# DOCKER TASK
# MUHAMMAD REHAN KHAN

**Q1.Run two Docker Containers using different menthods.**

## IMPERATIVE METHOD:

```
controlplane $ docker run -d --name nginx-cont -p 81:80 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
faef57eae888: Pull complete
76579e9ed380: Pull complete
cf707e233955: Pull complete
91bb7937700d: Pull complete
4b962717ba55: Pull complete
f46d7b05649a: Pull complete
103501419a0a: Pull complete
Digest: sha256:08bc36ad52474e528cc1ea3426b5e3f4bad8a130318e3140d6cfe29c8892c7ef
Status: Downloaded newer image for nginx:latest
92727bf01e2fdeabb1200e428e440ee1e55a79ed98c43438877981439d29c153
controlplane $ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS
NAMES
92727bf01e2f   nginx     "/docker-entrypoint.…"   7 seconds ago   Up 6 seconds
0.0.0.0:81->80/tcp, :::81->80/tcp   nginx-cont
controlplane $ ^C
controlplane $
```

**DECLARATIVE METHOD:**

```
controlplane $ docker-compose up -d --build
Creating root_nginx_1 ... done
controlplane $ docker-compose down
Stopping root_nginx_1 ... done
Removing root_nginx_1 ... done
Removing network root_default
controlplane $ sudo nano docker-compose.yml
controlplane $ cat docker-compose.yml
version: '3'
services:
  nginx:
    image: nginx
    ports:
      - "82:80"
```

```
controlplane $ docker-compose up -d
Creating network "root_default" with the default driver
Creating root_nginx_1 ... done
```

**check container running condition:**

```
controlplane $ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS
NAMES
58f45be9310a   nginx     "/docker-entrypoint.…"   4 seconds ago   Up 4 seconds
0.0.0.0:82->80/tcp, :::82->80/tcp   root_nginx_1
92727bf01e2f   nginx     "/docker-entrypoint.…"   6 minutes ago   Up 6 minutes
0.0.0.0:81->80/tcp, :::81->80/tcp   nginx-cont
controlplane $
```

**2. Make Dockerfile for any 2 of well-known image (i.e Nginx, Busybox, alpine or etc) , run container and delete anyone.**

**create index.html file:**

```
controlplane $ sudo nano index.html
```

**create Dockerfile for nginx webserver:**

```
controlplane $ sudo nano Dockerfile.webserver
controlplane $ ls
Dockerfile.webserver  Y  filesystem  index.html
controlplane $ cat index.html
<!-- index.html -->
<!DOCTYPE html>
<html>
<head>
  <title>Hello from Nginx</title>
</head>
<body>
  <h1>Rehan Devops sherdil it accademy</h1>
</body>
</html>
controlplane $
controlplane $ cat Dockerfile.webserver
FROM nginx
COPY index.html /usr/share/nginx/html/index.html
EXPOSE 80
```

**create nginx image from Dockerfile:**

```
controlplane $ docker build -t my-nginx -f Dockerfile
"docker build" requires exactly 1 argument.
See 'docker build --help'.

Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile

controlplane $ docker build -t my-nginx -f Dockerfile.webserver .
Sending build context to Docker daemon  6.715MB
Step 1/3 : FROM nginx
latest: Pulling from library/nginx
faef57eae888: Pull complete
76579e9ed380: Pull complete
cf707e233955: Pull complete
91bb7937700d: Pull complete
4b962717ba55: Pull complete
f46d7b05649a: Pull complete
103501419a0a: Pull complete
Digest: sha256:08bc36ad52474e528cc1ea3426b5e3f4bad8a130318e3140d6cfe29c8892c7ef
Status: Downloaded newer image for nginx:latest
 ---> 021283c8eb95
Step 2/3 : COPY index.html /usr/share/nginx/html/index.html
 ---> bd9a483c5f84
Step 3/3 : EXPOSE 80
 ---> Running in 269dcafee03d
Removing intermediate container 269dcafee03d
 ---> 9bcd3f784c70
Successfully built 9bcd3f784c70
Successfully tagged my-nginx:latest
```

**check docker images:**

```
controlplane $ docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
my-nginx     latest    9bcd3f784c70   21 seconds ago   187MB
nginx        latest    021283c8eb95   2 weeks ago      187MB
```

**create container from nginx image:**

```
controlplane $ docker run -d --name nginx-cont -p 8080:80 my-nginx
1f01a31c25babcd3504c98017ba35c428c13c0fdf6fcefbf182876b16d2a4e4c
```

**check container in running condition:**

```
controlplane $ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS
NAMES
1f01a31c25ba   my-nginx   "/docker-entrypoint.…"   27 seconds ago   Up 15 seconds
0.0.0.0:8080->80/tcp, :::8080->80/tcp   nginx-cont
```

**stop container:**

```
controlplane $ docker stop nginx-cont
nginx-cont
```

**delete container:**

```
controlplane $ docker rm nginx-cont
nginx-cont
```

**check container:**

```
controlplane $ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES

controlplane $
```

**create alpine image from Dockerfile:**

```
controlplane $ sudo nano Dockerfile.alpine
controlplane $ ls
Dockerfile.alpine  Dockerfile.busybox  filesystem  myfile.txt
controlplane $ cat Dockerfile.alpine
FROM alpine
RUN apk update && apk add --no-cache curl
CMD ["sleep", "infinity"]
```

**create alpine image from dockerfile:**

```
controlplane $ docker build -t my-alpine-image -f Dockerfile.alpine .
Sending build context to Docker daemon  6.715MB
Step 1/3 : FROM alpine
latest: Pulling from library/alpine
31e352740f53: Pull complete
Digest: sha256:82d1e9d7ed48a7523bdebc18cf6290bdb97b82302a8a9c27d4fe885949ea94d1
Status: Downloaded newer image for alpine:latest
 ---> c1aabb73d233
Step 2/3 : RUN apk update && apk add --no-cache curl
 ---> Running in 050c2393acb6
fetch https://dl-cdn.alpinelinux.org/alpine/v3.18/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.18/community/x86_64/APKINDEX.tar.gz
```

```
v3.18.2-424-gec5fe355ee5 [https://dl-cdn.alpinelinux.org/alpine/v3.18/main]
v3.18.2-426-g815aaae6d98 [https://dl-cdn.alpinelinux.org/alpine/v3.18/community]
OK: 20068 distinct packages available
fetch https://dl-cdn.alpinelinux.org/alpine/v3.18/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.18/community/x86_64/APKINDEX.tar.gz
(1/7) Installing ca-certificates (20230506-r0)
(2/7) Installing brotli-libs (1.0.9-r14)
(3/7) Installing libunistring (1.1-r1)
(4/7) Installing libidn2 (2.3.4-r1)
(5/7) Installing nghttp2-libs (1.55.1-r0)
(6/7) Installing libcurl (8.2.0-r0)
(7/7) Installing curl (8.2.0-r0)
Executing busybox-1.36.1-r0.trigger
Executing ca-certificates-20230506-r0.trigger
OK: 12 MiB in 22 packages
Removing intermediate container 050c2393acb6
 ---> b240852f790e
Step 3/3 : CMD ["sleep", "infinity"]
 ---> Running in d141c33f957e
Removing intermediate container d141c33f957e
 ---> 18071a80a0cf
Successfully built 18071a80a0cf
Successfully tagged my-alpine-image:latest
```

**check image:**

```
controlplane $ docker images
REPOSITORY       TAG       IMAGE ID       CREATED         SIZE
my-alpine-image   latest    18071a80a0cf   47 seconds ago   13.4MB
my-busybox       latest    5a95f46ede5f   7 minutes ago    4.26MB
busybox          latest    a416a98b71e2   34 hours ago     4.26MB
alpine           latest    c1aabb73d233   5 weeks ago      7.33MB
```

**create container from alpine image:**

```
controlplane $ docker run -d --name my-alpine-container my-alpine-image

62734ac4d1c2817c96e8b73de9b04fde0af50b2d3e40df2eeb171237c8af9850
controlplane $
```

**check container:**

```
controlplane $ docker ps
CONTAINER ID   IMAGE            COMMAND          CREATED          STATUS         PORTS     NAMES
62734ac4d1c2   my-alpine-image   "sleep infinity"   26 seconds ago   Up 10 seconds             my-alpine-
container
controlplane $
```

## 4. Create 2 Network of Bridge Type named as network01 and network02 and communicate containers on both of them

**create network01:**

```
controlplane $ docker network create --driver bridge network01
bd29e3d8d59be790e0a9c58e2a84bb7f57a8ce096e168ef7bd838af524aeb486
```

**create network02:**

```
controlplane $ docker network create --driver bridge network02
603a675934320becb1f26a80bd02d2cec1ecd17dad4e54413a0d5db6d70e11b0
```

**check network list:**

```
controlplane $ 
controlplane $ docker network ls
NETWORK ID     NAME        DRIVER    SCOPE
0f1a82eec396   bridge      bridge    local
e812261b4a8f   host        host      local
f636b144915f   network01   bridge    local
314102e44812   network02   bridge    local
2e4431617ab9   none        null      local
controlplane $ 
```

**create container from network01:**

```
controlplane $ docker run -d --name container01 --network network01 nginx:latest
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
faef57eae888: Pull complete 
76579e9ed380: Pull complete 
cf707e233955: Pull complete 
91bb7937700d: Pull complete 
4b962717ba55: Pull complete 
f46d7b05649a: Pull complete 
103501419a0a: Pull complete 
Digest: sha256:08bc36ad52474e528cc1ea3426b5e3f4bad8a130318e3140d6cfe29c8892c7ef
Status: Downloaded newer image for nginx:latest
cb286aae33f5a5d3943058be891402140261e85531100f3830e35050d258e991
```

**create container from network02:**

```
controlplane $ docker run -d --name container02 --network network02 nginx:latest
4f098f5c501112fae8b3094382935c561f5595ea02e6b7fd57deb95e92786175
```

**communicate container between network:**

```
controlplane $ docker network create --driver bridge shared_bridge_network --attachable
831520e38546bc3002f7c3f0c76d0394520fbf96c55e57d8b4000bd5e03b4cb5
controlplane $ docker network connect shared_bridge_network container01
controlplane $ docker network connect shared_bridge_network container02
```

**check network communicate:**

```
controlplane $ docker exec -it container01 /bin/bash
root@33cd7b0cf344:/# curl container02
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@33cd7b0cf344:/# exit
exit
controlplane $
```

**5. Save index file in log conatainer and access from web container**

**create volume:**

```
controlplane $ docker create -v /data --name log_container busybox
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
3f4d90098f5b: Pull complete
Digest: sha256:3fbc632167424a6d997e74f52b878d7cc478225cffac6bc977eedfe51c7f4e79
Status: Downloaded newer image for busybox:latest
f6e1c96b93d7d7134f08a3ee1f3fb6576ad02e932ff0d26c4fd5df0ab217009f
```

**copy index.html In log_container:/data/:**

```
controlplane $ sudo nano index.html
controlplane $ docker cp index.html log_container:/data/
controlplane $ docker run -d --name rehan-web_container --volumes-from log_container -p 81:80
nginx:latest
fa652ed9e4451a9011fe7b73f23568382b656a92c67dfd9394cc99b97805a0a0
```

**check index.html in nginx to check curl command:**

```
controlplane $ curl http://localhost:81
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
controlplane $
```

## 7.Create a Volume and store current date from conatainer01 and access from container02

**create volume:**

```
controlplane $ docker volume create date_volume
date_volume
```

**Create container & save current date in volume:**

```
controlplane $ docker run --name container1 -v date_volume:/data busybox sh -c "date > /data/date.txt"
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
3f4d90098f5b: Pull complete
Digest: sha256:3fbc632167424a6d997e74f52b878d7cc478225cffac6bc977eedfe51c7f4e79
Status: Downloaded newer image for busybox:latest
```

**Create container and access date from volume:**

```
controlplane $ docker run --rm -it -v date_volume:/data busybox sh -c "cat /data/date.txt"
Thu Jul 20 10:59:54 UTC 2023
controlplane $
```