Docker Hands-On Lab 4 — Volumes and Data Persistence

Trainer: Sandeep Kumar Sharma

Duration: 45–60 minutes

Goal: In this lab, you will understand Docker Volumes — what they are, why we use them, different types of volumes, and how to use the default volume for persistent data storage. Every container we create in this lab will be named **sandeep-container**.

---

1) Understanding Volumes in Docker

By default, Docker containers are **ephemeral**, meaning any data you create inside a container is lost when the container is removed. To solve this, Docker provides **volumes**, which are special directories stored outside of the container's writable layer.

In simple terms:

> Volumes allow your container data to persist even if the container is deleted or recreated.

Analogy: Think of a container as a temporary workspace and a volume as a permanent hard drive that stays safe even if the workspace is destroyed.

---

2) Why Do We Need Volumes?

- Containers are designed to be **stateless** — once they stop or are deleted, their data vanishes.
- Volumes are used to **store persistent data**, such as:
- Database files (MySQL, PostgreSQL, etc.)
- Application logs
- Uploaded user content
- Volumes allow **data sharing** between multiple containers.
- They improve **performance** and **manageability** compared to storing data inside the container layer.

---

3) Types of Docker Volumes

Docker supports three main types of volumes:

1. **Named Volume** — Created and managed by Docker, stored under `/var/lib/docker/volumes`.

```
docker volume create mydata
docker run -v mydata:/data --name sandeep-container ubuntu
```

2. **Anonymous Volume** — Created automatically when you use `-v /path` without a name. Docker assigns a random name.

3. **Bind Mount** — Links a specific host directory to a container directory.

```
docker run -v /home/user/data:/data --name sandeep-container ubuntu
```

In this lab, we'll work with **default (anonymous) volumes**.

---

4) Lab Exercise — Using Default Volume

Step 1: Run a Container with a Default Volume

We'll use Ubuntu as a base image and name our container **sandeep-container**.

```
docker run -it -v /data --name sandeep-container ubuntu:latest /bin/bash
```

This command: - Runs an interactive container from Ubuntu - Creates an **anonymous volume** attached to `/data` inside the container - Names the container `sandeep-container`

Step 2: Create a File Inside the Volume

Inside the container:

```
cd /data
echo "Hello from Docker Volume!" > volume_test.txt
ls -l
cat volume_test.txt
```

You should see your file content.

Step 3: Exit the Container

```
exit
```

Step 4: Verify the Volume Exists

Run:

```
docker volume ls
```

You'll see an automatically created volume like:

```
DRIVER    VOLUME NAME
local     2a8e1a7b7a4c90d7b1f...
```

Inspect this volume:

```
docker volume inspect <volume_name>
```

You'll see the mount path (usually `/var/lib/docker/volumes/<volume_name>/_data`).

Step 5: Remove the Container but Keep the Volume

```
docker stop sandeep-container
docker rm sandeep-container
```

Even though the container is deleted, the volume still exists. Confirm again:

```
docker volume ls
```

Step 6: Attach the Same Volume to a New Container

Run a new container using the same volume name and again name it **sandeep-container**:

```
docker run -it -v <volume_name>:/data --name sandeep-container ubuntu:latest /
bin/bash
```

Inside the container:

```
cd /data
cat volume_test.txt
```

You'll still see your file — proving **data persistence**.

Step 7: Cleanup (Optional)

```
docker stop sandeep-container
docker rm sandeep-container
docker volume prune
```

This removes all stopped containers and unused volumes.

---

5) Practice Tasks

1. Create a new container named `sandeep-container` with a volume mounted to `/mydata`.
2. Inside the container, create a text file.
3. Delete the container and create a new one using the same volume — verify the file still exists.
4. Inspect the volume's host path and explore its contents.

---

6) Summary

- Volumes provide persistent storage for containers.
- Default (anonymous) volumes are created automatically when you use `-v /path`.
- Volume data remains even after containers are removed.
- Use `docker volume ls` and `docker volume inspect` to manage and view details.
- Every container created in this lab was named **sandeep-container** for consistency.

✅**Checkpoint:** You've successfully created, used, and verified a default Docker volume for data persistence using your container **sandeep-container**!

Next, we'll explore **Named Volumes and Bind Mounts** to understand more advanced volume management.