.

# Lesson 04 Demo 06

# Mounting Volumes via Swarm Services

**Objective:** To mount volumes via swarm services for efficient data management and persistent storage across containers within the Docker swarm environment

**Tools required:** Ubuntu OS and Docker

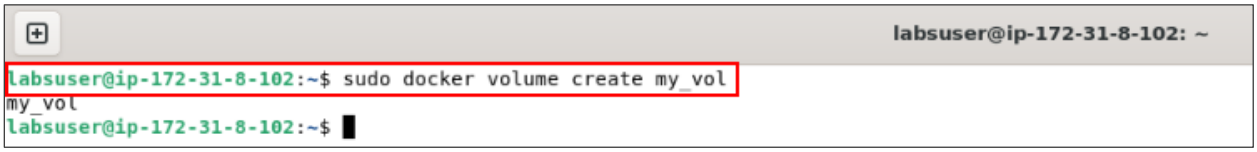
**Prerequisites:** None

Steps to be followed:

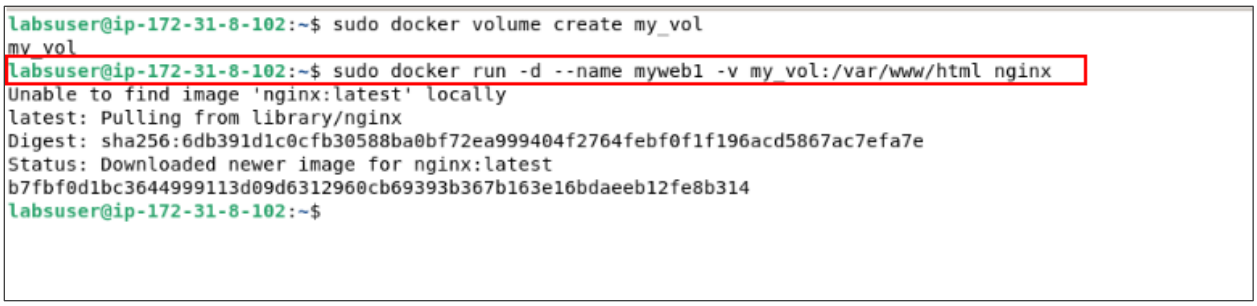
1. Mount volumes in Docker
2. Manage data and temporary file systems within Docker containers

## Step 1: Mount volumes in Docker

1.1 Execute the following command to create a Docker volume:
**sudo docker volume create my_vol**




1.2 Start the container with a volume using the following commands:
**sudo docker run -d --name myweb1 -v my_vol:/var/www/html nginx**
**sudo docker run -d --name myweb2 --mount **
**src=my_vol,dst=/var/www/html nginx**



.

```
labsuser@ip-172-31-8-102:~$ sudo docker volume create my_vol
my_vol
labsuser@ip-172-31-8-102:~$ sudo docker run -d --name myweb1 -v my_vol:/var/www/html nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
Digest: sha256:6db391d1c0cfb30588ba0bf72ea999404f2764febf0f1f196acd5867ac7efa7e
Status: Downloaded newer image for nginx:latest
b7fbf0d1bc3644999113d09d6312960cb69393b367b163e16bdaeeb12fe8b314
labsuser@ip-172-31-8-102:~$ sudo docker run -d --name myweb2 --mount \
> src=my_vol,dst=/var/www/html nginx
b9ce26ae10b0517f488035a2607eb41f28dcde24966ab67c3be1212adc64b5ff
labsuser@ip-172-31-8-102:~$
```

1.3 Run the following command to remove the containers:
**sudo docker rm -f myweb1 myweb2**

```
labsuser@ip-172-31-8-102:~$ sudo docker run -d --name myweb2 --mount \
> src=my_vol,dst=/var/www/html nginx
b9ce26ae10b0517f488035a2607eb41f28dcde24966ab67c3be1212adc64b5ff
labsuser@ip-172-31-8-102:~$ sudo docker rm -f myweb1 myweb2
myweb1
myweb2
labsuser@ip-172-31-8-102:~$
```

1.4 Execute the following commands to run the Docker in detached mode and mount the local **html** directory to the **/var/www/html** directory in the container:
**sudo docker run -d --name myweb1 -v **
**type=bind,source="$(pwd)"/html,target=/var/www/html nginx**

**sudo docker run -d --name myweb2 -v **
**"$(pwd)"/html:/var/www/html nginx**

**sudo docker run -d --name myweb3 -v **
**"$(pwd)"/html:/var/www/html:ro nginx**

```
labsuser@ip-172-31-8-102:~$ sudo docker run -d --name myweb1 -v \
> type=bind,source="$(pwd)"/html,target=/var/www/html nginx
6472b81af1c6ee1c7fd76eeffada319e0d3a76180bfcccf634f9b4f097d4efb4
labsuser@ip-172-31-8-102:~$
```

```
labsuser@ip-172-31-8-102:~$ sudo docker run -d --name myweb2 -v \
> "$(pwd)"/html:/var/www/html nginx
6d15e555d53c0bfec6c637de0d8029d4179b2aceb3dbab82a05defc9daefe4a2
labsuser@ip-172-31-8-102:~$
```

.

```
labsuser@ip-172-31-8-102:~$ sudo docker run -d --name myweb3 -v \
> "$(pwd)"/html:/var/www/html:ro nginx
fec4e1aab58c4494a71da07490aad9d759eb7a0e85c003df57b73b8ecd5c4b8f
labsuser@ip-172-31-8-102:~$
```

1.5 Run the following command to remove the containers:
**sudo docker rm -f myweb1 myweb2 myweb3**

```
labsuser@ip-172-31-8-102:~$ sudo docker rm -f myweb1 myweb2 myweb3
myweb1
myweb2
myweb3
labsuser@ip-172-31-8-102:~$
```

## Step 2: Manage data and temporary file systems within Docker containers

2.1 Execute the following command to create a temporary RAM-based file system:
**sudo docker run -d --name myweb1 **
**--mount type=tmpfs,destination=/app nginx**

**sudo docker run -d --name myweb2 --tmpfs /app nginx**

```
labsuser@ip-172-31-8-102:~$ sudo docker run -d --name myweb1 \
> --mount type=tmpfs,destination=/app nginx
5d59e60bbca6f0570e196f23e605e693193a699eb435a1d63dfc56768abe4a4a
labsuser@ip-172-31-8-102:~$
```

```
labsuser@ip-172-31-8-102:~$ sudo docker run -d --name myweb2 --tmpfs /app nginx
c1e0ada745149b3a313afda4e4c87fe4a2350af35afdc987022b8658979d0419
labsuser@ip-172-31-8-102:~$
```

2.2 Run the following command to remove the containers:
**sudo docker rm -f myweb1 myweb2**

```
c1e0ada745149b3a313afda4e4c87fe4a2350af35afdc987022b8658979d0419
labsuser@ip-172-31-8-102:~$ sudo docker rm -f myweb1 myweb2
myweb1
myweb2
```

2.3 Create a container from the Alpine image with **my_vol** volume using the following command:
**sudo docker run -it --name webapp -v my_vol:/var/www/html alpine**

.

```
myweb2
labsuser@ip-172-31-8-102:~$ sudo docker run -it --name webapp -v my_vol:/var/www/html alpine
Unable to find image 'alpine:latest' locally
latest: Pulling from library/alpine
4abcf2066143: Pull complete
Digest: sha256:c5b1261d6d3e43071626931fc004f70149baeba2c8ec672bd4f27761f8e1ad6b
Status: Downloaded newer image for alpine:latest
/ #
```

2.4 Execute the following commands to create three files inside the **/var/www/html** directory:

**cd /var/www/html**

**touch test1 test2 test3 && mkdir testdir**

**exit**

```
labsuser@ip-172-31-8-102:~$ sudo docker run -it --name webapp -v my_vol:/var/www/html alpine
Unable to find image 'alpine:latest' locally
latest: Pulling from library/alpine
4abcf2066143: Pull complete
Digest: sha256:c5b1261d6d3e43071626931fc004f70149baeba2c8ec672bd4f27761f8e1ad6b
Status: Downloaded newer image for alpine:latest
/ # cd /var/www/html
/var/www/html # touch test1 test2 test3 && mkdir testdir
/var/www/html # exit
labsuser@ip-172-31-8-102:~$
```

By following these steps, you have successfully mounted volumes via swarm services, enabling efficient data management and persistent storage across containers within the Docker swarm environment.