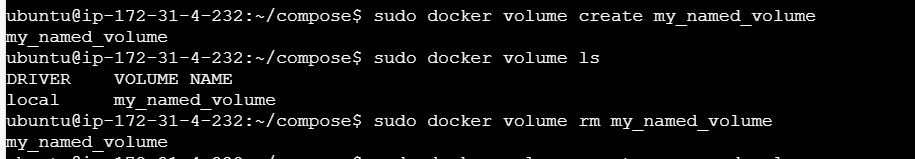
**Docker Volumes and Docker networking**

* A Docker volume is a way to store data outside the container filesystem, enabling data persistence across container restarts and deployments.
* Volumes are easier to back up or migrate than bind mounts.

**Types of Docker Volumes**

**Named Volumes**

* Named volumes are created and managed by Docker, providing a simple way to persist data. They are not tied to the lifecycle of a specific container and can be shared among multiple containers.



# Create a named volume

linux$ docker volume create my\_named\_volume

# Run a container with the named volume

linux$ docker run -d -v my\_named\_volume:/data my\_image:latest

NB: In this example, the volume my\_named\_volume is mounted to the /data directory inside the container.

**Anonymous Volumes**

* Anonymous volumes are similar to named volumes but do not have a specific name. They are often used for temporary data storage that does not need to be referenced by name.

# Run a container with an anonymous volume

linux$ docker run -d -v /data my\_image

NB: In this case, Docker automatically creates an anonymous volume and mounts it to the /data directory inside the container.

**Host Volumes (Bind Mounts)**

* Host volumes, or bind mounts, map a directory on the host machine to a directory in the container. Changes made to the directory on the host are reflected in the container and vice versa.

# Run a container with a bind mount

docker run -d -v /path/on/host:/data my\_image

NB: This command mounts the /path/on/host directory from the host machine to the /data directory inside the container.

* Docker volume inspect <volume\_name>



**Docker volume: Task**

* Create a Named Volume:

$ docker volume create shared\_data

shared\_data

$ docker volume ls

DRIVER VOLUME NAME

local shared\_data

* Run Two Containers Sharing the Same Volume:
* docker run -d --name container1 --mount source=shared\_data,target=/shared busybox
* docker run -d --name container2 --mount source=shared\_data,target=/shared busybox
* Write Data to the Volume from container1,
* docker exec container1 sh -c 'echo "Hello from container1" > /shared/hello.txt'
* Read Data from the Volume in container2,
* docker exec container2 cat /shared/hello.txt

**Docker Network**

* Container networking refers to the ability for containers to connect to and communicate with each other, or to non-Docker workloads.
* Containers have networking enabled by default, and they can make outgoing connections.

**User-defined networks**

* You can create custom, user-defined networks, and connect multiple containers to the same network. Once connected to a user-defined network, containers can communicate with each other using container IP addresses or container names.
* Docker network ls: command to list docker networks

# Creating custom network

linux$ docker network create -d bridge my-net

linux$ docker run --network=my-net -itd --name=container3 busboy