Aarna Bafna
Devops Lab
T11
07

Lab Assignment 7

AIM: To understand Docker architecture and container life cycle, install docker , deploy container in docker.

LAB OUTCOME:
LO1, LO5 Mapped.

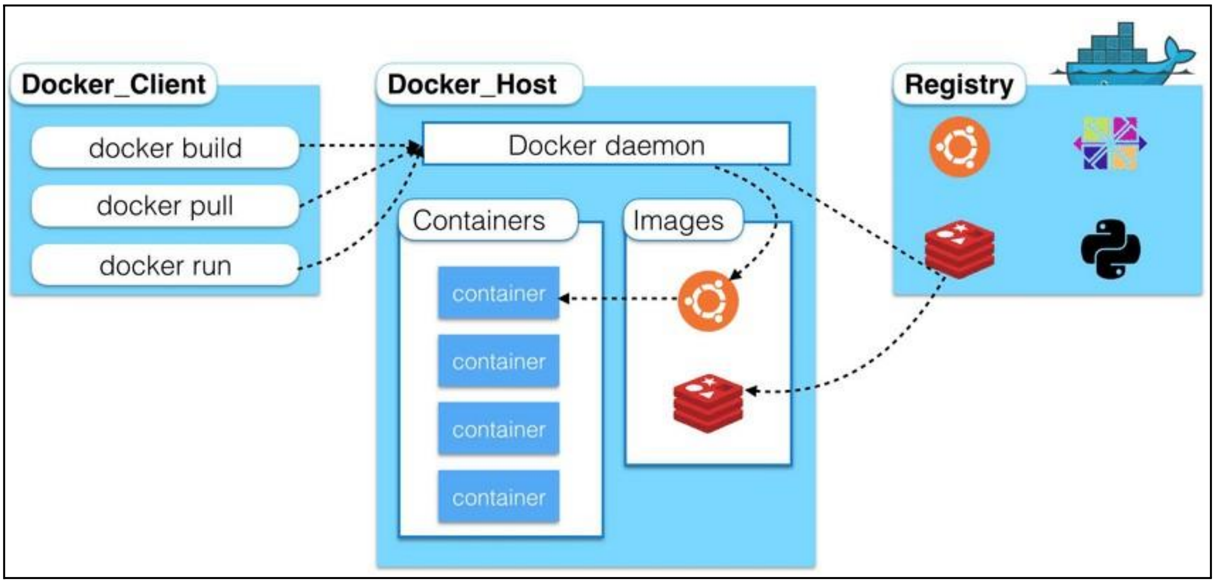
THEORY:
Docker is a technology that allows you to package and run applications and their dependencies in a consistent and isolated environment called a container. Think of it like a shipping container for your software – it contains everything your application needs to run, such as code, libraries, and settings, all bundled together.

Docker Architecture:
1. Docker Engine: This is like the core of Docker. It's a program that runs on your computer or server and manages containers. It consists of the Docker daemon (a background service) and the Docker command-line interface (CLI).

2. Images: Containers start from images. An image is like a blueprint or template for a container. It includes all the files and instructions needed to create a container. Images can be shared and used to create multiple containers.

3. Containers: These are the instances of images. When you run an image, it becomes a container. Containers are isolated environments that contain your application and its dependencies, making sure it runs consistently across different systems.

4. Registry: A registry is like a library of Docker images. Docker Hub is a popular public registry, but you can also set up private registries. You can push (upload) and pull (download) images to/from registries.



Life Cycle of a Container:

1. Create: You start by creating a container from an image using the `docker run` command. This creates an isolated instance of your application.

2. Run: Once created, you can start the container with `docker start`. Your application runs within the container as if it's on its own little computer.

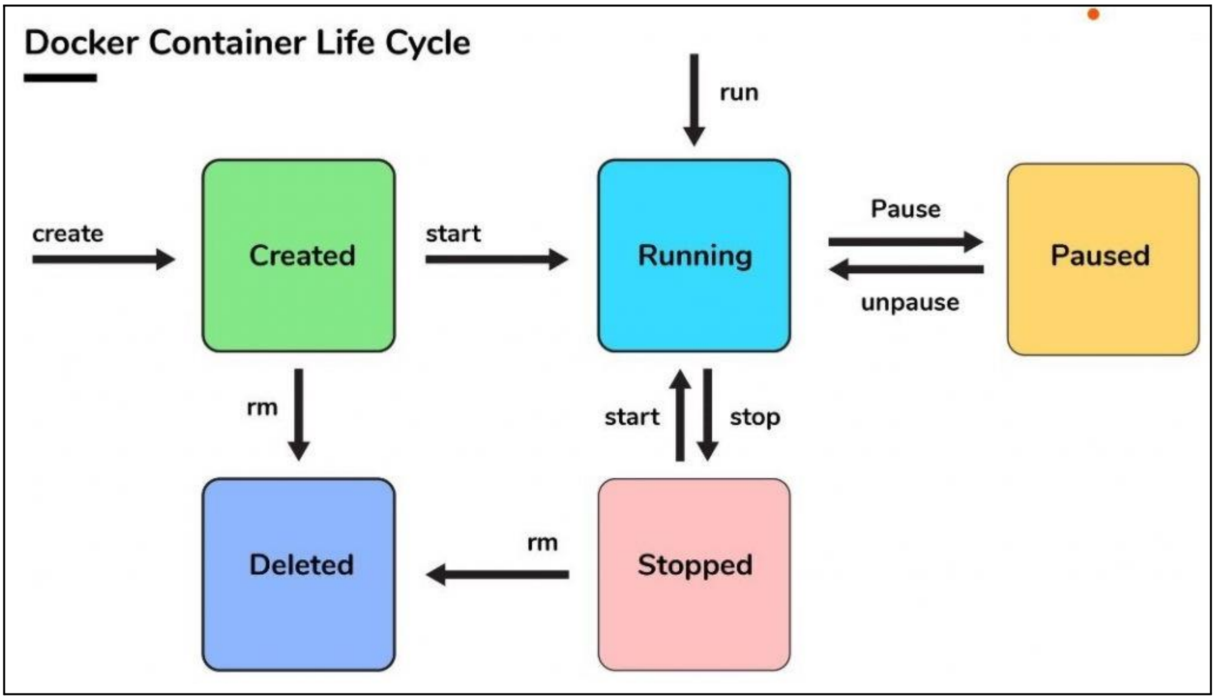
3. Pause and Resume: You can pause a running container with `docker pause` and then resume it with `docker unpause`. This can be handy for saving resources when a container isn't actively in use.

4. Stop: When you're done with a container, you can stop it with `docker stop`. This gracefully shuts down your application.

5. Start: You can later start the container again with `docker start`, and it will resume from where it left off.

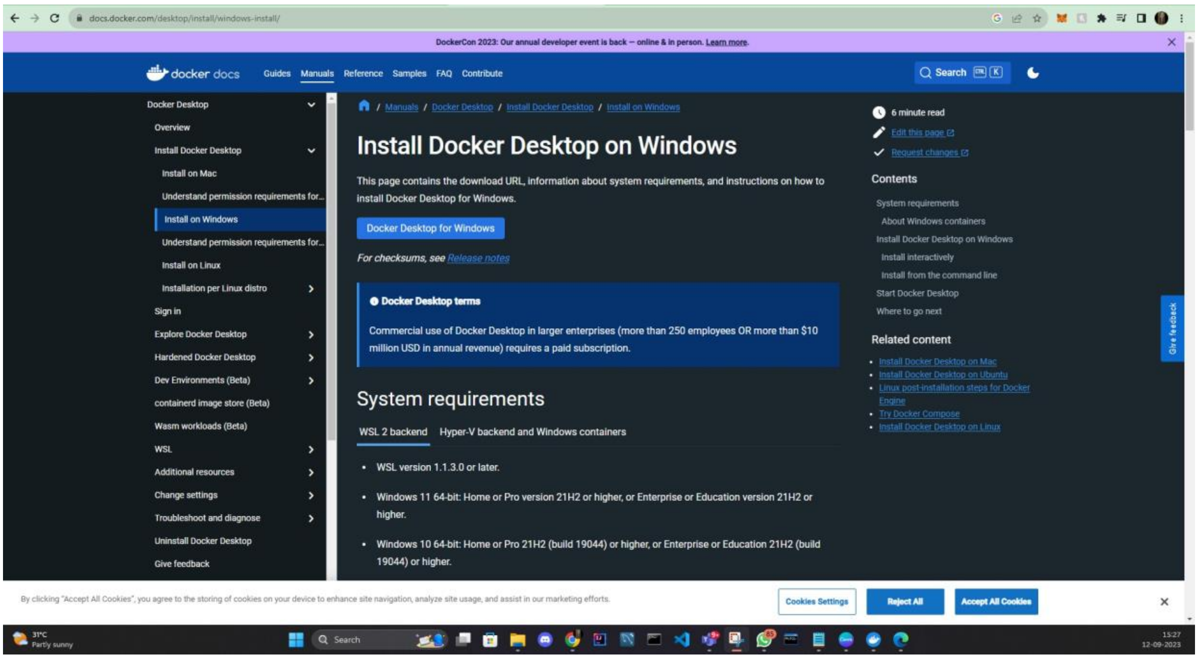
6. Remove: If you no longer need a container, you can remove it with `docker rm`. This deletes the container, but not the image it was created from.

7. Cleanup: You can also clean up unused images with `docker image prune` to free up storage space.



Installation steps of docker with screenshot.

1. Download Docker Desktop for Windows:
- Visit https://www.docker.com/products/docker-desktop and download the installer.



Install Docker Desktop on Windows

This page contains the download URL, information about system requirements, and instructions on how to install Docker Desktop for Windows.

Docker Desktop for Windows

For checksums, see Release notes

Docker Desktop terms

Commercial use of Docker Desktop in larger enterprises (more than 250 employees OR more than $10 million USD in annual revenue) requires a paid subscription.

System requirements

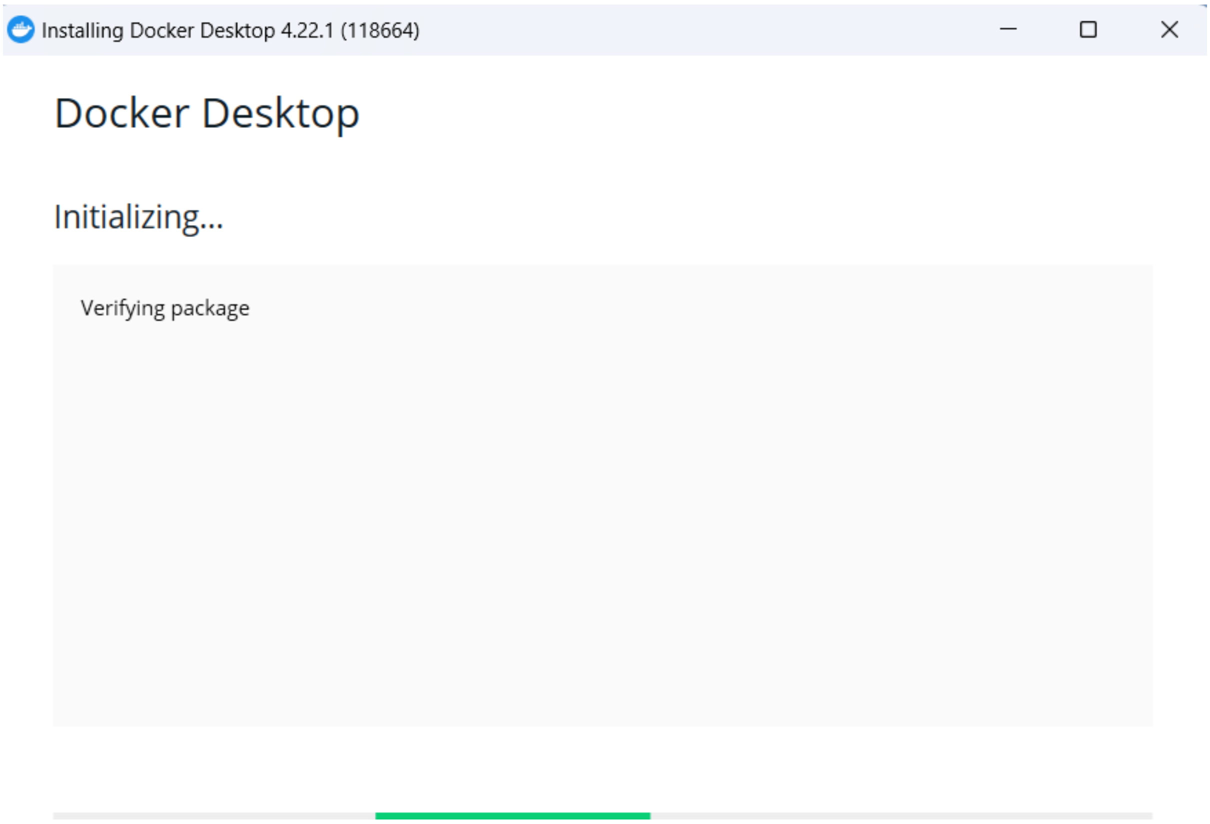
WSL 2 backend | Hyper-V backend and Windows containers

- WSL version 1.1.3.0 or later.
- Windows 11 64-bit: Home or Pro version 21H2 or higher, or Enterprise or Education version 21H2 or higher.
- Windows 10 64-bit: Home or Pro 21H2 (build 19044) or higher, or Enterprise or Education 21H2 (build 19044) or higher.

By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.

2. Run the Installer:

- Double-click the installer file to begin installation.



Installing Docker Desktop 4.22.1 (118664)

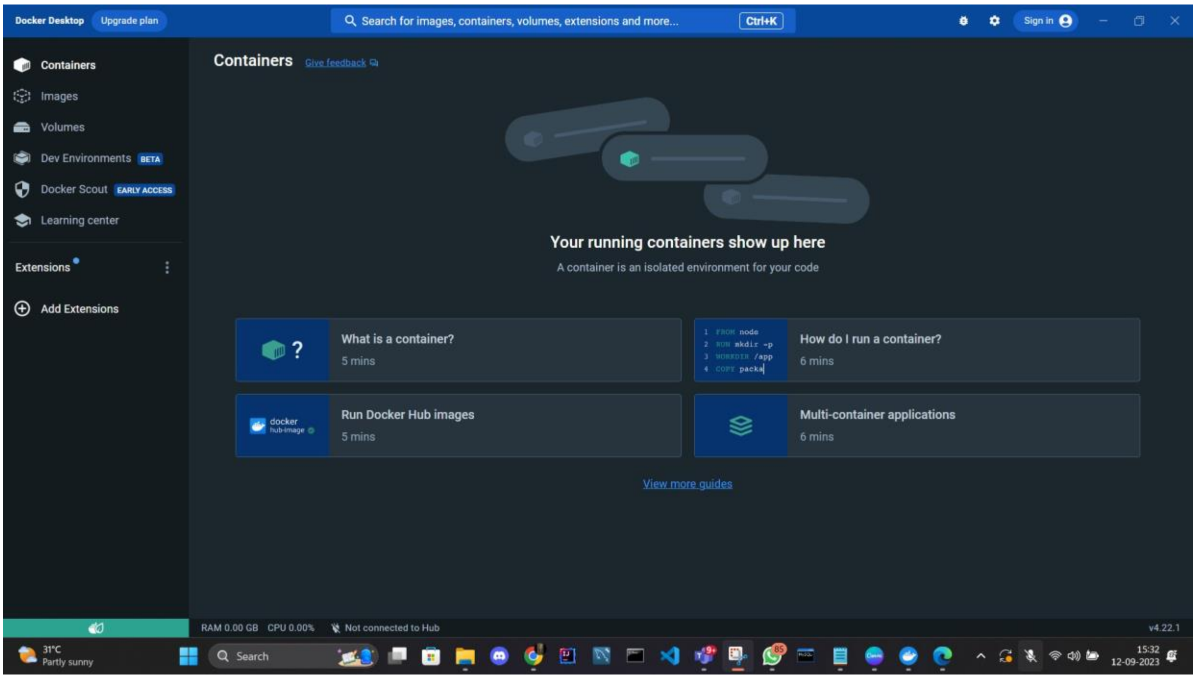
Docker Desktop

Initializing...

Verifying package

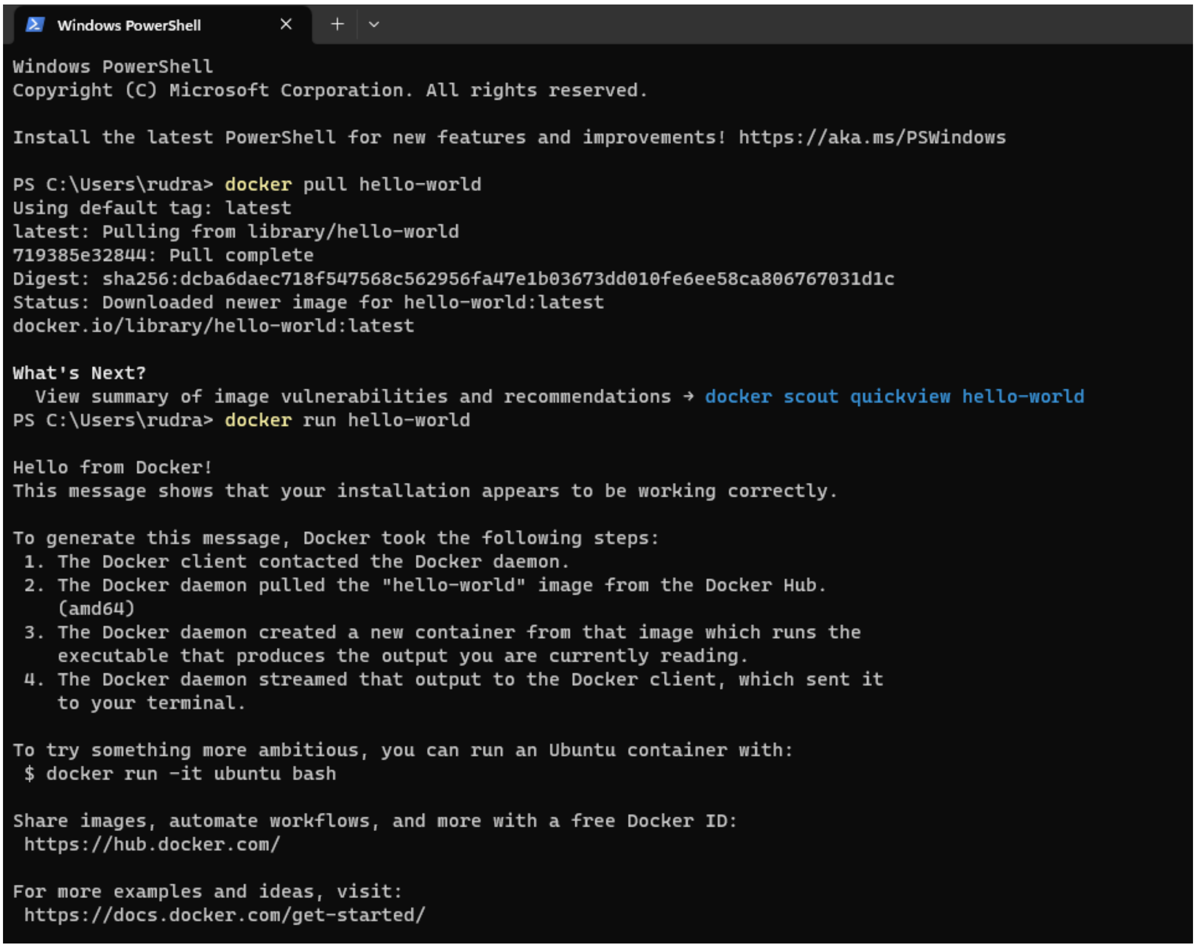
3. Configuration Options:

- After installation, access Docker settings by right-clicking the Docker icon on the desktop. Now, you're ready to configure Docker Desktop for Windows.

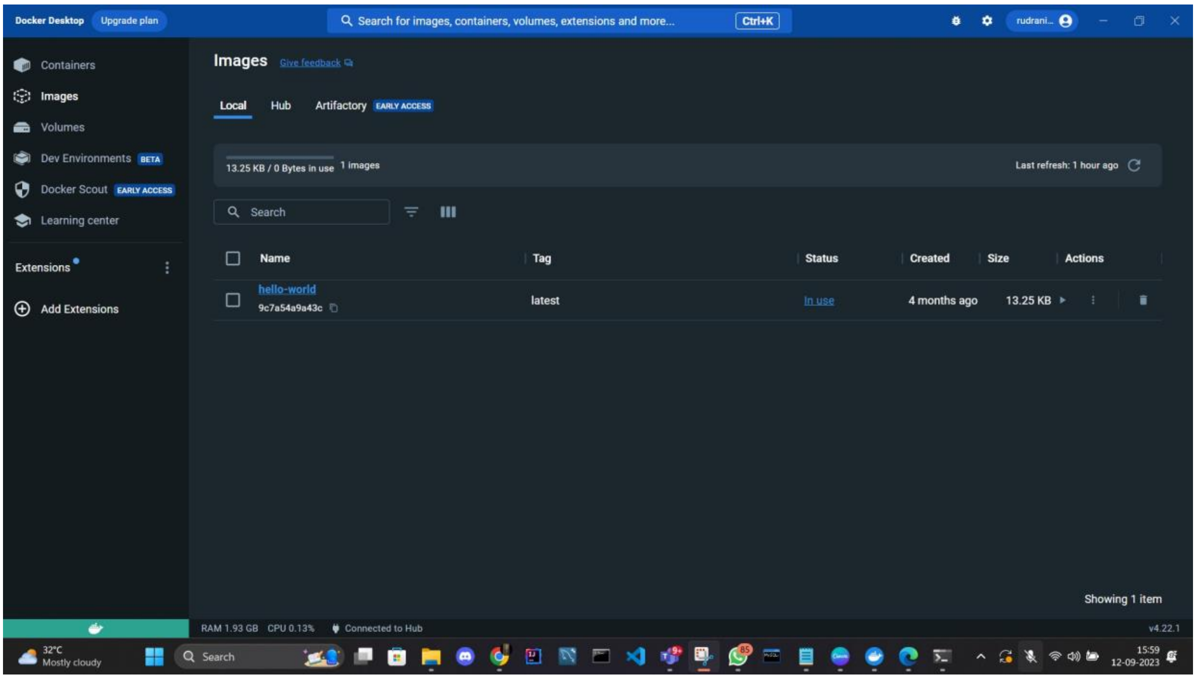


Your running containers show up here

A container is an isolated environment for your code



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\rudra> docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:dcba6daec718f547568c562956fa47e1b03673dd010fe6ee58ca806767031d1c
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest

What's Next?
  View summary of image vulnerabilities and recommendations → docker scout quickview hello-world
PS C:\Users\rudra> docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```



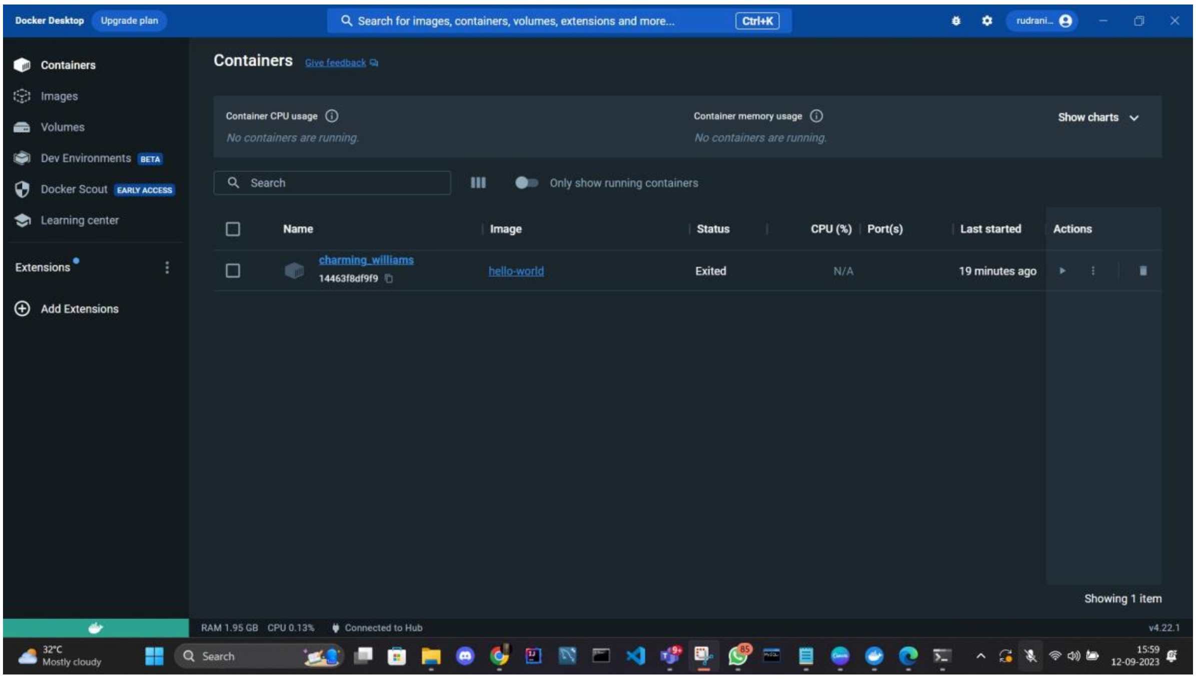
13.25 KB / 0 Bytes in use 1 images

Last refresh: 1 hour ago

☐	Name	Tag	Status	Created	Size	Actions
☐	hello-world 9c7a54a9a43c	latest	In use	4 months ago	13.25 KB	

Showing 1 item

RAM 1.93 GB CPU 0.13% Connected to Hub v4.22.1



Container CPU usage: No containers are running.

Container memory usage: No containers are running.

☐	Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
☐	charming_williams 14463f8df9f9	hello-world	Exited	N/A		19 minutes ago	

Showing 1 item

RAM 1.95 GB CPU 0.13% Connected to Hub v4.22.1

CONCLUSION:

Docker simplifies the process of developing, testing, and deploying applications because it ensures that what works on your development machine will also work in other environments, like a production server, without the "it works on my machine" problem. It's especially valuable in modern software development and deployment workflows, where consistency and scalability are essential.