

**Experiment No : 08**

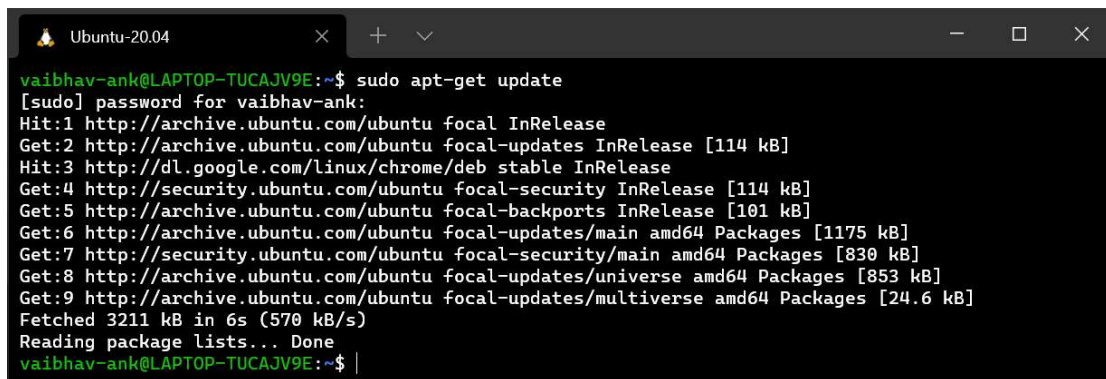
**Aim :** To understand Docker Architecture and Container Life Cycle, install Docker and execute docker commands to manage images and interact with containers.

**Theory :**

### **Option 1: Installing Docker from Official Repository :**

#### **Step 1. Update the apt package index:**

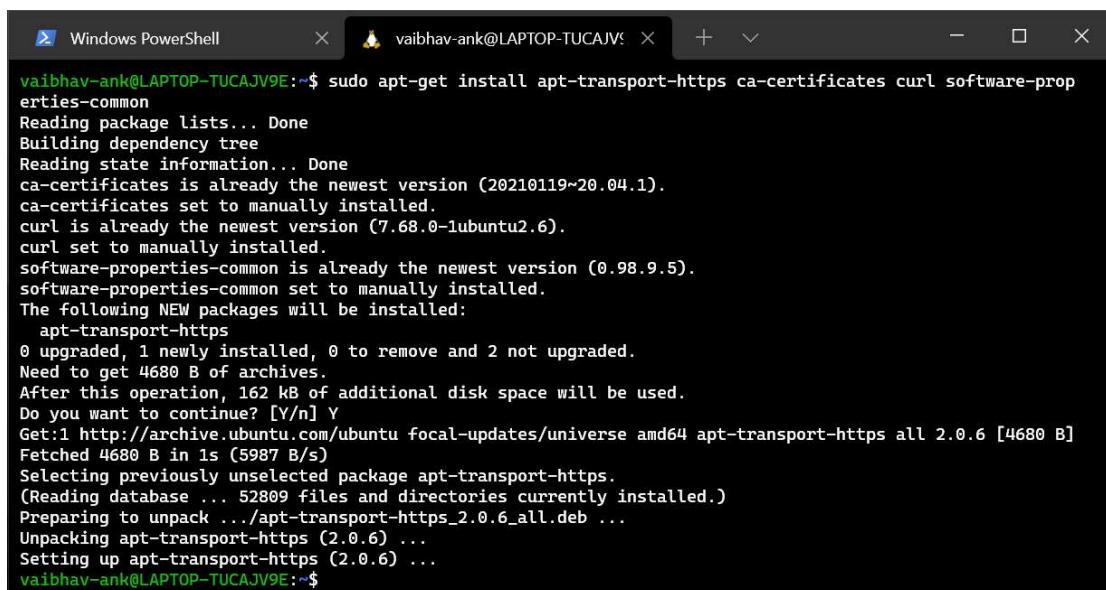
```
$ sudo apt-get update
```



```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo apt-get update
[sudo] password for vaibhav-ank:
Hit:1 http://archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Hit:3 http://dl.google.com/linux/chrome/deb stable InRelease
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-backports InRelease [101 kB]
Get:6 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1175 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [830 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [853 kB]
Get:9 http://archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 Packages [24.6 kB]
Fetched 3211 kB in 6s (570 kB/s)
Reading package lists... Done
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

#### **Step 2. Install packages to allow apt to use a repository over HTTPS:**

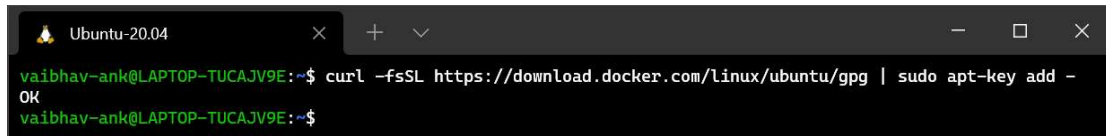
```
$ sudo apt-get install apt-transport-https ca-certificates curl
software-properties-common
```



```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo apt-get install apt-transport-https ca-certificates curl software-prop
erties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
ca-certificates is already the newest version (20210119~20.04.1).
ca-certificates set to manually installed.
curl is already the newest version (7.68.0-1ubuntu2.6).
curl set to manually installed.
software-properties-common is already the newest version (0.98.9.5).
software-properties-common set to manually installed.
The following NEW packages will be installed:
  apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 2 not upgraded.
Need to get 4680 B of archives.
After this operation, 162 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 apt-transport-https all 2.0.6 [4680 B]
Fetched 4680 B in 1s (5987 B/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 52809 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.0.6_all.deb ...
Unpacking apt-transport-https (2.0.6) ...
Setting up apt-transport-https (2.0.6) ...
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

**Step 3. Add Docker's official GPG key:**

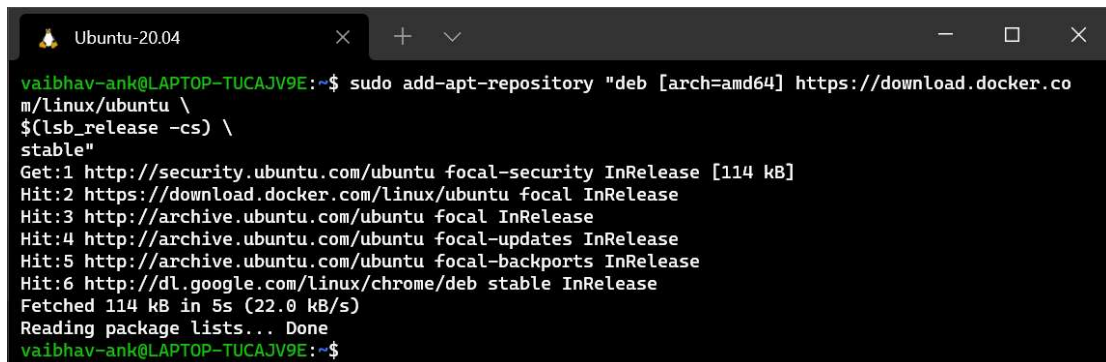
```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```



```
Ubuntu-20.04
vaibhav-ank@LAPTOP-TUCAJV9E:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
OK
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

**Step 4. Set up the stable repository (command depends on system architecture):**

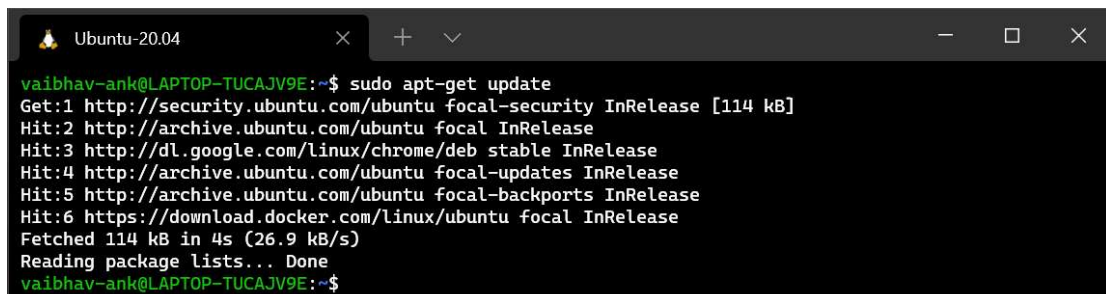
```
$ sudo add-apt-repository \
> "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
> $(lsb_release -cs) \
> stable"
```



```
Ubuntu-20.04
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.co
m/linux/ubuntu \
$(lsb_release -cs) \
stable"
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Hit:2 https://download.docker.com/linux/ubuntu focal InRelease
Hit:3 http://archive.ubuntu.com/ubuntu focal InRelease
Hit:4 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:5 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:6 http://dl.google.com/linux/chrome/deb stable InRelease
Fetched 114 kB in 5s (22.0 kB/s)
Reading package lists... Done
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

**Step 5. Update the package index again:**

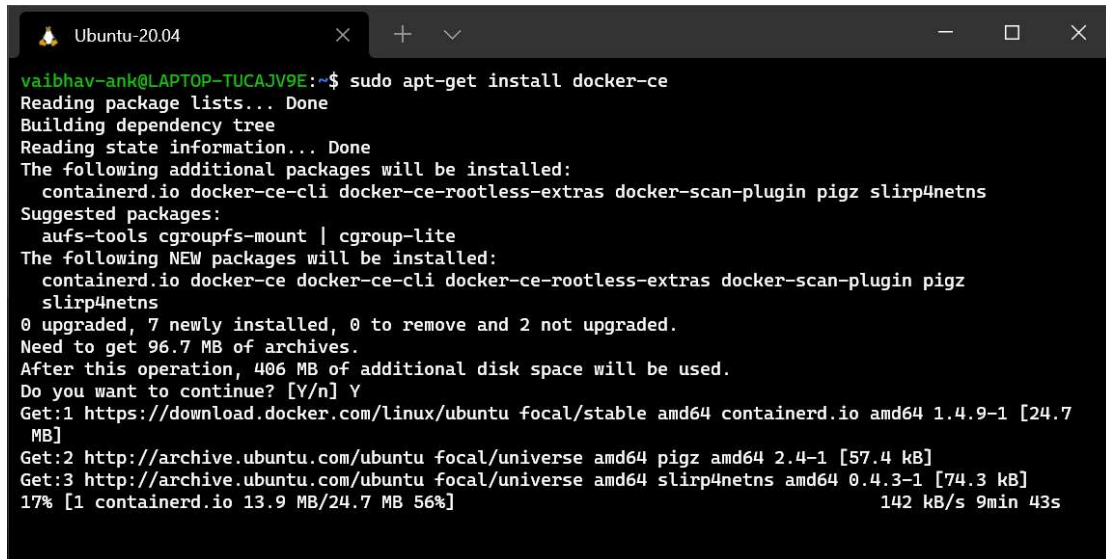
```
$ sudo apt-get update
```



```
Ubuntu-20.04
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo apt-get update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Hit:2 http://archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://dl.google.com/linux/chrome/deb stable InRelease
Hit:4 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:5 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:6 https://download.docker.com/linux/ubuntu focal InRelease
Fetched 114 kB in 4s (26.9 kB/s)
Reading package lists... Done
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

**Step 6. Install the docker-ce package (the service should start automatically after installation):**

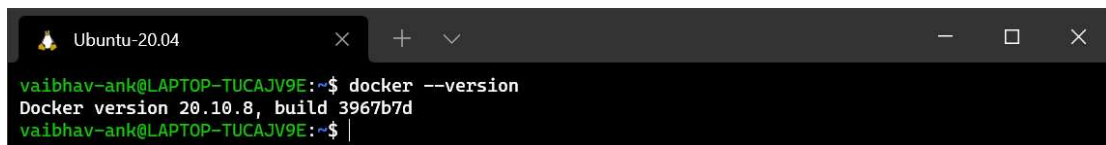
```
$ sudo apt-get install docker-ce
```

A terminal window titled 'Ubuntu-20.04' showing the command 'sudo apt-get install docker-ce' being executed. The output shows the package lists being read, the dependency tree being built, and the state information being read. It lists additional packages to be installed: containerd.io, docker-ce-cli, docker-ce-rootless-extras, docker-scan-plugin, pigz, and slirp4netns. It also suggests packages aufs-tools, cgroupfs-mount, and cgroup-lite. The output shows that 0 packages are upgraded, 7 are newly installed, and 0 are to be removed. The total size of the archives is 96.7 MB, and 406 MB of additional disk space will be used. The user is prompted to continue, and they press 'Y'. The terminal shows the progress of downloading the packages from the Docker repository and the Ubuntu archive, with a progress bar at the bottom indicating 17% completion for containerd.io.

```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo apt-get install docker-ce
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-ce-cli docker-ce-rootless-extras docker-scan-plugin pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-ce docker-ce-cli docker-ce-rootless-extras docker-scan-plugin pigz
  slirp4netns
0 upgraded, 7 newly installed, 0 to remove and 2 not upgraded.
Need to get 96.7 MB of archives.
After this operation, 406 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 https://download.docker.com/linux/ubuntu focal/stable amd64 containerd.io amd64 1.4.9-1 [24.7 MB]
Get:2 http://archive.ubuntu.com/ubuntu focal/universe amd64 pigz amd64 2.4-1 [57.4 kB]
Get:3 http://archive.ubuntu.com/ubuntu focal/universe amd64 slirp4netns amd64 0.4.3-1 [74.3 kB]
17% [1 containerd.io 13.9 MB/24.7 MB 56%] 142 kB/s 9min 43s
```

**Step 7. To confirm the installation, check the version of Docker:**

```
$ docker --version
```

A terminal window titled 'Ubuntu-20.04' showing the command 'docker --version' being executed. The output is 'Docker version 20.10.8, build 3967b7d'.

```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ docker --version
Docker version 20.10.8, build 3967b7d
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

**Option 2: Installing Docker from Default Repositories :**

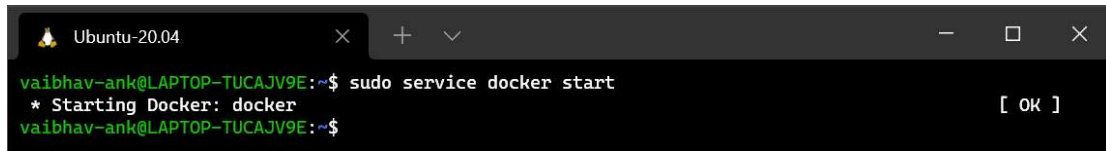
To install docker run:

```
$ sudo apt install docker.io
```

## Running Docker Commands with sudo :

**Step 1. To start the Docker service run the following commands:**

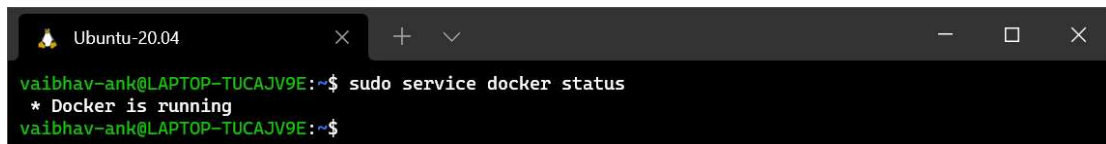
```
$ sudo systemctl start docker OR $ sudo service docker start
```

A terminal window titled 'Ubuntu-20.04' showing the command 'sudo service docker start' being executed. The output is '\* Starting Docker: docker' followed by a '[ OK ]' status message in a green box.

```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo service docker start
* Starting Docker: docker
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

**To check the status of the service, use the command:**

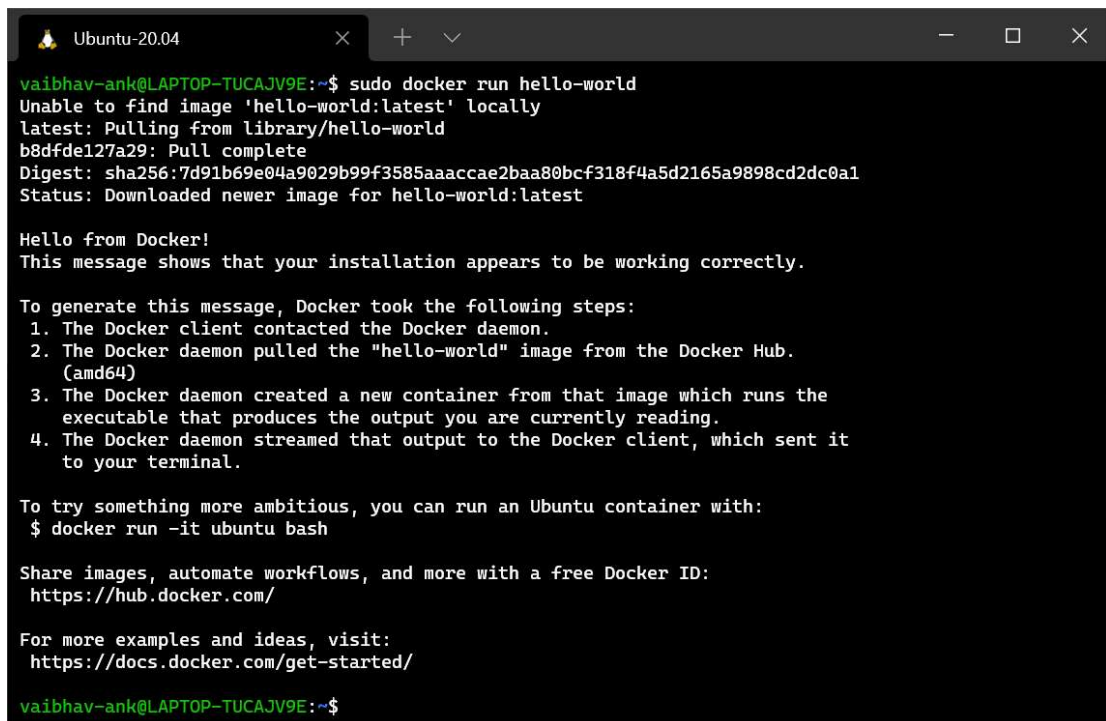
```
$ sudo systemctl status docker OR $ sudo service docker status
```

A terminal window titled 'Ubuntu-20.04' showing the command 'sudo service docker status' being executed. The output is '\* Docker is running'.

```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo service docker status
* Docker is running
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

**Step 2. Verify that docker installed properly:**

```
$ sudo docker run hello-world
```

A terminal window titled 'Ubuntu-20.04' showing the command 'sudo docker run hello-world' being executed. The output shows the Docker client pulling the 'hello-world:latest' image from the Docker Hub, creating a new container, and running the executable. It then displays a 'Hello from Docker!' message and a list of steps taken by Docker to generate the message. Finally, it provides a link to Docker Hub and a link to Docker documentation.

```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
b8dfde127a29: Pull complete
Digest: sha256:7d91b69e04a9029b99f3585aaaccae2baa80bcf318f4a5d2165a9898cd2dc0a1
Status: Downloaded newer image for hello-world:latest

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

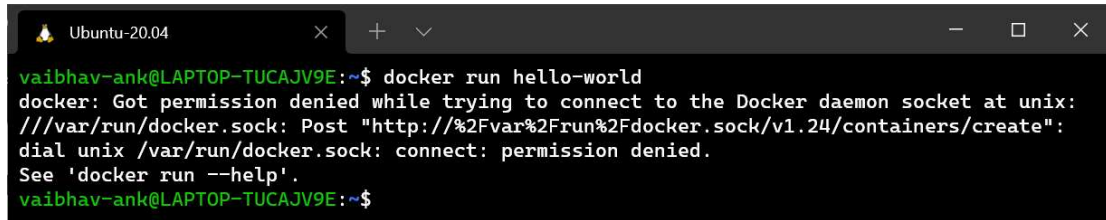
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

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

vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

## Running Docker Commands without sudo :

By default, you can only use the docker commands with root privileges. Ubuntu requires the use of the sudo prefix. For example, if you try to run a hello-world container, the output displays permission was denied.



```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ docker run hello-world
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:
///var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/containers/create":
dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

It is advisable to keep the settings as is. However, you can bypass typing sudo every time. Adding the user to the docker group grants privileges equivalent to root.

**Step 1. First, create the docker group with the command:**

```
$ sudo groupadd docker
```

**Step 2. Then, type the following command (making sure to replace [user] with your username):**

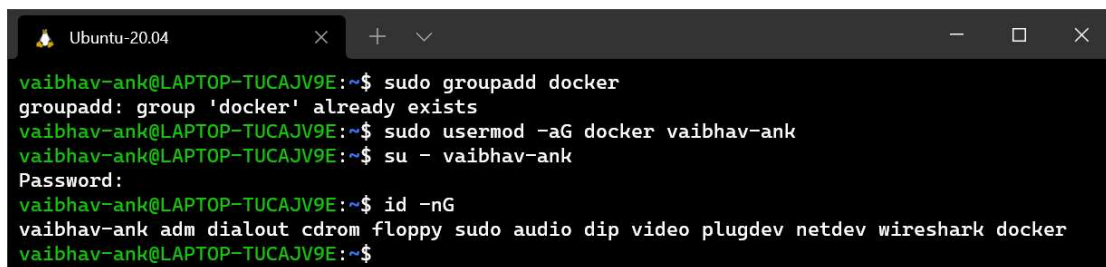
```
$ sudo usermod -aG docker [user]
```

**Step 3. Enable the new settings with:**

```
$ su - [user]
```

**Step 4. Lastly, check to confirm the user is now a part of the docker group by running:**

```
$ id -nG
```



```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo groupadd docker
groupadd: group 'docker' already exists
vaibhav-ank@LAPTOP-TUCAJV9E:~$ sudo usermod -aG docker vaibhav-ank
vaibhav-ank@LAPTOP-TUCAJV9E:~$ su - vaibhav-ank
Password:
vaibhav-ank@LAPTOP-TUCAJV9E:~$ id -nG
vaibhav-ank adm dialout cdrom floppy sudo audio dip video plugdev netdev wireshark docker
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

**Step 5. Now you can run the docker run hello-world command without the sudo prefix**

A terminal window titled 'Ubuntu-20.04' showing the command 'docker run hello-world' being executed. The output is 'Hello from Docker! This message shows that your installation appears to be working correctly.'

```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ docker run hello-world

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

## Working With Docker Images

Docker images are files that contain the source code, libraries, dependencies, tools, and other files a container need. You can create Docker images with Docker files or use existing ones available on Docker Hub.

To download a new Docker image, use the command:

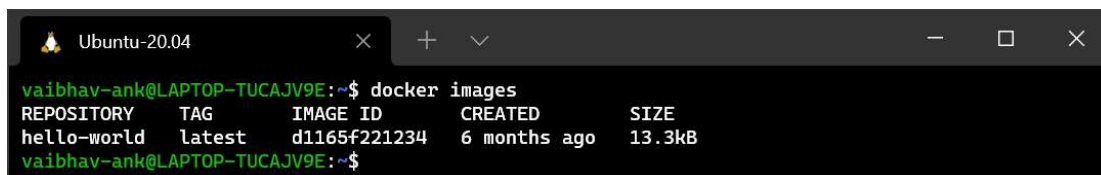
```
$ docker pull [image-name]
```

If you don't know the exact name of the image, search for it in Docker's repository with:

```
$ docker search ubuntu
```

After working with Docker for some time, you will collect a local registry of images. Display a list of all Docker images on the system with:

```
$ docker images
```

A terminal window titled 'Ubuntu-20.04' showing the command 'docker images' being executed. The output is a table listing the 'hello-world' image.

```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world    latest    d1165f221234   6 months ago   13.3kB
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

## Working With Docker Containers

Docker containers are isolated virtual environments that run based on the Docker image assigned to them.

To run a container based on an existing Docker image, use the command:

```
$ docker run [image_name]
```

Using the command above runs a container but doesn't move you inside of it. To run a container in interactive mode

and change to the container command prompt, run:

```
$ docker run -it [image_name]
```

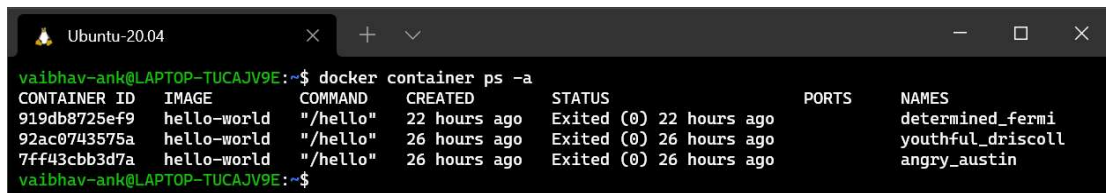
Note: Learn how to run a container in [How to Use Docker Run Command with Examples](#).

Another useful docker command is listing all the containers on the system. To list all active containers, type:

```
$ docker container ps
```

To view all containers (active and inactive), run:

```
$ docker container ps -a
```



```
vaibhav-ank@LAPTOP-TUCAJV9E:~$ docker container ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
919db8725ef9	hello-world	"/hello"	22 hours ago	Exited (0) 22 hours ago		determined_fermi
92ac0743575a	hello-world	"/hello"	26 hours ago	Exited (0) 26 hours ago		youthful_driscoll
7ff43cbb3d7a	hello-world	"/hello"	26 hours ago	Exited (0) 26 hours ago		angry_austin

```
vaibhav-ank@LAPTOP-TUCAJV9E:~$
```

## Working With Docker Volumes

The best way to preserve data generated within a container is mounting Docker volumes on to them. Mounted volumes don't depend on the container life cycle and can share data between containers.

Create a new Docker volume with:

```
$ docker volume create [volume_name]
```

To create a container and mount a volume to it, follow the syntax:

```
$ docker run [docker_image] \  
> --mount source=[volume_name],destination=[path_in_container]
```

## Conclusion :

Docker is essentially a toolkit that enables developers to build, deploy, run, update, and stop containers using simple commands and work-saving automation through a single API. Thus, we have successfully installed docker and executed docker commands to manage images and interact with containers.