Docker Hands-On Lab 6 — Named Volumes

Trainer: Sandeep Kumar Sharma

Duration: 45–60 minutes

Goal: In this lab, you will learn what **Named Volumes** are, how they differ from bind mounts and anonymous volumes, and how to use them for data persistence using a container named **sandeep-container**.

---

1) Understanding Named Volumes

Named volumes are **Docker-managed persistent storage locations**. Unlike bind mounts, which use absolute host paths, named volumes are stored under Docker's own directory (typically `/var/lib/docker/volumes`) and are managed entirely by Docker.

In simple terms:

> A named volume is like a dedicated folder created and maintained by Docker for your container's data.

Analogy: Think of a named volume as a labeled storage box managed by Docker — you can reuse it with any container, and it's easy to back up and migrate.

---

2) Why Do We Use Named Volumes?

- To persist application data even after a container is deleted.
- To allow multiple containers to share the same data easily.
- To let Docker manage the storage location automatically.
- To make backup, restore, and migration easier than with bind mounts.

Common use cases: - Databases like MySQL, PostgreSQL - Application logs and configuration files

---

3) Difference Between Bind Mounts and Named Volumes

Feature	Bind Mount	Named Volume
Storage Location	Any host directory	`/var/lib/docker/volumes`
Managed By	User/Host	Docker

Feature	Bind Mount	Named Volume
Use Case	Development & file sync	Data persistence & portability
Backup & Restore	Manual	Easy (via Docker commands)
Security	Host-dependent	More isolated & safer

---

4) Lab Exercise — Creating and Using a Named Volume

Step 1: Create a Named Volume

```
docker volume create sandeep-data
```

Verify the volume:

```
docker volume ls
```

You should see something like:

```
DRIVER    VOLUME NAME
local     sandeep-data
```

Step 2: Run a Container with the Named Volume

Now, create and run an **Ubuntu** container named **sandeep-container**, attaching the named volume to `/data` inside the container.

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

Inside the container, navigate to `/data` and create a file:

```
cd /data
echo "This file is stored in a named volume." > named_volume.txt
ls -l
```

Exit the container:

```
exit
```

Step 3: Verify Volume Data Persistence

Now, remove the container but **do not remove the volume**.

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

List volumes again:

```
docker volume ls
```

The `sandeep-data` volume will still be present.

Inspect it:

```
docker volume inspect sandeep-data
```

You'll see details like its mount path (usually under `/var/lib/docker/volumes/sandeep-data/_data`).

Step 4: Reuse the Same Volume with a New Container

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

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

Inside the container:

```
cd /data
cat named_volume.txt
```

You'll still see the file created earlier, showing **data persistence**.

Step 5: Mount the Same Volume to Multiple Containers

You can also share the same volume between containers.

Run another container:

```
docker run -it --name sandeep-container-2 -v sandeep-data:/shared-data
ubuntu:latest /bin/bash
```

Inside the new container:

```
cd /shared-data
ls
```

You'll see the same file available in both containers — **shared storage**.

Step 6: Cleanup (Optional)

Stop and remove the containers and volume:

```
docker stop sandeep-container sandeep-container-2
docker rm sandeep-container sandeep-container-2
docker volume rm sandeep-data
```

---

5) Practice Tasks

1. Create a named volume called `sandeep-db` and mount it to `/db` inside a container.
2. Create a file in `/db` and verify its persistence after deleting and recreating the container.
3. Share the same volume with two containers and confirm data synchronization.

---

6) Summary

- Named volumes are managed by Docker and stored under `/var/lib/docker/volumes`.
- They are easy to back up, migrate, and share across containers.
- Data inside named volumes persists even when containers are deleted.
- They provide better data isolation and management compared to bind mounts.
- In this lab, all containers were consistently named **sandeep-container** for clarity.

✅**Checkpoint:** You've successfully created, used, and verified a **Named Volume** for data persistence and sharing using your container **sandeep-container**!

Next, we can explore **Docker Compose** to manage multi-container applications efficiently.