**Khushi chauhan**

**Sap 500105956**

**Rollno R2142220261**

**Batch b1 H**

## Lab Exercise 3: Working with Docker Volumes

Objective:

· Learn how to create and manage Docker volumes.

· Understand how Docker volumes can be used to persist data across container restarts.

· Practice mounting Docker volumes to containers.

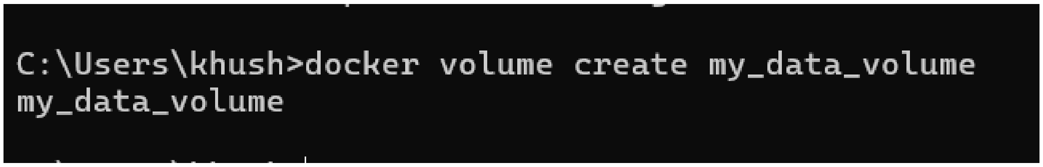
Prerequisites:

· Docker installed on your system.

· Basic understanding of Docker commands and container concepts.

Step 1: Create a Docker Volume

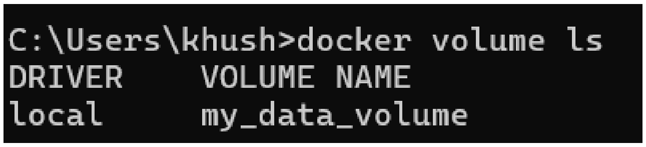
Create a new Docker volume:



```
C:\Users\khush>docker volume create my_data_volume
my_data_volume
```

This command creates a Docker volume named my_data_volume.

Verify that the volume was created:

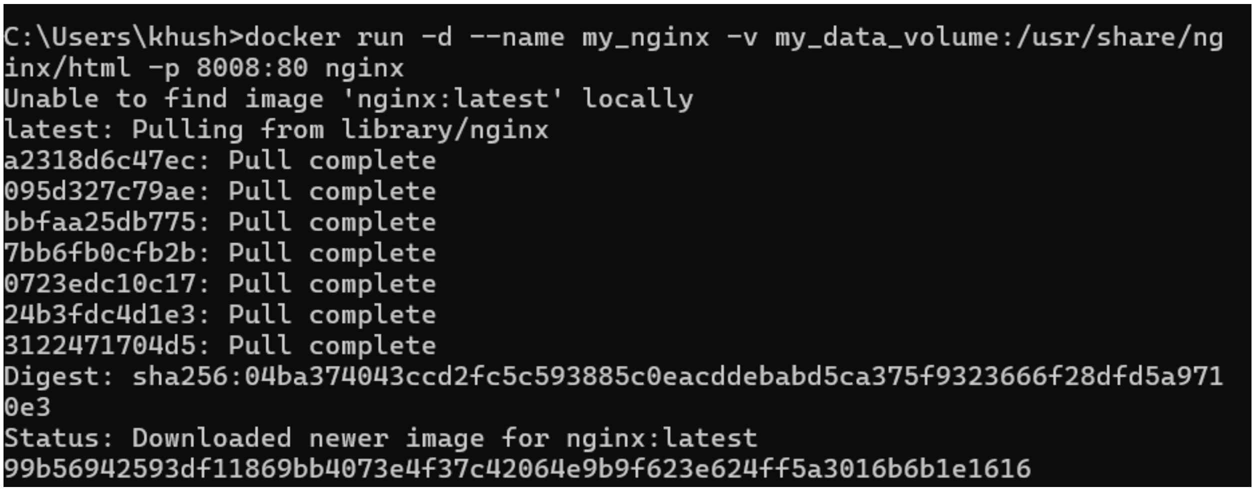


```
C:\Users\khush>docker volume ls
DRIVER    VOLUME NAME
local     my_data_volume
```

You should see my_data_volume listed among the volumes.

Step 2: Run a Container with the Volume Mounted

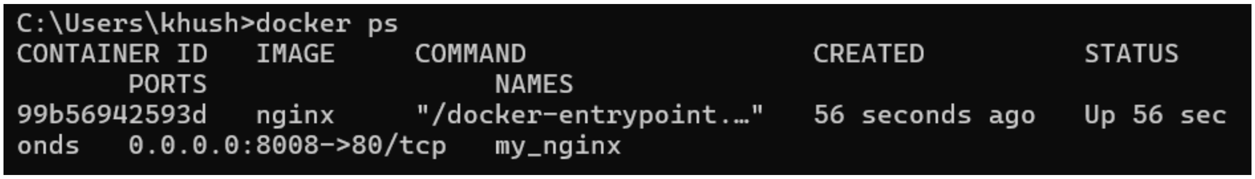
Run an Nginx container with the volume mounted:



```
C:\Users\khush>docker run -d --name my_nginx -v my_data_volume:/usr/share/ng
inx/html -p 8008:80 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
a2318d6c47ec: Pull complete
095d327c79ae: Pull complete
bbfaa25db775: Pull complete
7bb6fb0cfb2b: Pull complete
0723edc10c17: Pull complete
24b3fdc4d1e3: Pull complete
3122471704d5: Pull complete
Digest: sha256:04ba374043ccd2fc5c593885c0eacddebabd5ca375f9323666f28dfd5a971
0e3
Status: Downloaded newer image for nginx:latest
99b56942593df11869bb4073e4f37c42064e9b9f623e624ff5a3016b6b1e1616
```

This command starts an Nginx container named my_nginx and mounts the my_data_volume volume to the /usr/share/nginx/html directory inside the container.

Verify that the container is running:

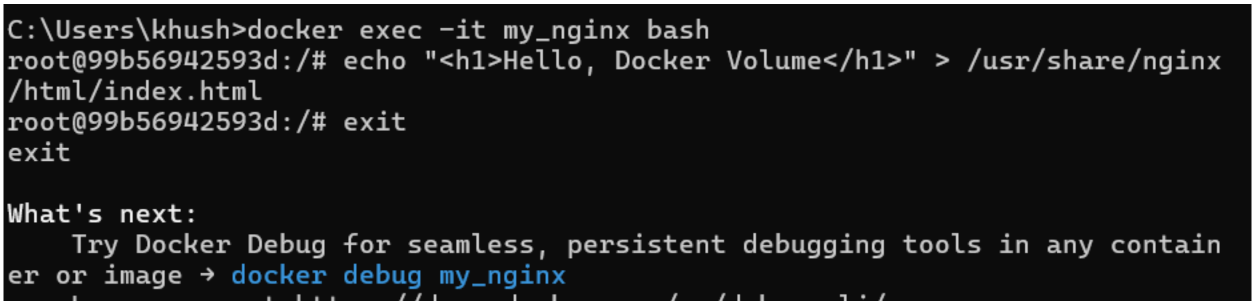


```
C:\Users\khush>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS
      PORTS                  NAMES
99b56942593d   nginx     "/docker-entrypoint.…"   56 seconds ago   Up 56 sec
onds   0.0.0.0:8008->80/tcp   my_nginx
```

You should see my_nginx listed as one of the running containers.

Step 3: Interact with the Volume

Create a simple HTML file in the volume:



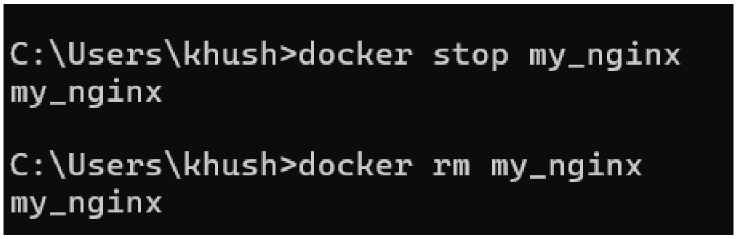
```
C:\Users\khush>docker exec -it my_nginx bash
root@99b56942593d:/# echo "<h1>Hello, Docker Volume</h1>" > /usr/share/nginx
/html/index.html
root@99b56942593d:/# exit
exit

What's next:
    Try Docker Debug for seamless, persistent debugging tools in any contain
er or image → docker debug my_nginx
```

This command creates an HTML file inside the /usr/share/nginx/html directory, which is backed by my_data_volume.

Access the Nginx server to see your file: Open a browser and navigate to http://localhost:8008. You should see the message "Hello, Docker Volume!" displayed on the page.

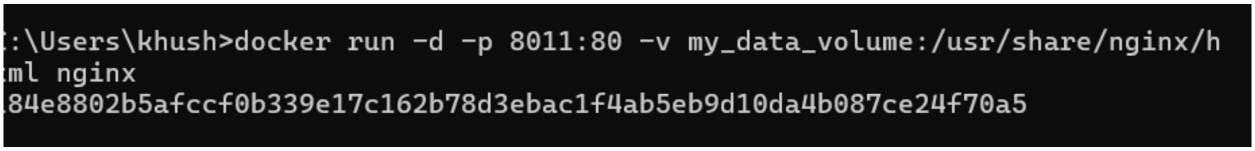
Step 4: Test Data Persistence

Stop and remove the container:



```
C:\Users\khush>docker stop my_nginx
my_nginx

C:\Users\khush>docker rm my_nginx
my_nginx
```

Run a new Nginx container using the same volume:



```
:\Users\khush>docker run -d -p 8011:80 -v my_data_volume:/usr/share/nginx/h
ml nginx
84e8802b5afccf0b339e17c162b78d3ebac1f4ab5eb9d10da4b087ce24f70a5
```

Access the Nginx server again: Navigate to http://localhost in your browser. You should still see the "Hello, Docker Volume!" message, demonstrating that the data persisted across container instances.