UPES

UNIVERSITY OF TOMORROW

Application Containerization And Orchestration Lab

Submitted By – Chitwan Singh

SAP ID – 500097009

Enrolment no. – R2142211291

Batch – DevOps B4

Submitted to

– Dr. Hitesh Kumar Sharma

Lab Experiment 2: Docker Volume

In this lab experiment, you will learn how to work with Docker volumes, which are used to
persist data across containers. Volumes enable data to be stored outside the container
filesystem and are crucial for managing data consistency and sharing data between containers.

PREREQUISITES:

Docker installed and running on your machine.

Objective:

Create a Docker volume, use it with a container, and observe how data persists across
container instances.

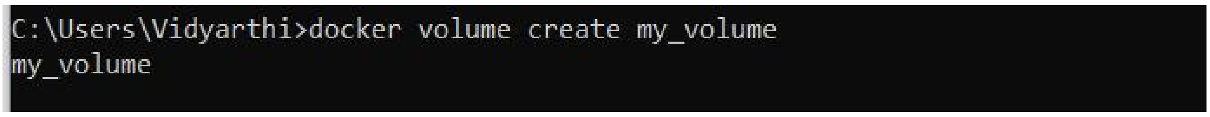
STEPS:

Step 1: Create a Docker Volume

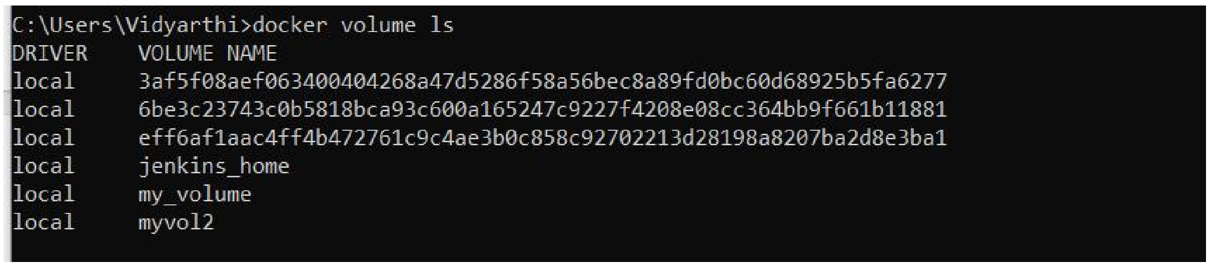
1.a Open a terminal on your machine.

1.b Run the following command to create a Docker volume named "my_volume":

docker volume create my_volume



```
C:\Users\Vidyarthi>docker volume create my_volume
my_volume
```

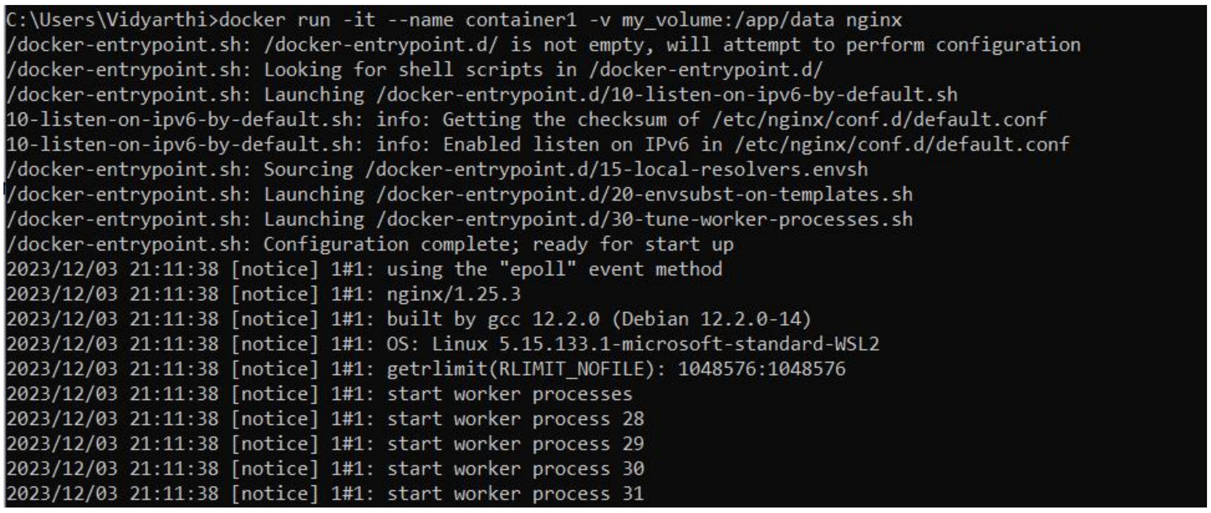


```
C:\Users\Vidyarthi>docker volume ls
DRIVER    VOLUME NAME
local     3af5f08aef063400404268a47d5286f58a56bec8a89fd0bc60d68925b5fa6277
local     6be3c23743c0b5818bca93c600a165247c9227f4208e08cc364bb9f661b11881
local     eff6af1aac4ff4b472761c9c4ae3b0c858c92702213d28198a8207ba2d8e3ba1
local     jenkins_home
local     my_volume
local     myvol2
```

Step 2: Launch Containers with the Volume

2.a Run a container using the volume you created:

docker run -it --name container1 -v my_volume:/app/data nginx



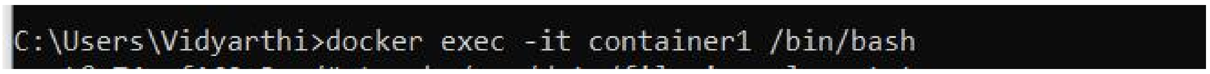
```
C:\Users\Vidyarthi>docker run -it --name container1 -v my_volume:/app/data nginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/12/03 21:11:38 [notice] 1#1: using the "epoll" event method
2023/12/03 21:11:38 [notice] 1#1: nginx/1.25.3
2023/12/03 21:11:38 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2023/12/03 21:11:38 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2023/12/03 21:11:38 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/12/03 21:11:38 [notice] 1#1: start worker processes
2023/12/03 21:11:38 [notice] 1#1: start worker process 28
2023/12/03 21:11:38 [notice] 1#1: start worker process 29
2023/12/03 21:11:38 [notice] 1#1: start worker process 30
2023/12/03 21:11:38 [notice] 1#1: start worker process 31
```

2.b Enter the container to observe the volume and create a file inside it:

touch /app/data/file_in_volume.txt

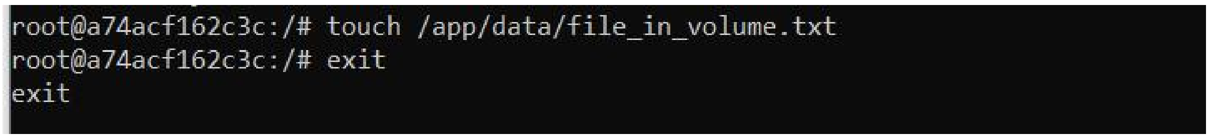
exit

Entering inside the container ‘container1’:



```
C:\Users\Vidyarthi>docker exec -it container1 /bin/bash
```

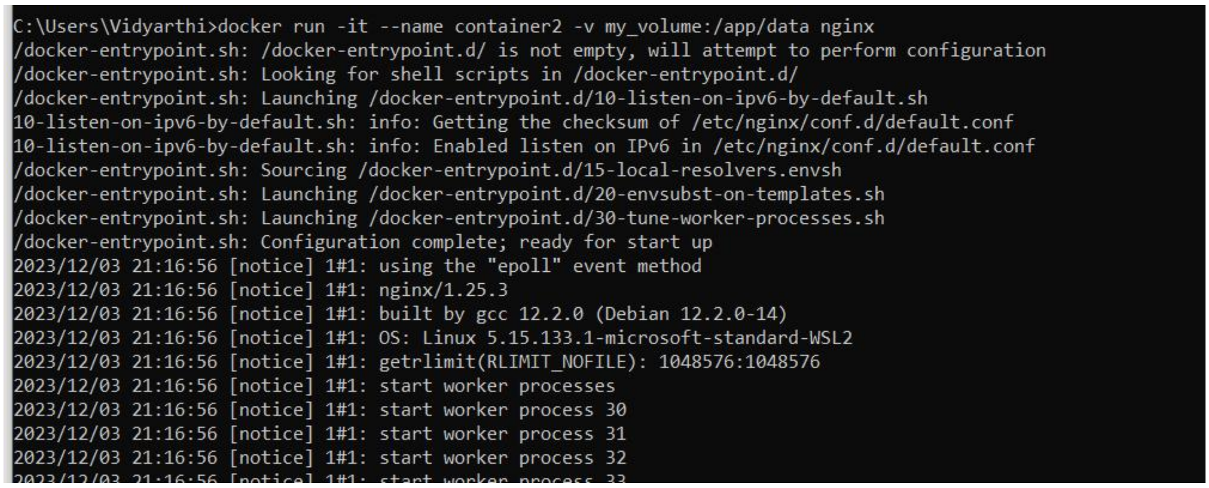
Creating a file inside the ‘container1’:



```
root@a74acf162c3c:/# touch /app/data/file_in_volume.txt
root@a74acf162c3c:/# exit
exit
```

2.c Run a second container, using the same volume, to verify data persistence:

docker run -it --name container2 -v my_volume:/app/data nginx



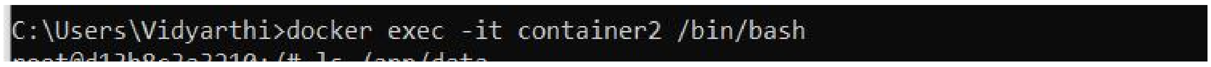
```
C:\Users\Vidyarthi>docker run -it --name container2 -v my_volume:/app/data nginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/12/03 21:16:56 [notice] 1#1: using the "epoll" event method
2023/12/03 21:16:56 [notice] 1#1: nginx/1.25.3
2023/12/03 21:16:56 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2023/12/03 21:16:56 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2023/12/03 21:16:56 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/12/03 21:16:56 [notice] 1#1: start worker processes
2023/12/03 21:16:56 [notice] 1#1: start worker process 30
2023/12/03 21:16:56 [notice] 1#1: start worker process 31
2023/12/03 21:16:56 [notice] 1#1: start worker process 32
2023/12/03 21:16:56 [notice] 1#1: start worker process 33
```

2.d Enter the second container and check if the file exists:

ls /app/data

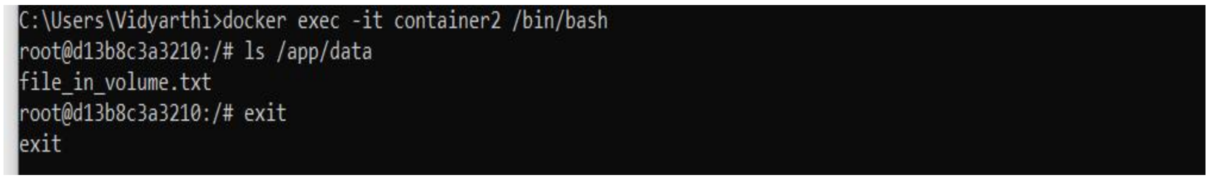
exit

Entering inside the container ‘container2’:



```
C:\Users\Vidyarthi>docker exec -it container2 /bin/bash
root@d13b8c3a3210:/# ls /app/data
```

Checking if the file ‘file_in_volume.txt’ exists in the ‘container2’ or not:



```
C:\Users\Vidyarthi>docker exec -it container2 /bin/bash
root@d13b8c3a3210:/# ls /app/data
file_in_volume.txt
root@d13b8c3a3210:/# exit
exit
```

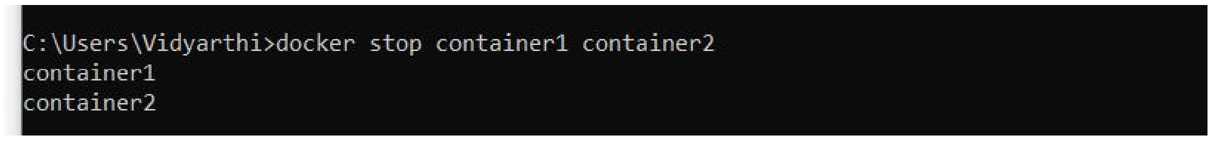
Step 3: Cleanup

3.a Stop and remove the containers:

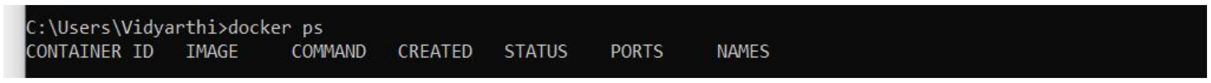
docker stop container1 container2

docker rm container1 container2

Stopping the containers ‘container1’ and ‘container2’:

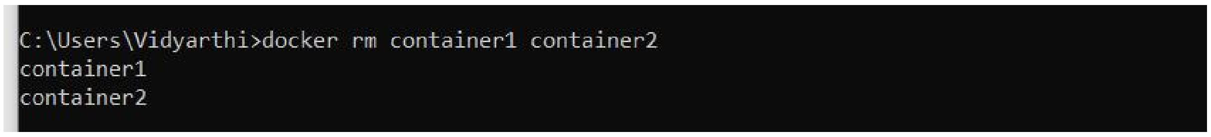


```
C:\Users\Vidyarthi>docker stop container1 container2
container1
container2
```



```
C:\Users\Vidyarthi>docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

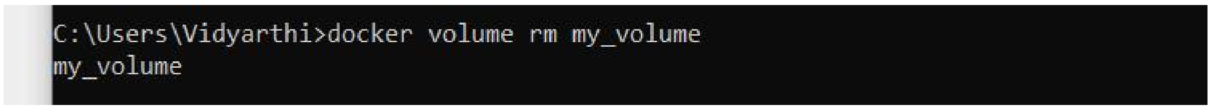
Removing the containers ‘container1’ and ‘container2’:



```
C:\Users\Vidyarthi>docker rm container1 container2
container1
container2
```

3.b Remove the volume:

docker volume rm my_volume



```
C:\Users\Vidyarthi>docker volume rm my_volume
my_volume
```

Conclusion:

In this experiment, you learned how to create a Docker volume, associate it with containers,
and observed how data persisted between different container instances. Docker volumes are
essential for maintaining data integrity, sharing data between containers, and ensuring data
persistence even when containers are removed or replaced. This skill is crucial for managing
stateful applications and databases within a Dockerized environment.