

Name-Aayush Mishra

sap id-500107141

Roll no- R2142220385

Batch-B1 Devops (NH)

Experiment: 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.

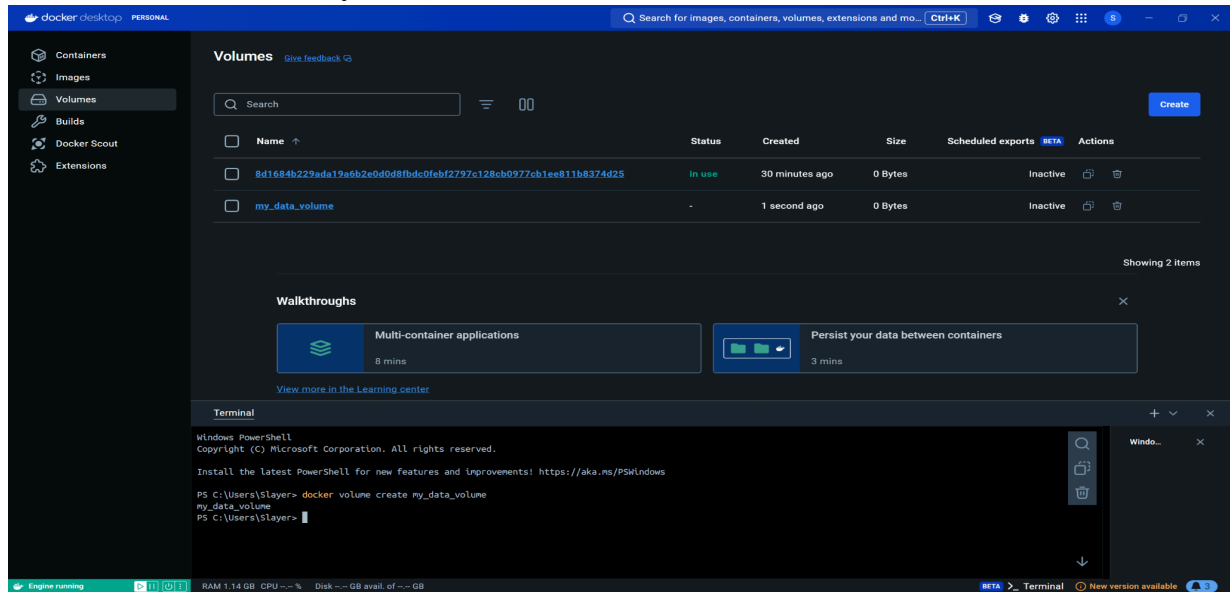
Prerequisites:

- Docker installed on your system.
- Basic understanding of Docker commands and container concepts.

Step 1: Create a Docker Volume

Create a new Docker volume:

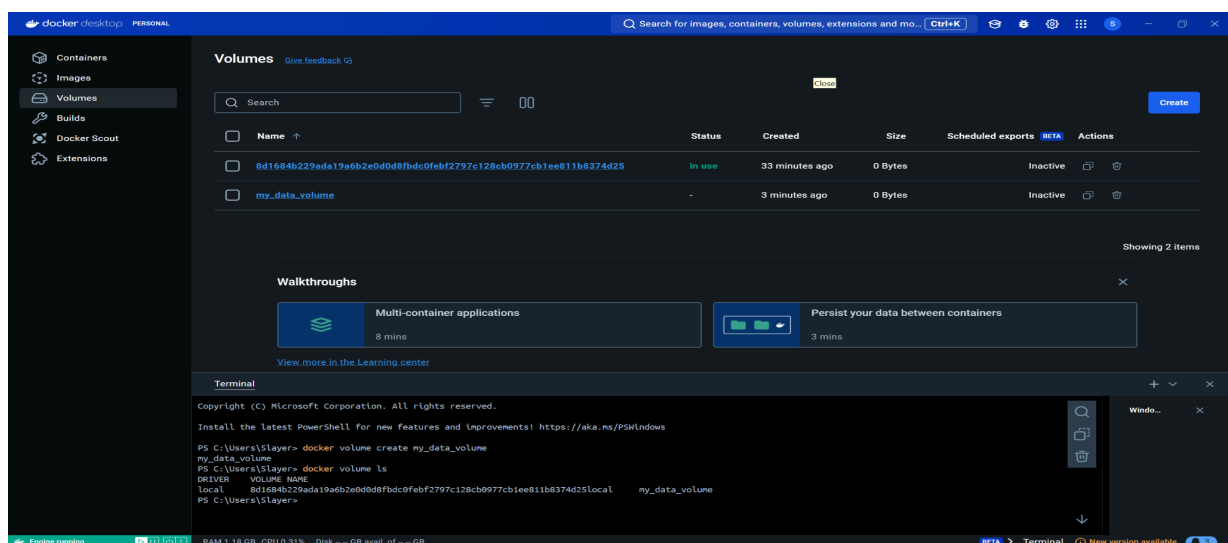
`docker volume create my_data_volume`



This command creates a Docker volume named `my_data_volume`.

Verify that the volume was created:

`docker volume ls`

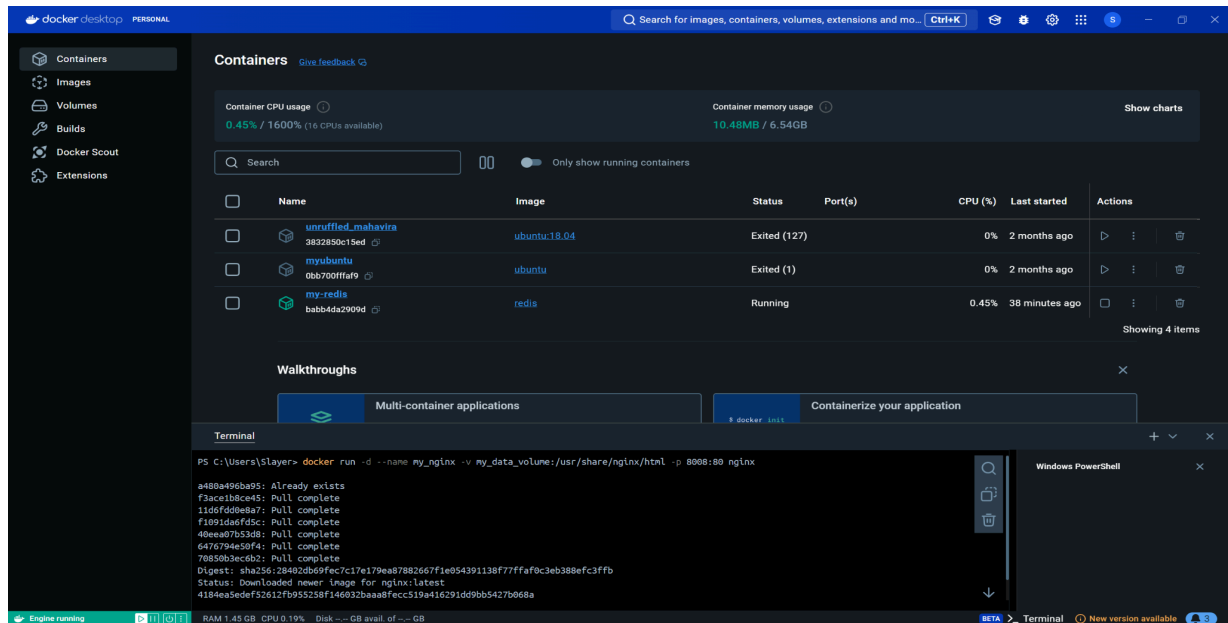


You should see `my_data_volume` listed among the volumes.

Step 2: Run a Container with the Volume Mounted

Run an Nginx container with the volume mounted:

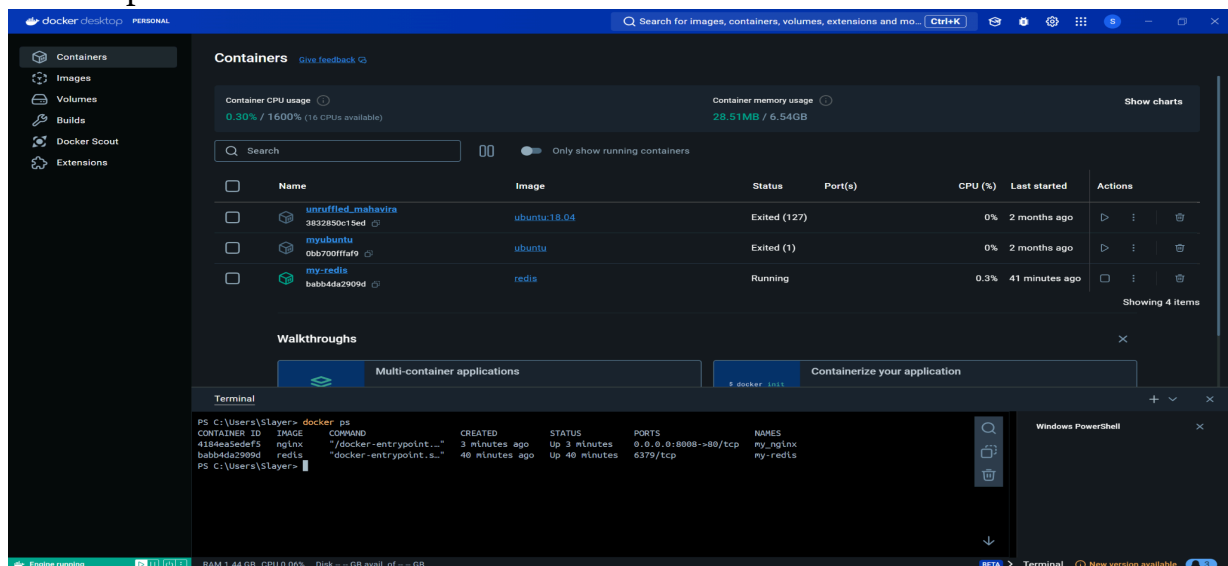
```
docker run -d --name my_nginx -v my_data_volume:/usr/share/nginx/html -p 8008:80 nginx
```



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

Verify that the container is running:

```
docker ps
```



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

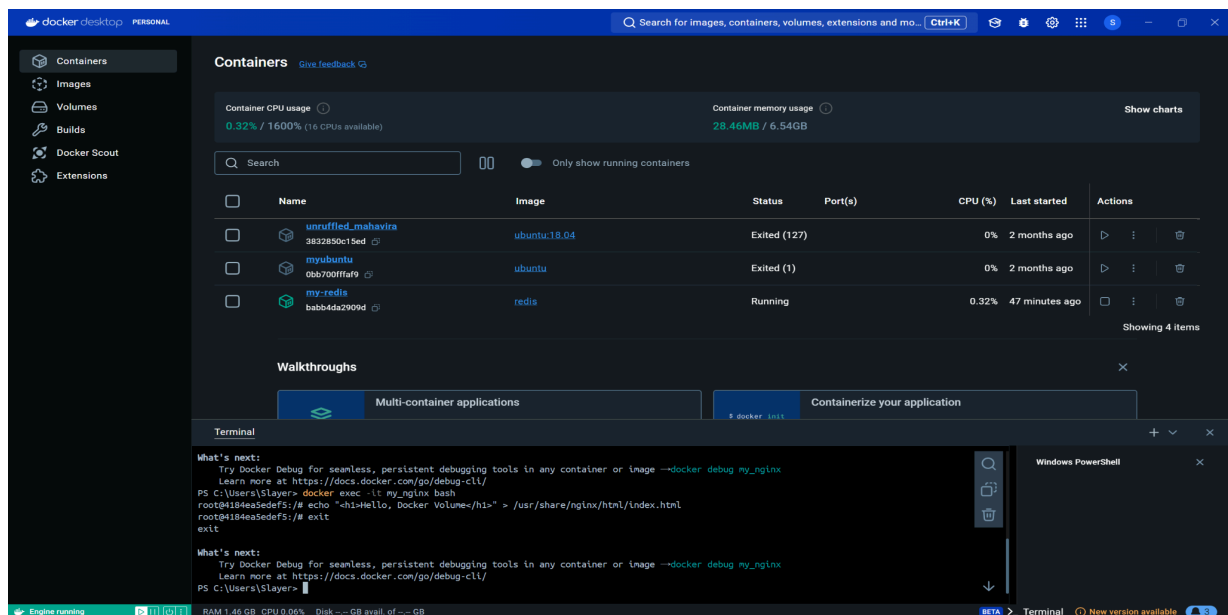
Step 3: Interact with the Volume

Create a simple HTML file in the volume:

```
docker exec -it my_nginx bash
```

```
echo "<h1>Hello, Docker Volume</h1>" > /usr/share/nginx/html/index.html
```

```
exit
```



Hello, Docker Volume

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

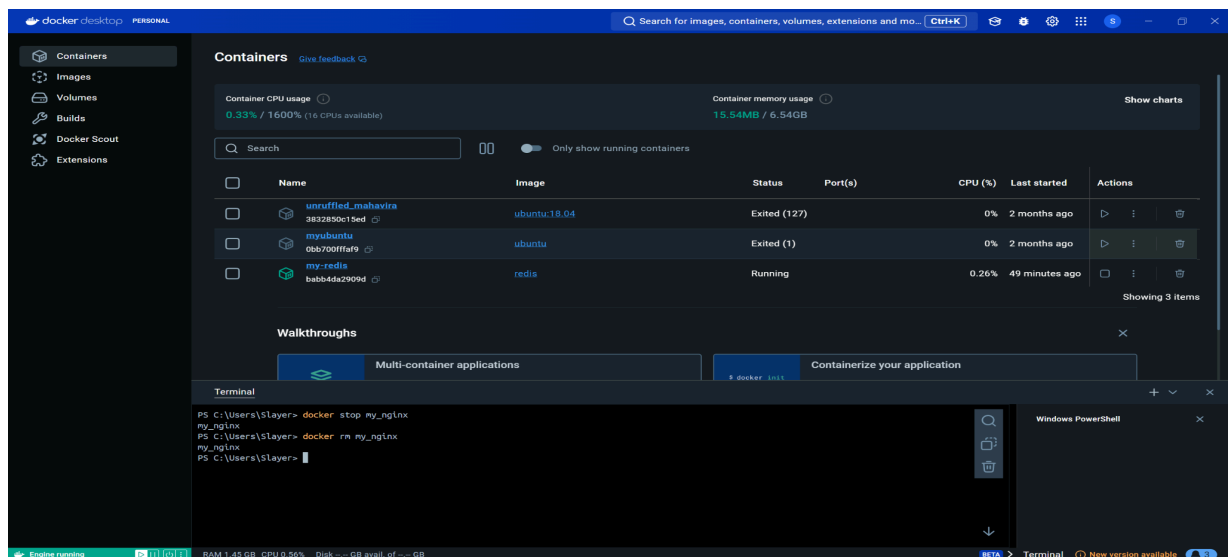
Access the Nginx server to see your file: Open a browser and navigate to `http://localhost:8008`. You should see the message "Hello, Docker Volume!" displayed on the page.

Step 4: Test Data Persistence

Stop and remove the container:

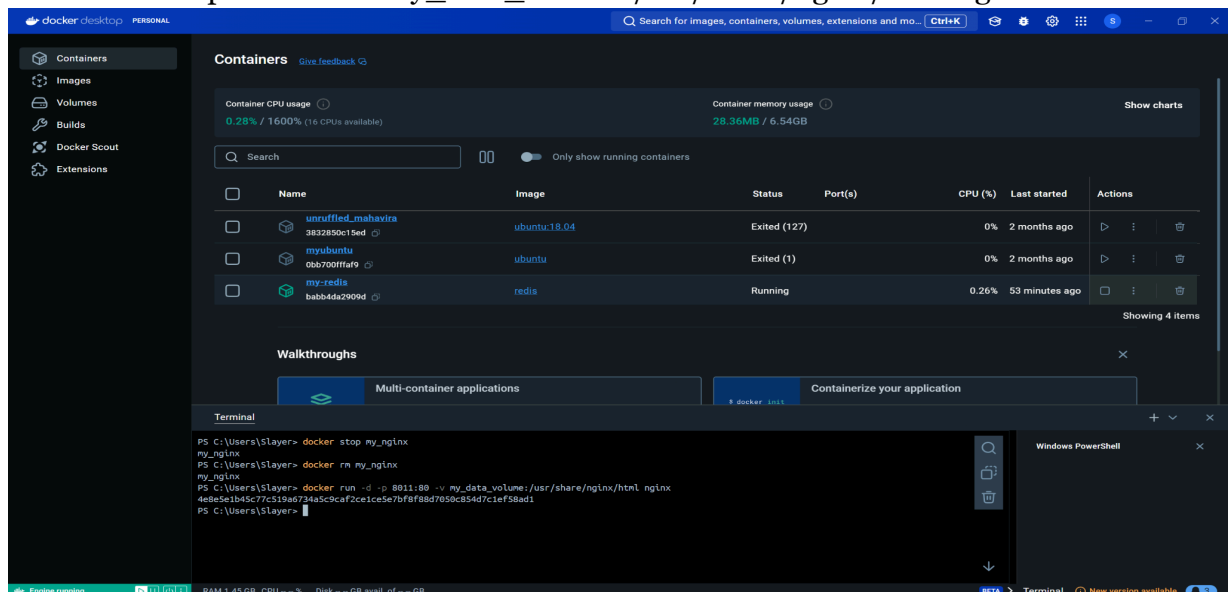
```
docker stop my_nginx
```

```
docker rm my_nginx
```



Run a new Nginx container using the same volume:

```
docker run -d -p 8011:80 -v my_data_volume:/usr/share/nginx/html nginx
```



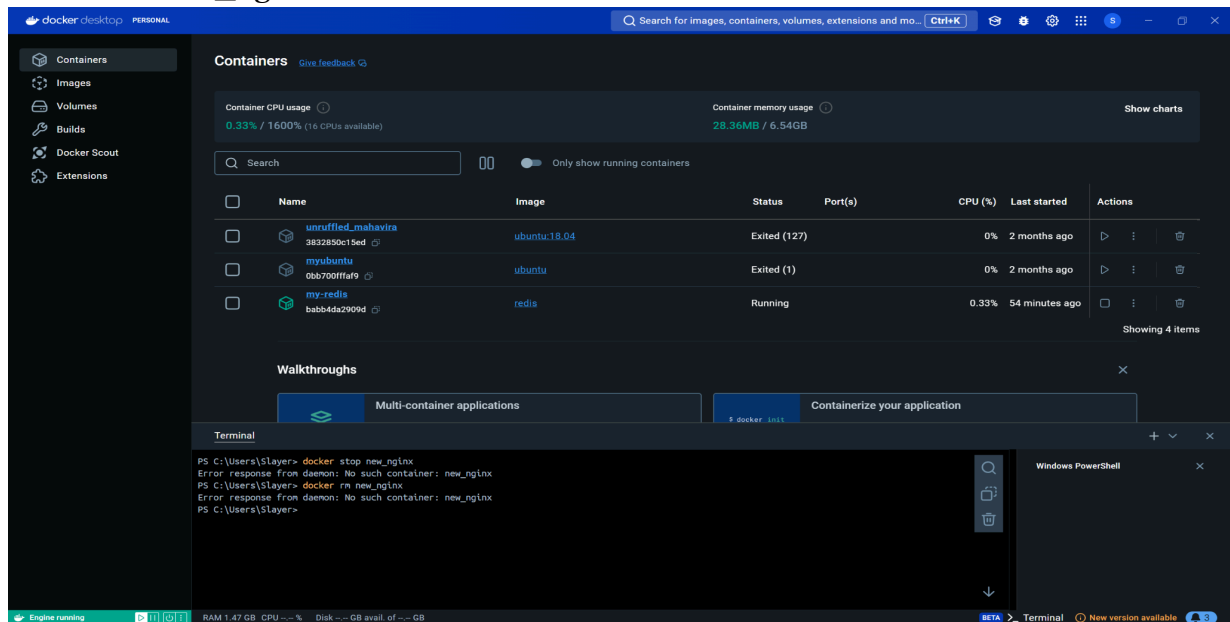
Access the Nginx server again: Navigate to `http://localhost` in your browser. You should still see the "Hello, Docker Volume!" message, demonstrating that the data persisted across container instances.

Step 5: Clean Up

Stop and remove the container:

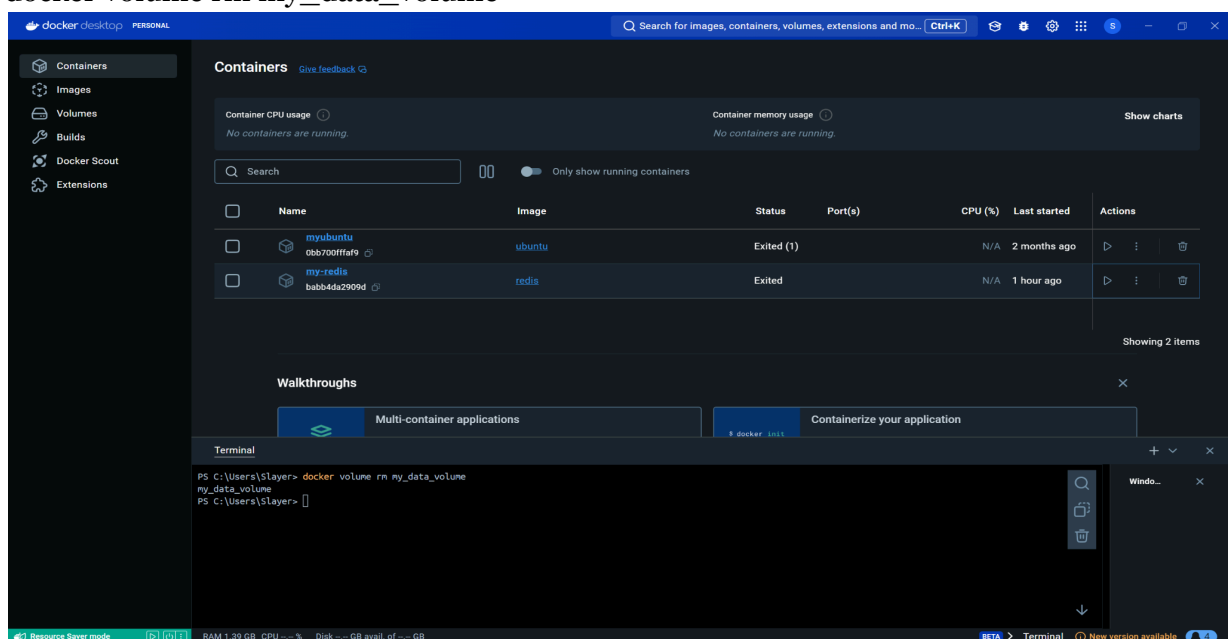
```
docker stop new_nginx
```

```
docker rm new_nginx
```



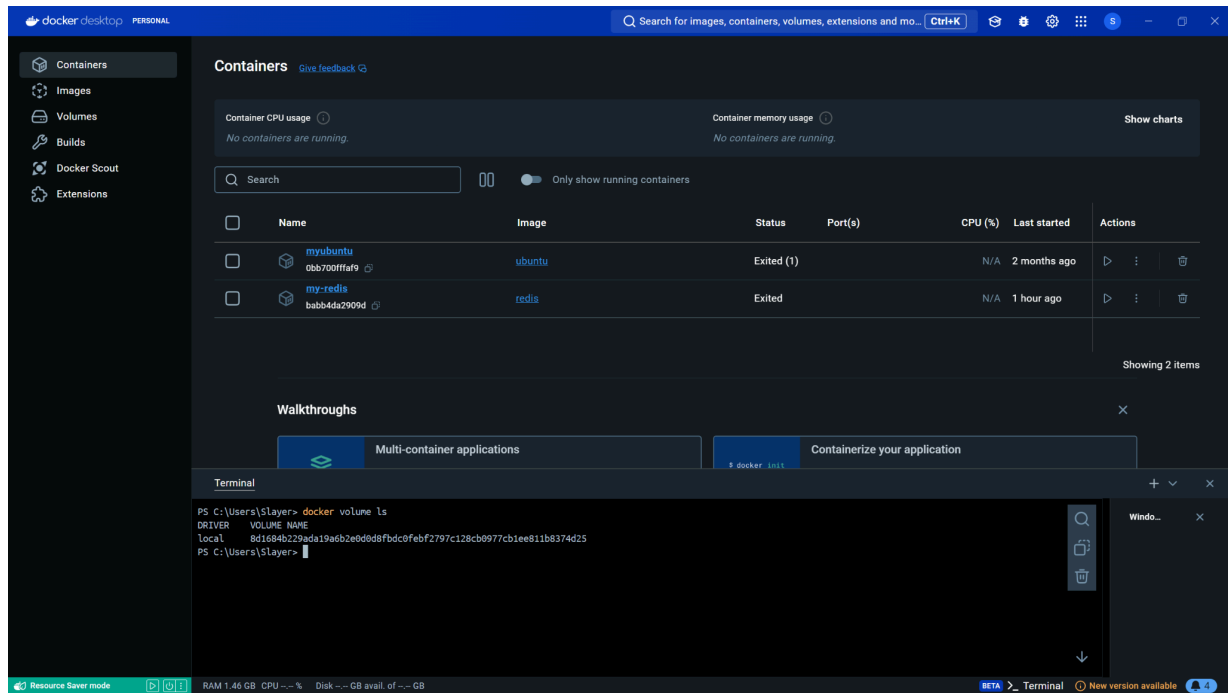
Remove the Docker volume:

```
docker volume rm my_data_volume
```



Verify that the volume is removed:

docker volume ls



The screenshot shows the Docker Desktop interface. On the left is a sidebar with navigation options: Containers, Images, Volumes, Builds, Docker Scout, and Extensions. The main area is titled 'Containers' and shows a table of containers. Below the table are 'Walkthroughs' and a 'Terminal' window.

Containers Table:

Name	Image	Status	Port(s)	CPU (%)	Last started	Actions
mysubuntu	ubuntu	Exited (1)		N/A	2 months ago	[Play] [More] [Delete]
my-redis	redis	Exited		N/A	1 hour ago	[Play] [More] [Delete]

Showing 2 items

Terminal Window:

```
PS C:\Users\Slayer> docker volume ls
DRIVER      VOLUME NAME
local      8d1684b229ada19a6b2e0d9d8fbd09feb2797c128cb9977cb1ee811b8374d25
PS C:\Users\Slayer>
```

The terminal output shows the command `docker volume ls` and its output, which lists a single volume with the name `8d1684b229ada19a6b2e0d9d8fbd09feb2797c128cb9977cb1ee811b8374d25`. This volume is associated with the `local` driver.

Ensure that my_data_volume is no longer listed.