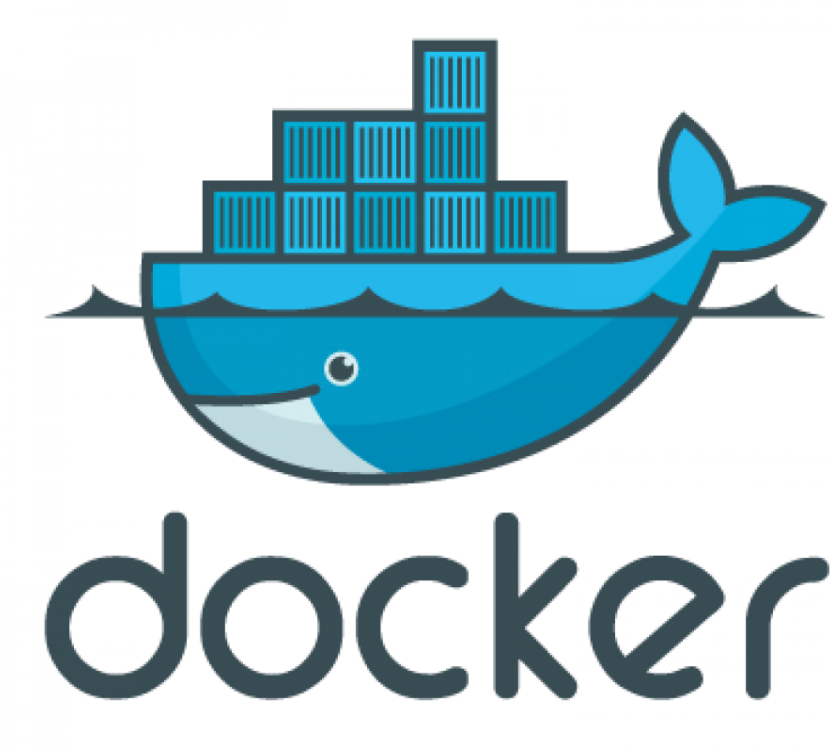
# Curso: Docker




## Homework 2 - Docker Extended Topics

Instructor: Caleb Espinoza Gutierrez

Estudiante: Ronald Torrico Ovando

## Exercise 1

1. List all Docker networks.

docker network ls

```
vboxuser@RTORRICOO-VH01:~$ docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
9e6679c0bcb6   bridge    bridge    local
a0a550b10097   host      host      local
db9828115d6d   none      null      local
```

2. Inspect the default bridge network.

docker network inspect bridge

```
vboxuser@RTORRICOO-VH01:~$ docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "9e6679c0bcb60ee9091af8f0d80973389ca21a2d7b0d26ec046ed9c52985e94e",
        "Created": "2025-05-02T20:06:15.223292104-04:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16",
                    "Gateway": "172.17.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {
            "com.docker.network.bridge.default_bridge": "true",
            "com.docker.network.bridge.enable_icc": "true",
            "com.docker.network.bridge.enable_ip_masquerade": "true",
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        },
        "Labels": {}
    }
]
vboxuser@RTORRICOO-VH01:~$
```

3. Create a new bridge user-defined network.

docker network create --driver bridge my_bridge_network

```
vboxuser@RTORRICOO-VH01:~$ docker network create --driver bridge my_bridge_network
a526ccff91a6dce32d7d0ea3e936416d55d133b249b3cac9a63e428fc347e108
vboxuser@RTORRICOO-VH01:~$ docker network ls
NETWORK ID     NAME                DRIVER    SCOPE
9e6679c0bcb6   bridge              bridge    local
a0a550b10097   host                host      local
a526ccff91a6   my_bridge_network   bridge    local
db9828115d6d   none                null      local
vboxuser@RTORRICOO-VH01:~$
```

4. Run a container attached to it and inspect its IP.

```
# Run a container (using Alpine as an example)
docker run -dit --name my_container --network my_bridge_network alpine sh

# Inspect its IP (either by inspecting the container or the network)
docker inspect my_container | grep IPAddress
# OR
docker network inspect my_bridge_network | grep -A 5 my_container
```

```
vboxuser@RTORRICOO-VH01:~$ docker run -dit --name my_container02 --network my_bridge_network alpine sh
3ab763896951104facbbca9f54ea070409cb3a2ad3ea2360f03910d9072c510b
vboxuser@RTORRICOO-VH01:~$ docker inspect my_container02 | grep IPAddress
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAddress": "172.18.0.2",
vboxuser@RTORRICOO-VH01:~$ docker network inspect my_bridge_network | grep -A 5 my_container02
                "Name": "my_container02",
                "EndpointID": "eac44ff39375b794f2aa4fa6afde1ceb77a3448cef9573f32b7491225baf7726",
                "MacAddress": "02:42:ac:12:00:02",
                "IPv4Address": "172.18.0.2/16",
                "IPv6Address": ""
            }
vboxuser@RTORRICOO-VH01:~$
```

## Exercise 2

1. Run two Nginx containers which have to be connected to that user-defined network created in Exercise 1.

```
# Run the first Nginx container
docker run -d --name nginx1 --network my_bridge_network nginx
# Run the second Nginx container
docker run -d --name nginx2 --network my_bridge_network nginx

docker ps
```

```
vboxuser@RTORRICOO-VH01:~$ docker run -d --name nginx1 --network my_bridge_network nginx
8d493b2497bc508d47f5ebdae5aade5eb8a0a1f5ff7d5e77f1a8b92fdcee9d93
vboxuser@RTORRICOO-VH01:~$ docker run -d --name nginx2 --network my_bridge_network nginx
5fe5720bc03d96f2df40e716f455b5e6dd614aa55178b3d80741cf04940d5376
vboxuser@RTORRICOO-VH01:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS     NAMES
5fe5720bc03d   nginx     "/docker-entrypoint.…"   15 seconds ago   Up 14 seconds   80/tcp    nginx2
8d493b2497bc   nginx     "/docker-entrypoint.…"   23 seconds ago   Up 22 seconds   80/tcp    nginx1
3ab763896951   alpine    "sh"                     32 minutes ago   Up 32 minutes             my_container02
```

```
# Verify both containers are running and connected to the network
docker network inspect my_bridge_network
```

```
vboxuser@RTORRICOO-VH01:~$ docker network inspect my_bridge_network
[
    {
        "Name": "my_bridge_network",
        "Id": "a526ccff91a6dce32d7d0ea3e936416d55d133b249b3cac9a63e428fc347e108",
        "Created": "2025-05-02T20:27:55.989951702-04:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "3ab763896951104facbbca9f54ea070409cb3a2ad3ea2360f03910d9072c510b": {
                "Name": "my_container02",
                "EndpointID": "eac44ff39375b794f2aa4fa6afde1ceb77a3448cef9573f32b7491225baf7726",
                "MacAddress": "02:42:ac:12:00:02",
                "IPv4Address": "172.18.0.2/16",
                "IPv6Address": ""
            },
            "5fe5720bc03d96f2df40e716f455b5e6dd614aa55178b3d80741cf04940d5376": {
                "Name": "nginx2",
                "EndpointID": "350628882a02224c481a6cee26cef9210d738ed3364bed003319d13e1cd34541",
                "MacAddress": "02:42:ac:12:00:04",
                "IPv4Address": "172.18.0.4/16",
                "IPv6Address": ""
            },
            "8d493b2497bc508d47f5ebdae5aade5eb8a0a1f5ff7d5e77f1a8b92fdcee9d93": {
                "Name": "nginx1",
                "EndpointID": "86684699fc86280c5902d14a715ec5f43548ce2c69908ddf80eb4a3b959380ac",
                "MacAddress": "02:42:ac:12:00:03",
                "IPv4Address": "172.18.0.3/16",
                "IPv6Address": ""
            }
        },
        "Options": {},
        "Labels": {}
    }
]
```

```
# Test connectivity between them (optional):
# Exec into nginx1 and ping nginx2 (requires ping utility)
docker exec -it nginx1 sh -c "apt-get update && apt-get install -y iputils-ping && ping nginx2"
```

```
vboxuser@RTORRICOO-VH01:~$ docker exec -it nginx1 sh -c "apt-get update && apt-get install -y iputils-ping && ping nginx2"
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8792 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [512 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [258 kB]
Fetched 9305 kB in 4s (2575 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
```

```
debconf: falling back to frontend: Teletype
Setting up iputils-ping (3:20221126-1+deb12u1) ...
PING nginx2 (172.18.0.4) 56(84) bytes of data.
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=1 ttl=64 time=0.073 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=2 ttl=64 time=0.188 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=3 ttl=64 time=0.875 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=4 ttl=64 time=0.142 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=5 ttl=64 time=0.096 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=6 ttl=64 time=0.098 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=7 ttl=64 time=0.055 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=8 ttl=64 time=0.086 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=9 ttl=64 time=0.059 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=10 ttl=64 time=0.056 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=11 ttl=64 time=0.055 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=12 ttl=64 time=0.051 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=13 ttl=64 time=1.23 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=14 ttl=64 time=0.063 ms
```

2. Use ping within both containers to test communication each other by container name.

```
# Install ping in nginx1 and nginx2
docker exec -it nginx1 bash -c "apt-get update && apt-get install -y iputils-ping"
docker exec -it nginx2 bash -c "apt-get update && apt-get install -y iputils-ping"
```

```
vboxuser@RTORRICOO-VH01:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS              PORTS     NAMES
5fe5720bc03d   nginx     "/docker-entrypoint.…"   43 minutes ago      Up 43 minutes       80/tcp    nginx2
8d493b2497bc   nginx     "/docker-entrypoint.…"   43 minutes ago      Up 43 minutes       80/tcp    nginx1
3ab763896951   alpine    "sh"                     About an hour ago   Up About an hour              my_container02
e530927e8c66   nginx     "/docker-entrypoint.…"   8 days ago          Dead                80/tcp    my_container
vboxuser@RTORRICOO-VH01:~$ docker exec -it nginx1 sh -c "apt-get update && apt-get install -y iputils-ping"
Hit:1 http://deb.debian.org/debian bookworm InRelease
Hit:2 http://deb.debian.org/debian bookworm-updates InRelease
Hit:3 http://deb.debian.org/debian-security bookworm-security InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
iputils-ping is already the newest version (3:20221126-1+deb12u1).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
vboxuser@RTORRICOO-VH01:~$ docker exec -it nginx2 sh -c "apt-get update && apt-get install -y iputils-ping"
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8792 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [512 B]
```

```
# Test communication using ping
docker exec -it nginx1 ping nginx2
```

```
debconf: falling back to frontend: Teletype
Setting up iputils-ping (3:20221126-1+deb12u1) ...
vboxuser@RTORRICOO-VH01:~$ docker exec -it nginx1 ping nginx2
PING nginx2 (172.18.0.4) 56(84) bytes of data.
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=1 ttl=64 time=0.044 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=2 ttl=64 time=0.059 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=3 ttl=64 time=0.066 ms
64 bytes from nginx2.my_bridge_network (172.18.0.4): icmp_seq=4 ttl=64 time=0.069 ms
^C
--- nginx2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3053ms
rtt min/avg/max/mdev = 0.044/0.059/0.069/0.009 ms
vboxuser@RTORRICOO-VH01:~$ docker exec -it nginx2 ping nginx1
PING nginx1 (172.18.0.3) 56(84) bytes of data.
64 bytes from nginx1.my_bridge_network (172.18.0.3): icmp_seq=1 ttl=64 time=0.084 ms
64 bytes from nginx1.my_bridge_network (172.18.0.3): icmp_seq=2 ttl=64 time=0.280 ms
64 bytes from nginx1.my_bridge_network (172.18.0.3): icmp_seq=3 ttl=64 time=0.065 ms
64 bytes from nginx1.my_bridge_network (172.18.0.3): icmp_seq=4 ttl=64 time=0.169 ms
^C
--- nginx1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3025ms
rtt min/avg/max/mdev = 0.065/0.149/0.280/0.084 ms
vboxuser@RTORRICOO-VH01:~$
```

## Exercise 3

1. Create a Docker volume: mydata .

```
docker volume create mydata
docker volume ls
```

```
vboxuser@RTORRICOO-VH01:~$ docker volume ls
DRIVER    VOLUME NAME
vboxuser@RTORRICOO-VH01:~$ docker volume create mydata
mydata
vboxuser@RTORRICOO-VH01:~$ docker volume ls
DRIVER    VOLUME NAME
local     mydata
```

2. Run a container using the volume.

```
# Mount mydata to /data inside the container
docker run -it --name mycontainer -v mydata:/data alpine sh
```

```
vboxuser@RTORRICOO-VH01:~$ docker run -it --name mycontainer -v mydata:/data alpine sh
/ #
/ # ls
bin    data   dev    etc    home   lib    media  mnt    opt    proc   root   run    sbin   srv    sys    tmp    usr
/ # pwd
/
/ # touch file01.txt
/ # ls
bin         dev         file01.txt  lib         mnt         proc        run         srv         tmp         var
data        etc         home        media       opt         root        sbin        sys         usr
/ #
```

3. Write a file inside /data from the container, then:

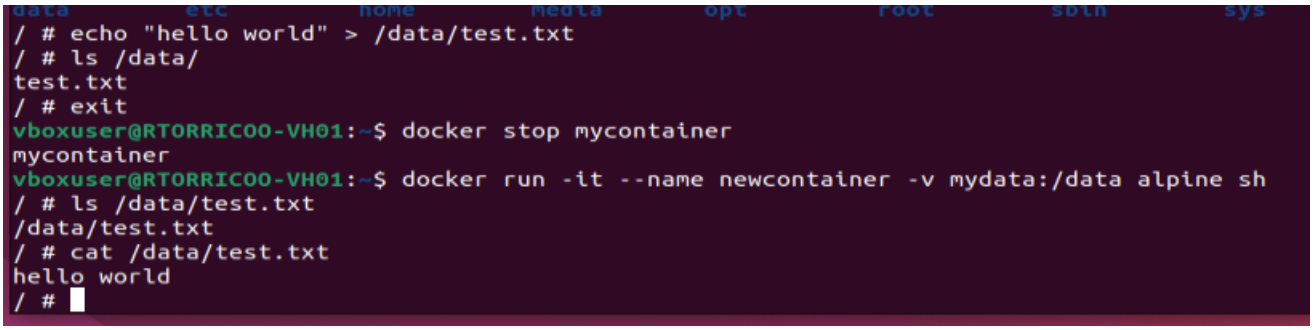
```
echo "Hello world" > /data/test.txt
cat /data/test.txt
exit
```

1. Stop the container.

```
docker stop mycontainer
```

2. Relaunch to verify persistence.

```
docker run -it --name newcontainer -v mydata:/data alpine sh
cat /data/test.txt
```




## Exercise 4

1. Create a directory on your host.

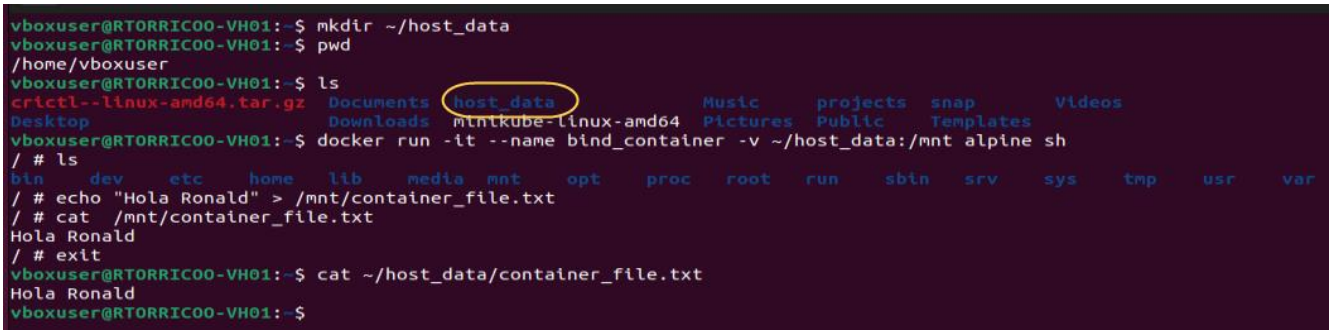
```
mkdir ~/host_data
```

2. Run a container with a bind mount.

```
docker run -it --name bind_container -v ~/host_data:/mnt alpine sh
```

3. Create a file in /mnt from the container and check the host.

```
echo "Hello from the container!" > /mnt/container_file.txt
cat /mnt/container_file.txt
exit

cat ~/host_data/container_file.txt
```




## Exercise 5

1. Create a file in a named volume.

```
# Create a named volume
docker volume create my_vol

# Run a container, mount the volume to /data, and create a file
docker run -it --name vol_container -v my_vol:/data alpine sh -c "echo 'Stored in a named volume' >
/data/vol_file.txt && cat /data/vol_file.txt"

# Inspect where Docker stores the volume data
docker volume inspect my_vol
```

```
vboxuser@RTORRICOO-VH01:~$ docker volume create my_vol
my_vol
vboxuser@RTORRICOO-VH01:~$ docker volume ls
DRIVER    VOLUME NAME
local     my_vol
local     mydata
vboxuser@RTORRICOO-VH01:~$ docker run -it --name vol_container -v my_vol:/data alpine sh -c "echo 'Hello Ronny' > /data/vol_file.txt && cat /data/
vol_file.txt"
Hello Ronny
vboxuser@RTORRICOO-VH01:~$ docker volume inspect my_vol
[
    {
        "CreatedAt": "2025-05-02T22:43:52-04:00",
        "Driver": "local",
        "Labels": null,
        "Mountpoint": "/var/snap/docker/common/var-lib-docker/volumes/my_vol/_data",
        "Name": "my_vol",
        "Options": null,
        "Scope": "local"
    }
]
vboxuser@RTORRICOO-VH01:~$
```

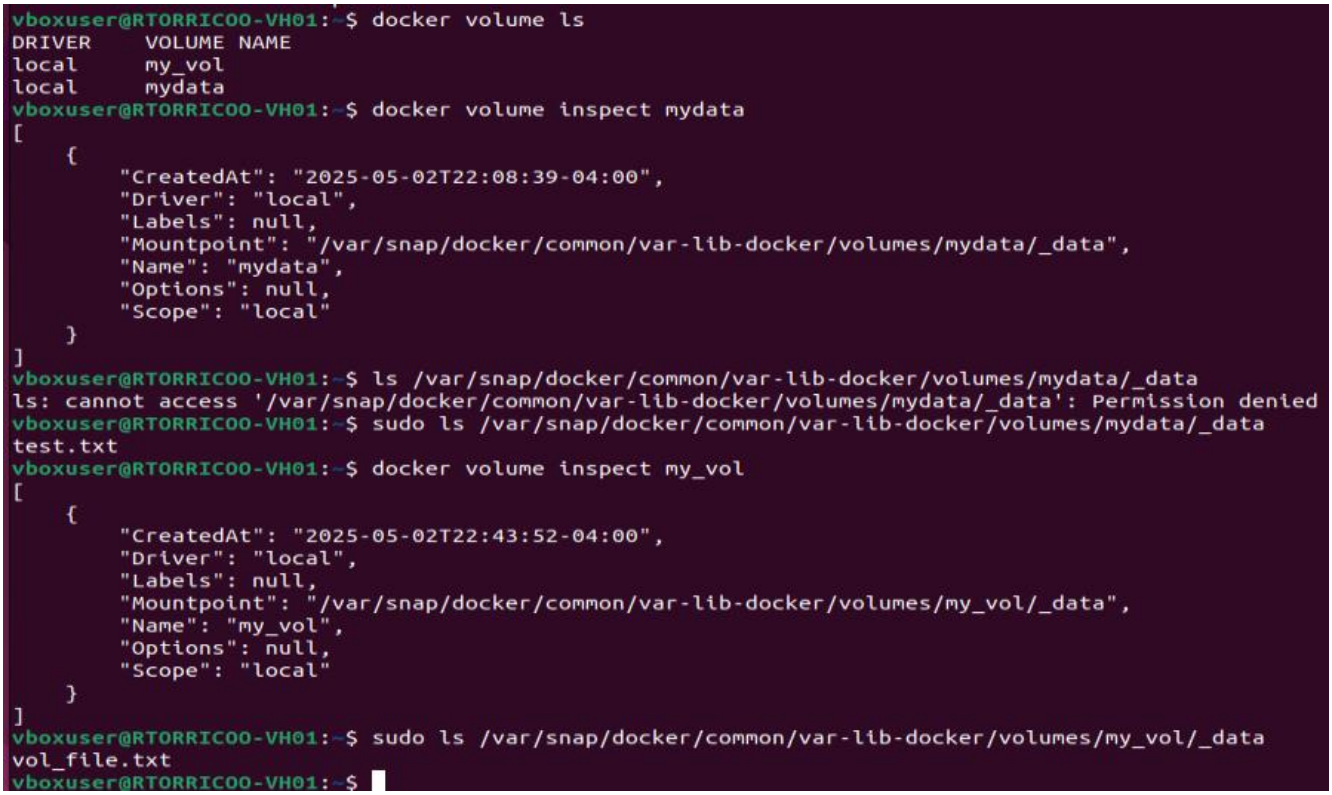
2. Create a file using a bind mount.

```
# Create a host directory
mkdir ~/bind_data

# Run a container, bind mount the host dir to /mnt, and create a file
docker run -it --name bind_container -v ~/bind_data:/mnt alpine sh -c "echo 'Stored in a bind mount'
> /mnt/bind_file.txt && cat /mnt/bind_file.txt"

# Check the file on the host
ls ~/bind_data && cat ~/bind_data/bind_file.txt
```

```
vboxuser@RTORRICOO-VH01:~$ mkdir ~/bind_data
vboxuser@RTORRICOO-VH01:~$ ls -l
total 122388
drwxrwxr-x  2 vboxuser vboxuser      4096 may  2 22:47 bind_data
-rw-rw-r--  1 vboxuser vboxuser         9 abr  7 22:43 crictl-linux-amd64.tar.gz
drwxr-xr-x  2 vboxuser vboxuser      4096 ago 27  2024 Desktop
drwxr-xr-x  2 vboxuser vboxuser      4096 sep 14  2023 Documents
drwxr-xr-x  2 vboxuser vboxuser      4096 ago 27  2024 Downloads
drwxrwxr-x  2 vboxuser vboxuser      4096 may  2 22:26 host_data
-rw-rw-r--  1 vboxuser vboxuser 125267037 abr 10 23:13 minikube-linux-amd64
drwxr-xr-x  2 vboxuser vboxuser      4096 ago 21  2023 Music
drwxr-xr-x  3 vboxuser vboxuser      4096 sep 11  2023 Pictures
drwxrwxr-x  5 vboxuser vboxuser      4096 sep  6  2023 projects
drwxr-xr-x  2 vboxuser vboxuser      4096 ago 21  2023 Public
drwx------ 10 vboxuser vboxuser      4096 ago 29  2024 snap
drwxr-xr-x  2 vboxuser vboxuser      4096 ago 21  2023 Templates
drwxr-xr-x  2 vboxuser vboxuser      4096 ago 21  2023 Videos
vboxuser@RTORRICOO-VH01:~$ docker run -it --name bind_container -v ~/bind_data:/mnt alpine sh -c "echo 'Bolivia 2025' > /mnt/bind_file.txt && cat
/mnt/bind_file.txt"
docker: Error response from daemon: Conflict. The container name "/bind_container" is already in use by container "df2ec91d7f7a803b78e6eedf20324e1
813dcb75dc9edf585565ec23b0b82fc25". You have to remove (or rename) that container to be able to reuse that name.

Run 'docker run --help' for more information
vboxuser@RTORRICOO-VH01:~$ docker run -it --name my_bind_container -v ~/bind_data:/mnt alpine sh -c "echo 'Bolivia 2025' > /mnt/bind_file.txt && c
at /mnt/bind_file.txt"
Bolivia 2025
vboxuser@RTORRICOO-VH01:~$ ls ~/bind_data && cat ~/bind_data/bind_file.txt
bind_file.txt
Bolivia 2025
vboxuser@RTORRICOO-VH01:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED       STATUS       PORTS     NAMES
5fe5720bc03d   nginx     "/docker-entrypoint.…"   2 hours ago   Up 2 hours   80/tcp    nginx2
8d493b2497bc   nginx     "/docker-entrypoint.…"   2 hours ago   Up 2 hours   80/tcp    nginx1
3ab763896951   alpine    "sh"                     2 hours ago   Up 2 hours             my_container02
vboxuser@RTORRICOO-VH01:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS                          PORTS     NAMES
3d5bd1d5f6e6   alpine    "sh -c 'echo 'Bolivi…"   About a minute ago   Exited (0) About a minute ago             my_bind_container
7f6f56b7035d   alpine    "sh -c 'echo 'Hello …"   6 minutes ago        Exited (0) 6 minutes ago                  vol_container
df2ec91d7f7a   alpine    "sh"                     26 minutes ago       Exited (0) 25 minutes ago                 bind_container
43b7a3372fe6   alpine    "sh"                     33 minutes ago       Exited (0) 27 minutes ago                 newcontainer
2e5f609ec42c   alpine    "sh"                     40 minutes ago       Exited (0) 34 minutes ago                 mycontainer
5fe5720bc03d   nginx     "/docker-entrypoint.…"   2 hours ago          Up 2 hours                      80/tcp    nginx2
8d493b2497bc   nginx     "/docker-entrypoint.…"   2 hours ago          Up 2 hours                      80/tcp    nginx1
3ab763896951   alpine    "sh"                     2 hours ago          Up 2 hours                                my_container02
e530927e8c66   nginx     "/docker-entrypoint.…"   9 days ago           Dead                            80/tcp    my_container
```

3. Observe where data is stored on the host with docker volume inspect and ls .

```
docker volumen ls
docker volumen inspect mydata
```




## Exercise 6

1. Run an Ubuntu container with the necessary options to enable Docker in Docker (DinD).

docker run --privileged --name dind_container -d docker:dind

- *--privileged: Grants full host system access (required for DinD).*
- *-d: Runs the container in detached mode.*

2. Exec into the container and run *docker version* .

docker exec -it dind_container sh -c "docker version"

```
vboxuser@RTORRICOO-VH01:~$ docker run --privileged --name dind_container -d docker:dind
Unable to find image 'docker:dind' locally
dind: Pulling from library/docker
f18232174bc9: Already exists
9c3b451c4c6d: Pull complete
4f4fb700ef54: Pull complete
2b64f750a092: Pull complete
3e19e08b20db: Pull complete
c9ff937cfa78: Pull complete
c53278fe5e6f: Pull complete
6e060defd84a: Pull complete
63dbc57c1a2a: Pull complete
1293ab979e28: Pull complete
f0a4127e99cf: Pull complete
f418ff940aa3: Pull complete
10d0fb59c4c5: Pull complete
aa6535ddaccd: Pull complete
8f8d00d6e5e8: Pull complete
bfabfe011b48: Pull complete
Digest: sha256:f49e1c71b5d9f8ebe53715f78996ce42b8be4b1ec03875d187dfe3c03de1dc00
Status: Downloaded newer image for docker:dind
11fda9a25ffac52ed6e3ad6d59c89e4a90f447da1705668c84fc03a37cb0869a
vboxuser@RTORRICOO-VH01:~$ docker exec -it dind_container sh -c "docker version"
Client:
 Version:           28.1.1
 API version:       1.49
 Go version:        go1.23.8
 Git commit:        4eba377
 Built:             Fri Apr 18 09:51:06 2025
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:          28.1.1
  API version:      1.49 (minimum version 1.24)
  Go version:       go1.23.8
  Git commit:       01f442b
  Built:            Fri Apr 18 09:52:21 2025
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          v1.7.27
  GitCommit:        05044ec0a9a75232cad458027ca83437aae3f4da
 runc:
  Version:          1.2.6
  GitCommit:        v1.2.6-0-ge89a299
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
vboxuser@RTORRICOO-VH01:~$
```

## Exercise 7

1. Run a container with memory and CPU limits:
    1. Memory = 256m
    2. CPU = 0.5

```
docker run -d --name limited_container \
 --memory="256m" \       # Memory limit: 256MB
 --cpus="0.5" \          # CPU limit: 0.5 cores (50% of a CPU)
 nginx                   # Using Nginx for demonstration
```

2. Check resource usage stats .

```
docker stats limited_container
```

```
vboxuser@RTORRICOO-VH01:~$ docker ps
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS          PORTS           NAMES
078c7996b7fe   nginx         "/docker-entrypoint.…"   3 minutes ago    Up 3 minutes    80/tcp          limited_container
11fda9a25ffa   docker:dind   "dockerd-entrypoint.…"   21 minutes ago   Up 21 minutes   2375-2376/tcp   dind_container
222360a1a126   ubuntu        "/bin/bash"              34 minutes ago   Up 34 minutes                   ubuntu-dind
vboxuser@RTORRICOO-VH01:~$ docker stats limited_container
```

```
CONTAINER ID   NAME                CPU %     MEM USAGE / LIMIT   MEM %     NET I/O       BLOCK I/O         PIDS
078c7996b7fe   limited_container   0.00%     15.13MiB / 256MiB   5.91%     3.49kB / 0B   12.6MB / 12.3kB   5
```

3. Check disk usage (docker system).

```
docker system df
docker inspect limited_container | grep -iE "memory|cpu"
```

```
vboxuser@RTORRICOO-VH01:~$ docker system df
TYPE            TOTAL     ACTIVE    SIZE      RECLAIMABLE
Images          4         4         665.5MB   7.834MB (1%)
Containers      12        3         43.79MB   43.77MB (99%)
Local Volumes   3         3         245.8kB   0B (0%)
Build Cache     0         0         0B        0B
vboxuser@RTORRICOO-VH01:~$ docker inspect limited_container | grep -iE "memory|cpu"
            "CpuShares": 0,
            "Memory": 268435456,
            "NanoCpus": 500000000,
            "CpuPeriod": 0,
            "CpuQuota": 0,
            "CpuRealtimePeriod": 0,
            "CpuRealtimeRuntime": 0,
            "CpusetCpus": "",
            "CpusetMems": "",
            "MemoryReservation": 0,
            "MemorySwap": 536870912,
            "MemorySwappiness": null,
            "CpuCount": 0,
            "CpuPercent": 0,
vboxuser@RTORRICOO-VH01:~$
```

## Exercise 8

1. Run a container with policy --restart on-failure .

```
docker run -d --name on_failure_container --restart on-failure alpine sleep 30
```

- *The sleep 30 command will exit after 30 seconds.*
- *Since it exits **successfully** (exit code 0), Docker **will not restart** it (only restarts on failures, i.e., non-zero exit codes).*

```
vboxuser@RTORRICOO-VH01:~$ docker run -d --name on_failure_container --restart on-failure alpine sleep 30
23226aaf8e7cbee54351771041659a9336d0752046144f447ee5c8a8210894e9
vboxuser@RTORRICOO-VH01:~$ docker ps
CONTAINER ID   IMAGE         COMMAND                  CREATED              STATUS              PORTS           NAMES
23226aaf8e7c   alpine        "sleep 30"               6 seconds ago        Up 5 seconds                        on_failure_container
078c7996b7fe   nginx         "/docker-entrypoint.…"   36 minutes ago       Up 36 minutes       80/tcp          limited_container
11fda9a25ffa   docker:dind   "dockerd-entrypoint.…"   53 minutes ago       Up 53 minutes       2375-2376/tcp   dind_container
222360a1a126   ubuntu        "/bin/bash"              About an hour ago    Up About an hour                    ubuntu-dind
vboxuser@RTORRICOO-VH01:~$ docker ps -a --filter "name=on_failure_container" --format "{{.Status}}"
Exited (0) 44 seconds ago
vboxuser@RTORRICOO-VH01:~$ docker ps -a --filter "name=on_failure_container" --format "{{.Status}}"
Exited (0) 2 minutes ago
```

2. Kill the container and observe how it restarts

```
docker start on_failure_container          # Restart it (if stopped)
docker exec on_failure_container sh -c "exit 1"   # Force a failure
docker kill on_failure_container           # Sends SIGKILL (immediate termination)
```

```
vboxuser@RTORRICOO-VH01:~$ docker start on_failure_container
on_failure_container
vboxuser@RTORRICOO-VH01:~$ docker exec on_failure_container sh -c "exit 1"
vboxuser@RTORRICOO-VH01:~$ docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED             STATUS                          PORTS           NAMES
23226aaf8e7c   alpine        "sleep 30"               10 minutes ago      Exited (0) 12 seconds ago                       on_failure_container
078c7996b7fe   nginx         "/docker-entrypoint.…"   46 minutes ago      Up 46 minutes                   80/tcp          limited_container
11fda9a25ffa   docker:dind   "dockerd-entrypoint.…"   About an hour ago   Up About an hour                2375-2376/tcp   dind_container
222360a1a126   ubuntu        "/bin/bash"              About an hour ago   Up About an hour                                ubuntu-dind
3d5bd1d5f6e6   alpine        "sh -c 'echo 'Bolivi…"   2 hours ago         Exited (0) 2 hours ago                          my_bind_container
7f6f56b7035d   alpine        "sh -c 'echo 'Hello …"   2 hours ago         Exited (0) 2 hours ago                          vol_container
df2ec91d7f7a   alpine        "sh"                     3 hours ago         Exited (0) 2 hours ago                          bind_container
43b7a3372fe6   alpine        "sh"                     3 hours ago         Exited (0) 3 hours ago                          newcontainer
2e5f609ec42c   alpine        "sh"                     3 hours ago         Exited (0) 3 hours ago                          mycontainer
5fe5720bc03d   nginx         "/docker-entrypoint.…"   4 hours ago         Exited (255) About an hour ago  80/tcp          nginx2
8d493b2497bc   nginx         "/docker-entrypoint.…"   4 hours ago         Exited (255) About an hour ago  80/tcp          nginx1
3ab763896951   alpine        "sh"                     4 hours ago         Exited (255) About an hour ago                  my_container02
e530927e8c66   nginx         "/docker-entrypoint.…"   9 days ago          Dead                            80/tcp          my_container
vboxuser@RTORRICOO-VH01:~$ docker ps -a --filter "name=on_failure_container" --format "{{.Status}}"
Exited (0) 39 seconds ago
vboxuser@RTORRICOO-VH01:~$ docker kill on_failure_container
Error response from daemon: cannot kill container: on_failure_container: container 23226aaf8e7cbee54351771041659a9336d0752046144f447ee5c8a8210894e9 is
 not running
vboxuser@RTORRICOO-VH01:~$ docker start on_failure_container
on_failure_container
vboxuser@RTORRICOO-VH01:~$ docker kill on_failure_container
Error response from daemon: cannot kill container: on_failure_container: permission denied
vboxuser@RTORRICOO-VH01:~$ docker ps -a --filter "name=on_failure_container" --format "{{.Status}}"
Up 21 seconds
vboxuser@RTORRICOO-VH01:~$
```

3. Try with the policy --restart unless-stopped

```
docker run -d --name unless_stopped_container --restart unless-stopped alpine sleep 30
docker logs unless_stopped_container                                    # Shows sleep exits after 30s
docker ps -a --filter "name=unless_stopped_container" --format "{{.Status}}"
```

```
vboxuser@RTORRICOO-VH01:~$ docker run -d --name unless_stopped_container --restart unless-stopped alpine sleep 30
c709cb0f82274fdcac183640f86ce247ed57e4d8823fad16a07b935ae850f183
vboxuser@RTORRICOO-VH01:~$
vboxuser@RTORRICOO-VH01:~$ docker logs unless_stopped_container
vboxuser@RTORRICOO-VH01:~$ docker ps -a --filter "name=unless_stopped_container" --format "{{.Status}}"
Up 25 seconds
vboxuser@RTORRICOO-VH01:~$ docker kill unless_stopped_container
Error response from daemon: cannot kill container: unless_stopped_container: permission denied
vboxuser@RTORRICOO-VH01:~$ docker ps -a --filter "name=unless_stopped_container" --format "{{.Status}}"
Up 22 seconds
vboxuser@RTORRICOO-VH01:~$ sudo docker kill unless_stopped_container
[sudo] password for vboxuser:
Error response from daemon: cannot kill container: unless_stopped_container: container c709cb0f82274fdcac183640f86ce247ed57e4d8823fad16a07b935ae850f
3 is not running
vboxuser@RTORRICOO-VH01:~$
```

4. Reboot the system and see what happens.

```
docker ps -a --filter "name=unless_stopped_container"
```

```
vboxuser@RTORRICOO-VH01:~$ docker ps -a --filter "name=unless_stopped_container"
CONTAINER ID   IMAGE     COMMAND      CREATED         STATUS                     PORTS     NAMES
c709cb0f8227   alpine    "sleep 30"   6 minutes ago   Exited (0) 4 minutes ago             unless_stopped_containe
vboxuser@RTORRICOO-VH01:~$ docker logs unless_stopped_container
vboxuser@RTORRICOO-VH01:~$ docker ps -a --filter "name=unless_stopped_container"
CONTAINER ID   IMAGE     COMMAND      CREATED         STATUS                     PORTS     NAMES
c709cb0f8227   alpine    "sleep 30"   10 minutes ago   Exited (0) 8 minutes ago            unless_stopped_contain
vboxuser@RTORRICOO-VH01:~$
```

## Exercise 9

1. Create a network dbnet .

```
docker network create dbnet
```

```
vboxuser@RTORRICOO-VH01:~$ docker network create dbnet
175a822ac96b9029afe4ca14abfcb0ea0ba5b6d72644303c43ee419462532ce9
vboxuser@RTORRICOO-VH01:~$ docker network ps
docker: unknown command: docker network ps

Usage:  docker network

Run 'docker network --help' for more information
vboxuser@RTORRICOO-VH01:~$ docker network ls
NETWORK ID     NAME                DRIVER    SCOPE
97c5fe062e42   bridge              bridge    local
175a822ac96b   dbnet               bridge    local
a0a550b10097   host                host      local
a526ccff91a6   my_bridge_network   bridge    local
db9828115d6d   none                null      local
vboxuser@RTORRICOO-VH01:~$
```

2. Create a volume dbdata .

```
docker volume create dbdata
```

```
vboxuser@RTORRICOO-VH01:~$ docker volume create dbdata
dbdata
vboxuser@RTORRICOO-VH01:~$ docker volume ls
DRIVER    VOLUME NAME
local     dbdata
local     ed1ae15e94819670f58c01f7c251ca64a519a21d772271b1d90a8ebab7f8aa99
local     my_vol
local     mydata
vboxuser@RTORRICOO-VH01:~$
```

3. Run a MariaDB container with the following requirements:
    1. Attached to volume dbdata .
    2. Attached to network dbnet .
    3. Do NOT expose ANY port.

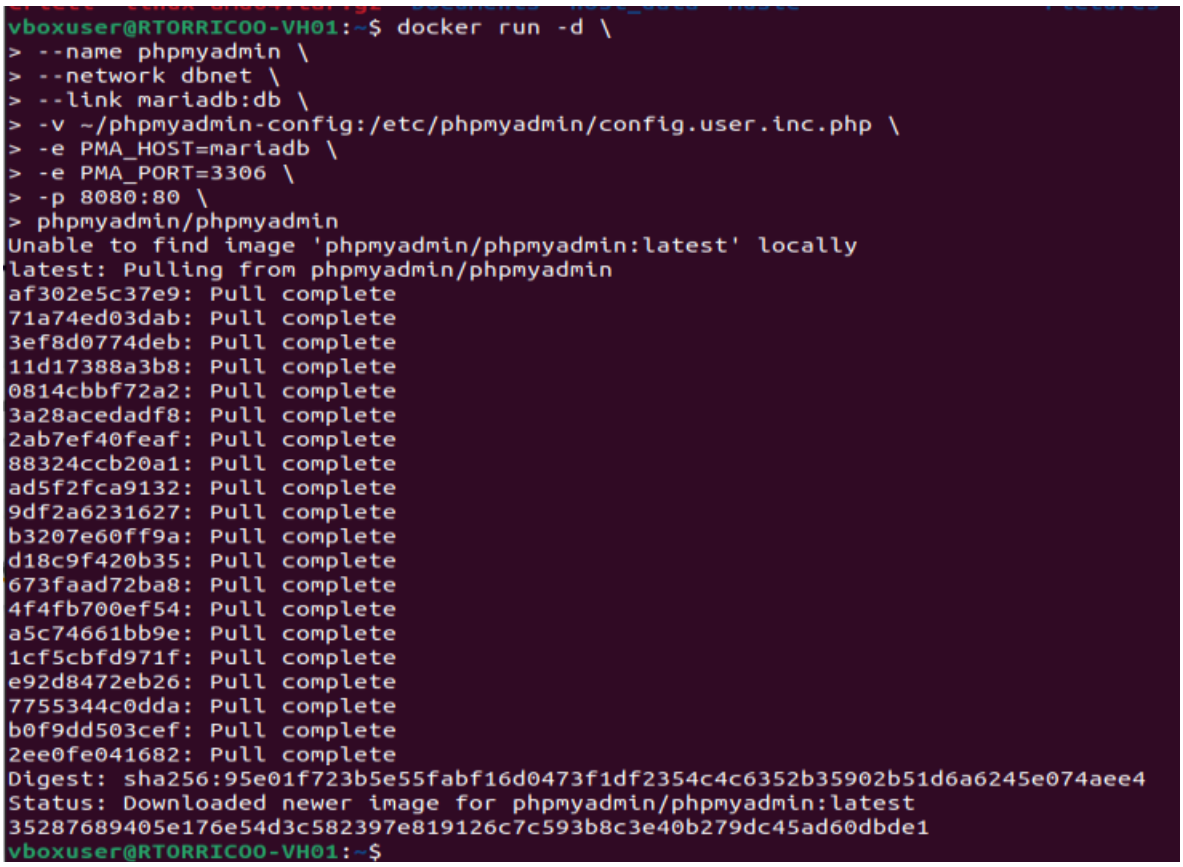
```
docker run -d --name mariadb --network dbnet \
 -v dbdata:/var/lib/mysql \
 -e MYSQL_ROOT_PASSWORD=my-secret-pw \
 Mariadb

docker inspect mariadb-container | grep dbnet
docker volume inspect dbdata
```

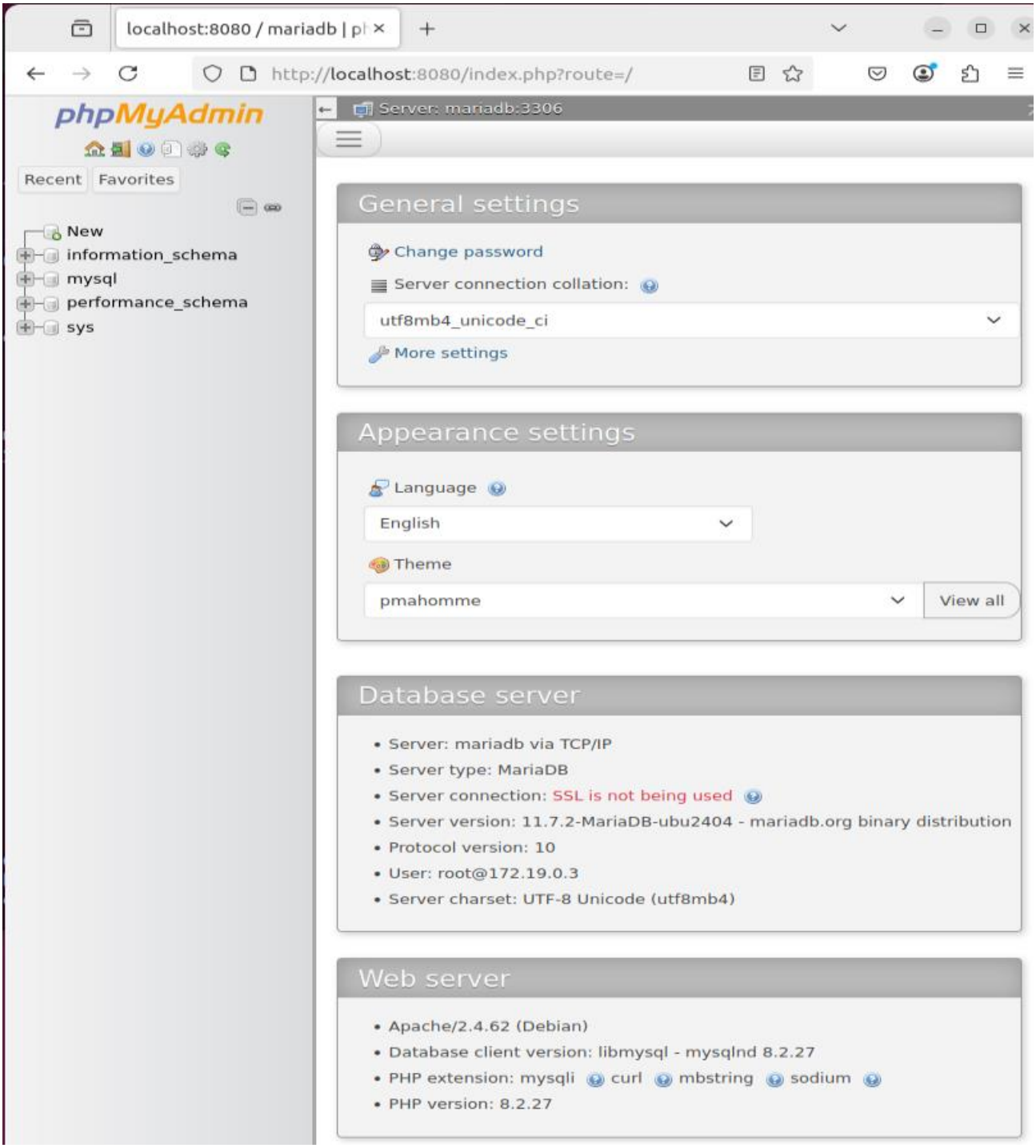
```
vboxuser@RTORRICOO-VH01:~$ docker run -d --name mariadb --network dbnet -v dbdata:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=my_pass mariadb
1259c898605db1441732ac7842db9cbb26fe7bbd87eff4e84a0947601b8b6f92
vboxuser@RTORRICOO-VH01:~$ docker ps --filter "name=mariadb"
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS      NAMES
1259c898605d   mariadb   "docker-entrypoint.s…"   12 seconds ago   Up 12 seconds   3306/tcp   mariadb
vboxuser@RTORRICOO-VH01:~$ docker inspect mariadb | grep dbnet
            "NetworkMode": "dbnet",
                "dbnet": {
vboxuser@RTORRICOO-VH01:~$ docker volume inspect dbdata
[
    {
        "CreatedAt": "2025-05-03T01:22:04-04:00",
        "Driver": "local",
        "Labels": null,
        "Mountpoint": "/var/snap/docker/common/var-lib-docker/volumes/dbdata/_data",
        "Name": "dbdata",
        "Options": null,
        "Scope": "local"
    }
]
vboxuser@RTORRICOO-VH01:~$
```

## Exercise 10

1. Run a PHPMyAdmin container with the following requirements:
    1. Attached to network dbnet (created in Exercise 9).
    2. Use a bind mount to persist the web app configuration.
    3. Linked to the previous MariaDB container (created in Exercise 9)
    4. Open a browser to display the PHPMyAdmin Login Form.
    5. Login with the DB credentials.

```
docker run -d \
 --name phpmyadmin \
 --network dbnet \
 --link mariadb-container:db \
 -v ~/phpmyadmin-config:/etc/phpmyadmin/config.user.inc.php \
 -e PMA_HOST=mariadb-container \
 -e PMA_PORT=3306 \
 -p 8080:80 \
 phpmyadmin/phpmyadmin
```




localhost:8080 / mariadb | pl
http://localhost:8080/index.php?route=/
phpMyAdmin
Recent
Favorites
New
information_schema
mysql
performance_schema
sys
Server: mariadb:3306
General settings
Change password
Server connection collation:
utf8mb4_unicode_ci
More settings
Appearance settings
Language
English
Theme
pmahomme
View all
Database server
Server: mariadb via TCP/IP
Server type: MariaDB
Server connection: SSL is not being used
Server version: 11.7.2-MariaDB-ubu2404 - mariadb.org binary distribution
Protocol version: 10
User: root@172.19.0.3
Server charset: UTF-8 Unicode (utf8mb4)
Web server
Apache/2.4.62 (Debian)
Database client version: libmysql - mysqlnd 8.2.27
PHP extension: mysqli curl mbstring sodium
PHP version: 8.2.27