# Placement Empowerment Program

***Cloud Computing and DevOps Centre***

Automate Docker Container Management: Create a script that starts, stops, or removes specific Docker containers based on user input.

Name: Samyuktha V Department:Ads



# Introduction

In modern DevOps and cloud environments, managing Docker containers efficiently is crucial. Manually starting, stopping, or removing containers can be time-consuming, especially when dealing with multiple instances. This Proof of Concept (PoC) aims to automate Docker container management using a Windows Batch script. The script allows users to start, stop, or remove specific containers with a simple command, making container administration more efficient and error-free.

# Overview

This PoC involves creating a **batch script (docker\_manager.bat)** that interacts with Docker CLI commands to manage container lifecycle operations. The script uses conditional statements (IF conditions) to determine user input and execute corresponding Docker commands. It provides an easy-to-use command-line interface for container operations.

Key functionalities include:

✅ **Starting** a Docker container.

✅ **Stopping** a running container.

✅ **Removing** an existing container.

# Objectives

The primary goals of this PoC are:

1. **Automate** Docker container lifecycle management using a Windows Batch script.
2. **Simplify** repetitive container management tasks (start, stop, remove).
3. **Improve Efficiency** by reducing manual effort in handling containers.
4. **Enhance Usability** by providing a simple command-line interface.
5. **Ensure Flexibility** so users can extend or modify the script for their needs.

# Importance

1. **Reduces Manual Work**: No need to manually type Docker commands every time.
2. **Speeds Up Operations**: One command automates container management.
3. **Prepares for Advanced Automation**: This PoC is a foundation for integrating container management into larger DevOps workflows.
4. **Boosts Productivity**: DevOps teams can focus on development rather than container administration.
5. **Scalability**: The script can be enhanced to support multiple containers or advanced Docker functionalities like logs, volume management, and networking.

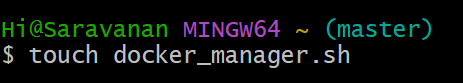
# Step-by-Step Overview

## Step 1:

We need to create a Bash script that will manage Docker containers. In Git Bash run:

**touch docker\_manager.sh**

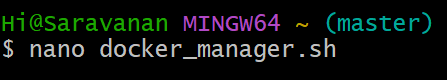
This will create an empty file named docker\_manager.sh.



## Step 2:

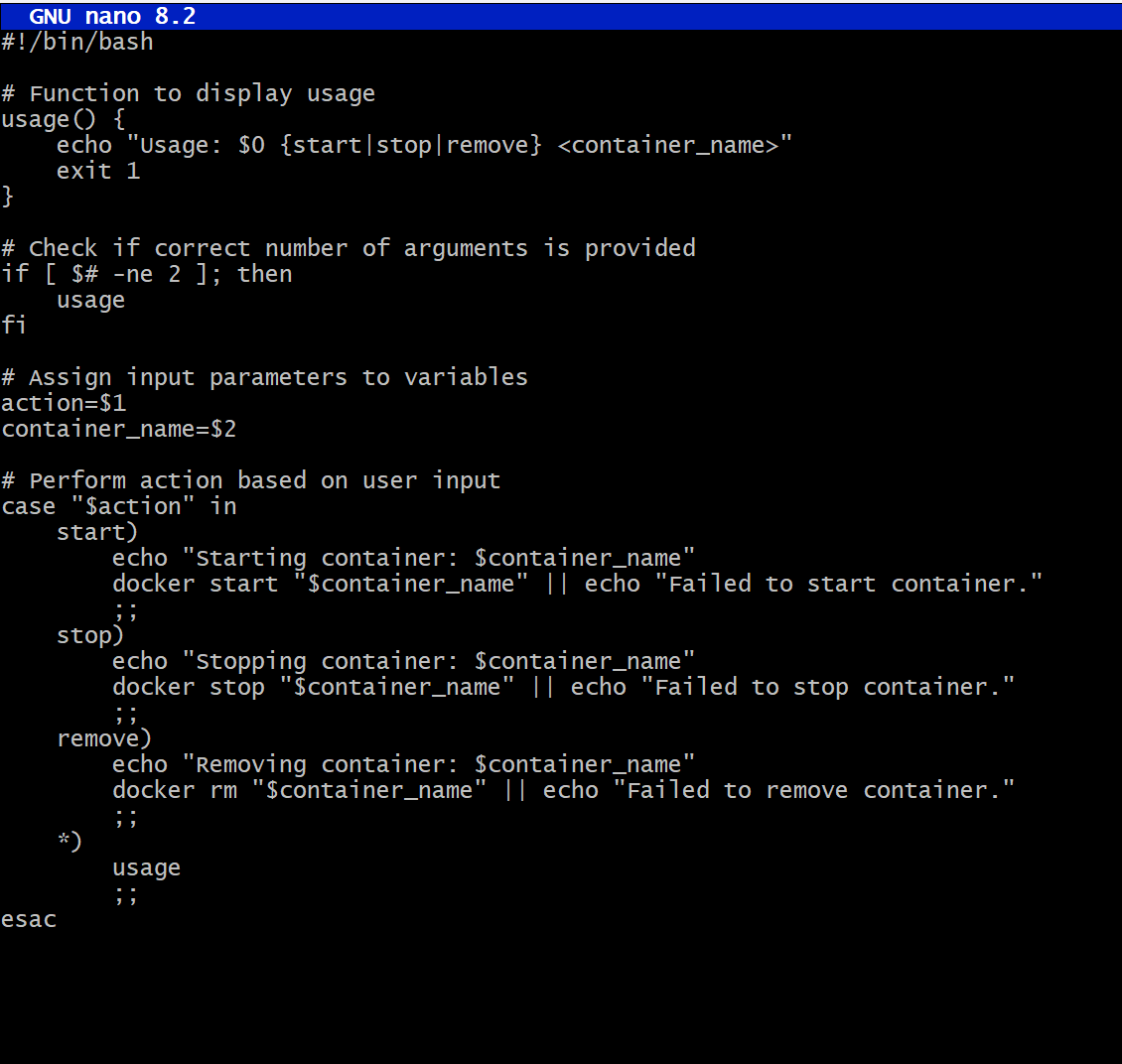
Next, open the file in a text editor.

**nano docker\_manager.sh**

****

## Step 3:

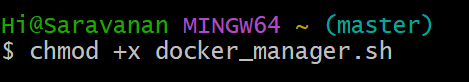
Add the Script Code into the docker\_manager.sh file and then press **Ctrl + O**, then **Enter**, and **Ctrl + X** to save.



## Step 4:

Before we can run the script, we need to make it executable.

**chmod +x docker\_manager.sh**

****

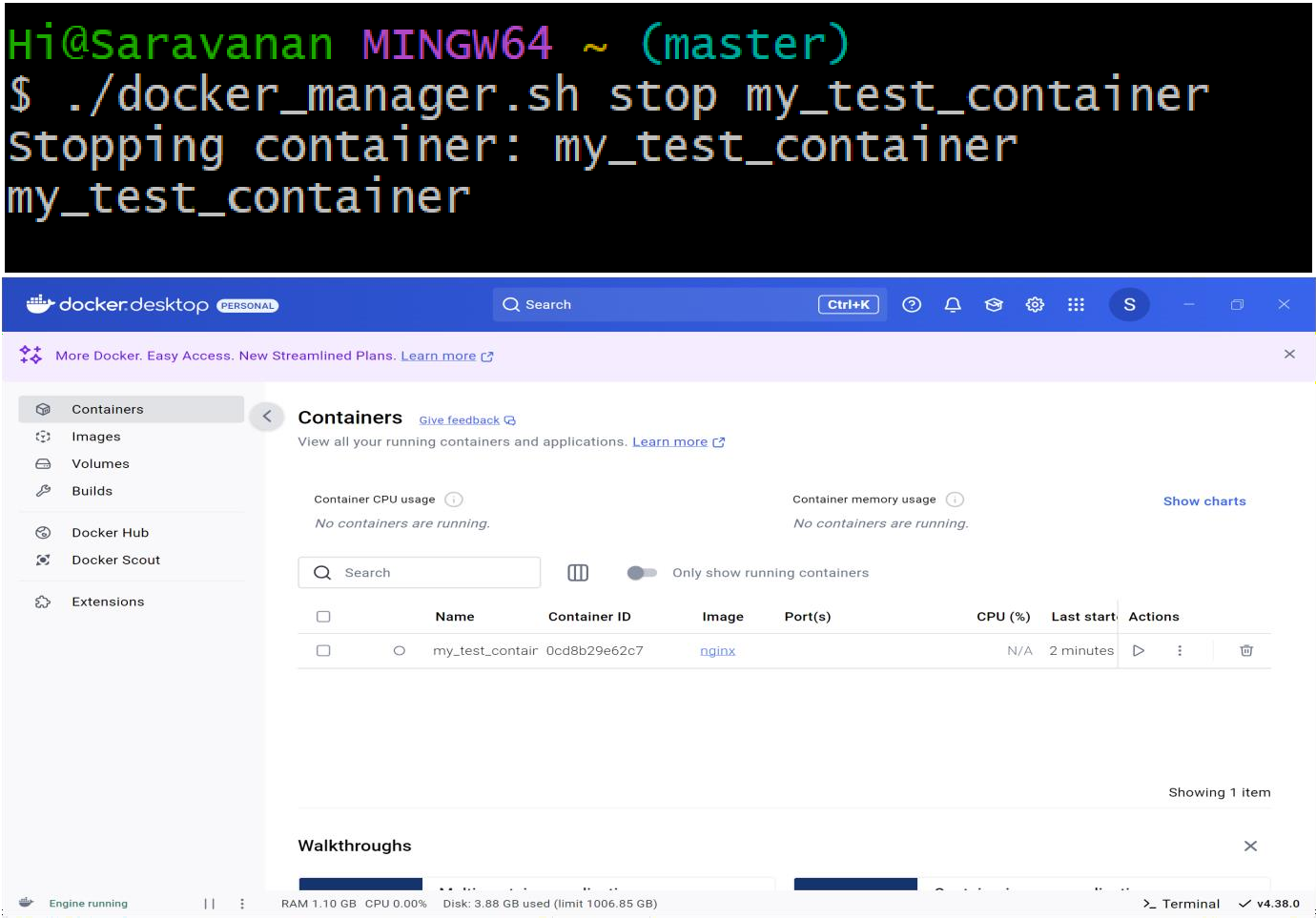
## Step 5:

Create a Test Container and view if it is running by: **docker run -d --name my\_test\_container nginx docker ps -a**

## Step 6:

Test the Script with the Correct Command Format To stop the container:

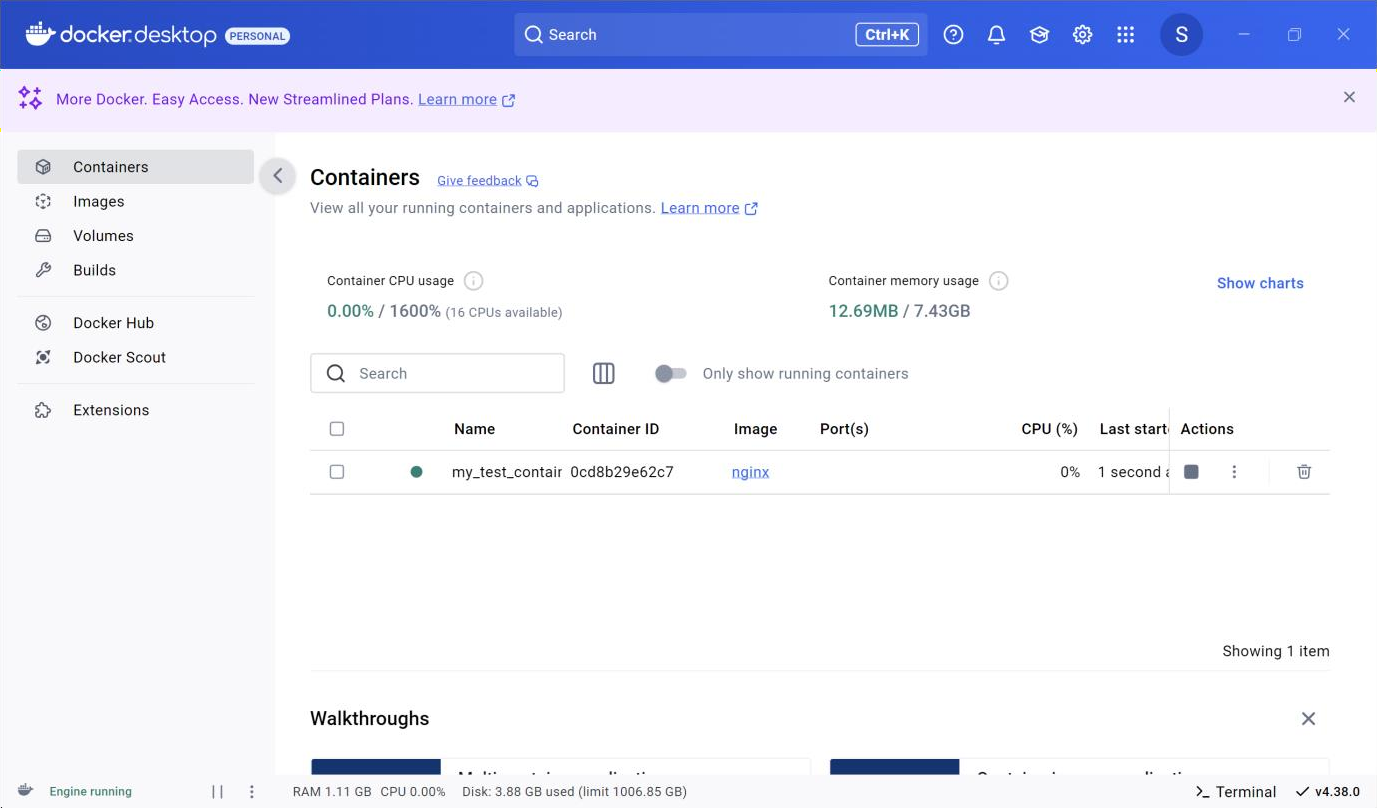
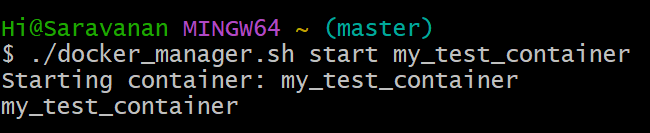
**./docker\_manager.sh stop my\_test\_container**

****

## Step 7:

To start the container:

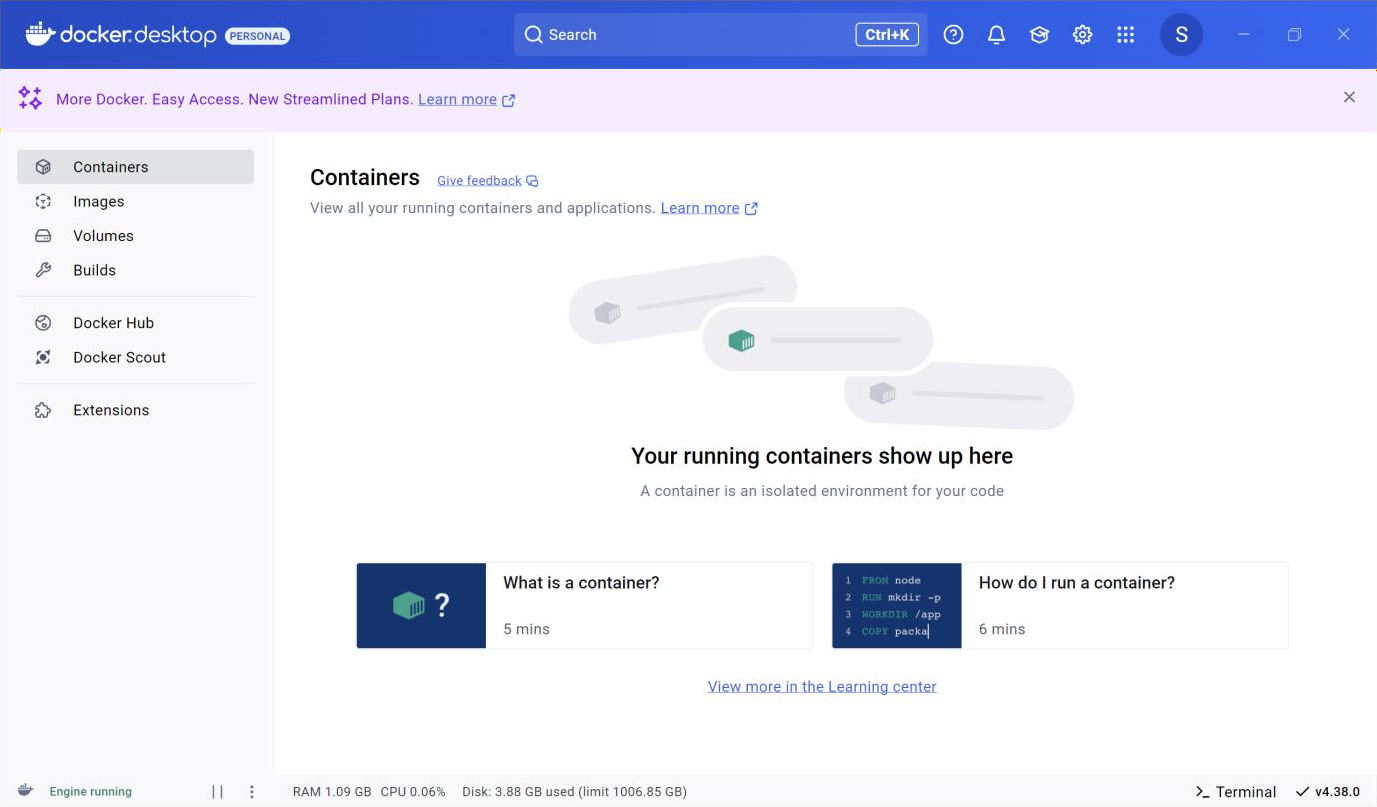
**./docker\_manager.sh start my\_test\_container**

****

## Step 8:

To remove the container (after stopping it):

**./docker\_manager.sh remove my\_test\_container**

****

Before wrapping up, ensure:

1. Your script correctly starts, stops, and removes the container.
2. Running docker ps -a reflects the expected status after each operation.

# Outcomes

By completing this PoC, you will:

1. **Automate Docker Container Management** – Develop a script to start, stop, and remove containers with a single command, reducing manual effort.
2. **Enhance Shell Scripting Skills** – Gain hands-on experience in writing and executing Bash scripts for automating Docker workflows.
3. **Improve Docker Command Proficiency** – Master essential commands like docker start, docker stop, and docker rm for efficient container lifecycle management.
4. **Simplify Deployment Processes** – Learn how scripting can streamline container operations, making it easier to manage applications in a real-world environment.
5. **Understand the Importance of Infrastructure as Code (IaC)** – Explore how automation using scripts enhances efficiency, reduces errors, and supports scalable containerized environments.