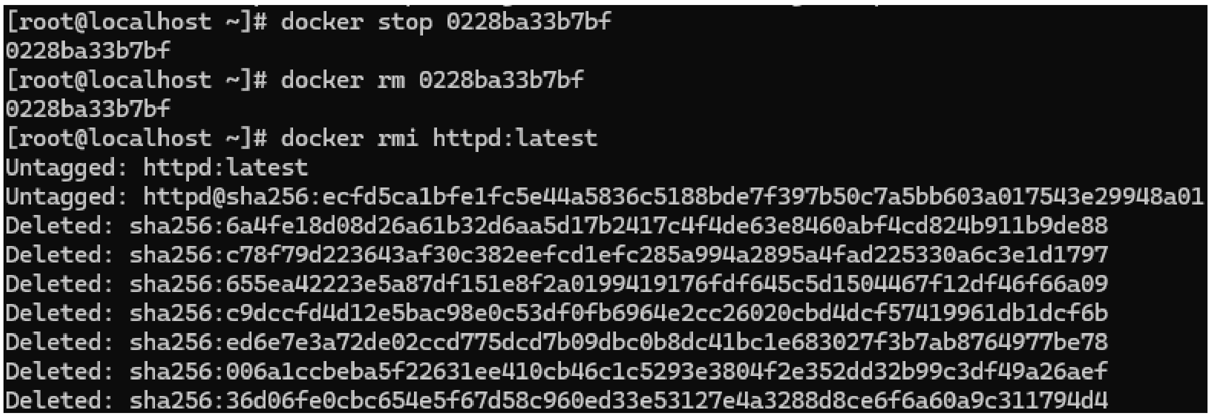
```
[root@localhost ~]# docker run -d --name httpd -p 8080:80 httpd
0228ba33b7bfd095d9573dc6cb89d4a259a862aa8c8829cc228c92670d21fc3d
```

Verify its state.



```
[root@localhost ~]# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED         STATUS         PORTS                                   NAMES
0228ba33b7bf   httpd     "httpd-foreground"   4 seconds ago   Up 3 seconds   0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   httpd
```

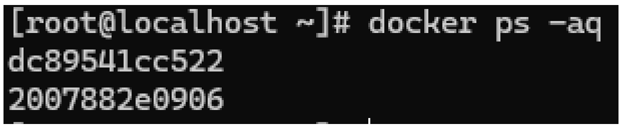
Remove both the container and its image from your system.



```
[root@localhost ~]# docker stop 0228ba33b7bf
0228ba33b7bf
[root@localhost ~]# docker rm 0228ba33b7bf
0228ba33b7bf
[root@localhost ~]# docker rmi httpd:latest
Untagged: httpd:latest
Untagged: httpd@sha256:ecfd5ca1bfe1fc5e44a5836c5188bde7f397b50c7a5bb603a017543e29948a01
Deleted: sha256:6a4fe18d08d26a61b32d6aa5d17b2417c4f4de63e8460abf4cd824b911b9de88
Deleted: sha256:c78f79d223643af30c382eefcd1efc285a994a2895a4fad225330a6c3e1d1797
Deleted: sha256:655ea42223e5a87df151e8f2a0199419176fdf645c5d1504467f12df46f66a09
Deleted: sha256:c9dccfd4d12e5bac98e0c53df0fb6964e2cc26020cbd4dcf57419961db1dcf6b
Deleted: sha256:ed6e7e3a72de02ccd775dcd7b09dbc0b8dc41bc1e683027f3b7ab8764977be78
Deleted: sha256:006a1ccbeba5f22631ee410cb46c1c5293e3804f2e352dd32b99c3df49a26aef
Deleted: sha256:36d06fe0cbc654e5f67d58c960ed33e53127e4a3288d8ce6f6a60a9c311794d4
```

7. List Short Container IDs

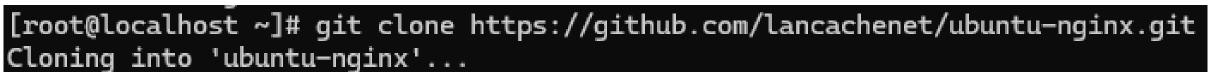
Execute a command that displays only the short container IDs.



```
[root@localhost ~]# docker ps -aq
dc89541cc522
2007882e0906
```

8. Clone & Build Custom Image

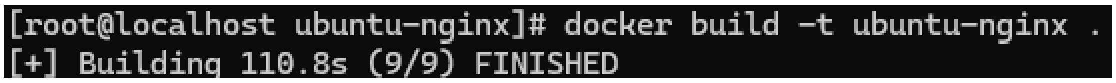
Clone the repository: lancachenet/ubuntu-nginx



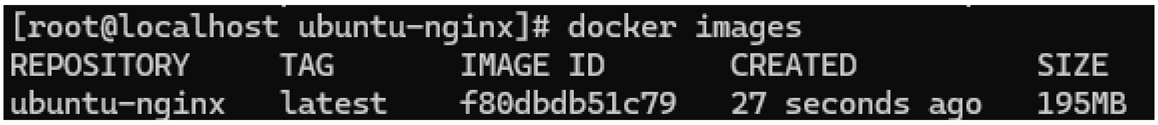
```
[root@localhost ~]# git clone https://github.com/lancachenet/ubuntu-nginx.git
Cloning into 'ubuntu-nginx'...
```

Build a Docker image using the provided Dockerfile.

Add relevant tags while building the image.



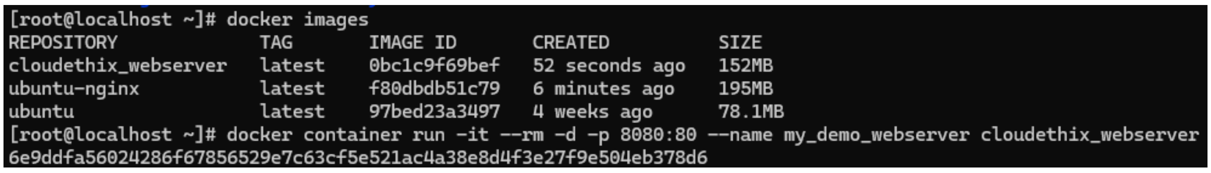
```
[root@localhost ubuntu-nginx]# docker build -t ubuntu-nginx .
[+] Building 110.8s (9/9) FINISHED
```



```
[root@localhost ubuntu-nginx]# docker images
REPOSITORY     TAG       IMAGE ID       CREATED          SIZE
ubuntu-nginx   latest    f80dbdb51c79   27 seconds ago   195MB
```

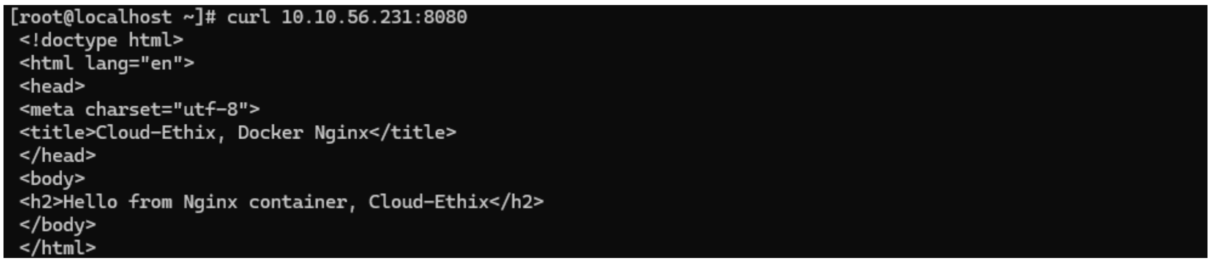
9. Run & Test the Webpage

Run a Docker container using the newly built image.



```
[root@localhost ~]# docker images
REPOSITORY            TAG       IMAGE ID       CREATED          SIZE
cloudethix_webserver  latest    0bc1c9f69bef   52 seconds ago   152MB
ubuntu-nginx          latest    f80dbdb51c79   6 minutes ago    195MB
ubuntu                latest    97bed23a3497   4 weeks ago      78.1MB
[root@localhost ~]# docker container run -it --rm -d -p 8080:80 --name my_demo_webserver cloudethix_webserver
6e9ddfa56024286f67856529e7c63cf5e521ac4a38e8d4f3e27f9e504eb378d6
```

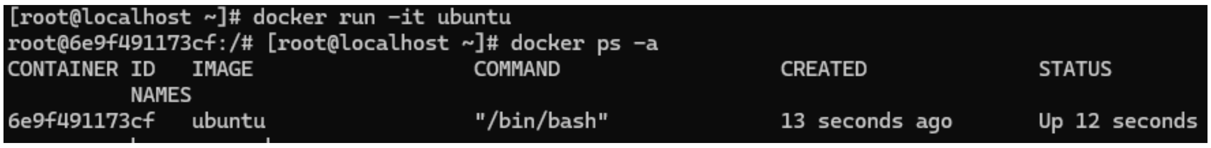
Use the curl command to verify that the webpage is accessible.



```
[root@localhost ~]# curl 10.10.56.231:8080
 <!doctype html>
 <html lang="en">
 <head>
 <meta charset="utf-8">
 <title>Cloud-Ethix, Docker Nginx</title>
 </head>
 <body>
 <h2>Hello from Nginx container, Cloud-Ethix</h2>
 </body>
 </html>
```

10. Container Detach & Reattach

Launch an Ubuntu container.



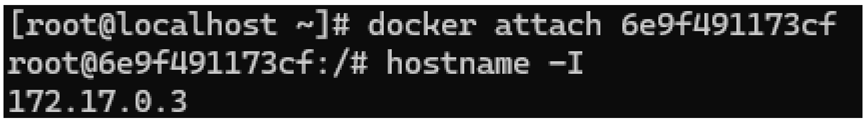
```
[root@localhost ~]# docker run -it ubuntu
root@6e9f491173cf:/# [root@localhost ~]# docker ps -a
CONTAINER ID   IMAGE                 COMMAND                  CREATED             STATUS
          NAMES
6e9f491173cf   ubuntu                "/bin/bash"              13 seconds ago      Up 12 seconds
```

Demonstrate the escape sequence (detach).

While inside the container, press the escape sequence: Ctrl +p Ctrl +q

Reattach the container after detaching.

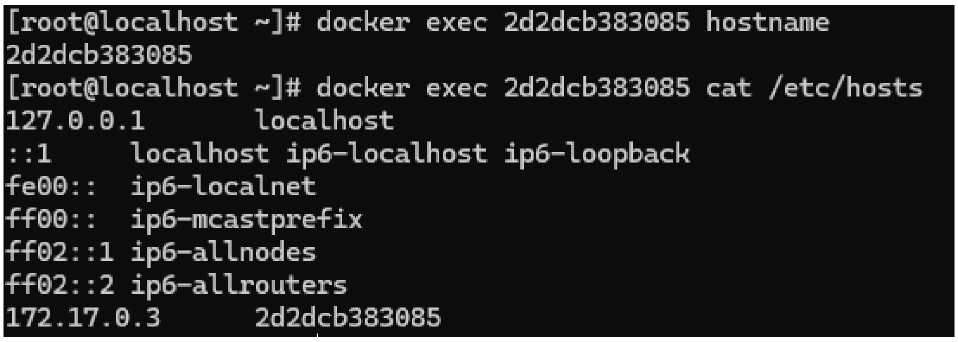
Check and note down the container's IP address.



```
[root@localhost ~]# docker attach 6e9f491173cf
root@6e9f491173cf:/# hostname -I
172.17.0.3
```

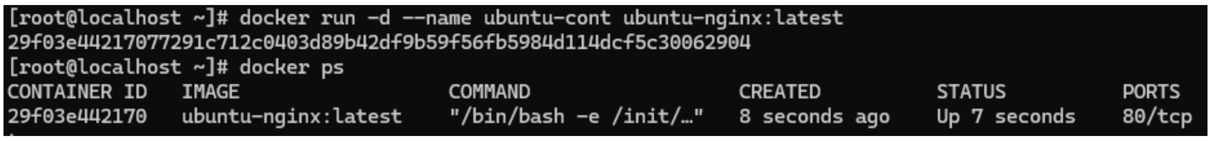
12. Inspect Host Configuration

Use a Docker command to inspect the hostname and the /etc/hosts file of the httpd container.



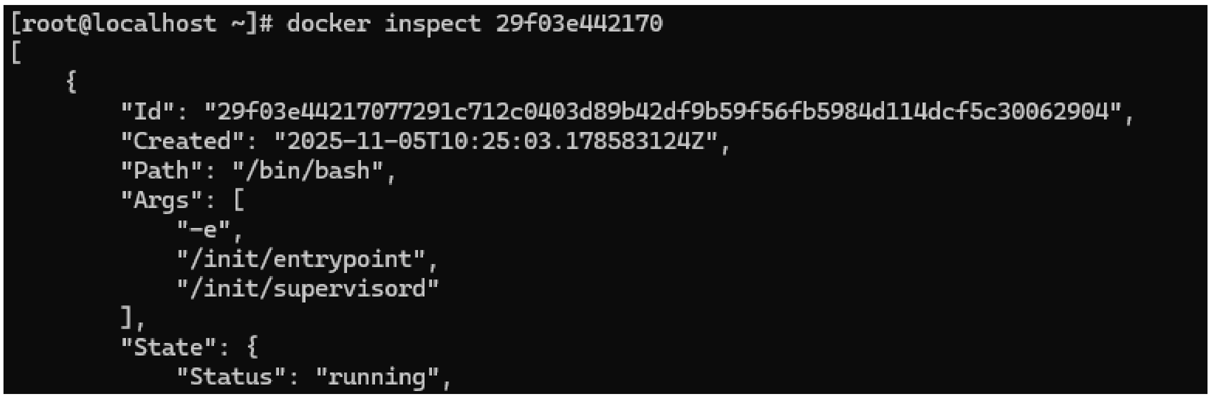
```
[root@localhost ~]# docker exec 2d2dcb383085 hostname
2d2dcb383085
[root@localhost ~]# docker exec 2d2dcb383085 cat /etc/hosts
127.0.0.1       localhost
::1     localhost ip6-localhost ip6-loopback
fe00::  ip6-localnet
ff00::  ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
172.17.0.3      2d2dcb383085
```

13. Monitor Container Stats



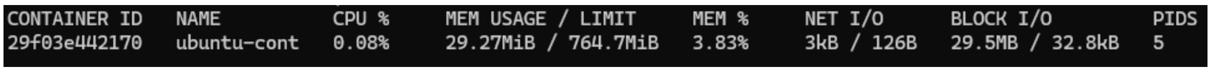
```
[root@localhost ~]# docker run -d --name ubuntu-cont ubuntu-nginx:latest
29f03e44217077291c712c0403d89b42df9b59f56fb5984d114dcf5c30062904
[root@localhost ~]# docker ps
CONTAINER ID   IMAGE                 COMMAND                  CREATED          STATUS         PORTS
29f03e442170   ubuntu-nginx:latest   "/bin/bash -e /init/…"   8 seconds ago    Up 7 seconds   80/tcp
```

Inspect a container created with the lancachenet/ubuntu-nginx image.

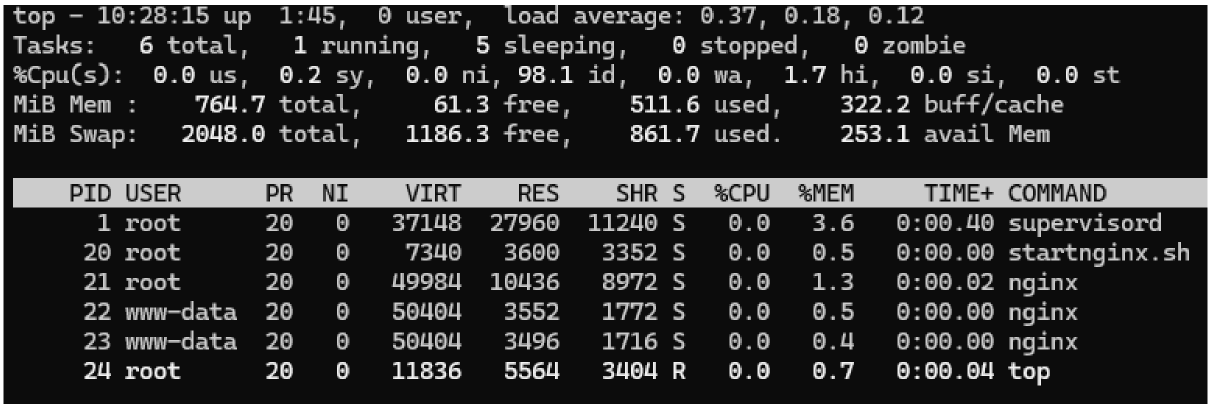


```
[root@localhost ~]# docker inspect 29f03e442170
[
    {
        "Id": "29f03e44217077291c712c0403d89b42df9b59f56fb5984d114dcf5c30062904",
        "Created": "2025-11-05T10:25:03.178583124Z",
        "Path": "/bin/bash",
        "Args": [
            "-e",
            "/init/entrypoint",
            "/init/supervisord"
        ],
        "State": {
            "Status": "running",
```

Monitor its resource usage and stats using the top command.



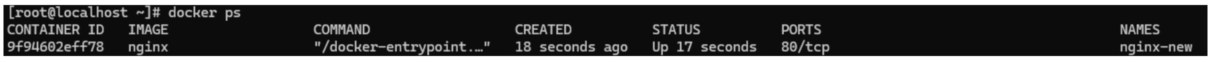
```
CONTAINER ID   NAME          CPU %     MEM USAGE / LIMIT     MEM %     NET I/O       BLOCK I/O         PIDS
29f03e442170   ubuntu-cont   0.08%     29.27MiB / 764.7MiB   3.83%     3kB / 126B    29.5MB / 32.8kB   5
```



```
top - 10:28:15 up  1:45,  0 user,  load average: 0.37, 0.18, 0.12
Tasks:   6 total,   1 running,   5 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.2 sy,  0.0 ni, 98.1 id,  0.0 wa,  1.7 hi,  0.0 si,  0.0 st
MiB Mem :    764.7 total,     61.3 free,    511.6 used,    322.2 buff/cache
MiB Swap:   2048.0 total,   1186.3 free,    861.7 used.    253.1 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
      1 root      20   0   37148  27960  11240 S   0.0   3.6   0:00.40 supervisord
     20 root      20   0    7340   3600   3352 S   0.0   0.5   0:00.00 startnginx.sh
     21 root      20   0   49984  10436   8972 S   0.0   1.3   0:00.02 nginx
     22 www-data  20   0   50404   3552   1772 S   0.0   0.5   0:00.00 nginx
     23 www-data  20   0   50404   3496   1716 S   0.0   0.4   0:00.00 nginx
     24 root      20   0   11836   5564   3404 R   0.0   0.7   0:00.04 top
```

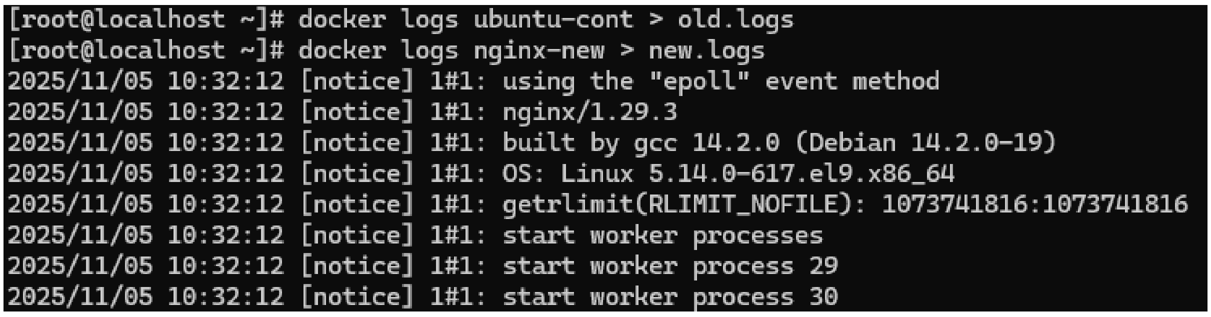
14. Nginx Containers Log Review

Start a new Nginx container.

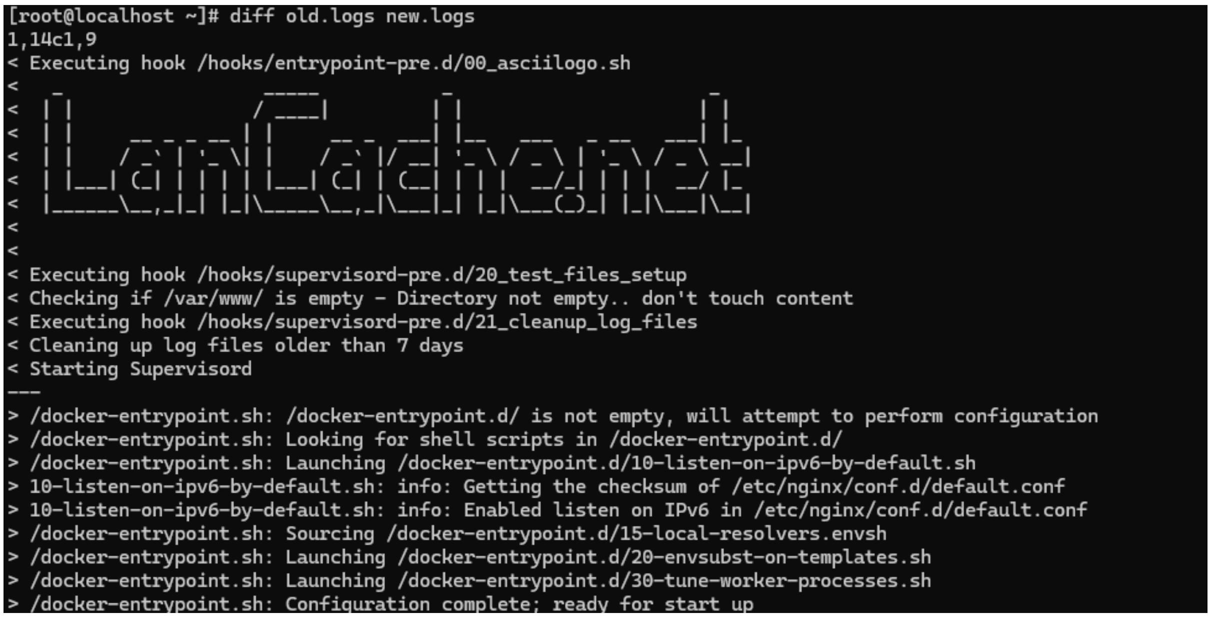


```
[root@localhost ~]# docker ps
CONTAINER ID   IMAGE                         COMMAND                  CREATED          STATUS          PORTS                                        NAMES
9f94602eff78   nginx                         "/docker-entrypoint.…"   18 seconds ago   Up 17 seconds   80/tcp                                       nginx-new
```

Review and compare the logs for both Nginx containers.



```
[root@localhost ~]# docker logs ubuntu-cont > old.logs
[root@localhost ~]# docker logs nginx-new > new.logs
2025/11/05 10:32:12 [notice] 1#1: using the "epoll" event method
2025/11/05 10:32:12 [notice] 1#1: nginx/1.29.3
2025/11/05 10:32:12 [notice] 1#1: built by gcc 14.2.0 (Debian 14.2.0-19)
2025/11/05 10:32:12 [notice] 1#1: OS: Linux 5.14.0-617.el9.x86_64
2025/11/05 10:32:12 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1073741816:1073741816
2025/11/05 10:32:12 [notice] 1#1: start worker processes
2025/11/05 10:32:12 [notice] 1#1: start worker process 29
2025/11/05 10:32:12 [notice] 1#1: start worker process 30
```

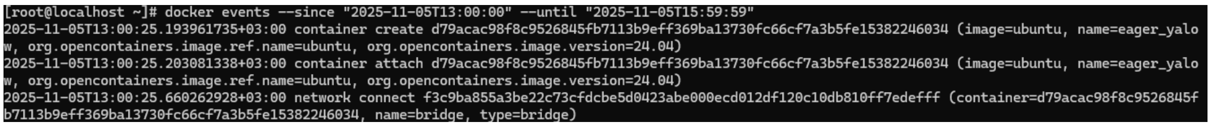


```
[root@localhost ~]# diff old.logs new.logs
1,14c1,9
< Executing hook /hooks/entrypoint-pre.d/00_asciilogo.sh
<   _                 _____           _                 _
<  | |               / ____|         | |               | |
<  | |     __ _ _ __ | |     __ _  ___| |__   ___   _ __   ___| |_
<  | |    / _` | '_ \| |    / _` |/ __| '_ \ / _ \ | '_ \ / _ \ __|
<  | |___| (_| | | | | |___| (_| | (__| | | |  __/_| | | |  __/ |_
<  |______\__,_|_| |_|\_____\__,_|\___|_| |_|\___(_)_| |_|\___|\__|
<
<
< Executing hook /hooks/supervisord-pre.d/20_test_files_setup
< Checking if /var/www/ is empty - Directory not empty.. don't touch content
< Executing hook /hooks/supervisord-pre.d/21_cleanup_log_files
< Cleaning up log files older than 7 days
< Starting Supervisord
---
> /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
```

15. System Events & Filters

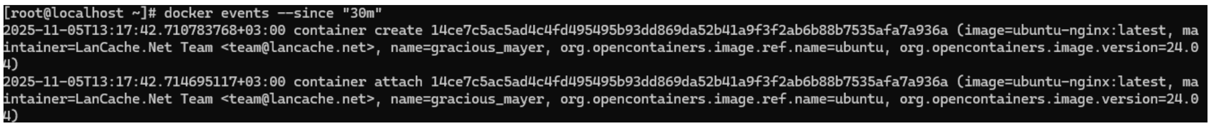
Examine Docker system events filtered by:

Specific date, and



```
[root@localhost ~]# docker events --since "2025-11-05T13:00:00" --until "2025-11-05T15:59:59"
2025-11-05T13:00:25.193961735+03:00 container create d79acac98f8c9526845fb7113b9eff369ba13730fc66cf7a3b5fe15382246034 (image=ubuntu, name=eager_yalo
w, org.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.04)
2025-11-05T13:00:25.203081338+03:00 container attach d79acac98f8c9526845fb7113b9eff369ba13730fc66cf7a3b5fe15382246034 (image=ubuntu, name=eager_yalo
w, org.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.04)
2025-11-05T13:00:25.660262928+03:00 network connect f3c9ba855a3be22c73cfdcbe5d0423abe000ecd012df120c10db810ff7edefff (container=d79acac98f8c9526845f
b7113b9eff369ba13730fc66cf7a3b5fe15382246034, name=bridge, type=bridge)
```

The last 30 minutes.

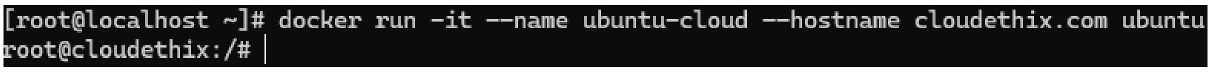


```
[root@localhost ~]# docker events --since "30m"
2025-11-05T13:17:42.710783768+03:00 container create 14ce7c5ac5ad4c4fd495495b93dd869da52b41a9f3f2ab6b88b7535afa7a936a (image=ubuntu-nginx:latest, ma
intainer=LanCache.Net Team <team@lancache.net>, name=gracious_mayer, org.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.0
4)
2025-11-05T13:17:42.714695117+03:00 container attach 14ce7c5ac5ad4c4fd495495b93dd869da52b41a9f3f2ab6b88b7535afa7a936a (image=ubuntu-nginx:latest, ma
intainer=LanCache.Net Team <team@lancache.net>, name=gracious_mayer, org.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.0
4)
```

Apply two filters simultaneously using name and event.

16. Custom Hostname & Logs

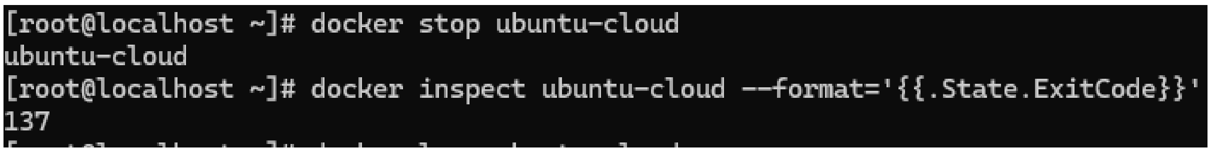
Create an Ubuntu container with: A meaningful name, and Hostname set to cloudethix.com.



```
[root@localhost ~]# docker run -it --name ubuntu-cloud --hostname cloudethix.com ubuntu
root@cloudethix:/# |
```

Stop or kill the container.

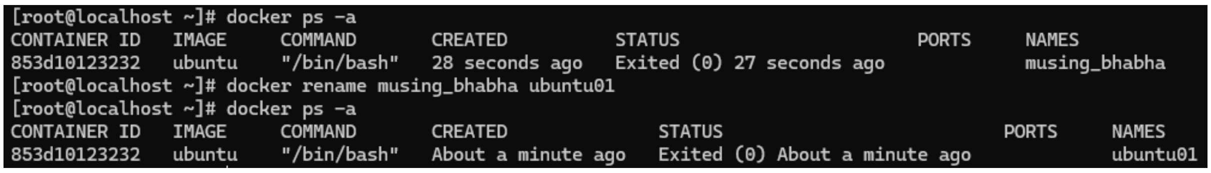
Check its exit code and review its logs for details.



```
[root@localhost ~]# docker stop ubuntu-cloud
ubuntu-cloud
[root@localhost ~]# docker inspect ubuntu-cloud --format='{{.State.ExitCode}}'
137
```

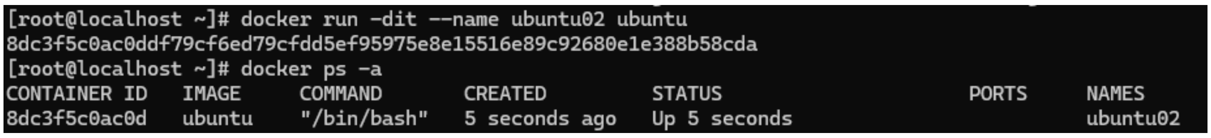
17. Container Management Operations

Create and start an Ubuntu container with a random name, then rename it to Ubuntu01.



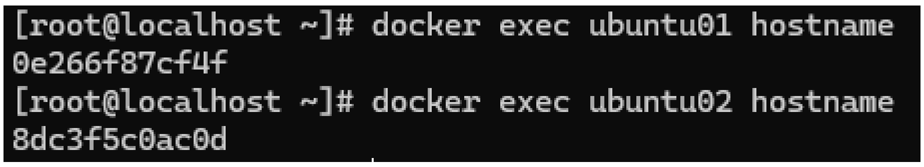
```
[root@localhost ~]# docker ps -a
CONTAINER ID   IMAGE     COMMAND       CREATED          STATUS                      PORTS     NAMES
853d10123232   ubuntu    "/bin/bash"   28 seconds ago   Exited (0) 27 seconds ago             musing_bhabha
[root@localhost ~]# docker rename musing_bhabha ubuntu01
[root@localhost ~]# docker ps -a
CONTAINER ID   IMAGE     COMMAND       CREATED              STATUS                          PORTS     NAMES
853d10123232   ubuntu    "/bin/bash"   About a minute ago   Exited (0) About a minute ago             ubuntu01
```

Launch another container named Ubuntu02.



```
[root@localhost ~]# docker run -dit --name ubuntu02 ubuntu
8dc3f5c0ac0ddf79cf6ed79cfdd5ef95975e8e15516e89c92680e1e388b58cda
[root@localhost ~]# docker ps -a
CONTAINER ID   IMAGE     COMMAND       CREATED         STATUS         PORTS     NAMES
8dc3f5c0ac0d   ubuntu    "/bin/bash"   5 seconds ago   Up 5 seconds             ubuntu02
```

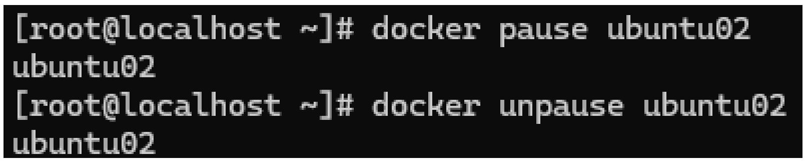
Check the hostname for both containers.



```
[root@localhost ~]# docker exec ubuntu01 hostname
0e266f87cf4f
[root@localhost ~]# docker exec ubuntu02 hostname
8dc3f5c0ac0d
```

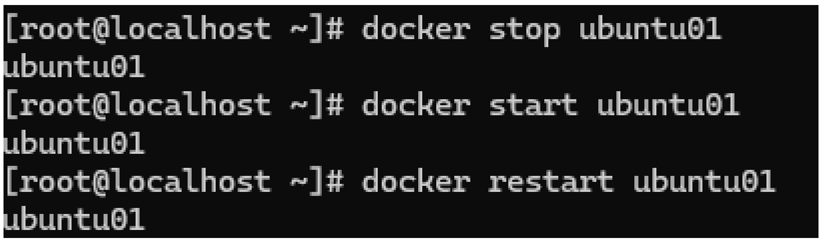
Perform the following:

Pause and unpause Ubuntu02.



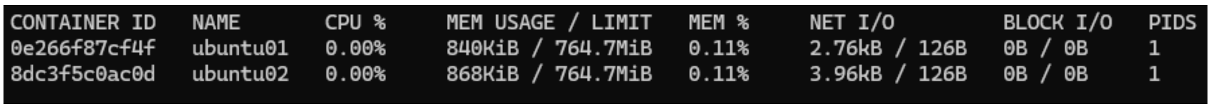
```
[root@localhost ~]# docker pause ubuntu02
ubuntu02
[root@localhost ~]# docker unpause ubuntu02
ubuntu02
```

Stop, start, and restart Ubuntu01.

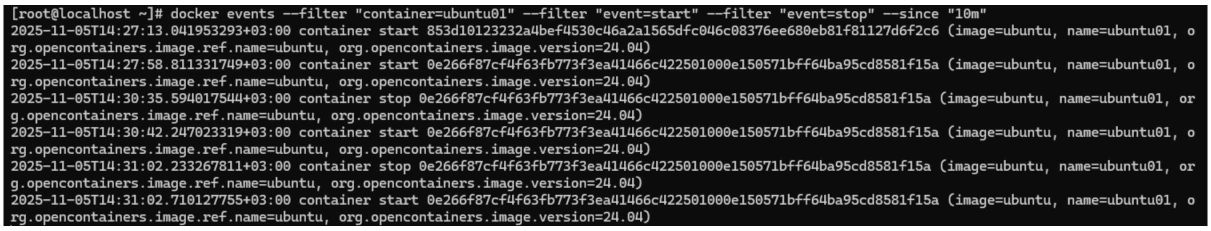


```
[root@localhost ~]# docker stop ubuntu01
ubuntu01
[root@localhost ~]# docker start ubuntu01
ubuntu01
[root@localhost ~]# docker restart ubuntu01
ubuntu01
```

Inspect stats and system events for both.

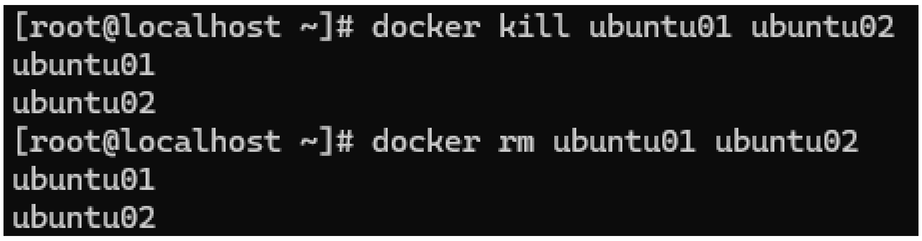


```
CONTAINER ID   NAME       CPU %     MEM USAGE / LIMIT     MEM %     NET I/O          BLOCK I/O   PIDS
0e266f87cf4f   ubuntu01   0.00%     840KiB / 764.7MiB     0.11%     2.76kB / 126B    0B / 0B     1
8dc3f5c0ac0d   ubuntu02   0.00%     868KiB / 764.7MiB     0.11%     3.96kB / 126B    0B / 0B     1
```



```
[root@localhost ~]# docker events --filter "container=ubuntu01" --filter "event=start" --filter "event=stop" --since "10m"
2025-11-05T14:27:13.041953293+03:00 container start 853d10123232a4bef4530c46a2a1565dfc046c08376ee680eb81f81127d6f2c6 (image=ubuntu, name=ubuntu01, o
rg.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.04)
2025-11-05T14:27:58.811331749+03:00 container start 0e266f87cf4f63fb773f3ea41466c422501000e150571bff64ba95cd8581f15a (image=ubuntu, name=ubuntu01, o
rg.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.04)
2025-11-05T14:30:35.594017544+03:00 container stop 0e266f87cf4f63fb773f3ea41466c422501000e150571bff64ba95cd8581f15a (image=ubuntu, name=ubuntu01, or
g.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.04)
2025-11-05T14:30:42.247023319+03:00 container start 0e266f87cf4f63fb773f3ea41466c422501000e150571bff64ba95cd8581f15a (image=ubuntu, name=ubuntu01, o
rg.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.04)
2025-11-05T14:31:02.233267811+03:00 container stop 0e266f87cf4f63fb773f3ea41466c422501000e150571bff64ba95cd8581f15a (image=ubuntu, name=ubuntu01, or
g.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.04)
2025-11-05T14:31:02.710127755+03:00 container start 0e266f87cf4f63fb773f3ea41466c422501000e150571bff64ba95cd8581f15a (image=ubuntu, name=ubuntu01, o
rg.opencontainers.image.ref.name=ubuntu, org.opencontainers.image.version=24.04)
```

Finally, kill and delete the containers, ensuring all created directories are removed.

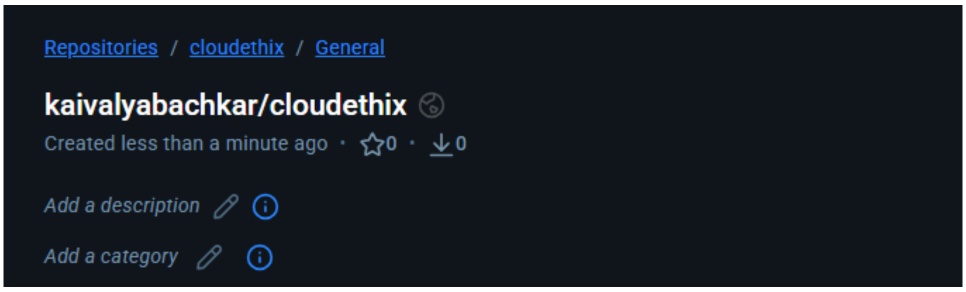


```
[root@localhost ~]# docker kill ubuntu01 ubuntu02
ubuntu01
ubuntu02
[root@localhost ~]# docker rm ubuntu01 ubuntu02
ubuntu01
ubuntu02
```

18. DockerHub Repository Setup

Create a DockerHub account.

Set up a repository with a meaningful name.


Repositories / cloudethix / General

kaivalyabachkar/cloudethix

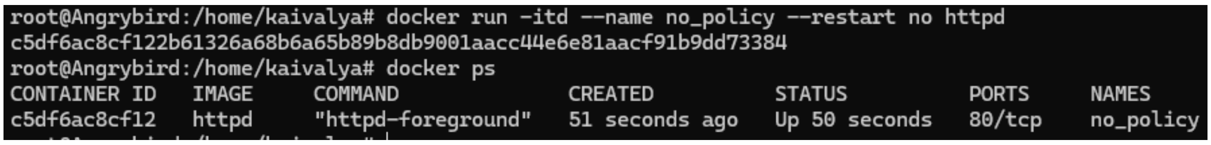
Created less than a minute ago · ☆0 · ↓0

Add a description

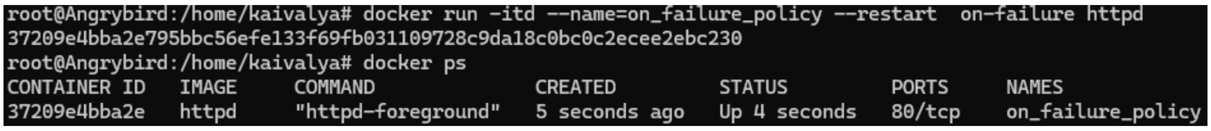
Add a category

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.

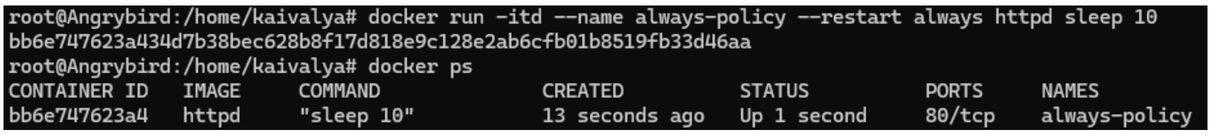
No_Policy:-

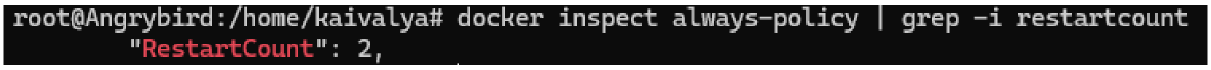

```
root@Angrybird:/home/kaivalya# docker run -itd --name no_policy --restart no httpd
c5df6ac8cf122b61326a68b6a65b89b8db9001aacc44e6e81aacf91b9dd73384
root@Angrybird:/home/kaivalya# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED          STATUS          PORTS     NAMES
c5df6ac8cf12   httpd     "httpd-foreground"   51 seconds ago   Up 50 seconds   80/tcp    no_policy
```

On-Failure:-

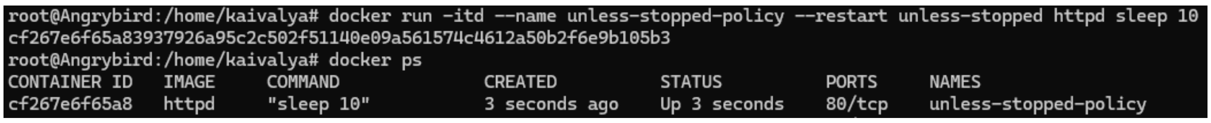

```
root@Angrybird:/home/kaivalya# docker run -itd --name=on_failure_policy --restart  on-failure httpd
37209e4bba2e795bbc56efe133f69fb031109728c9da18c0bc0c2ecee2ebc230
root@Angrybird:/home/kaivalya# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED         STATUS         PORTS     NAMES
37209e4bba2e   httpd     "httpd-foreground"   5 seconds ago   Up 4 seconds   80/tcp    on_failure_policy
```

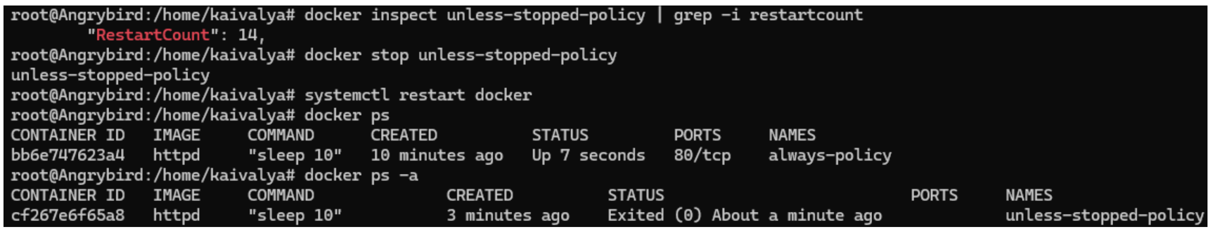
Restart-Always:-


```
root@Angrybird:/home/kaivalya# docker run -itd --name always-policy --restart always httpd sleep 10
bb6e747623a434d7b38bec628b8f17d818e9c128e2ab6cfb01b8519fb33d46aa
root@Angrybird:/home/kaivalya# docker ps
CONTAINER ID   IMAGE     COMMAND             CREATED          STATUS         PORTS     NAMES
bb6e747623a4   httpd     "sleep 10"          13 seconds ago   Up 1 second    80/tcp    always-policy
```


```
root@Angrybird:/home/kaivalya# docker inspect always-policy | grep -i restartcount
            "RestartCount": 2,
```

Unless-Stopped:-


```
root@Angrybird:/home/kaivalya# docker run -itd --name unless-stopped-policy --restart unless-stopped httpd sleep 10
cf267e6f65a83937926a95c2c502f51140e09a561574c4612a50b2f6e9b105b3
root@Angrybird:/home/kaivalya# docker ps
CONTAINER ID   IMAGE     COMMAND             CREATED         STATUS         PORTS     NAMES
cf267e6f65a8   httpd     "sleep 10"          3 seconds ago   Up 3 seconds   80/tcp    unless-stopped-policy
```


```
root@Angrybird:/home/kaivalya# docker inspect unless-stopped-policy | grep -i restartcount
            "RestartCount": 14,
root@Angrybird:/home/kaivalya# docker stop unless-stopped-policy
unless-stopped-policy
root@Angrybird:/home/kaivalya# systemctl restart docker
root@Angrybird:/home/kaivalya# docker ps
CONTAINER ID   IMAGE     COMMAND      CREATED          STATUS         PORTS     NAMES
bb6e747623a4   httpd     "sleep 10"   10 minutes ago   Up 7 seconds   80/tcp    always-policy
root@Angrybird:/home/kaivalya# docker ps -a
CONTAINER ID   IMAGE     COMMAND             CREATED         STATUS                          PORTS     NAMES
cf267e6f65a8   httpd     "sleep 10"          3 minutes ago   Exited (0) About a minute ago             unless-stopped-policy
```

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

e.g. docker update --help

```
root@Angrybird:/home/kaivalya# docker update --restart no unless-stopped-policy
unless-stopped-policy
root@Angrybird:/home/kaivalya# docker update --restart on-failure always-policy
always-policy
root@Angrybird:/home/kaivalya# docker update --restart always on_failure_policy
on_failure_policy
root@Angrybird:/home/kaivalya# docker update --restart unless-stopped no_policy
no_policy
```

```
root@Angrybird:/home/kaivalya# docker inspect --format '{{.HostConfig.RestartPolicy.Name}}' no_policy
unless-stopped
root@Angrybird:/home/kaivalya# docker inspect --format '{{.HostConfig.RestartPolicy.Name}}' on_failure_policy
always
root@Angrybird:/home/kaivalya# docker inspect --format '{{.HostConfig.RestartPolicy.Name}}' always-policy
on-failure
root@Angrybird:/home/kaivalya# docker inspect --format '{{.HostConfig.RestartPolicy.Name}}' unless-stopped-policy
no
```

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.

NGINX container with a cloudethix and expose it on the host's port 8080:-

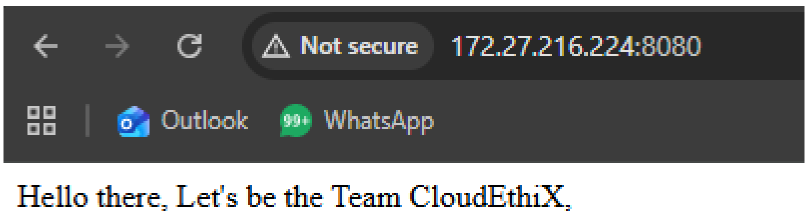
```
root@Angrybird:/home/kaivalya# docker run -itd --name cloudethix -p 8080:80 nginx
628ba6607e1d30944102391fd4896e67c076deed70a1a9828f307762f56f6e23
```

Created a index.html file and copied the file cloudethix Container:-

```
root@Angrybird:/home/kaivalya# echo "Hello there, Let's be the Team CloudEthiX," > /home/kaivalya/index.html
root@Angrybird:/home/kaivalya# docker cp /home/kaivalya/index.html cloudethix:/usr/share/nginx/html/
Successfully copied 2.05kB to cloudethix:/usr/share/nginx/html/
```

Acces the container in a Browser to verify that the webpage displays correctly:-

```
root@Angrybird:/home/kaivalya# ip r l
default via 172.27.208.1 dev eth0 proto kernel
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
172.27.208.0/20 dev eth0 proto kernel scope link src 172.27.216.224
```



Hello there, Let's be the Team CloudEthiX,

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

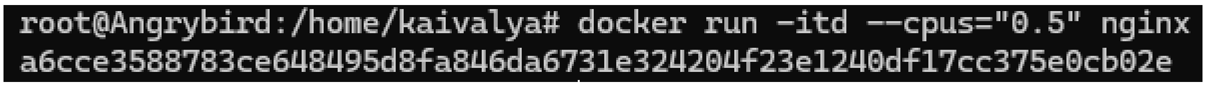
docker container run --help

ref links :-

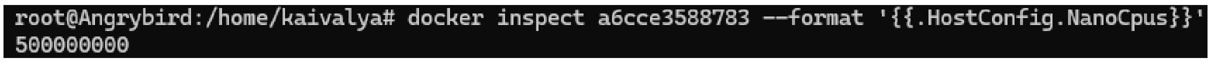
https://phoenixnap.com/kb/docker-memory-and-cpu-limit

https://www.baeldung.com/ops/docker-memory-limit

Limiting CPU Usage:

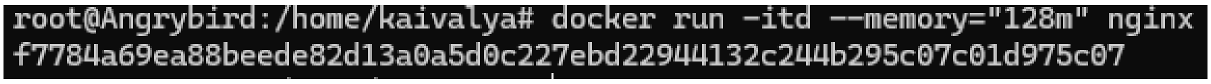


```
root@Angrybird:/home/kaivalya# docker run -itd --cpus="0.5" nginx
a6cce3588783ce648495d8fa846da6731e324204f23e1240df17cc375e0cb02e
```

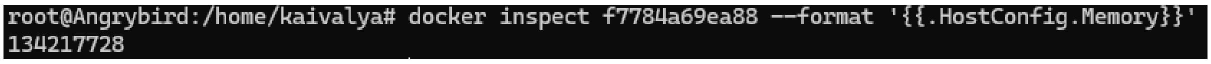


```
root@Angrybird:/home/kaivalya# docker inspect a6cce3588783 --format '{{.HostConfig.NanoCpus}}'
500000000
```

Limiting Memory Usage:

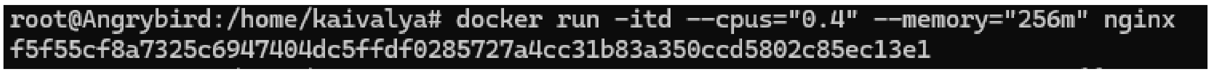


```
root@Angrybird:/home/kaivalya# docker run -itd --memory="128m" nginx
f7784a69ea88beede82d13a0a5d0c227ebd22944132c244b295c07c01d975c07
```

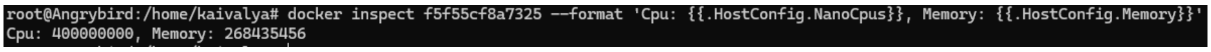


```
root@Angrybird:/home/kaivalya# docker inspect f7784a69ea88 --format '{{.HostConfig.Memory}}'
134217728
```

Combining CPU and Memory Limits:



```
root@Angrybird:/home/kaivalya# docker run -itd --cpus="0.4" --memory="256m" nginx
f5f55cf8a7325c6947404dc5ffdf0285727a4cc31b83a350ccd5802c85ec13e1
```



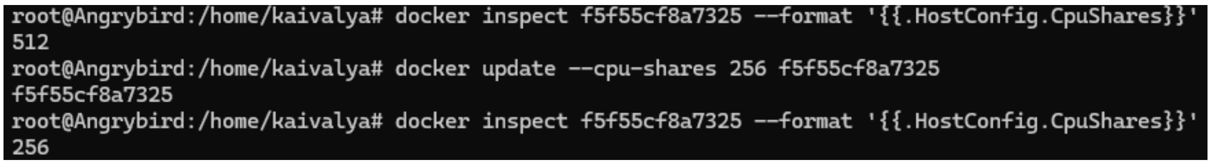
```
root@Angrybird:/home/kaivalya# docker inspect f5f55cf8a7325 --format 'Cpu: {{.HostConfig.NanoCpus}}, Memory: {{.HostConfig.Memory}}'
Cpu: 400000000, Memory: 268435456
```

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

ref links :-

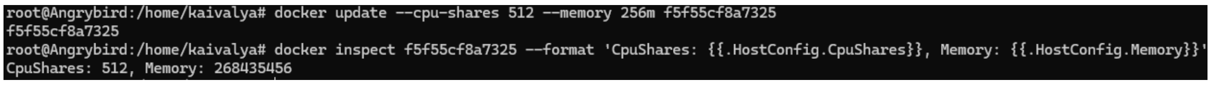
https://docs.docker.com/engine/reference/commandline/update/

Update a container's cpu-shares (--cpu-shares):-



```
root@Angrybird:/home/kaivalya# docker inspect f5f55cf8a7325 --format '{{.HostConfig.CpuShares}}'
512
root@Angrybird:/home/kaivalya# docker update --cpu-shares 256 f5f55cf8a7325
f5f55cf8a7325
root@Angrybird:/home/kaivalya# docker inspect f5f55cf8a7325 --format '{{.HostConfig.CpuShares}}'
256
```

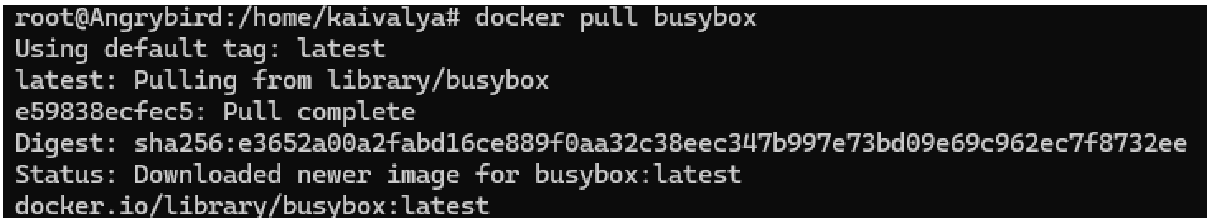
Update a container with cpu-shares and memory (-m, --memory):-



```
root@Angrybird:/home/kaivalya# docker update --cpu-shares 512 --memory 256m f5f55cf8a7325
f5f55cf8a7325
root@Angrybird:/home/kaivalya# docker inspect f5f55cf8a7325 --format 'CpuShares: {{.HostConfig.CpuShares}}, Memory: {{.HostConfig.Memory}}'
CpuShares: 512, Memory: 268435456
```

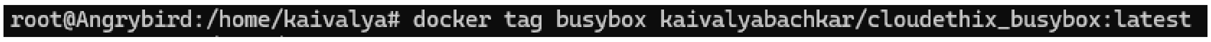
24. Pull the Busy-box image to your local system, tag it, and push it to the Docker Hub repository "yourname_cloudethix_busybox."

Pull the BusyBox image:-



```
root@Angrybird:/home/kaivalya# 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
```

Tag the image:-



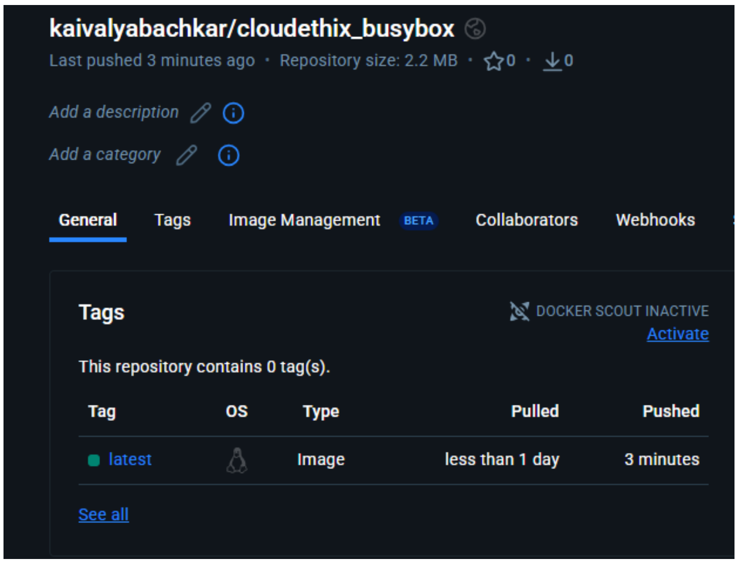
```
root@Angrybird:/home/kaivalya# docker tag busybox kaivalyabachkar/cloudethix_busybox:latest
```

Log in to Docker Hub:-

```
root@Angrybird:/home/kaivalya# docker login -u kaivalyabachkar
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

Login Succeeded
```

Push the tagged image to Docker Hub:-

```
root@Angrybird:/home/kaivalya# docker push kaivalyabachkar/cloudethix_busybox:latest
The push refers to repository [docker.io/kaivalyabachkar/cloudethix_busybox]
e14542cc0629: Pushed
latest: digest: sha256:be49435f6288f9c5cce0357c2006cc266cb5c450dbd6dc8e3a3baec10c46b065 size: 527
```



kaivalyabachkar/cloudethix_busybox

Last pushed 3 minutes ago · Repository size: 2.2 MB · 0 · 0

Add a description

Add a category

General Tags Image Management BETA Collaborators Webhooks

Tags

DOCKER SCOUT INACTIVE Activate

This repository contains 0 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	less than 1 day	3 minutes

See all

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.

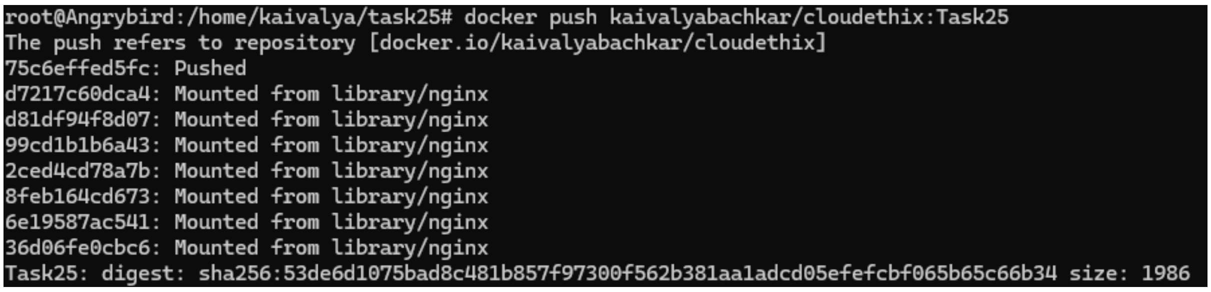
Ensure files in the same directory:-

```
root@Angrybird:/home/kaivalya/task25# ls
Dockerfile  custom-index.html
```

Build the Docker image using the Dockerfile:-

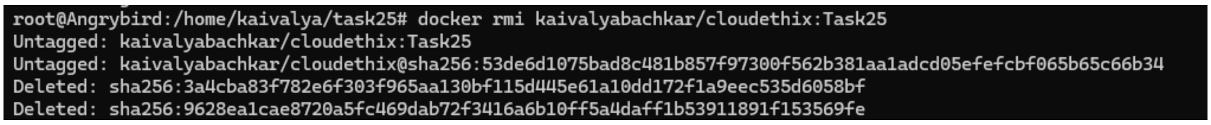
```
root@Angrybird:/home/kaivalya/task25# docker build -t kaivalyabachkar/cloudethix:Task25
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.072kB
Step 1/2 : FROM nginx:latest
 ---> d261fd19cb63
Step 2/2 : COPY custom-index.html /usr/share/nginx/html/index.html
 ---> 3a4cba83f782
Successfully built 3a4cba83f782
Successfully tagged kaivalyabachkar/cloudethix:Task25
```

Push it your repository:-



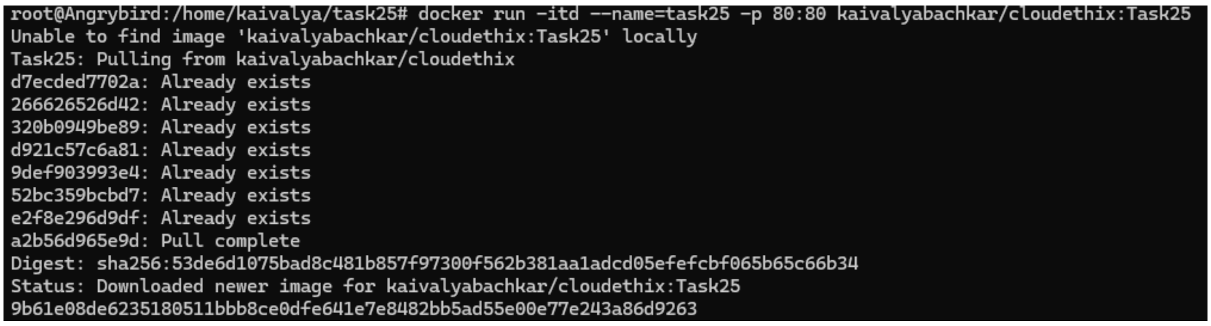
```
root@Angrybird:/home/kaivalya/task25# docker push kaivalyabachkar/cloudethix:Task25
The push refers to repository [docker.io/kaivalyabachkar/cloudethix]
75c6effed5fc: Pushed
d7217c60dca4: Mounted from library/nginx
d81df94f8d07: Mounted from library/nginx
99cd1b1b6a43: Mounted from library/nginx
2ced4cd78a7b: Mounted from library/nginx
8feb164cd673: Mounted from library/nginx
6e19587ac541: Mounted from library/nginx
36d06fe0cbc6: Mounted from library/nginx
Task25: digest: sha256:53de6d1075bad8c481b857f97300f562b381aa1adcd05efefcbf065b65c66b34 size: 1986
```

Delete the local image:-

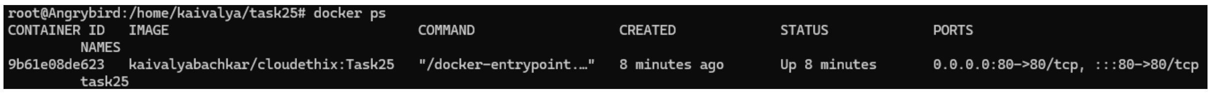


```
root@Angrybird:/home/kaivalya/task25# docker rmi kaivalyabachkar/cloudethix:Task25
Untagged: kaivalyabachkar/cloudethix:Task25
Untagged: kaivalyabachkar/cloudethix@sha256:53de6d1075bad8c481b857f97300f562b381aa1adcd05efefcbf065b65c66b34
Deleted: sha256:3a4cba83f782e6f303f965aa130bf115d445e61a10dd172f1a9eec535d6058bf
Deleted: sha256:9628ea1cae8720a5fc469dab72f3416a6b10ff5a4daff1b53911891f153569fe
```

Start a new container using the custom Nginx image was pushed and assigned Port 80:-

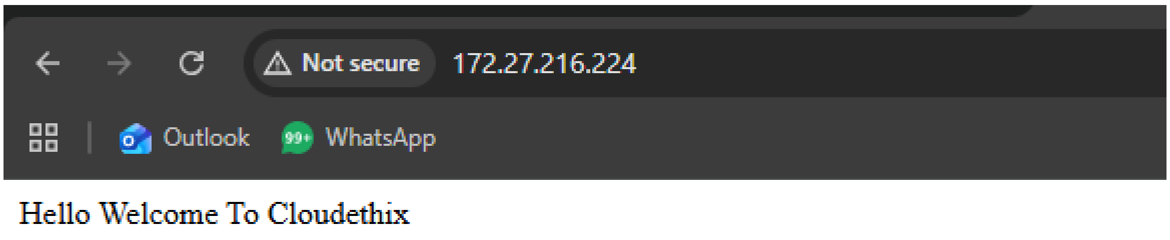


```
root@Angrybird:/home/kaivalya/task25# docker run -itd --name=task25 -p 80:80 kaivalyabachkar/cloudethix:Task25
Unable to find image 'kaivalyabachkar/cloudethix:Task25' locally
Task25: Pulling from kaivalyabachkar/cloudethix
d7ecded7702a: Already exists
266626526d42: Already exists
320b0949be89: Already exists
d921c57c6a81: Already exists
9def903993e4: Already exists
52bc359bcbd7: Already exists
e2f8e296d9df: Already exists
a2b56d965e9d: Pull complete
Digest: sha256:53de6d1075bad8c481b857f97300f562b381aa1adcd05efefcbf065b65c66b34
Status: Downloaded newer image for kaivalyabachkar/cloudethix:Task25
9b61e08de6235180511bbb8ce0dfe641e7e8482bb5ad55e00e77e243a86d9263
```



```
root@Angrybird:/home/kaivalya/task25# docker ps
CONTAINER ID   IMAGE                               COMMAND                  CREATED         STATUS         PORTS
         NAMES
9b61e08de623   kaivalyabachkar/cloudethix:Task25   "/docker-entrypoint.…"   8 minutes ago   Up 8 minutes   0.0.0.0:80->80/tcp, :::80->80/tcp
         task25
```

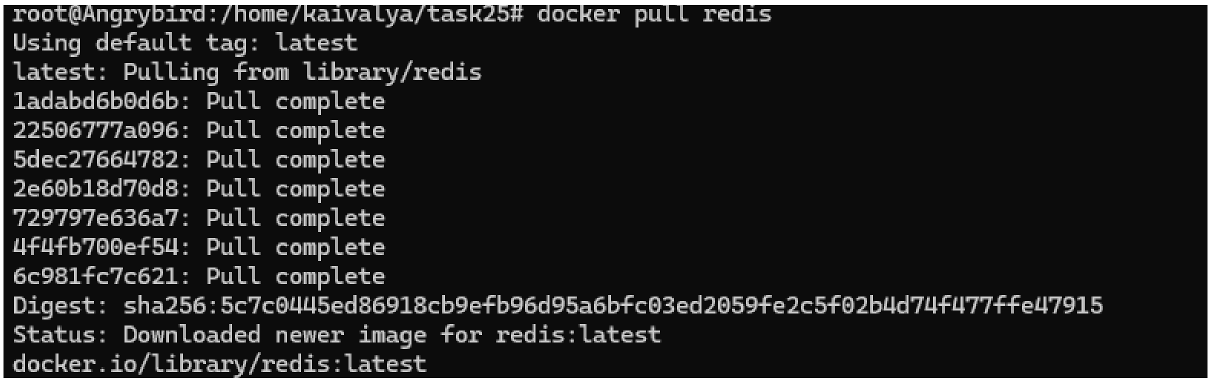
Check the container page in browser:-



Hello Welcome To Cloudethix

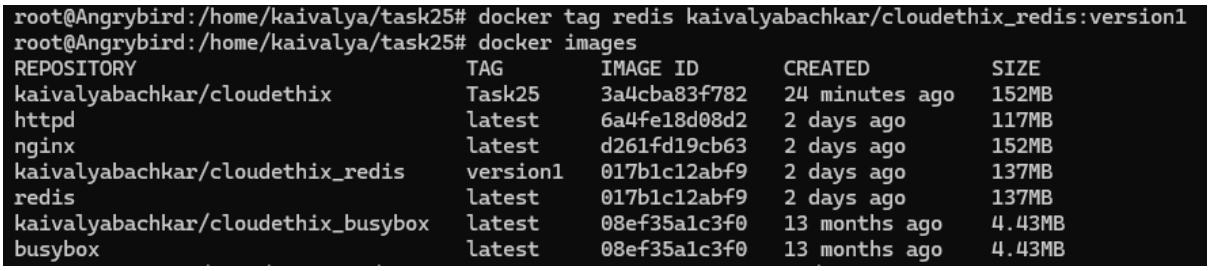
26. Push the Redis images tagged as version 1 and 3 to your Docker Hub repository, named "yourname_cloudethix_redis."

Pull the Redis image:-



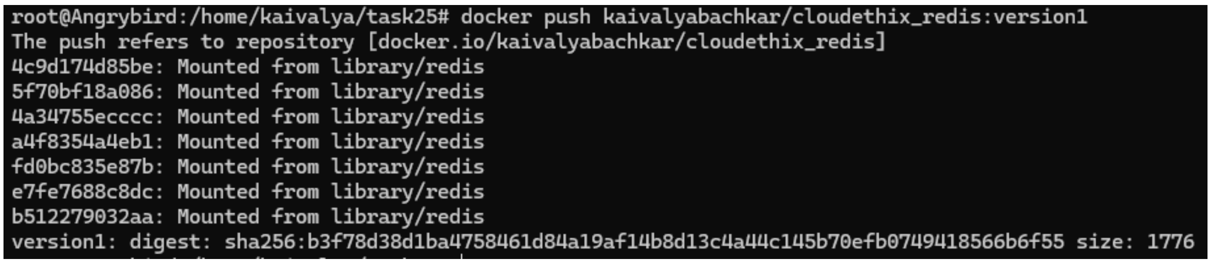
```
root@Angrybird:/home/kaivalya/task25# docker pull redis
Using default tag: latest
latest: Pulling from library/redis
1adabd6b0d6b: Pull complete
22506777a096: Pull complete
5dec27664782: Pull complete
2e60b18d70d8: Pull complete
729797e636a7: Pull complete
4f4fb700ef54: Pull complete
6c981fc7c621: Pull complete
Digest: sha256:5c7c0445ed86918cb9efb96d95a6bfc03ed2059fe2c5f02b4d74f477ffe47915
Status: Downloaded newer image for redis:latest
docker.io/library/redis:latest
```

Tagged as version 1:-



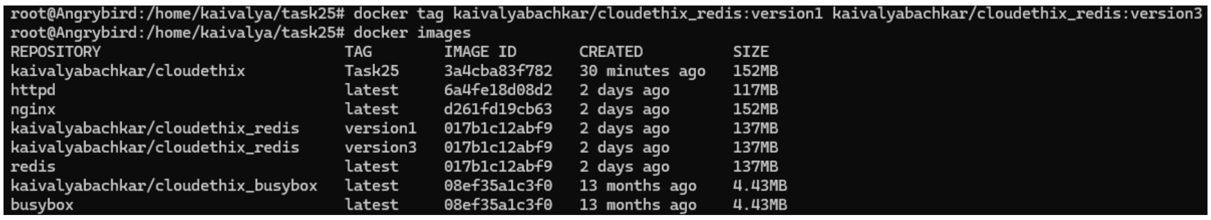
```
root@Angrybird:/home/kaivalya/task25# docker tag redis kaivalyabachkar/cloudethix_redis:version1
root@Angrybird:/home/kaivalya/task25# docker images
REPOSITORY                          TAG        IMAGE ID       CREATED          SIZE
kaivalyabachkar/cloudethix          Task25     3a4cba83f782   24 minutes ago   152MB
httpd                               latest     6a4fe18d08d2   2 days ago       117MB
nginx                               latest     d261fd19cb63   2 days ago       152MB
kaivalyabachkar/cloudethix_redis    version1   017b1c12abf9   2 days ago       137MB
redis                               latest     017b1c12abf9   2 days ago       137MB
kaivalyabachkar/cloudethix_busybox  latest     08ef35a1c3f0   13 months ago    4.43MB
busybox                             latest     08ef35a1c3f0   13 months ago    4.43MB
```

Pushed on Docker Hub repository:-



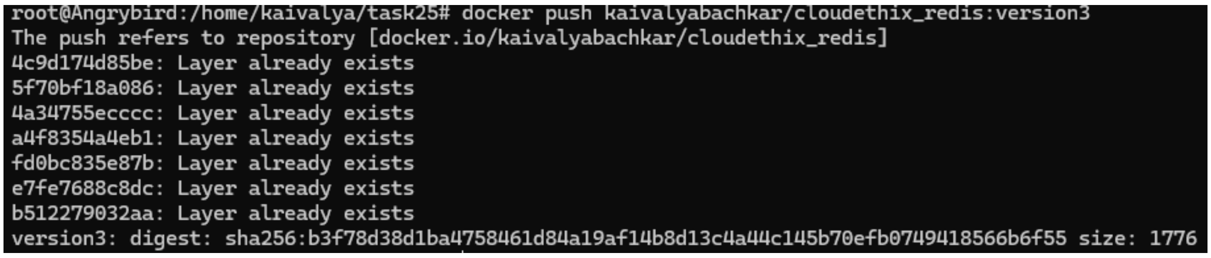
```
root@Angrybird:/home/kaivalya/task25# docker push kaivalyabachkar/cloudethix_redis:version1
The push refers to repository [docker.io/kaivalyabachkar/cloudethix_redis]
4c9d174d85be: Mounted from library/redis
5f70bf18a086: Mounted from library/redis
4a34755ecccc: Mounted from library/redis
a4f8354a4eb1: Mounted from library/redis
fd0bc835e87b: Mounted from library/redis
e7fe7688c8dc: Mounted from library/redis
b512279032aa: Mounted from library/redis
version1: digest: sha256:b3f78d38d1ba4758461d84a19af14b8d13c4a44c145b70efb0749418566b6f55 size: 1776
```

Tagged as version 3:-

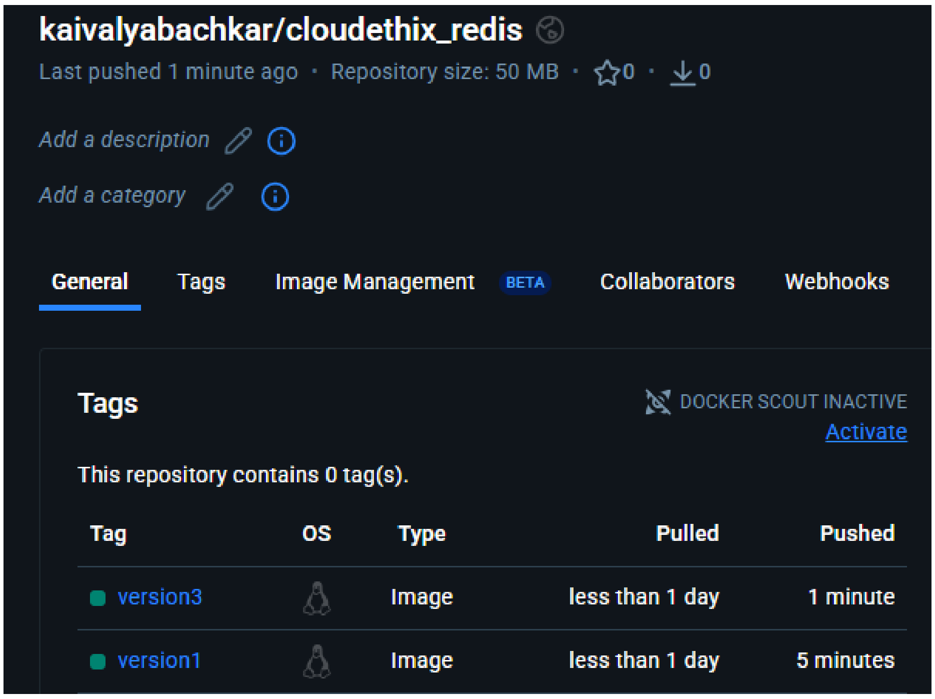


```
root@Angrybird:/home/kaivalya/task25# docker tag kaivalyabachkar/cloudethix_redis:version1 kaivalyabachkar/cloudethix_redis:version3
root@Angrybird:/home/kaivalya/task25# docker images
REPOSITORY                          TAG        IMAGE ID       CREATED          SIZE
kaivalyabachkar/cloudethix          Task25     3a4cba83f782   30 minutes ago   152MB
httpd                               latest     6a4fe18d08d2   2 days ago       117MB
nginx                               latest     d261fd19cb63   2 days ago       152MB
kaivalyabachkar/cloudethix_redis    version1   017b1c12abf9   2 days ago       137MB
kaivalyabachkar/cloudethix_redis    version3   017b1c12abf9   2 days ago       137MB
redis                               latest     017b1c12abf9   2 days ago       137MB
kaivalyabachkar/cloudethix_busybox  latest     08ef35a1c3f0   13 months ago    4.43MB
busybox                             latest     08ef35a1c3f0   13 months ago    4.43MB
```

Pushed on Docker Hub repository:-



```
root@Angrybird:/home/kaivalya/task25# docker push kaivalyabachkar/cloudethix_redis:version3
The push refers to repository [docker.io/kaivalyabachkar/cloudethix_redis]
4c9d174d85be: Layer already exists
5f70bf18a086: Layer already exists
4a34755ecccc: Layer already exists
a4f8354a4eb1: Layer already exists
fd0bc835e87b: Layer already exists
e7fe7688c8dc: Layer already exists
b512279032aa: Layer already exists
version3: digest: sha256:b3f78d38d1ba4758461d84a19af14b8d13c4a44c145b70efb0749418566b6f55 size: 1776
```



kaivalyabachkar/cloudethix_redis

Last pushed 1 minute ago · Repository size: 50 MB · 0 · 0

Add a description

Add a category

General | Tags | Image Management BETA | Collaborators | Webhooks

Tags

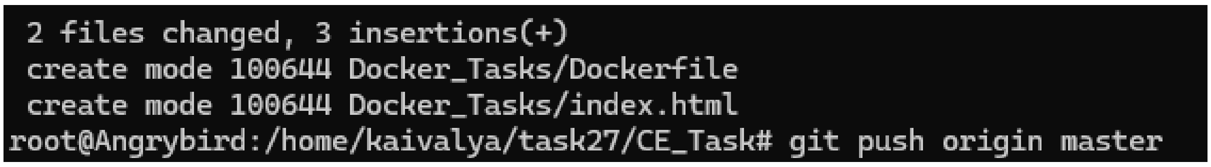
DOCKER SCOUT INACTIVE
Activate

This repository contains 0 tag(s).

Tag	OS	Type	Pulled	Pushed
version3		Image	less than 1 day	1 minute
version1		Image	less than 1 day	5 minutes

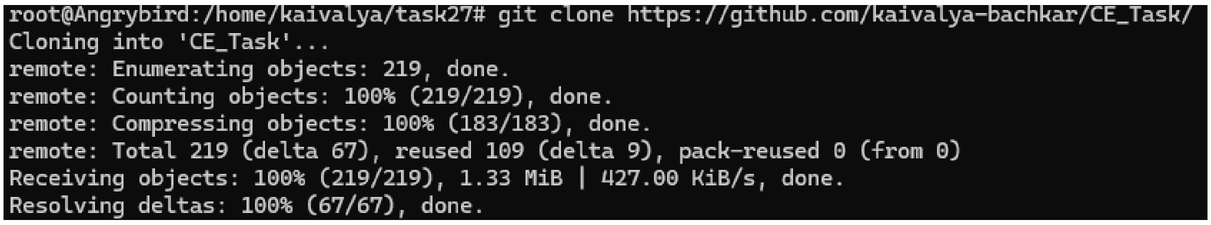
27. Create a remote Git repository and add the Dockerfile and index.html files. Clone the repository locally and create a release branch. Build the Docker image from the release branch with a meaningful tag, then run a container from the image and expose it on host port 8383. Check the webpage in a browser, 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.

Create a remote Git repository and add the Dockerfile and index.html files:-



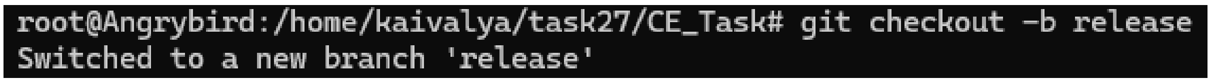
```
 2 files changed, 3 insertions(+)
 create mode 100644 Docker_Tasks/Dockerfile
 create mode 100644 Docker_Tasks/index.html
root@Angrybird:/home/kaivalya/task27/CE_Task# git push origin master
```

Clone the repository locally:-



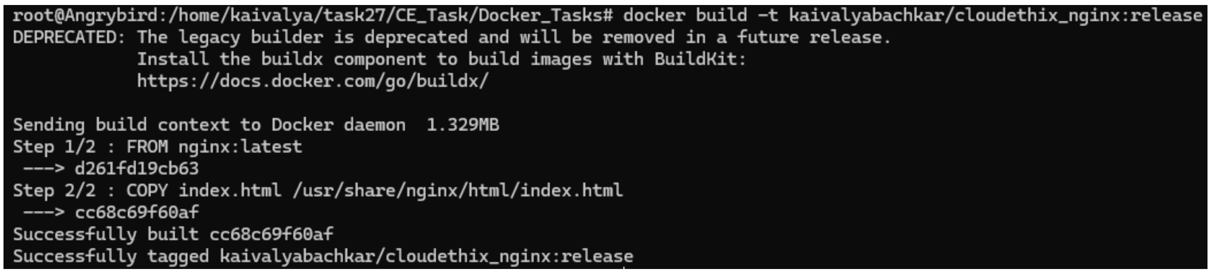
```
root@Angrybird:/home/kaivalya/task27# git clone https://github.com/kaivalya-bachkar/CE_Task/
Cloning into 'CE_Task'...
remote: Enumerating objects: 219, done.
remote: Counting objects: 100% (219/219), done.
remote: Compressing objects: 100% (183/183), done.
remote: Total 219 (delta 67), reused 109 (delta 9), pack-reused 0 (from 0)
Receiving objects: 100% (219/219), 1.33 MiB | 427.00 KiB/s, done.
Resolving deltas: 100% (67/67), done.
```

Create a release branch:-



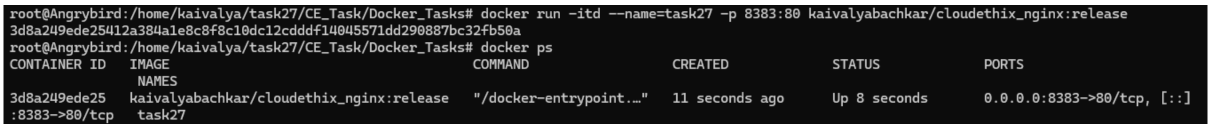
```
root@Angrybird:/home/kaivalya/task27/CE_Task# git checkout -b release
Switched to a new branch 'release'
```

Build the Docker image from the release branch with a tag:-



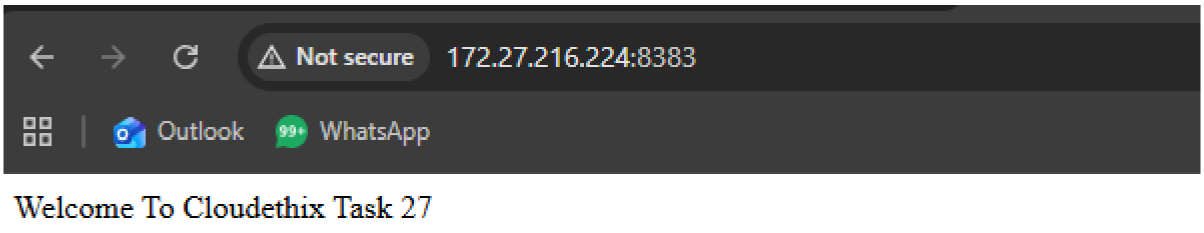
```
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker build -t kaivalyabachkar/cloudethix_nginx:release
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  1.329MB
Step 1/2 : FROM nginx:latest
 ---> d261fd19cb63
Step 2/2 : COPY index.html /usr/share/nginx/html/index.html
 ---> cc68c69f60af
Successfully built cc68c69f60af
Successfully tagged kaivalyabachkar/cloudethix_nginx:release
```

Run a container from the image and expose it on host port 8383:-



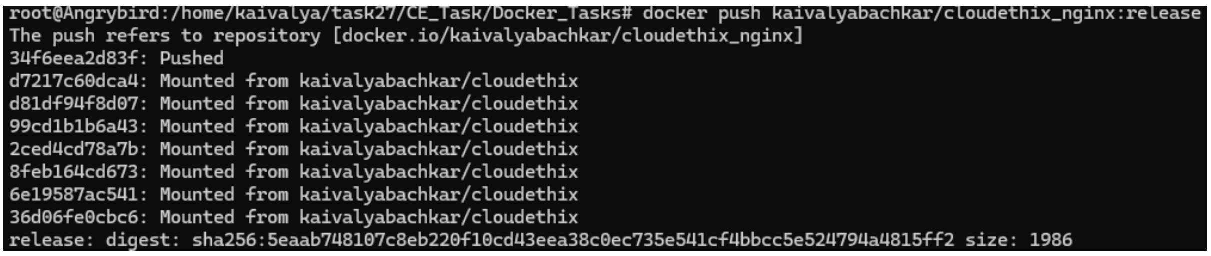
```
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker run -itd --name=task27 -p 8383:80 kaivalyabachkar/cloudethix_nginx:release
3d8a249ede25412a384a1e8c8f8c10dc12cdddf14045571dd290887bc32fb50a
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker ps
CONTAINER ID   IMAGE                                      COMMAND                  CREATED          STATUS         PORTS
               NAMES
3d8a249ede25   kaivalyabachkar/cloudethix_nginx:release   "/docker-entrypoint.…"   11 seconds ago   Up 8 seconds   0.0.0.0:8383->80/tcp, [::]
:8383->80/tcp   task27
```

Check the webpage in a browser:-



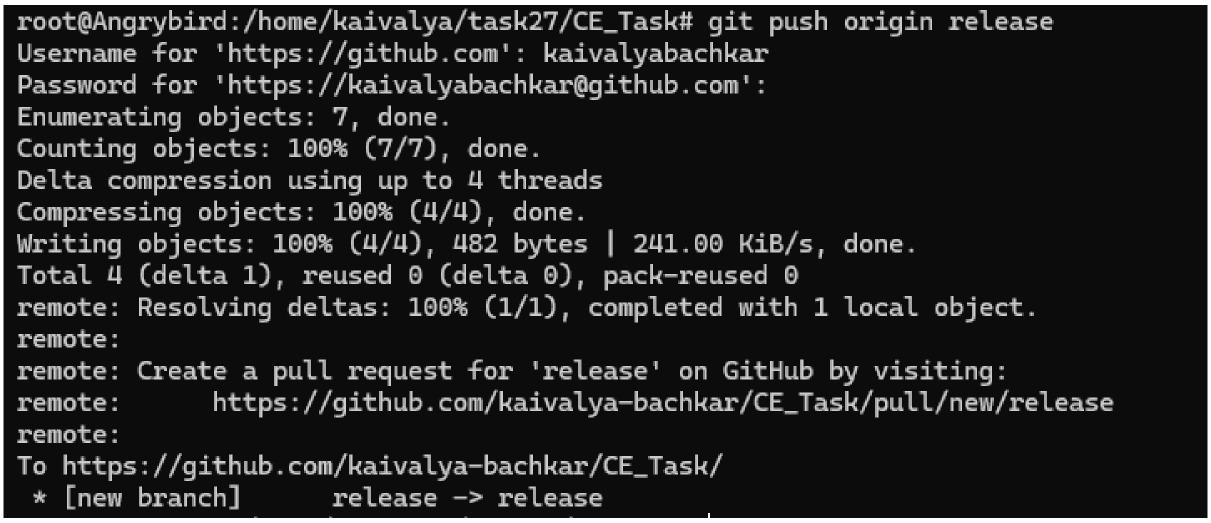
Welcome To Cloudethix Task 27

Push the image to your Docker Hub repository named "kaivalyabachkar/cloudethix_nginx:-



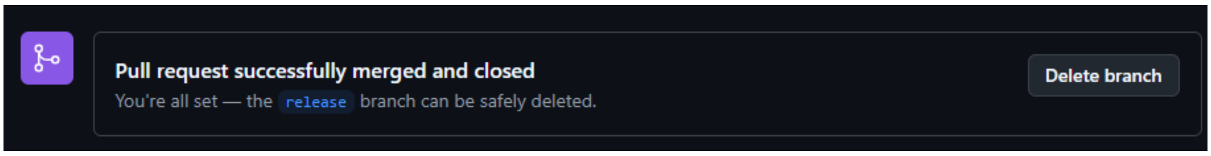
```
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker push kaivalyabachkar/cloudethix_nginx:release
The push refers to repository [docker.io/kaivalyabachkar/cloudethix_nginx]
34f6eea2d83f: Pushed
d7217c60dca4: Mounted from kaivalyabachkar/cloudethix
d81df94f8d07: Mounted from kaivalyabachkar/cloudethix
99cd1b1b6a43: Mounted from kaivalyabachkar/cloudethix
2ced4cd78a7b: Mounted from kaivalyabachkar/cloudethix
8feb164cd673: Mounted from kaivalyabachkar/cloudethix
6e19587ac541: Mounted from kaivalyabachkar/cloudethix
36d06fe0cbc6: Mounted from kaivalyabachkar/cloudethix
release: digest: sha256:5eaab748107c8eb220f10cd43eea38c0ec735e541cf4bbcc5e524794a4815ff2 size: 1986
```

Push the release branch to the remote Git repository:-



```
root@Angrybird:/home/kaivalya/task27/CE_Task# git push origin release
Username for 'https://github.com': kaivalyabachkar
Password for 'https://kaivalyabachkar@github.com':
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 482 bytes | 241.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'release' on GitHub by visiting:
remote:      https://github.com/kaivalya-bachkar/CE_Task/pull/new/release
remote:
To https://github.com/kaivalya-bachkar/CE_Task/
 * [new branch]      release -> release
```

Merge it by creating a pull request:-

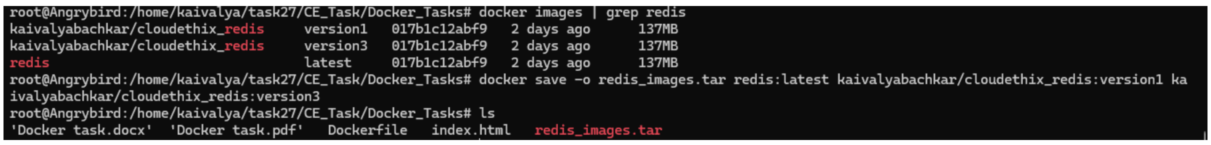


Pull request successfully merged and closed

You're all set — the release branch can be safely deleted.

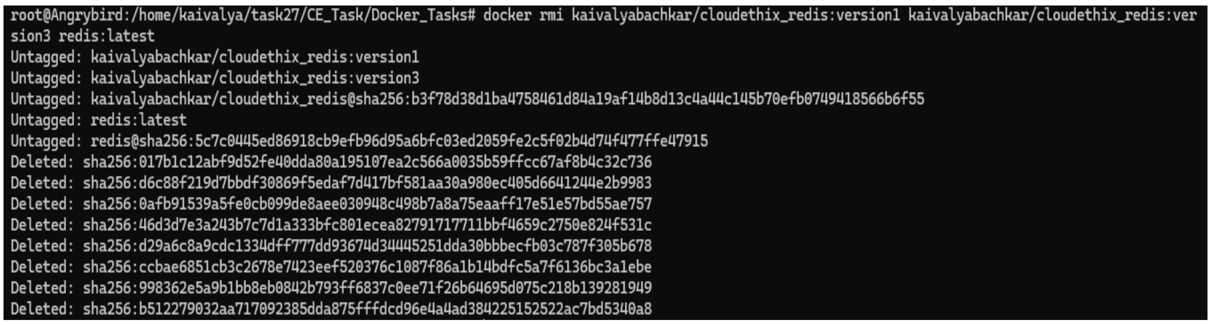
28. Save all local Redis images as a .tar file in the master branch of your local repository. Delete all Redis images from your local system and push the master branch to the remote repository. Load the Redis images from the tar file to your local system, and verify that all images have been loaded correctly.

Save all local Redis images as a .tar file in the master branch of your local repository:-



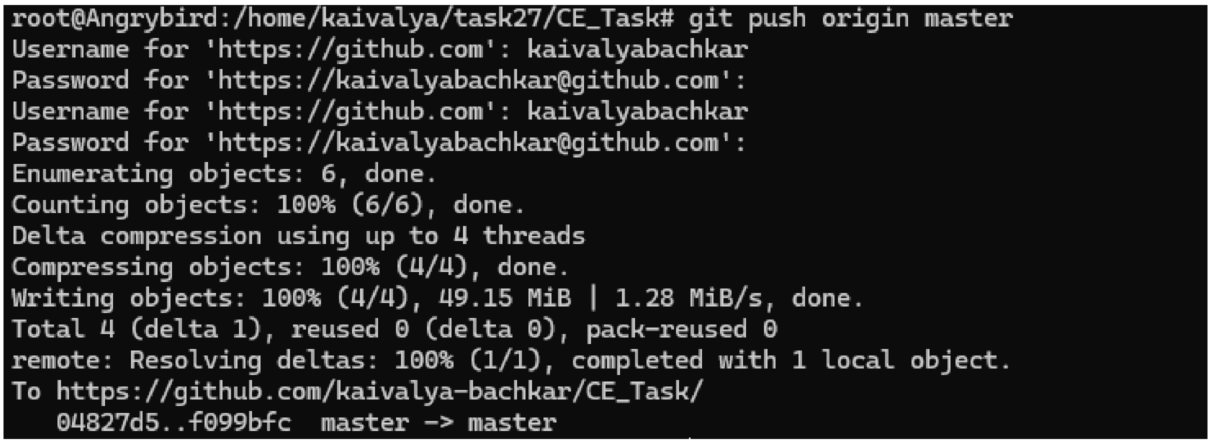
```
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker images | grep redis
kaivalyabachkar/cloudethix_redis     version1   017b1c12abf9   2 days ago      137MB
kaivalyabachkar/cloudethix_redis     version3   017b1c12abf9   2 days ago      137MB
redis                                latest     017b1c12abf9   2 days ago      137MB
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker save -o redis_images.tar redis:latest kaivalyabachkar/cloudethix_redis:version1 ka
ivalyabachkar/cloudethix_redis:version3
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# ls
'Docker task.docx'  'Docker task.pdf'   Dockerfile   index.html   redis_images.tar
```

Delete all Redis images from your local system:-



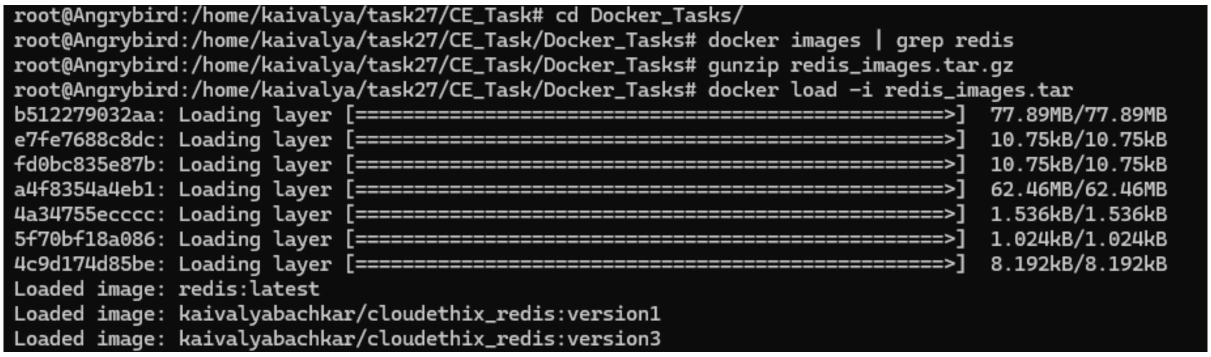
```
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker rmi kaivalyabachkar/cloudethix_redis:version1 kaivalyabachkar/cloudethix_redis:ver
sion3 redis:latest
Untagged: kaivalyabachkar/cloudethix_redis:version1
Untagged: kaivalyabachkar/cloudethix_redis:version3
Untagged: kaivalyabachkar/cloudethix_redis@sha256:b3f78d38d1ba4758461d84a19af14b8d13c4a44c145b70efb0749418566b6f55
Untagged: redis:latest
Untagged: redis@sha256:5c7c0445ed86918cb9efb96d95a6bfc03ed2059fe2c5f02b4d74f477ffe47915
Deleted: sha256:017b1c12abf9d52fe40dda80a195107ea2c566a0035b59ffcc67af8b4c32c736
Deleted: sha256:d6c88f219d7bbdf30869f5edaf7d417bf581aa30a980ec405d6641244e2b9983
Deleted: sha256:0afb91539a5fe0cb099de8aee030948c498b7a8a75eaaff17e51e57bd55ae757
Deleted: sha256:46d3d7e3a243b7c7d1a333bfc801ecea82791717711bbf4659c2750e824f531c
Deleted: sha256:d29a6c8a9cdc1334dff777dd93674d34445251dda30bbbecfb03c787f305b678
Deleted: sha256:ccbae6851cb3c2678e7423eef520376c1087f86a1b14bdfc5a7f6136bc3a1ebe
Deleted: sha256:998362e5a9b1bb8eb0842b793ff6837c0ee71f26b64695d075c218b139281949
Deleted: sha256:b512279032aa717092385dda875fffdcd96e4a4ad384225152522ac7bd5340a8
```

Push the master branch to the remote repository:-



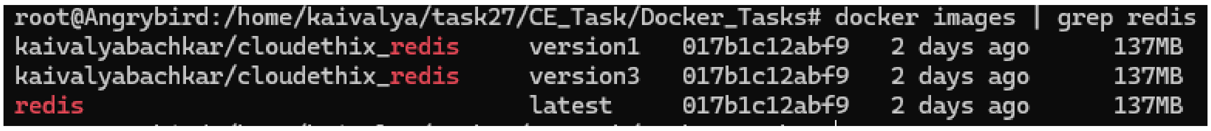
```
root@Angrybird:/home/kaivalya/task27/CE_Task# git push origin master
Username for 'https://github.com': kaivalyabachkar
Password for 'https://kaivalyabachkar@github.com':
Username for 'https://github.com': kaivalyabachkar
Password for 'https://kaivalyabachkar@github.com':
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 49.15 MiB | 1.28 MiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/kaivalya-bachkar/CE_Task/
   04827d5..f099bfc  master -> master
```

Load the Redis images from the tar file to your local system:-



```
root@Angrybird:/home/kaivalya/task27/CE_Task# cd Docker_Tasks/
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker images | grep redis
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# gunzip redis_images.tar.gz
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker load -i redis_images.tar
b512279032aa: Loading layer [==================================================>]  77.89MB/77.89MB
e7fe7688c8dc: Loading layer [==================================================>]  10.75kB/10.75kB
fd0bc835e87b: Loading layer [==================================================>]  10.75kB/10.75kB
a4f8354a4eb1: Loading layer [==================================================>]  62.46MB/62.46MB
4a34755ecccc: Loading layer [==================================================>]  1.536kB/1.536kB
5f70bf18a086: Loading layer [==================================================>]  1.024kB/1.024kB
4c9d174d85be: Loading layer [==================================================>]  8.192kB/8.192kB
Loaded image: redis:latest
Loaded image: kaivalyabachkar/cloudethix_redis:version1
Loaded image: kaivalyabachkar/cloudethix_redis:version3
```

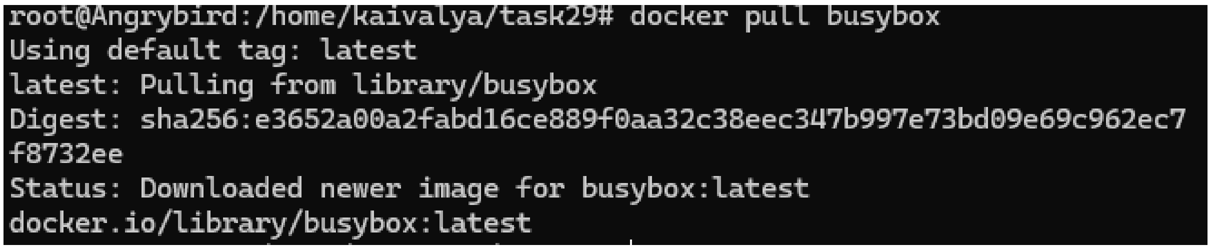
Verify that all images have been loaded correctly:-



```
root@Angrybird:/home/kaivalya/task27/CE_Task/Docker_Tasks# docker images | grep redis
kaivalyabachkar/cloudethix_redis      version1     017b1c12abf9   2 days ago       137MB
kaivalyabachkar/cloudethix_redis      version3     017b1c12abf9   2 days ago       137MB
redis                                 latest       017b1c12abf9   2 days ago       137MB
```

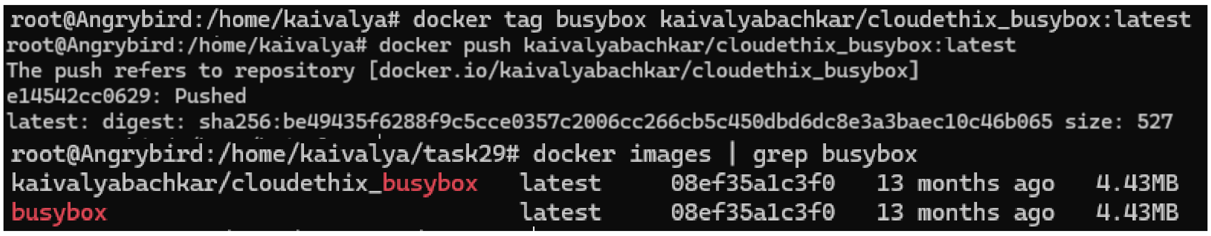
29. Pull the Busy-box image to your local system, tag it, and push it to the Docker Hub repository "yourname_cloudethix_busybox." Export the Docker image from the NGINX container, create a .tar file, and import the tar file to create a Docker image with a meaningful name. After importing the image, tag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

Pull the Busy-box image to your local system:-



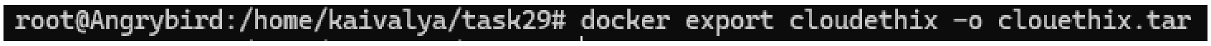
```
root@Angrybird:/home/kaivalya/task29# docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
Digest: sha256:e3652a00a2fabd16ce889f0aa32c38eec347b997e73bd09e69c962ec7
f8732ee
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
```

Tag it, and push it to the Docker Hub repository "yourname_cloudethix_busybox:-



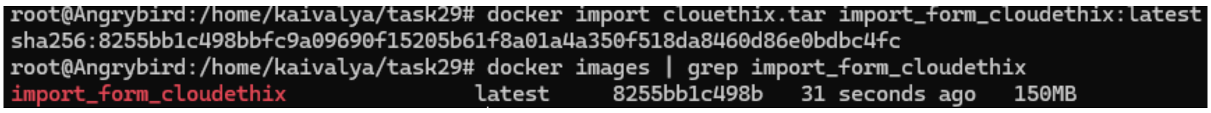
```
root@Angrybird:/home/kaivalya# docker tag busybox kaivalyabachkar/cloudethix_busybox:latest
root@Angrybird:/home/kaivalya# docker push kaivalyabachkar/cloudethix_busybox:latest
The push refers to repository [docker.io/kaivalyabachkar/cloudethix_busybox]
e14542cc0629: Pushed
latest: digest: sha256:be49435f6288f9c5cce0357c2006cc266cb5c450dbd6dc8e3a3baec10c46b065 size: 527
root@Angrybird:/home/kaivalya/task29# docker images | grep busybox
kaivalyabachkar/cloudethix_busybox   latest     08ef35a1c3f0   13 months ago    4.43MB
busybox                              latest     08ef35a1c3f0   13 months ago    4.43MB
```

Export the Docker image from the NGINX container Create a .tar file:-



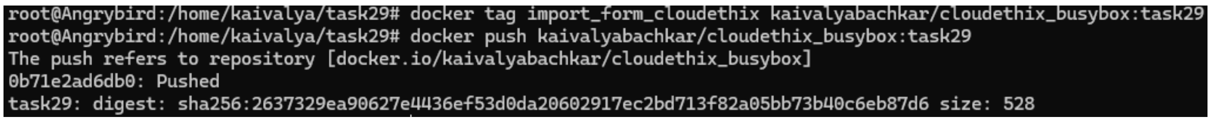
```
root@Angrybird:/home/kaivalya/task29# docker export cloudethix -o clouethix.tar
```

Import the tar file to create a Docker image with a meaningful name:-

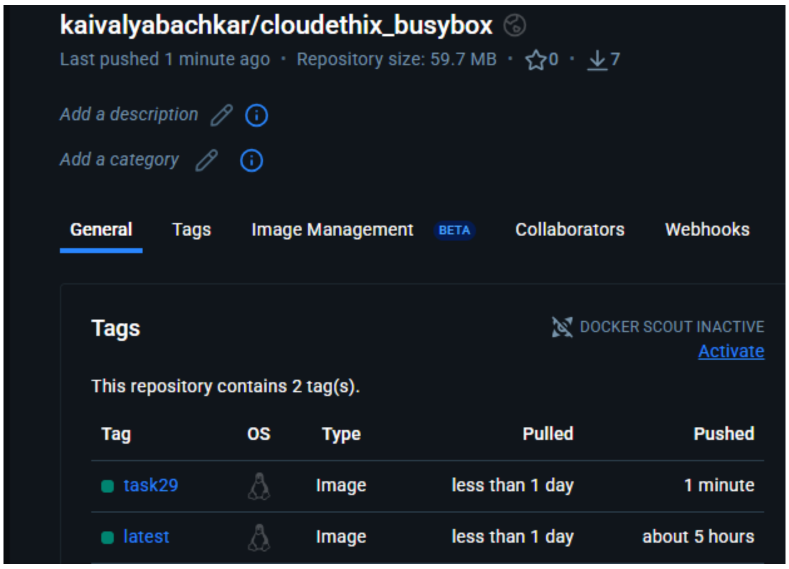


```
root@Angrybird:/home/kaivalya/task29# docker import clouethix.tar import_form_cloudethix:latest
sha256:8255bb1c498bbfc9a09690f15205b61f8a01a4a350f518da8460d86e0bdbc4fc
root@Angrybird:/home/kaivalya/task29# docker images | grep import_form_cloudethix
import_form_cloudethix                latest      8255bb1c498b    31 seconds ago    150MB
```

After importing the image, tag it and push it to the "yourname_cloudethix_busybox" Docker Hub repository:-



```
root@Angrybird:/home/kaivalya/task29# docker tag import_form_cloudethix kaivalyabachkar/cloudethix_busybox:task29
root@Angrybird:/home/kaivalya/task29# docker push kaivalyabachkar/cloudethix_busybox:task29
The push refers to repository [docker.io/kaivalyabachkar/cloudethix_busybox]
0b71e2ad6db0: Pushed
task29: digest: sha256:2637329ea90627e4436ef53d0da20602917ec2bd713f82a05bb73b40c6eb87d6 size: 528
```



kaivalyabachkar/cloudethix_busybox

Last pushed 1 minute ago · Repository size: 59.7 MB · ☆0 · ↓7

Add a description

Add a category

General | Tags | Image Management BETA | Collaborators | Webhooks

Tags

DOCKER SCOUT INACTIVE Activate

This repository contains 2 tag(s).

Tag	OS	Type	Pulled	Pushed
task29		Image	less than 1 day	1 minute
latest		Image	less than 1 day	about 5 hours

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

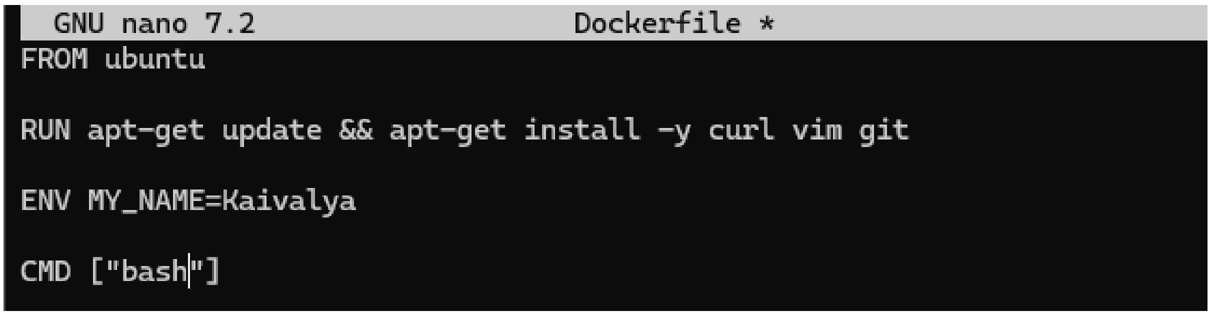
Base image: Ubuntu

Install packages: curl, vim, and git

Set an environment variable: MY_NAME=Your_Name

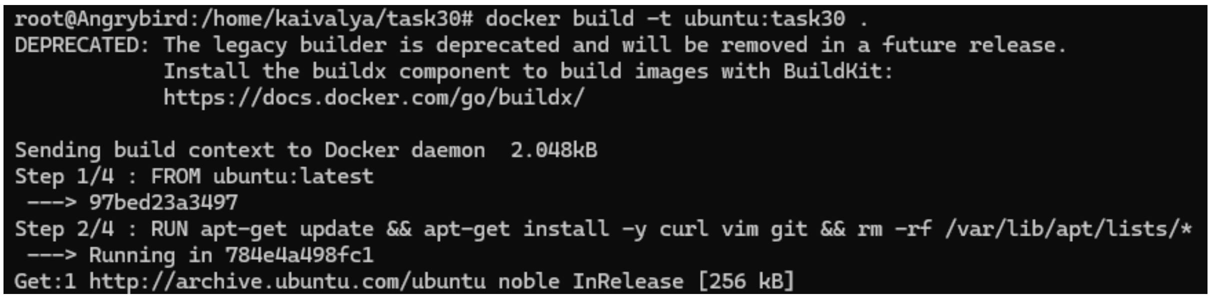
Build the custom image using docker build and list all available images using docker images.

Dockerfile creation :-



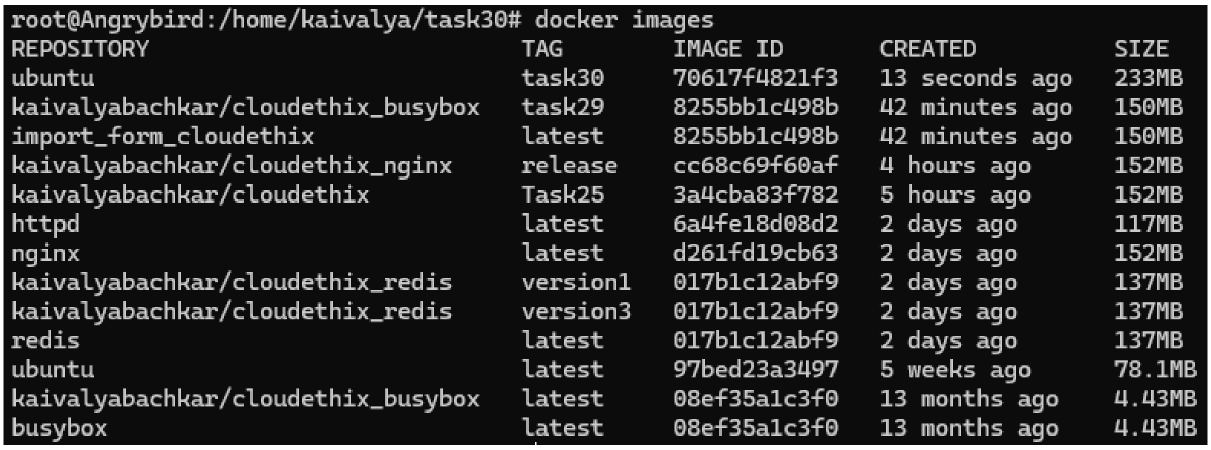
```
  GNU nano 7.2                          Dockerfile *
FROM ubuntu

RUN apt-get update && apt-get install -y curl vim git

ENV MY_NAME=Kaivalya

CMD ["bash"]
```

Build the custom image using docker build:-



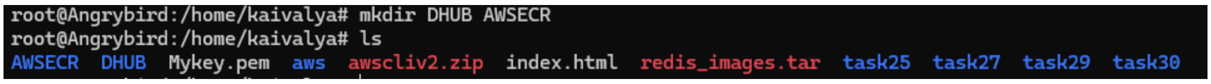
```
root@Angrybird:/home/kaivalya/task30# docker build -t ubuntu:task30 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  2.048kB
Step 1/4 : FROM ubuntu:latest
 ---> 97bed23a3497
Step 2/4 : RUN apt-get update && apt-get install -y curl vim git && rm -rf /var/lib/apt/lists/*
 ---> Running in 784e4a498fc1
Get:1 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
```

list all available images using docker images:-



```
root@Angrybird:/home/kaivalya/task30# docker images
REPOSITORY                          TAG        IMAGE ID       CREATED          SIZE
ubuntu                              task30     70617f4821f3   13 seconds ago   233MB
kaivalyabachkar/cloudethix_busybox  task29     8255bb1c498b   42 minutes ago   150MB
import_form_cloudethix              latest     8255bb1c498b   42 minutes ago   150MB
kaivalyabachkar/cloudethix_nginx    release    cc68c69f60af   4 hours ago      152MB
kaivalyabachkar/cloudethix          Task25     3a4cba83f782   5 hours ago      152MB
httpd                               latest     6a4fe18d08d2   2 days ago       117MB
nginx                               latest     d261fd19cb63   2 days ago       152MB
kaivalyabachkar/cloudethix_redis    version1   017b1c12abf9   2 days ago       137MB
kaivalyabachkar/cloudethix_redis    version3   017b1c12abf9   2 days ago       137MB
redis                               latest     017b1c12abf9   2 days ago       137MB
ubuntu                              latest     97bed23a3497   5 weeks ago      78.1MB
kaivalyabachkar/cloudethix_busybox  latest     08ef35a1c3f0   13 months ago    4.43MB
busybox                             latest     08ef35a1c3f0   13 months ago    4.43MB
```

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

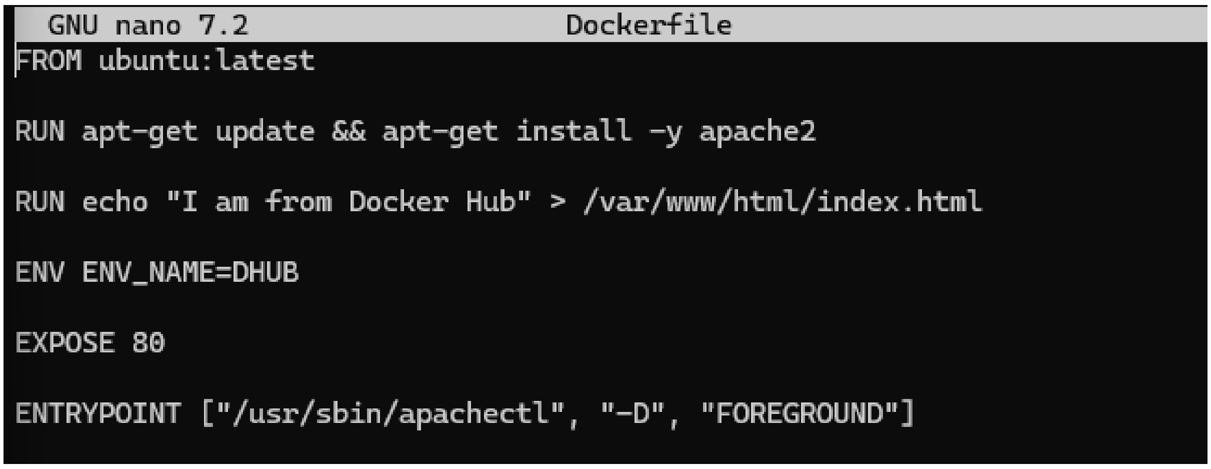


```
root@Angrybird:/home/kaivalya# mkdir DHUB AWSECR
root@Angrybird:/home/kaivalya# ls
AWSECR  DHUB  Mykey.pem  aws  awscliv2.zip  index.html  redis_images.tar  task25  task27  task29  task30
```

32. Dockerfile Creation: Develop two Dockerfiles to construct custom images with the following specifications:

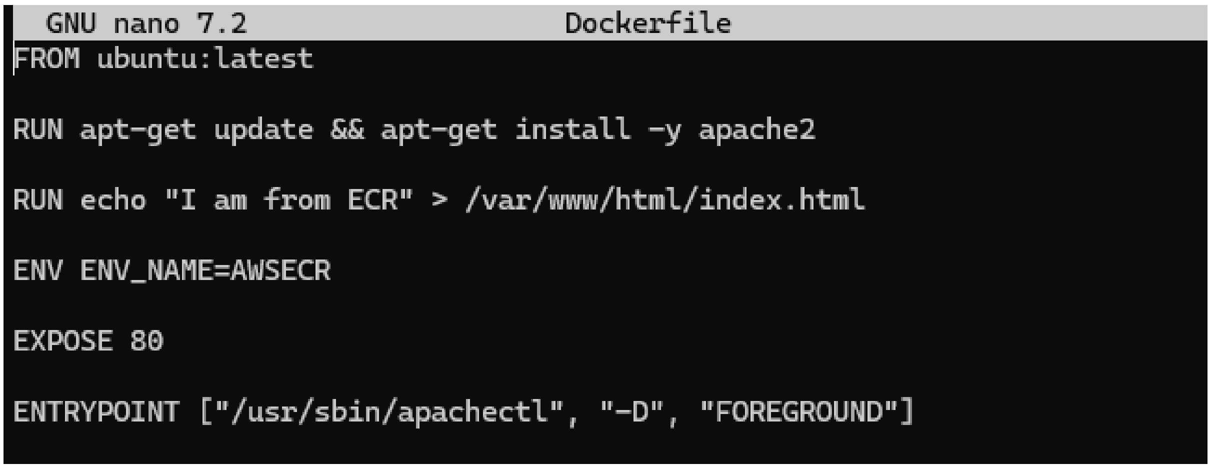
- Base image: Ubuntu
- Install packages: httpd
- Add "I am from Docker Hub" to the index.html file for DHUB directory and "I am from ECR" for AWSECR directory.
- Set environment variable ENV_NAME=DHUB for the DHUB directory and ENV_NAME=AWSECR for the AWSECR directory.
- Start http service using ENTRYPOINT
- Expose port 80.

DockerFile For Docker Hub:-



```
  GNU nano 7.2                     Dockerfile
FROM ubuntu:latest

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

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

ENV ENV_NAME=DHUB

EXPOSE 80

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

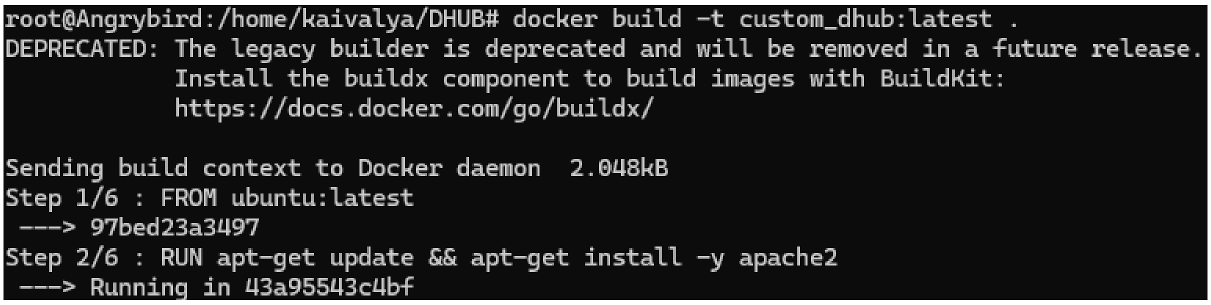
DockerFile For AWSECR:-



```
  GNU nano 7.2                     Dockerfile
FROM ubuntu:latest

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

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

ENV ENV_NAME=AWSECR

EXPOSE 80

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

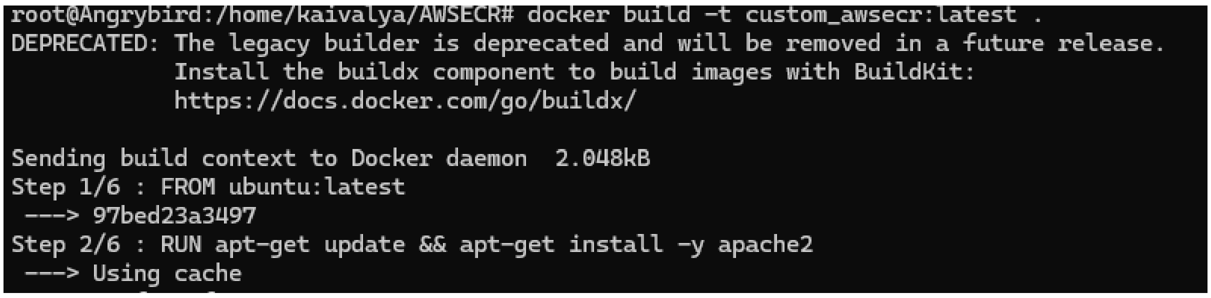
33. Build Custom Images: Utilize the docker build command to build the custom images. List all available images using docker images.

Image for Docker Hub:-



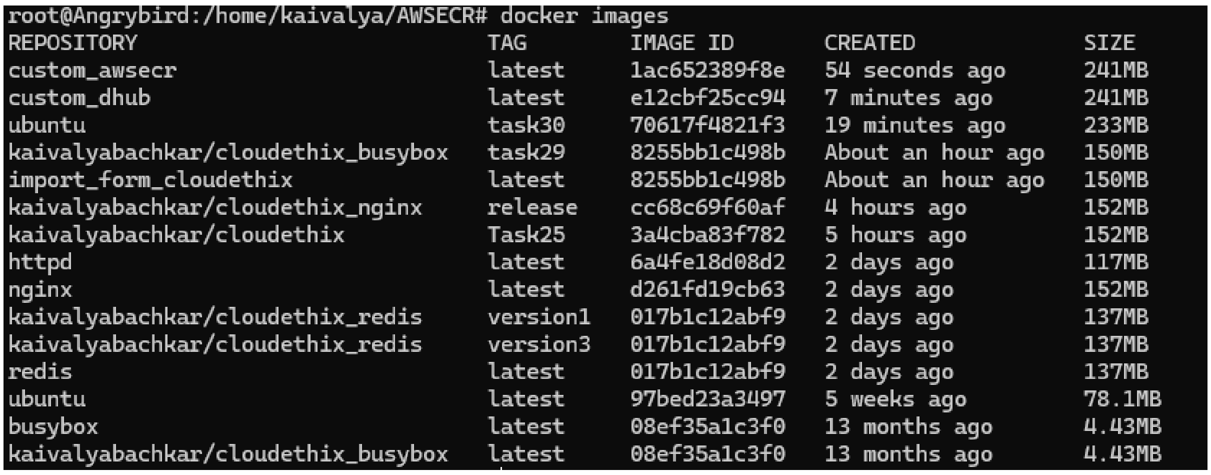
```
root@Angrybird:/home/kaivalya/DHUB# docker build -t custom_dhub:latest .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  2.048kB
Step 1/6 : FROM ubuntu:latest
 ---> 97bed23a3497
Step 2/6 : RUN apt-get update && apt-get install -y apache2
 ---> Running in 43a95543c4bf
```

Image For AWSECR:-



```
root@Angrybird:/home/kaivalya/AWSECR# docker build -t custom_awsecr:latest .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  2.048kB
Step 1/6 : FROM ubuntu:latest
 ---> 97bed23a3497
Step 2/6 : RUN apt-get update && apt-get install -y apache2
 ---> Using cache
```

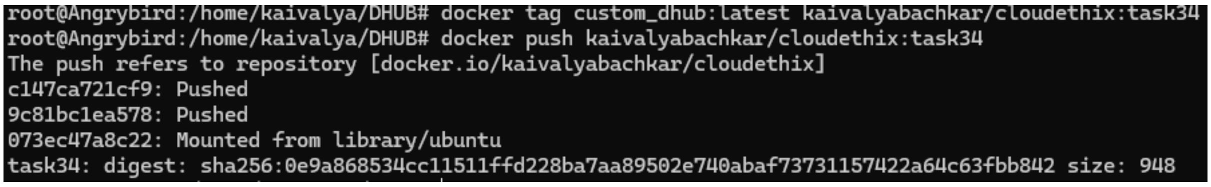
List all available images using docker images:-



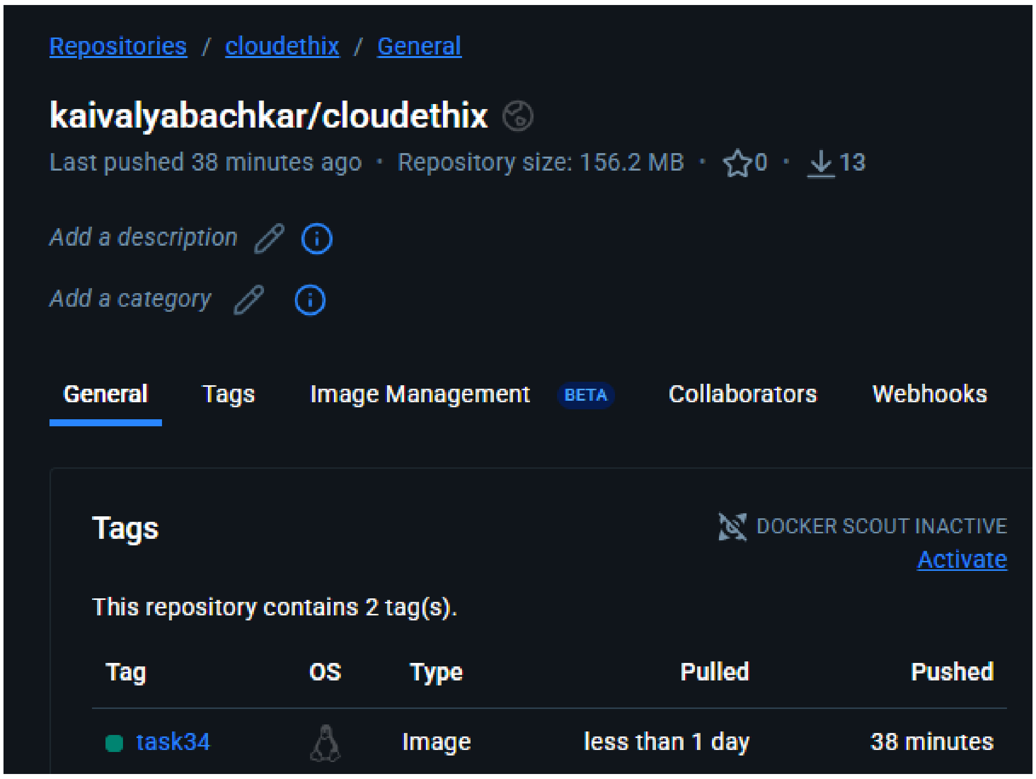
```
root@Angrybird:/home/kaivalya/AWSECR# docker images
REPOSITORY                            TAG        IMAGE ID       CREATED            SIZE
custom_awsecr                         latest     1ac652389f8e   54 seconds ago     241MB
custom_dhub                           latest     e12cbf25cc94   7 minutes ago      241MB
ubuntu                                task30     70617f4821f3   19 minutes ago     233MB
kaivalyabachkar/cloudethix_busybox    task29     8255bb1c498b   About an hour ago  150MB
import_form_cloudethix                latest     8255bb1c498b   About an hour ago  150MB
kaivalyabachkar/cloudethix_nginx      release    cc68c69f60af   4 hours ago        152MB
kaivalyabachkar/cloudethix            Task25     3a4cba83f782   5 hours ago        152MB
httpd                                 latest     6a4fe18d08d2   2 days ago         117MB
nginx                                 latest     d261fd19cb63   2 days ago         152MB
kaivalyabachkar/cloudethix_redis      version1   017b1c12abf9   2 days ago         137MB
kaivalyabachkar/cloudethix_redis      version3   017b1c12abf9   2 days ago         137MB
redis                                 latest     017b1c12abf9   2 days ago         137MB
ubuntu                                latest     97bed23a3497   5 weeks ago        78.1MB
busybox                               latest     08ef35a1c3f0   13 months ago      4.43MB
kaivalyabachkar/cloudethix_busybox    latest     08ef35a1c3f0   13 months ago      4.43MB
```

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

Push the Docker image to Docker Hub:-



```
root@Angrybird:/home/kaivalya/DHUB# docker tag custom_dhub:latest kaivalyabachkar/cloudethix:task34
root@Angrybird:/home/kaivalya/DHUB# docker push kaivalyabachkar/cloudethix:task34
The push refers to repository [docker.io/kaivalyabachkar/cloudethix]
c147ca721cf9: Pushed
9c81bc1ea578: Pushed
073ec47a8c22: Mounted from library/ubuntu
task34: digest: sha256:0e9a868534cc11511ffd228ba7aa89502e740abaf73731157422a64c63fbb842 size: 948
```



Repositories / cloudethix / General

kaivalyabachkar/cloudethix

Last pushed 38 minutes ago · Repository size: 156.2 MB · ☆0 · ⤓13

Add a description

Add a category

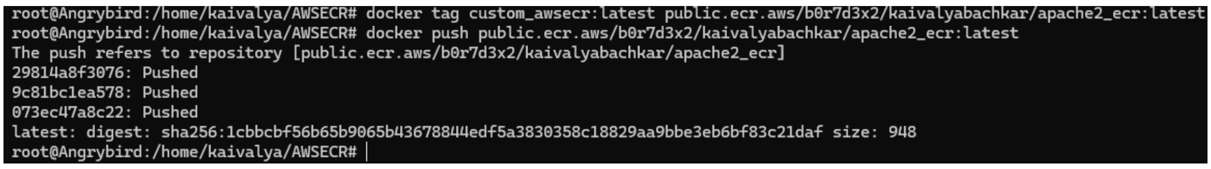
Tags

DOCKER SCOUT INACTIVE
Activate

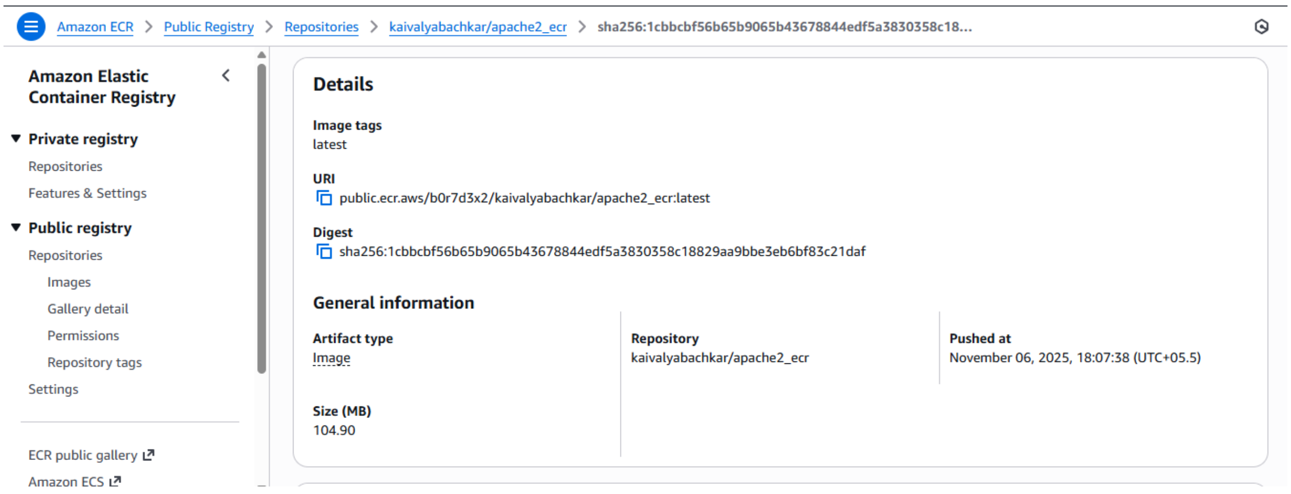
This repository contains 2 tag(s).

Tag	OS	Type	Pulled	Pushed
task34	🐧	Image	less than 1 day	38 minutes

Push the Docker image to AWS ECR:-



```
root@Angrybird:/home/kaivalya/AWSECR# docker tag custom_awsecr:latest public.ecr.aws/b0r7d3x2/kaivalyabachkar/apache2_ecr:latest
root@Angrybird:/home/kaivalya/AWSECR# docker push public.ecr.aws/b0r7d3x2/kaivalyabachkar/apache2_ecr:latest
The push refers to repository [public.ecr.aws/b0r7d3x2/kaivalyabachkar/apache2_ecr]
29814a8f3076: Pushed
9c81bc1ea578: Pushed
073ec47a8c22: Pushed
latest: digest: sha256:1cbbcbf56b65b9065b43678844edf5a3830358c18829aa9bbe3eb6bf83c21daf size: 948
root@Angrybird:/home/kaivalya/AWSECR# |
```


Amazon ECR > Public Registry > Repositories > kaivalyabachkar/apache2_ecr > sha256:1cbbcbf56b65b9065b43678844edf5a3830358c18...

Details

Image tags
latest

URI
public.ecr.aws/b0r7d3x2/kaivalyabachkar/apache2_ecr:latest

Digest
sha256:1cbbcbf56b65b9065b43678844edf5a3830358c18829aa9bbe3eb6bf83c21daf

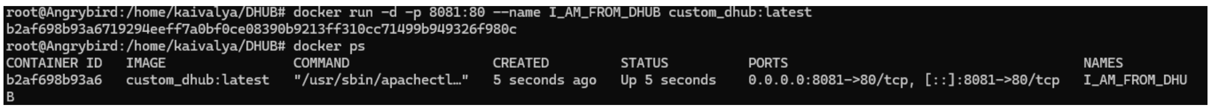
General information

Artifact type	Repository	Pushed at
Image	kaivalyabachkar/apache2_ecr	November 06, 2025, 18:07:38 (UTC+05.5)

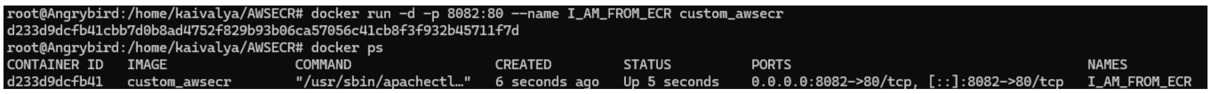
Size (MB)
104.90

35. Run Containers:

docker run -d -p 8081:80 --name I_AM_FROM_DHUB your_docker_hub_image


```
root@Angrybird:/home/kaivalya/DHUB# docker run -d -p 8081:80 --name I_AM_FROM_DHUB custom_dhub:latest
b2af698b93a6719294eeff7a0bf0ce08390b9213ff310cc71499b949326f980c
root@Angrybird:/home/kaivalya/DHUB# docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATED         STATUS         PORTS                                   NAMES
b2af698b93a6   custom_dhub:latest   "/usr/sbin/apachectl…"   5 seconds ago   Up 5 seconds   0.0.0.0:8081->80/tcp, [::]:8081->80/tcp   I_AM_FROM_DHU
B
```

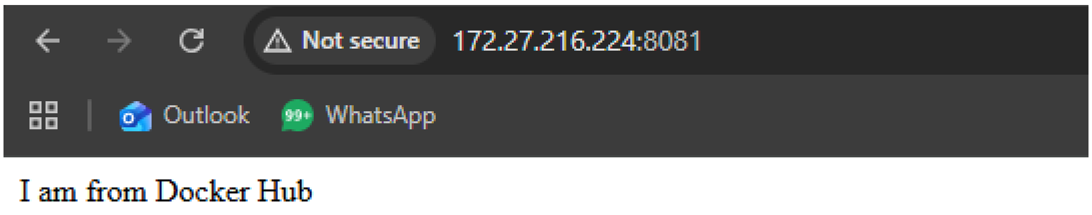
docker run -d -p 8082:80 --name I_AM_FROM_ECR your_aws_ecr_image


```
root@Angrybird:/home/kaivalya/AWSECR# docker run -d -p 8082:80 --name I_AM_FROM_ECR custom_awsecr
d233d9dcfb41cbb7d0b8ad4752f829b93b06ca57056c41cb8f3f932b45711f7d
root@Angrybird:/home/kaivalya/AWSECR# docker ps
CONTAINER ID   IMAGE           COMMAND                  CREATED         STATUS         PORTS                                   NAMES
d233d9dcfb41   custom_awsecr   "/usr/sbin/apachectl…"   6 seconds ago   Up 5 seconds   0.0.0.0:8082->80/tcp, [::]:8082->80/tcp   I_AM_FROM_ECR
```

36. Access Pages from Browser:

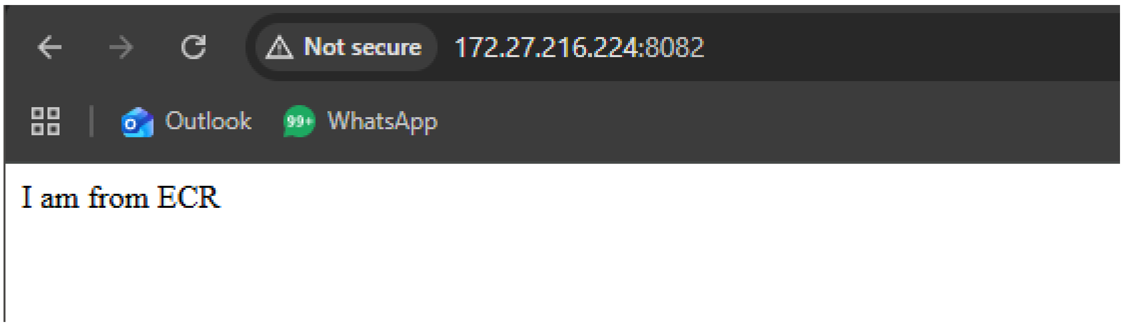
Open a web browser and access the pages:

For Docker Hub: http://localhost:8081



I am from Docker Hub

For AWS ECR: http://localhost:8082



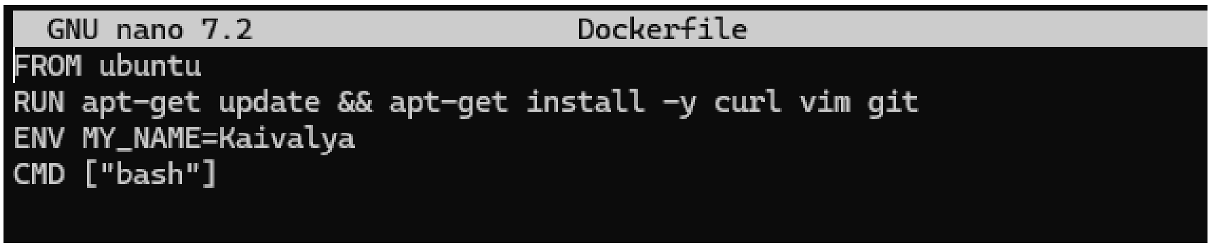
I am from ECR

Task-3

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

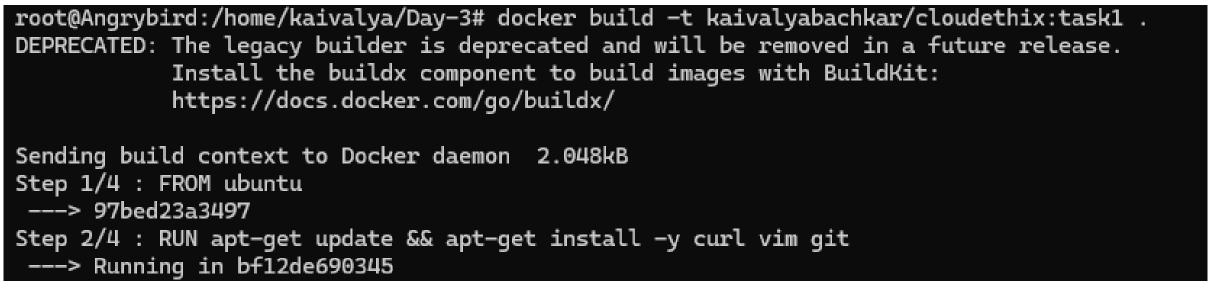
    - Base image: Ubuntu
    - Install packages: curl, vim, and git
    - Set an environment variable: MY_NAME=Your_Name
    - Build the custom image using docker build and list all available images using docker images.

Dockerfile:-



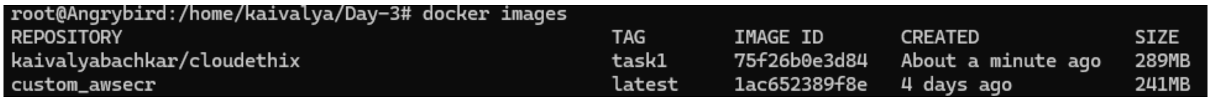
```
  GNU nano 7.2                          Dockerfile
FROM ubuntu
RUN apt-get update && apt-get install -y curl vim git
ENV MY_NAME=Kaivalya
CMD ["bash"]
```

Docker Build:-



```
root@Angrybird:/home/kaivalya/Day-3# docker build -t kaivalyabachkar/cloudethix:task1 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  2.048kB
Step 1/4 : FROM ubuntu
 ---> 97bed23a3497
Step 2/4 : RUN apt-get update && apt-get install -y curl vim git
 ---> Running in bf12de690345
```

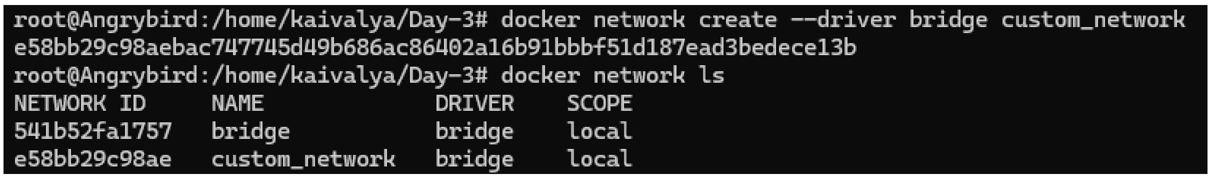
Docker Images:-



```
root@Angrybird:/home/kaivalya/Day-3# docker images
REPOSITORY                                              TAG       IMAGE ID       CREATED              SIZE
kaivalyabachkar/cloudethix                              task1     75f26b0e3d84   About a minute ago   289MB
custom_awsecr                                           latest    1ac652389f8e   4 days ago           241MB
```

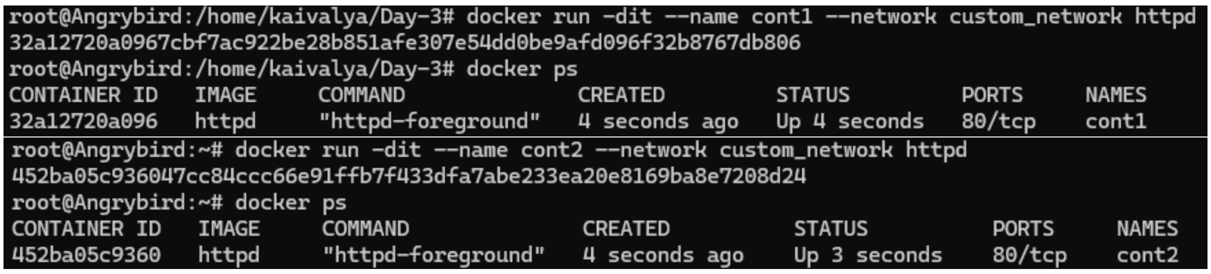
38. **Docker networking**: Create two Docker containers with different names, and add them to the same custom network. Verify that the containers can communicate with each other using their names as hostnames.

Custom-Network:-



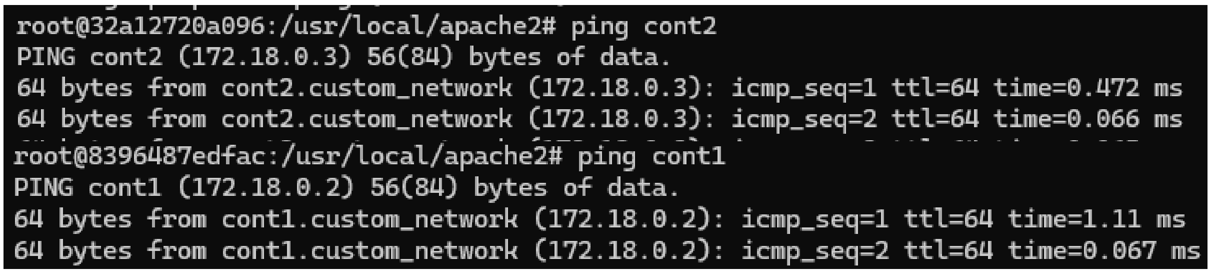
```
root@Angrybird:/home/kaivalya/Day-3# docker network create --driver bridge custom_network
e58bb29c98aebac747745d49b686ac86402a16b91bbbf51d187ead3bedece13b
root@Angrybird:/home/kaivalya/Day-3# docker network ls
NETWORK ID     NAME             DRIVER    SCOPE
541b52fa1757   bridge           bridge    local
e58bb29c98ae   custom_network   bridge    local
```

Two Docker Container:-



```
root@Angrybird:/home/kaivalya/Day-3# docker run -dit --name cont1 --network custom_network httpd
32a12720a0967cbf7ac922be28b851afe307e54dd0be9afd096f32b8767db806
root@Angrybird:/home/kaivalya/Day-3# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED         STATUS         PORTS     NAMES
32a12720a096   httpd     "httpd-foreground"   4 seconds ago   Up 4 seconds   80/tcp    cont1
root@Angrybird:~# docker run -dit --name cont2 --network custom_network httpd
452ba05c936047cc84ccc66e91ffb7f433dfa7abe233ea20e8169ba8e7208d24
root@Angrybird:~# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED         STATUS         PORTS     NAMES
452ba05c9360   httpd     "httpd-foreground"   4 seconds ago   Up 3 seconds   80/tcp    cont2
```

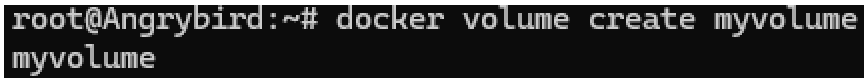
Verify the Containers can Communicate with each other using hostnames:-



```
root@32a12720a096:/usr/local/apache2# ping cont2
PING cont2 (172.18.0.3) 56(84) bytes of data.
64 bytes from cont2.custom_network (172.18.0.3): icmp_seq=1 ttl=64 time=0.472 ms
64 bytes from cont2.custom_network (172.18.0.3): icmp_seq=2 ttl=64 time=0.066 ms
root@8396487edfac:/usr/local/apache2# ping cont1
PING cont1 (172.18.0.2) 56(84) bytes of data.
64 bytes from cont1.custom_network (172.18.0.2): icmp_seq=1 ttl=64 time=1.11 ms
64 bytes from cont1.custom_network (172.18.0.2): icmp_seq=2 ttl=64 time=0.067 ms
```

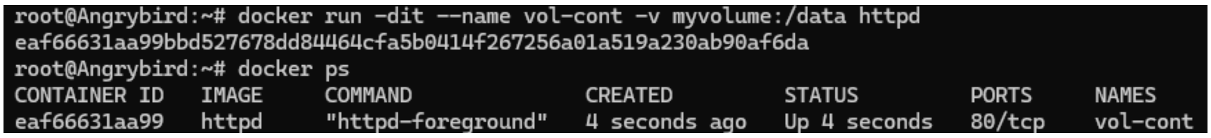
39. **Data persistence with volumes**: Run a Docker container with an attached volume, create a file inside the volume, and verify that the file persists after the container is stopped and removed.

New-Volume:-



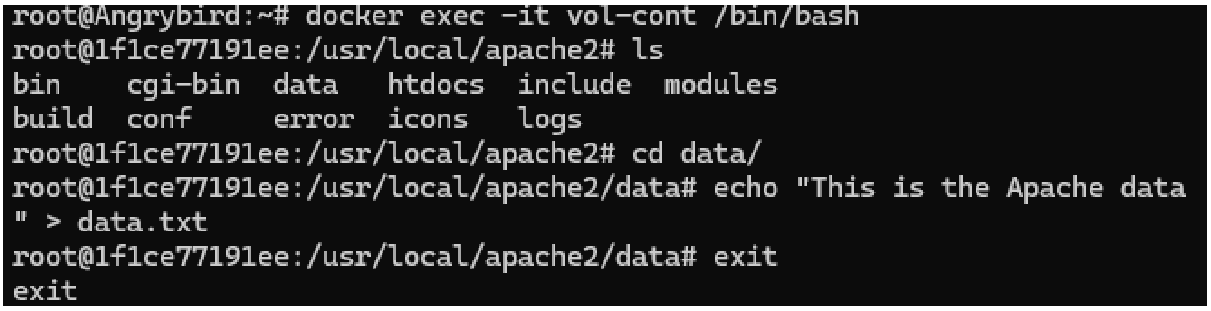
```
root@Angrybird:~# docker volume create myvolume
myvolume
```

Docker container with an attached volume:-



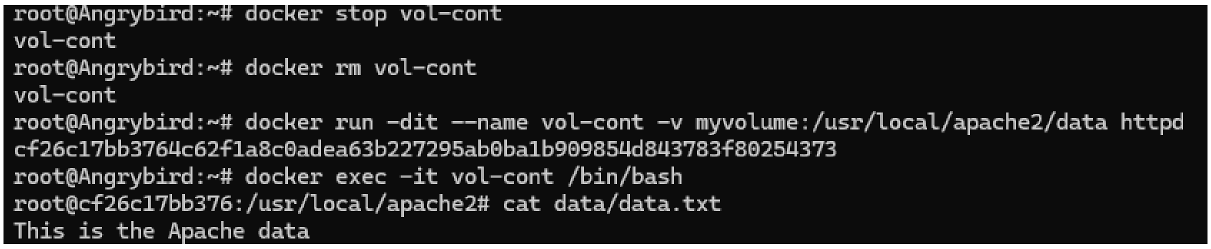
```
root@Angrybird:~# docker run -dit --name vol-cont -v myvolume:/data httpd
eaf66631aa99bbd527678dd84464cfa5b0414f267256a01a519a230ab90af6da
root@Angrybird:~# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED         STATUS         PORTS     NAMES
eaf66631aa99   httpd     "httpd-foreground"   4 seconds ago   Up 4 seconds   80/tcp    vol-cont
```

Create a file inside the volume:-



```
root@Angrybird:~# docker exec -it vol-cont /bin/bash
root@1f1ce77191ee:/usr/local/apache2# ls
bin    cgi-bin  data   htdocs  include  modules
build  conf     error  icons   logs
root@1f1ce77191ee:/usr/local/apache2# cd data/
root@1f1ce77191ee:/usr/local/apache2/data# echo "This is the Apache data
" > data.txt
root@1f1ce77191ee:/usr/local/apache2/data# exit
exit
```

Verify that the file persists after the container is stopped and removed:-

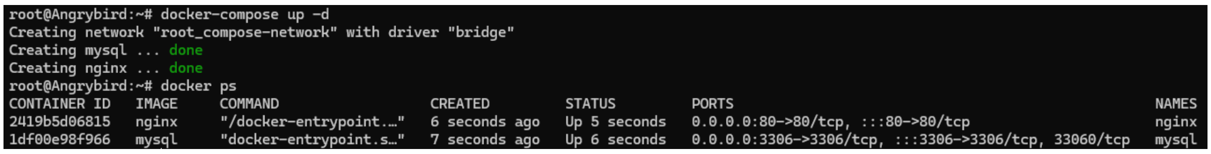


```
root@Angrybird:~# docker stop vol-cont
vol-cont
root@Angrybird:~# docker rm vol-cont
vol-cont
root@Angrybird:~# docker run -dit --name vol-cont -v myvolume:/usr/local/apache2/data httpd
cf26c17bb3764c62f1a8c0adea63b227295ab0ba1b909854d843783f80254373
root@Angrybird:~# docker exec -it vol-cont /bin/bash
root@cf26c17bb376:/usr/local/apache2# cat data/data.txt
This is the Apache data
```

40. **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.

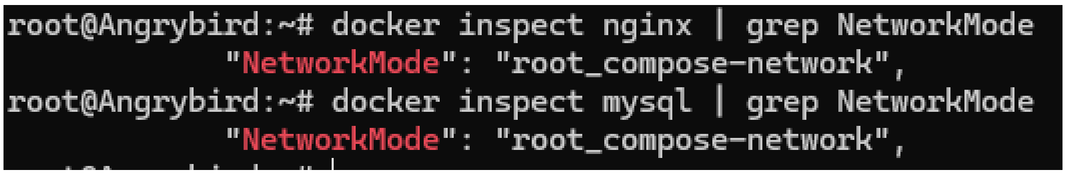
Create a docker-compose.yml file:-

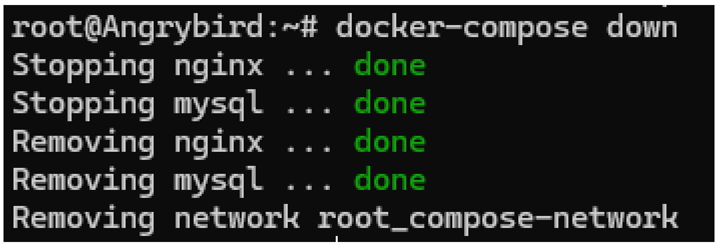
```
services:
  web:
    image: nginx
    container_name: nginx
    ports:
      - "80:80"
    networks:
      - compose-network
    volumes:
      - mydir:/usr/share/nginx/html
    depends_on:
      - database

  database:
    image: mysql
    container_name: mysql
    ports:
      - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: Pass@123
      MYSQL_DATABASE: mydb
    networks:
      - compose-network
    volumes:
      - mydata:/var/lib/mysql

networks:
  compose-network:
    driver: bridge


volumes:
  mydir:
  mydata:
```



```
root@Angrybird:~# docker-compose up -d
Creating network "root_compose-network" with driver "bridge"
Creating mysql ... done
Creating nginx ... done
root@Angrybird:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                                                  NAMES
2419b5d06815   nginx     "/docker-entrypoint.…"   6 seconds ago   Up 5 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp                      nginx
1df00e98f966   mysql     "docker-entrypoint.s…"   7 seconds ago   Up 6 seconds   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   mysql
```

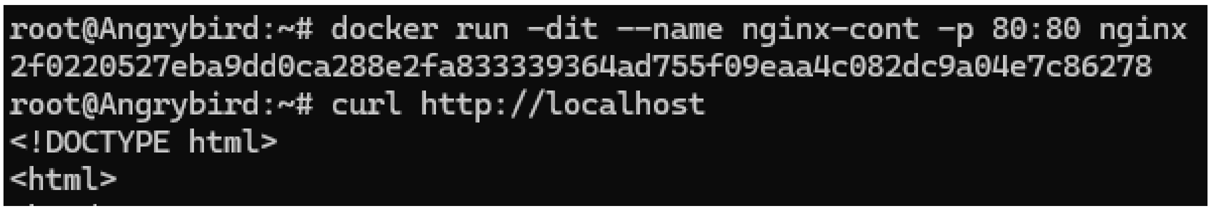
Ensure that both containers are connected to the same network:-


```
root@Angrybird:~# docker inspect nginx | grep NetworkMode
            "NetworkMode": "root_compose-network",
root@Angrybird:~# docker inspect mysql | grep NetworkMode
            "NetworkMode": "root_compose-network",
```


```
root@Angrybird:~# docker-compose down
Stopping nginx ... done
Stopping mysql ... done
Removing nginx ... done
Removing mysql ... done
Removing network root_compose-network
```

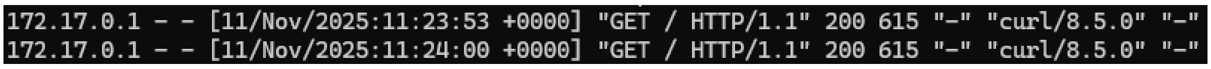
41. **Container logging**: Run a Docker container that generates log output, and practice using docker logs to view and analyze the logs.

Run a Docker container that generates log output:-


```
root@Angrybird:~# docker run -dit --name nginx-cont -p 80:80 nginx
2f0220527eba9dd0ca288e2fa833339364ad755f09eaa4c082dc9a04e7c86278
root@Angrybird:~# curl http://localhost
<!DOCTYPE html>
<html>
```

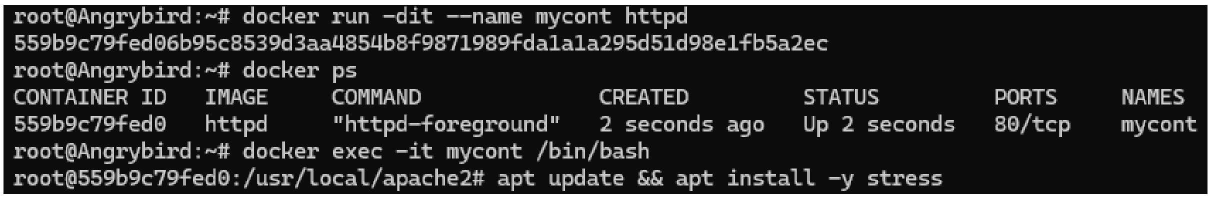
Docker logs to view and analyze the logs:-

View All Logs for the Container:-


```
172.17.0.1 - - [11/Nov/2025:11:23:53 +0000] "GET / HTTP/1.1" 200 615 "-" "curl/8.5.0" "-"
172.17.0.1 - - [11/Nov/2025:11:24:00 +0000] "GET / HTTP/1.1" 200 615 "-" "curl/8.5.0" "-"
```

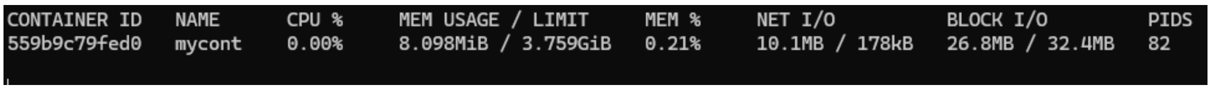
42. Container resource monitoring: Run a Docker container that consumes system resources (e.g., CPU or memory) and practice using docker stats and docker top to monitor the container's resource usage.

Run a Docker container:-

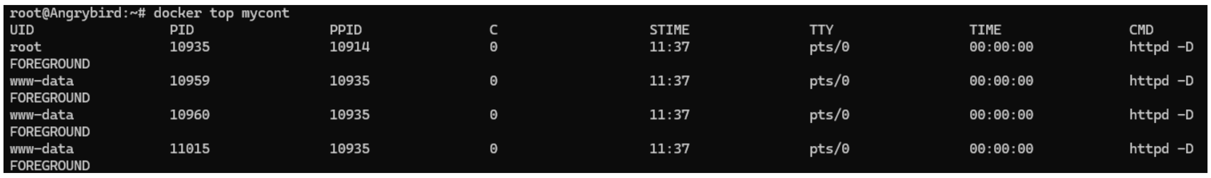

```
root@Angrybird:~# docker run -dit --name mycont httpd
559b9c79fed06b95c8539d3aa4854b8f9871989fda1a1a295d51d98e1fb5a2ec
root@Angrybird:~# docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED         STATUS         PORTS     NAMES
559b9c79fed0   httpd     "httpd-foreground"   2 seconds ago   Up 2 seconds   80/tcp    mycont
root@Angrybird:~# docker exec -it mycont /bin/bash
root@559b9c79fed0:/usr/local/apache2# apt update && apt install -y stress
```

Monitor the container's resource usage:-

Using docker stats:-

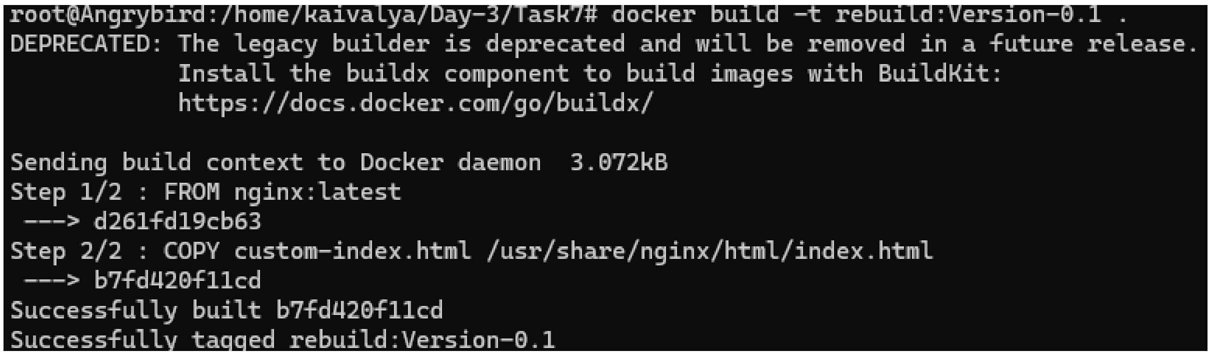

```
CONTAINER ID   NAME     CPU %     MEM USAGE / LIMIT     MEM %     NET I/O          BLOCK I/O         PIDS
559b9c79fed0   mycont   0.00%     8.098MiB / 3.759GiB   0.21%     10.1MB / 178kB   26.8MB / 32.4MB   82
```

Using docker top:-

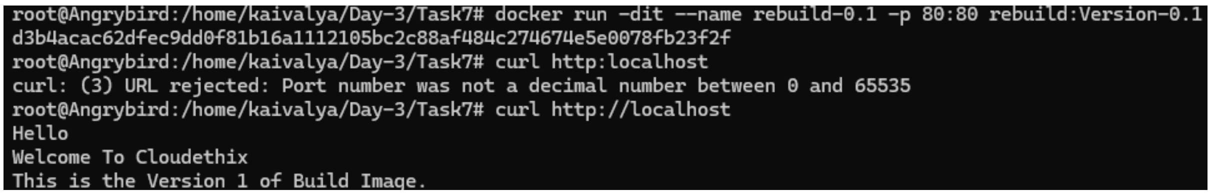

```
root@Angrybird:~# docker top mycont
UID                 PID                 PPID                C                   STIME               TTY                 TIME                CMD
root                10935               10914               0                   11:37               pts/0               00:00:00            httpd -D
FOREGROUND
www-data            10959               10935               0                   11:37               pts/0               00:00:00            httpd -D
FOREGROUND
www-data            10960               10935               0                   11:37               pts/0               00:00:00            httpd -D
FOREGROUND
www-data            11015               10935               0                   11:37               pts/0               00:00:00            httpd -D
FOREGROUND
```

43. **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 Image:-

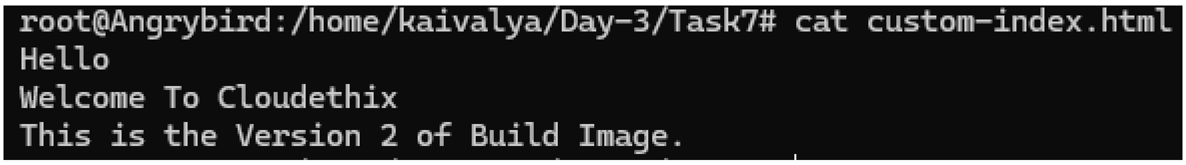


```
root@Angrybird:/home/kaivalya/Day-3/Task7# docker build -t rebuild:Version-0.1 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.072kB
Step 1/2 : FROM nginx:latest
 ---> d261fd19cb63
Step 2/2 : COPY custom-index.html /usr/share/nginx/html/index.html
 ---> b7fd420f11cd
Successfully built b7fd420f11cd
Successfully tagged rebuild:Version-0.1
```



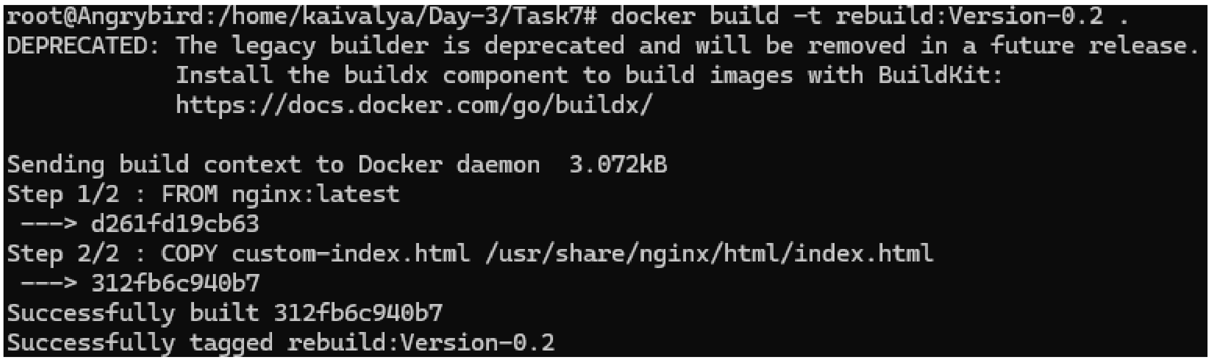
```
root@Angrybird:/home/kaivalya/Day-3/Task7# docker run -dit --name rebuild-0.1 -p 80:80 rebuild:Version-0.1
d3b4acac62dfec9dd0f81b16a1112105bc2c88af484c274674e5e0078fb23f2f
root@Angrybird:/home/kaivalya/Day-3/Task7# curl http:localhost
curl: (3) URL rejected: Port number was not a decimal number between 0 and 65535
root@Angrybird:/home/kaivalya/Day-3/Task7# curl http://localhost
Hello
Welcome To Cloudethix
This is the Version 1 of Build Image.
```

Update the source:-



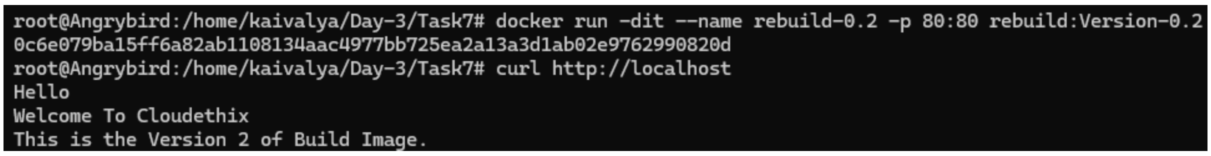
```
root@Angrybird:/home/kaivalya/Day-3/Task7# cat custom-index.html
Hello
Welcome To Cloudethix
This is the Version 2 of Build Image.
```

Rebuild the Docker Image:-



```
root@Angrybird:/home/kaivalya/Day-3/Task7# docker build -t rebuild:Version-0.2 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  3.072kB
Step 1/2 : FROM nginx:latest
 ---> d261fd19cb63
Step 2/2 : COPY custom-index.html /usr/share/nginx/html/index.html
 ---> 312fb6c940b7
Successfully built 312fb6c940b7
Successfully tagged rebuild:Version-0.2
```

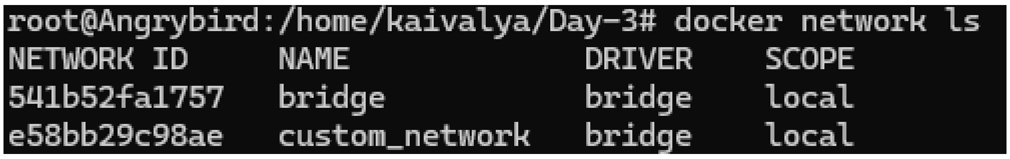
Deploy the updated Version:-



```
root@Angrybird:/home/kaivalya/Day-3/Task7# docker run -dit --name rebuild-0.2 -p 80:80 rebuild:Version-0.2
0c6e079ba15ff6a82ab1108134aac4977bb725ea2a13a3d1ab02e9762990820d
root@Angrybird:/home/kaivalya/Day-3/Task7# curl http://localhost
Hello
Welcome To Cloudethix
This is the Version 2 of Build Image.
```

44. **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.

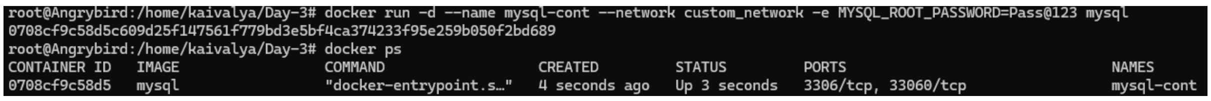
Create a Custom Bridge Network:-



```
root@Angrybird:/home/kaivalya/Day-3# docker network ls
NETWORK ID     NAME             DRIVER    SCOPE
541b52fa1757   bridge           bridge    local
e58bb29c98ae   custom_network   bridge    local
```

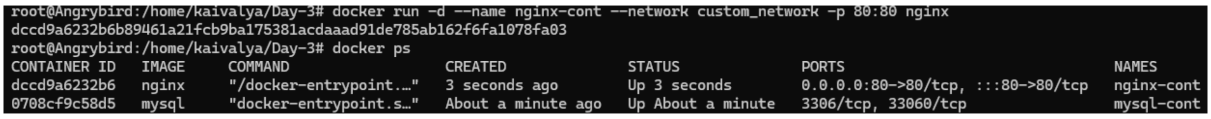
Create two Docker containers using Custom-network:-

MySQL:-



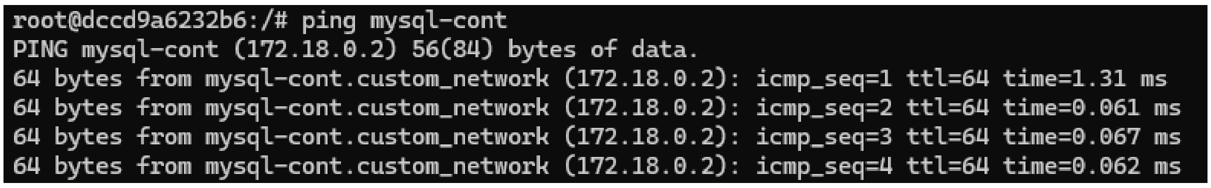
```
root@Angrybird:/home/kaivalya/Day-3# docker run -d --name mysql-cont --network custom_network -e MYSQL_ROOT_PASSWORD=Pass@123 mysql
0708cf9c58d5c609d25f147561f779bd3e5bf4ca374233f95e259b050f2bd689
root@Angrybird:/home/kaivalya/Day-3# docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATED         STATUS         PORTS                           NAMES
0708cf9c58d5   mysql                "docker-entrypoint.s…"   4 seconds ago   Up 3 seconds   3306/tcp, 33060/tcp             mysql-cont
```

NGINX:-



```
root@Angrybird:/home/kaivalya/Day-3# docker run -d --name nginx-cont --network custom_network -p 80:80 nginx
dccd9a6232b6b89461a21fcb9ba175381acdaaad91de785ab162f6fa1078fa03
root@Angrybird:/home/kaivalya/Day-3# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS              PORTS                                NAMES
dccd9a6232b6   nginx     "/docker-entrypoint.…"   3 seconds ago        Up 3 seconds        0.0.0.0:80->80/tcp, :::80->80/tcp    nginx-cont
0708cf9c58d5   mysql     "docker-entrypoint.s…"   About a minute ago   Up About a minute   3306/tcp, 33060/tcp                  mysql-cont
```

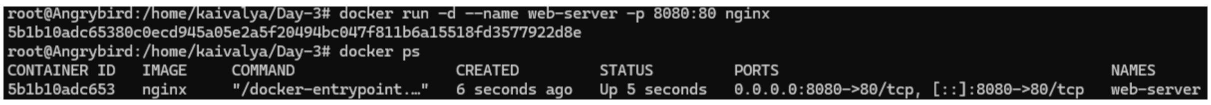
Communicate with each other:-



```
root@dccd9a6232b6:/# ping mysql-cont
PING mysql-cont (172.18.0.2) 56(84) bytes of data.
64 bytes from mysql-cont.custom_network (172.18.0.2): icmp_seq=1 ttl=64 time=1.31 ms
64 bytes from mysql-cont.custom_network (172.18.0.2): icmp_seq=2 ttl=64 time=0.061 ms
64 bytes from mysql-cont.custom_network (172.18.0.2): icmp_seq=3 ttl=64 time=0.067 ms
64 bytes from mysql-cont.custom_network (172.18.0.2): icmp_seq=4 ttl=64 time=0.062 ms
```

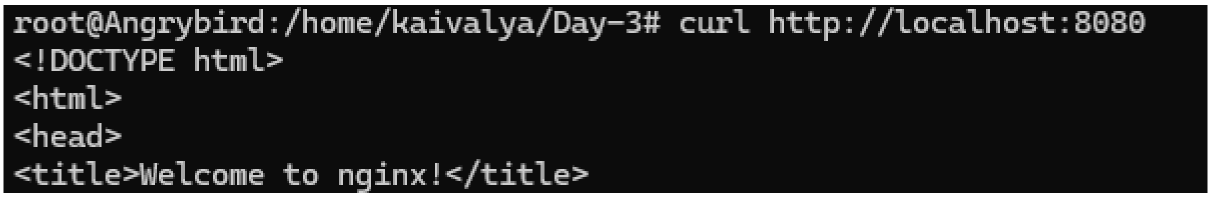
45. **Port mapping**: Run a containerized web server (e.g., NGINX or Apache) and map its default port (80 or 443) to a custom port on the host machine. Verify that the web server is accessible through the custom port on the host.

Created a web-server using nginx and map the default port 80 to custom port 8080:-

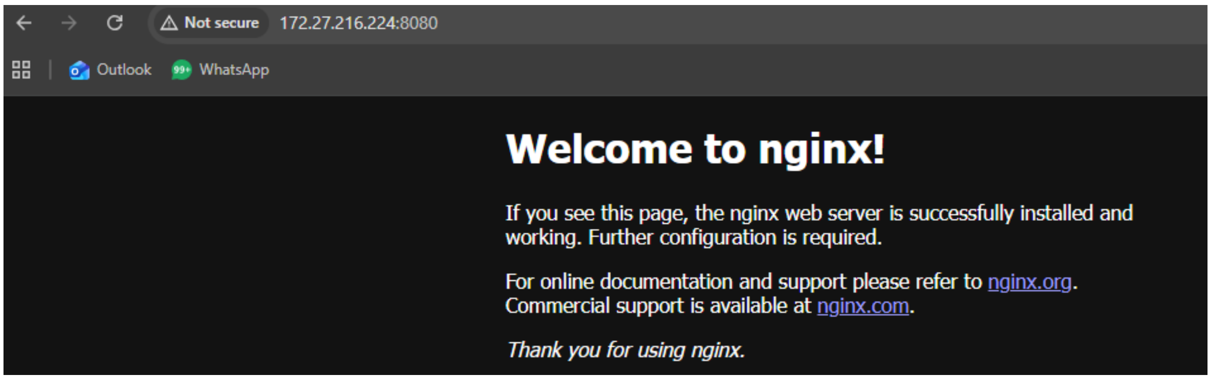


```
root@Angrybird:/home/kaivalya/Day-3# docker run -d --name web-server -p 8080:80 nginx
5b1b10adc65380c0ecd945a05e2a5f20494bc047f811b6a15518fd3577922d8e
root@Angrybird:/home/kaivalya/Day-3# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                                      NAMES
5b1b10adc653   nginx     "/docker-entrypoint.…"   6 seconds ago   Up 5 seconds   0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   web-server
```

Accessing through the custom port on the host machine:-



```
root@Angrybird:/home/kaivalya/Day-3# curl http://localhost:8080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
```



Welcome to nginx!

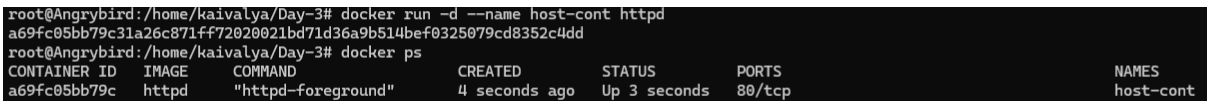
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

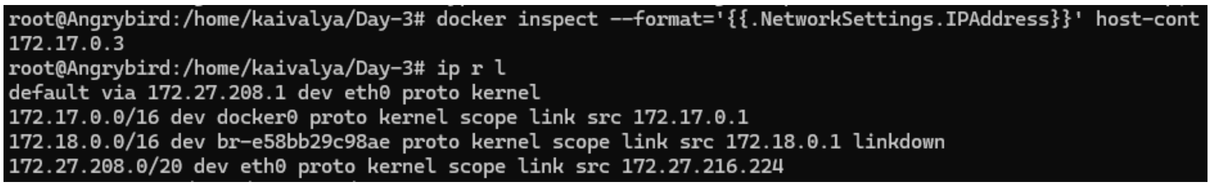
46. **Host networking**: Run a container with the host networking mode and compare its network configuration with that of the host machine. Explain the advantages and disadvantages of using host networking for containers.

Run a container with the host networking mode:-



```
root@Angrybird:/home/kaivalya/Day-3# docker run -d --name host-cont httpd
a69fc05bb79c31a26c871ff72020021bd71d36a9b514bef0325079cd8352c4dd
root@Angrybird:/home/kaivalya/Day-3# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                                      NAMES
a69fc05bb79c   httpd     "httpd-foreground"       4 seconds ago   Up 3 seconds   80/tcp                                     host-cont
```

Compare its network configuration with that of the host machine:-



```
root@Angrybird:/home/kaivalya/Day-3# docker inspect --format='{{.NetworkSettings.IPAddress}}' host-cont
172.17.0.3
root@Angrybird:/home/kaivalya/Day-3# ip r l
default via 172.27.208.1 dev eth0 proto kernel
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
172.18.0.0/16 dev br-e58bb29c98ae proto kernel scope link src 172.18.0.1 linkdown
172.27.208.0/20 dev eth0 proto kernel scope link src 172.27.216.224
```

Advantages and disadvantages of using host networking for containers:-

Advantages of Host Networking

- **Performance:**

Host networking can offer slight performance advantages by removing the overhead of network address translation (NAT) and virtual network interfaces, as the container directly uses the host's network stack.

- **Simplified Port Management:**

When an application in a container binds to a port, it is directly accessible on that same port on the host's IP address, eliminating the need for port mapping. This can simplify configurations for applications requiring access to a large range of ports.

- **Direct Access to Host Network Services:**

Containers can directly access services running on the host's network interfaces without any routing or NAT in between.

Disadvantages of Host Networking

- **Reduced Isolation and Security:**

The primary disadvantage is the loss of network isolation. The container shares the host's network namespace, potentially exposing the host to vulnerabilities within the container and vice versa. Malicious containers could potentially interfere with host network services or other containers.

- **Port Conflicts:**

If multiple containers or a host process try to bind to the same port, conflicts can arise, leading to application failures.

- **Limited Scalability and Portability:**

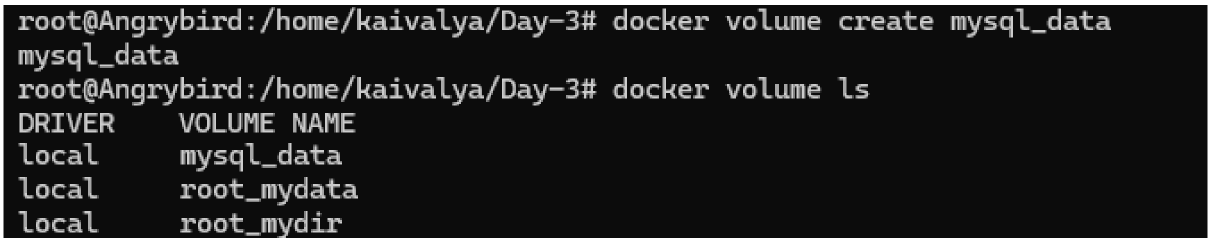
Host networking makes it harder to scale applications across multiple hosts or migrate them, as port availability and network configurations are tied to the specific host.

- **No Container IP Address:**

Containers using host networking do not have their own distinct IP addresses, making it challenging for service discovery or for other containers to communicate with them directly using IP addresses.

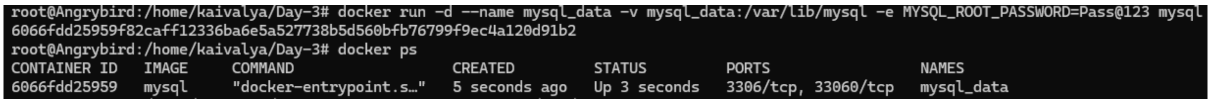
47. **Named volume**: Launch a containerized database (e.g., MySQL or PostgreSQL) using a named volume to store its data. Stop and remove the container, then recreate it using the same named volume. Verify that the data persists across container recreations.

Create a Volume for Database:-



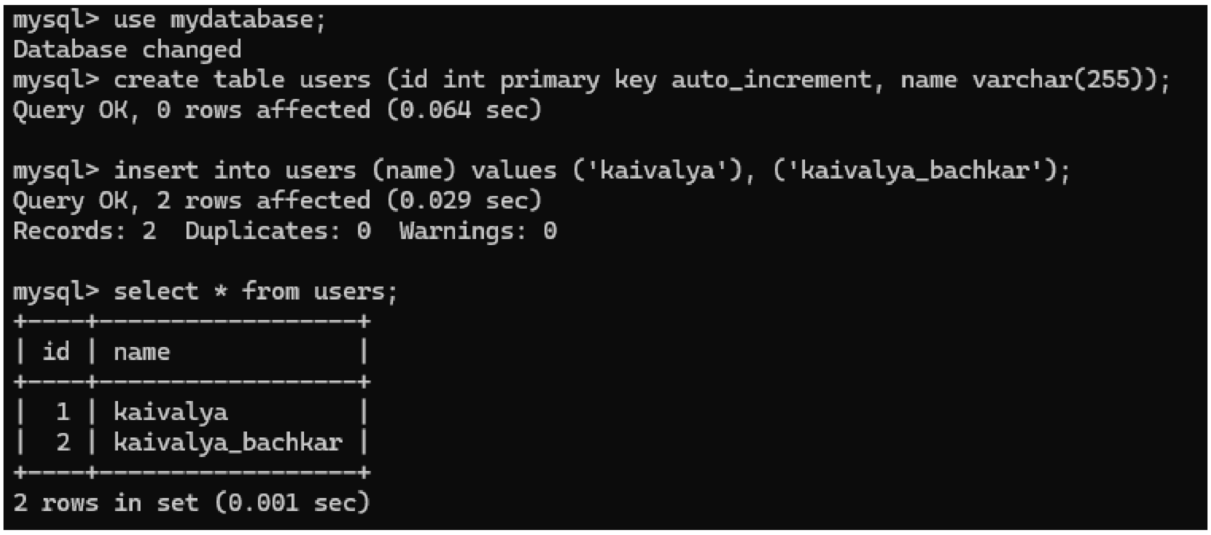
```
root@Angrybird:/home/kaivalya/Day-3# docker volume create mysql_data
mysql_data
root@Angrybird:/home/kaivalya/Day-3# docker volume ls
DRIVER    VOLUME NAME
local     mysql_data
local     root_mydata
local     root_mydir
```

Create a Container for Database with Volume:-



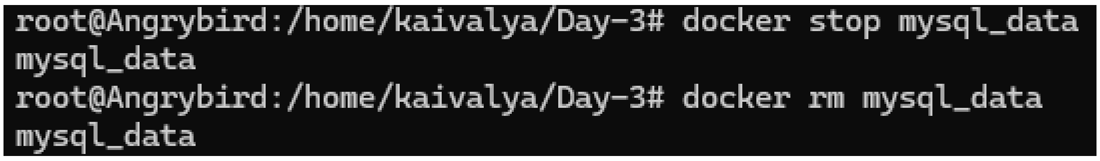
```
root@Angrybird:/home/kaivalya/Day-3# docker run -d --name mysql_data -v mysql_data:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=Pass@123 mysql
6066fdd25959f82caff12336ba6e5a527738b5d560bfb76799f9ec4a120d91b2
root@Angrybird:/home/kaivalya/Day-3# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                 NAMES
6066fdd25959   mysql     "docker-entrypoint.s…"   5 seconds ago   Up 3 seconds   3306/tcp, 33060/tcp   mysql_data
```

Create a Some data inside the Database:-



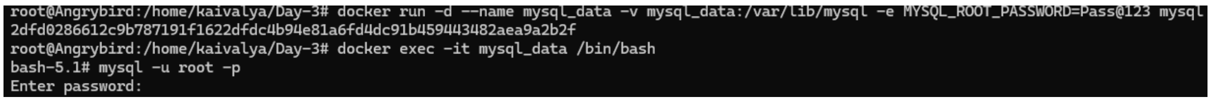
```
mysql> use mydatabase;
Database changed
mysql> create table users (id int primary key auto_increment, name varchar(255));
Query OK, 0 rows affected (0.064 sec)

mysql> insert into users (name) values ('kaivalya'), ('kaivalya_bachkar');
Query OK, 2 rows affected (0.029 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from users;
+----+------------------+
| id | name             |
+----+------------------+
|  1 | kaivalya         |
|  2 | kaivalya_bachkar |
+----+------------------+
2 rows in set (0.001 sec)
```

Stop and remove the container:-



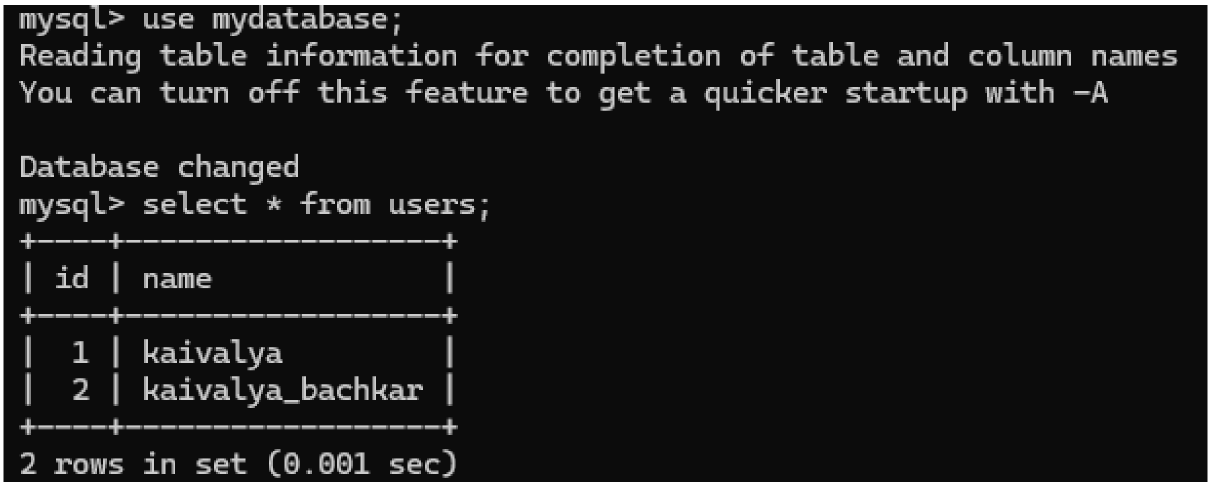
```
root@Angrybird:/home/kaivalya/Day-3# docker stop mysql_data
mysql_data
root@Angrybird:/home/kaivalya/Day-3# docker rm mysql_data
mysql_data
```

Recreate it using the same named volume:-



```
root@Angrybird:/home/kaivalya/Day-3# docker run -d --name mysql_data -v mysql_data:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=Pass@123 mysql
2dfd0286612c9b787191f1622dfdc4b94e81a6fd4dc91b459443482aea9a2b2f
root@Angrybird:/home/kaivalya/Day-3# docker exec -it mysql_data /bin/bash
bash-5.1# mysql -u root -p
Enter password:
```

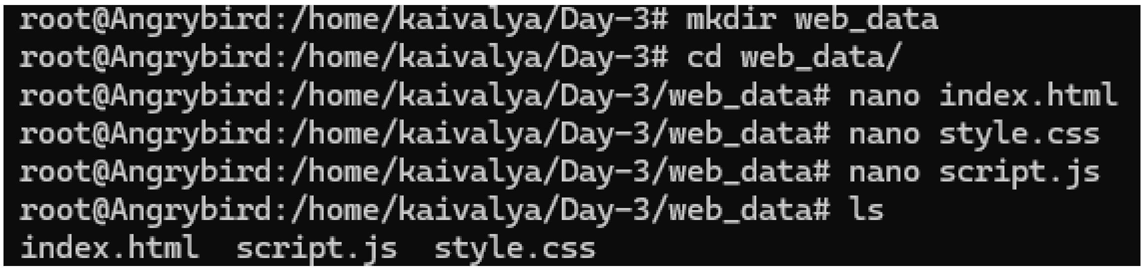
Verify that the data persists across container recreations:-



```
mysql> use mydatabase;
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> select * from users;
+----+------------------+
| id | name             |
+----+------------------+
|  1 | kaivalya         |
|  2 | kaivalya_bachkar |
+----+------------------+
2 rows in set (0.001 sec)
```

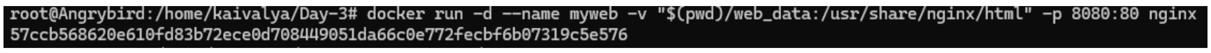
48. **Bind mount**: Create a simple web application with a local directory containing HTML, CSS, and JavaScript files. Run a web server container (e.g., NGINX or Apache) and bind mount the local directory to the container's webroot. Verify that the web application is accessible and update the local files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

Create a web application with a local directory containing HTML, CSS, & JavaScript files:-



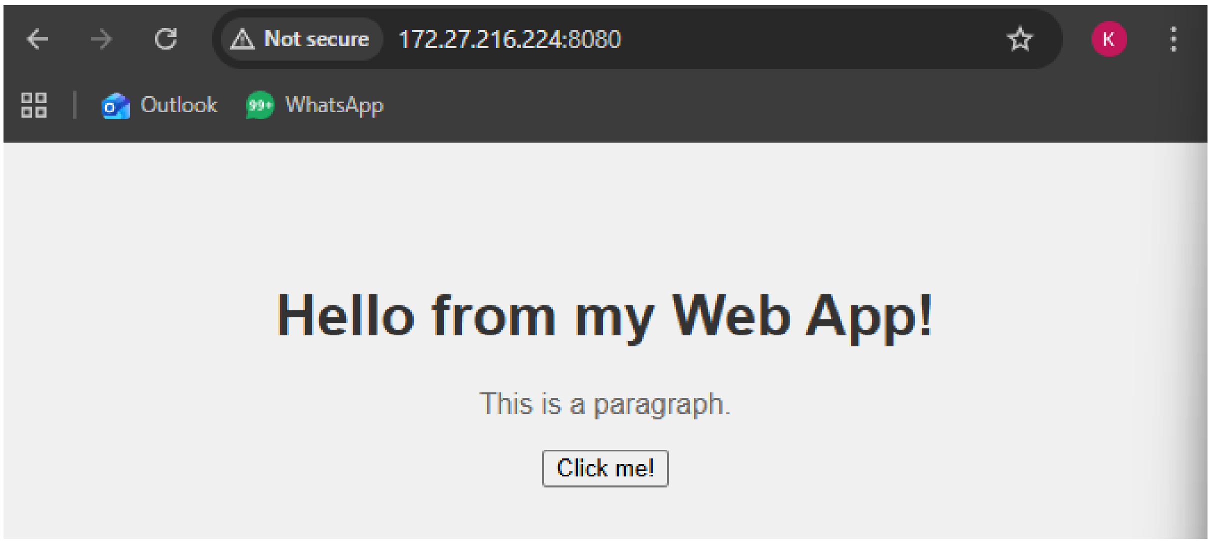
```
root@Angrybird:/home/kaivalya/Day-3# mkdir web_data
root@Angrybird:/home/kaivalya/Day-3# cd web_data/
root@Angrybird:/home/kaivalya/Day-3/web_data# nano index.html
root@Angrybird:/home/kaivalya/Day-3/web_data# nano style.css
root@Angrybird:/home/kaivalya/Day-3/web_data# nano script.js
root@Angrybird:/home/kaivalya/Day-3/web_data# ls
index.html  script.js  style.css
```

Run a container using nginx and bind mount the local directory to container's Webroot:-



```
root@Angrybird:/home/kaivalya/Day-3# docker run -d --name myweb -v "$(pwd)/web_data:/usr/share/nginx/html" -p 8080:80 nginx
57ccb568620e610fd83b72ece0d708449051da66c0e772fecbf6b07319c5e576
```

Verify that the web application is accessible:-

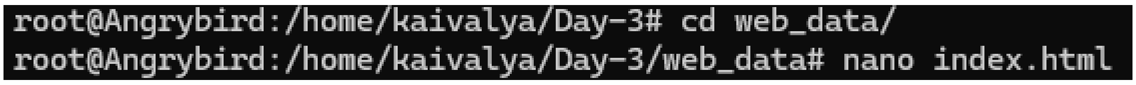


Not secure 172.27.216.224:8080

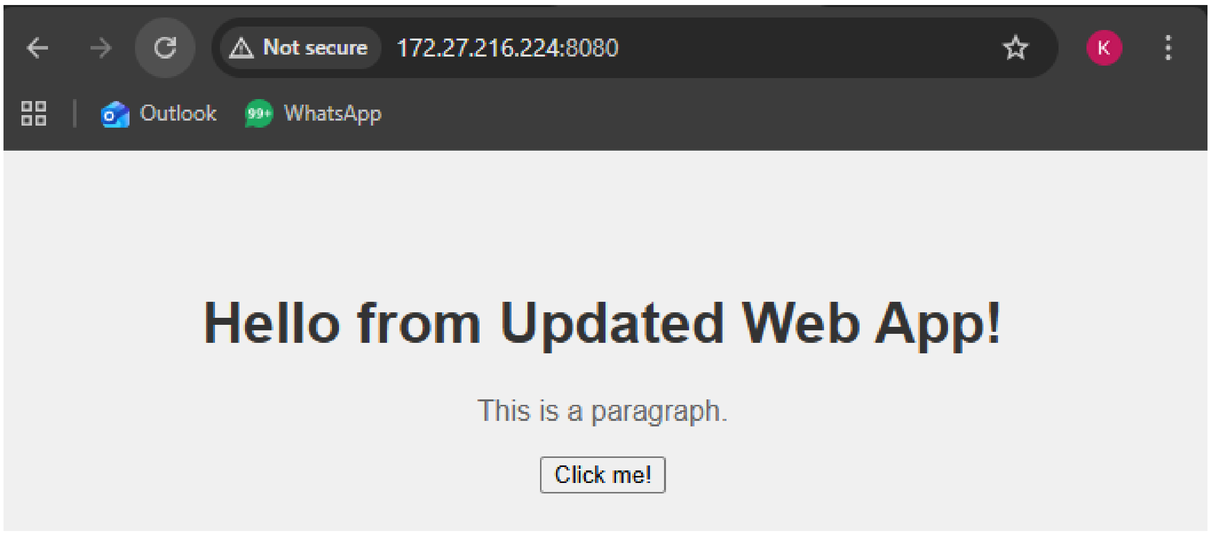
Hello from my Web App!

This is a paragraph.

update the local files to see if changes are reflected in the container:-



```
root@Angrybird:/home/kaivalya/Day-3# cd web_data/
root@Angrybird:/home/kaivalya/Day-3/web_data# nano index.html
```



Not secure 172.27.216.224:8080

Hello from Updated Web App!

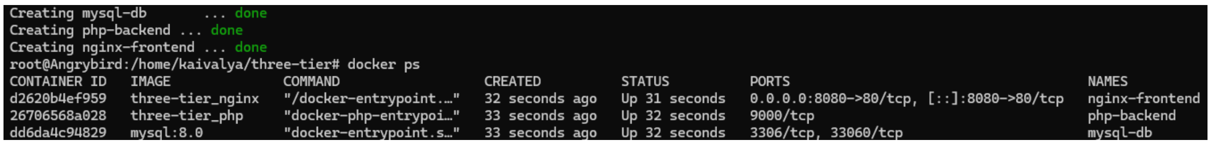
This is a paragraph.

50. **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

Docker Compose File:-

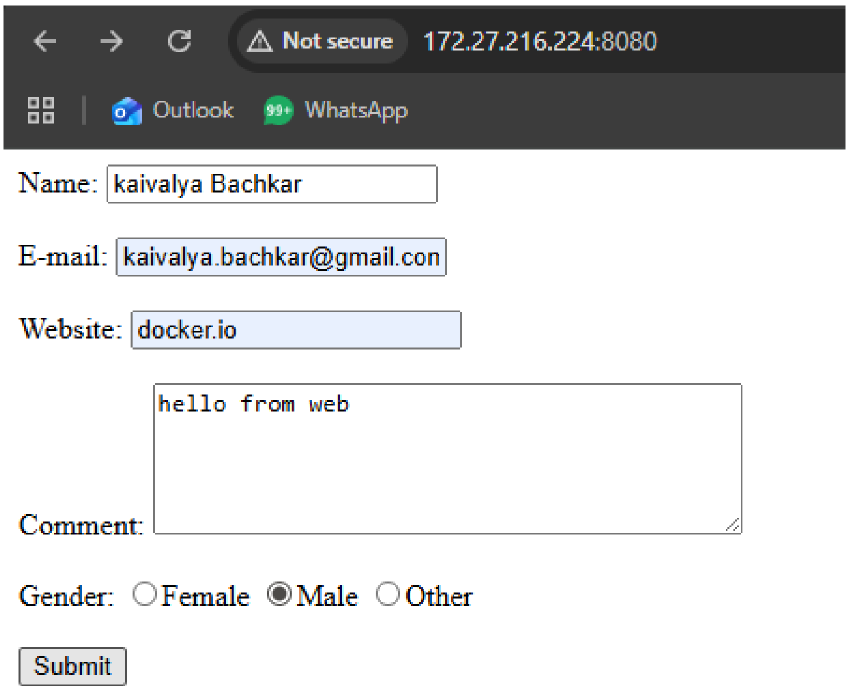
```
version: "3.9"


services:
  nginx:
    build: ./nginx
    container_name: nginx-frontend
    ports:
      - "8080:80"
    depends_on:
      - php
    volumes:
      - ./nginx/index.html:/usr/share/nginx/html/index.html
      - ./nginx/default.conf:/etc/nginx/conf.d/default.conf
    networks:
      - app-network


  php:
    build: ./php
    container_name: php-backend
    volumes:
      - ./php:/var/www/html
    networks:
      - app-network


  mysql:
    image: mysql:8.0
    container_name: mysql-db
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: Pass@123
      MYSQL_DATABASE: mydatabse
    volumes:
      - db_data:/var/lib/mysql
      - ./mysql/init.sql:/docker-entrypoint-initdb.d/init.sql
    networks:
      - app-network

networks:
  app-network:

volumes:
  db_data:
```

Docker Compose up:-



```
Creating mysql-db       ... done
Creating php-backend ... done
Creating nginx-frontend ... done
root@Angrybird:/home/kaivalya/three-tier# docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS          PORTS                                   NAMES
d2620b4ef959   three-tier_nginx   "/docker-entrypoint.…"   32 seconds ago   Up 31 seconds   0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   nginx-frontend
26706568a028   three-tier_php     "docker-php-entrypoi…"   33 seconds ago   Up 32 seconds   9000/tcp                                php-backend
dd6da4c94829   mysql:8.0          "docker-entrypoint.s…"   33 seconds ago   Up 32 seconds   3306/tcp, 33060/tcp                     mysql-db
```

Access a Frontend and Enter some data:-



172.27.216.224:8080

Name: kaivalya Bachkar

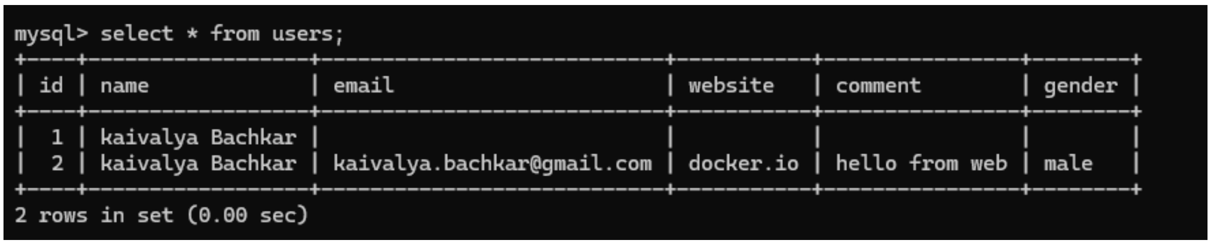
E-mail: kaivalya.bachkar@gmail.con

Website: docker.io

Comment: hello from web

Gender: Female [✓ Male] Other

Verify that data is stored in Database:-



```
mysql> select * from users;
+----+------------------+---------------------------+------------+-----------------+--------+
| id | name             | email                     | website    | comment         | gender |
+----+------------------+---------------------------+------------+-----------------+--------+
|  1 | kaivalya Bachkar |                           |            |                 |        |
|  2 | kaivalya Bachkar | kaivalya.bachkar@gmail.com | docker.io | hello from web  | male   |
+----+------------------+---------------------------+------------+-----------------+--------+
2 rows in set (0.00 sec)
```