Name-Anushka Chamoli

Batch-4 (DevOps)

Roll No- R2142211336

Sap Id-500097354

Lab Experiment 3: Docker Volume

Steps:

Step 1: Create a Docker Volume

Open a terminal on machine.

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

```
anushka@anushka-VirtualBox:~$ docker volume create my_volume
my_volume
```

Step 2: Launch Containers with the Volume

Run a container using the volume you created:

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

```
anushka@anushka-VirtualBox:~$ 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
2023/12/02 18:52:13 [notice] 1#1: using the "epoll" event method
2023/12/02 18:52:13 [notice] 1#1: nginx/1.25.2
2023/12/02 18:52:13 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2023/12/02 18:52:13 [notice] 1#1: OS: Linux 6.2.0-35-generic
2023/12/02 18:52:13 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/12/02 18:52:13 [notice] 1#1: start worker processes
2023/12/02 18:52:13 [notice] 1#1: start worker process 29
2023/12/02 18:52:13 [notice] 1#1: start worker process 30
2023/12/02 18:52:13 [notice] 1#1: start worker process 31
2023/12/02 18:52:13 [notice] 1#1: start worker process 32
```

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

```
anushka@anushka-VirtualBox:~$ sudo touch /app/data/file_in_volume.txt
anushka@anushka-VirtualBox:~$ docker exec -it container5  /bin/bash
root@371a6f22d520:/# touch /app/data/file_in_volume.txt
root@371a6f22d520:/# exit
```

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

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

Enter the second container and check if the file exists:

```
anushka@anushka-VirtualBox:~$ docker exec -it container6  /bin/bash
root@456caa9da255:/# ls /app/data
file_in_volume.txt
root@456caa9da255:/# exit
```

Step 3: Cleanup

Stop and remove the containers:

```
anushka@anushka-VirtualBox:~$ docker stop container5 container6
container5
container6
anushka@anushka-VirtualBox:~$ docker rm container5 container6
container5
container6
```

Remove the volume:

```
anushka@anushka-VirtualBox:~$ docker volume rm my_volume
my_volume
anushka@anushka-VirtualBox:~$
```

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.