**School of Computer Science**

**UNIVERSITY OF PETROLEUM AND ENERGY STUDIES**

**DEHRADUN, UTTARAKHAND**



**Containers & Docker Security**

**Lab File (2022-2026)**

# **5th Semester**

|  |  |
| --- | --- |
| *Submitted To:*  ***Dr. Hitesh Kumar Sharma*** | *Submitted By:*  *Anuj Pawadia*  *(500107126)*  *B Tech CSE*  *DevOps[5th Semester]*  *R2142220311*  *Batch - 1* |

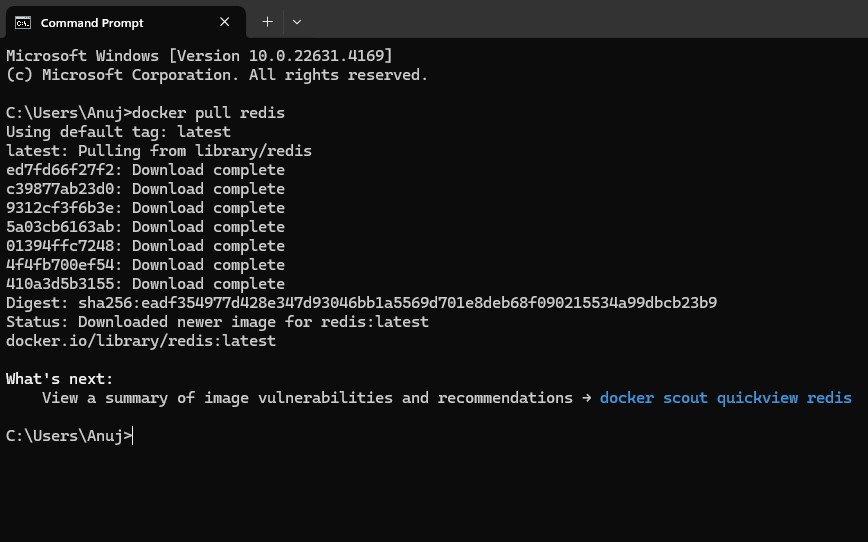
**EXPERIMENT 2**

**AIM: Working with Docker (Basic Commands)**

**Steps to Complete:**

1. Search a Redis image on Docker Hub

* **docker pull Redis:**downloads the Redis image from Docker-hub to your local machine. Docker-hub is a repository where Docker images are stored. The Redis image is used to run a Redis server. By pulling the image, you're preparing your system to create containers based on this image.

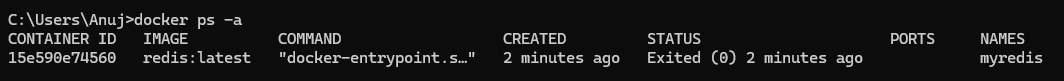


1. Run Docker Container of Redis Image in background

* **docker run:** This command creates and starts a new container from an image.
* **it:** The -i flag keeps the STDIN open, and the -t flag allocates a pseudo-TTY, allowing you to interact with the container via a terminal session.
* **--name myredis:** This names the container "*myredis*" for easy reference.
* **redis: latest:** Specifies the Redis image (using the latest version) to use for the container.
* **/bin/bash:** This opens a Bash shell inside the container, allowing you to interact with it directly. While this command starts the container interactively.

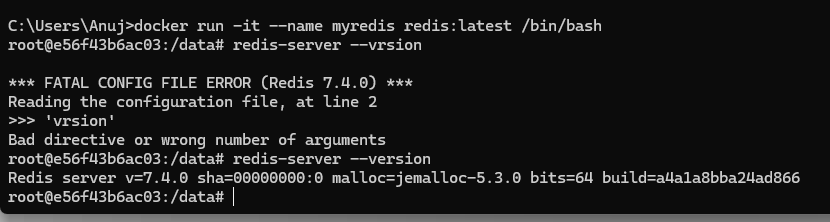
1. Run Docker PS and Docker PS –a

* **docker ps:** This lists all running Docker containers. It shows container IDs, names, and other details about active containers.
* **docker ps -a:** This lists all containers, including those that are stopped. This is useful to see all containers that exist on your system, not just the ones that are currently running.



1. Run Docker Container and take its console

* The command to start the container (*docker run -it --name myredis redis:latest /bin/bash*) is the same as in step 2, creating an interactive terminal session inside the container.



1. Create a Docker Volume and connect it

* **--mount source=myredisdata2,target=/data :** This option mounts the myredisdata2 volume to the /data directory inside the container. Any data written to /data inside the container will be stored in the myredisdata2 volume, persisting even if the container is removed.
* **touch /data/lab2.txt:** This creates a new file named lab2.txt in the /data directory, demonstrating that the volume is correctly mounted.
* **Ls:** This lists the contents of the /data directory, showing that **lab2.txt** has been successfully created, confirming that the volume is connected and functioning.

b