

# IMPLEMENTATION OF DOCKER CONTAINERS

## Experiment Title

**Implementation of Different Docker Containers**

## Aim

To install Docker and create, execute, and analyze different types of Docker containers such as **Hello-World**, **Ubuntu OS**, **Web Server (Nginx)**, and **Programming Environment (Python)** in order to understand containerization concepts.

## Software Requirements

- **Operating System:** Kali Linux
- **Container Platform:** Docker Engine
- **Internet Connection**

## Docker

Docker is a **containerization platform** that allows applications to run in isolated environments called **containers**. Unlike Virtual Machines, Docker containers do **not require a full guest operating system**.

Containers share the host OS kernel, making them:

- Lightweight
- Fast to start, Resource efficient

# Procedure

## Step 1: Installation of Docker

Docker Engine was installed on Kali Linux using the following commands:

```
sudo apt update
sudo apt install docker.io -y
sudo systemctl start docker
sudo systemctl enable docker
```

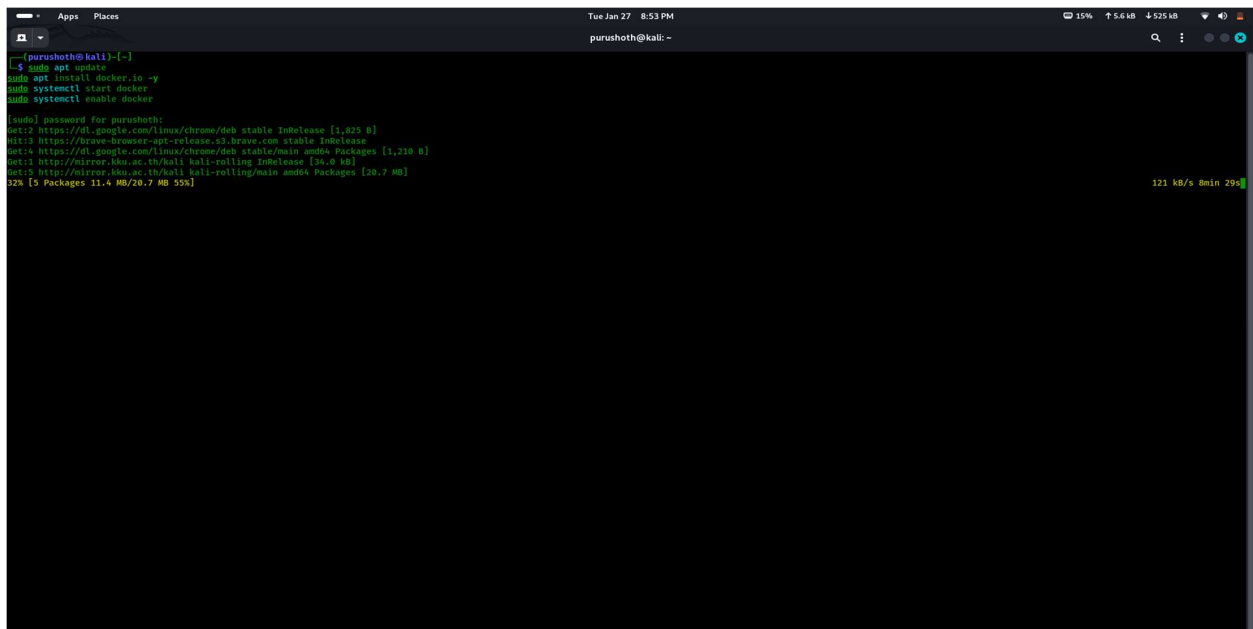
### Explanation:

- `apt update` → Updates package list
- `apt install docker.io` → Installs Docker Engine
- `systemctl start docker` → Starts Docker service
- `systemctl enable docker` → Enables Docker at boot

### Observation:

Docker service started successfully.

### Screenshot-1: Docker installation / Docker version output



```
purushoth@kali: ~  
$ sudo apt update  
Get:1 https://dl.google.com/linux/chrome/deb stable InRelease [1,825 B]  
Hit:2 https://brave-browser-apt-release.s3.brave.com stable InRelease  
Get:3 https://dl.google.com/linux/chrome/deb stable/main amd64 Packages [1,210 B]  
Get:4 https://mirror.kku.ac.th/kali kali-rolling InRelease [24.6 kB]  
Get:5 http://mirror.kku.ac.th/kali kali-rolling/main amd64 Packages [20.7 MB]  
32% [5 Packages 11.4 MB/20.7 MB 55%]  
121 kB/s 8min 29s
```

```
Apps Places Tue Jan 27 9:03 PM 23% 6.6 kB 4.5 kB
purushoth@kali: ~
Fetched 72.2 MB in 2min 7s (567 kB/s)
Selecting previously unselected package runc.
(Reading database ... 45989 files and directories currently installed.)
Preparing to unpack .../0-runc_1.3.3+ds1-2_amd64.deb ...
Unpacking runc (1.3.3+ds1-2) ...
Selecting previously unselected package containerd.
Preparing to unpack .../1-containerd_1.7.24-ds1-10_amd64.deb ...
Unpacking containerd (1.7.24-ds1-10) ...
Selecting previously unselected package tini-static.
Preparing to unpack .../2-tini-static_0.19.0-6_amd64.deb ...
Unpacking tini-static (0.19.0-6) ...
Selecting previously unselected package docker.io.
Preparing to unpack .../3-docker.io_27.5.1+dfsg4-1_amd64.deb ...
Unpacking docker.io (27.5.1+dfsg4-1) ...
Selecting previously unselected package libcgroup1:amd64.
Preparing to unpack .../4-libcgroup1_4.2-1_amd64.deb ...
Unpacking libcgroup1:amd64 (4.2-1) ...
Selecting previously unselected package criu.
Preparing to unpack .../5-criu_4.2-1_amd64.deb ...
Unpacking criu (4.2-1) ...
Selecting previously unselected package docker-buildx.
Preparing to unpack .../6-docker-buildx_0.19.3+ds1-4_amd64.deb ...
Unpacking docker-buildx (0.19.3+ds1-4) ...
Selecting previously unselected package docker-cli.
Preparing to unpack .../7-docker-cli_27.5.1+dfsg4-1_amd64.deb ...
Unpacking docker-cli (27.5.1+dfsg4-1) ...
Selecting previously unselected package python3-protobuf.
Preparing to unpack .../8-python3-protobuf_3.21.12-15_amd64.deb ...
Unpacking python3-protobuf (3.21.12-15) ...
Selecting previously unselected package python3-pycriu.
Preparing to unpack .../9-python3-pycriu_4.2-1_all.deb ...
Unpacking python3-pycriu (4.2-1) ...
Setting up docker-cli (27.5.1+dfsg4-1) ...
Setting up tini-static (0.19.0-6) ...
Setting up runc (1.3.3+ds1-2) ...
Setting up libcgroup1:amd64 (4.2-1) ...
Setting up criu (4.2-1) ...
Setting up python3-protobuf (3.21.12-15) ...
Setting up containerd (1.7.24-ds1-10) ...
containerd.service is a disabled or a static unit not running, not starting it.
Setting up docker-buildx (0.19.3+ds1-4) ...
Setting up python3-pycriu (4.2-1) ...
Setting up docker.io (27.5.1+dfsg4-1) ...
update-rc.d: We have no instructions for the docker init script.
update-rc.d: It looks like a non-network service, we enable it.
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service' → '/usr/lib/systemd/system/docker.service'.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket' → '/usr/lib/systemd/system/docker.socket'.
Processing triggers for kali-menu (2025.4.3) ...
Processing triggers for libe-bin (2.42-5) ...
Processing triggers for man-db (2.13.1-1) ...
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
purushoth@kali:~$
```

## Step 2: Verify Docker Installation

```
docker --version
```

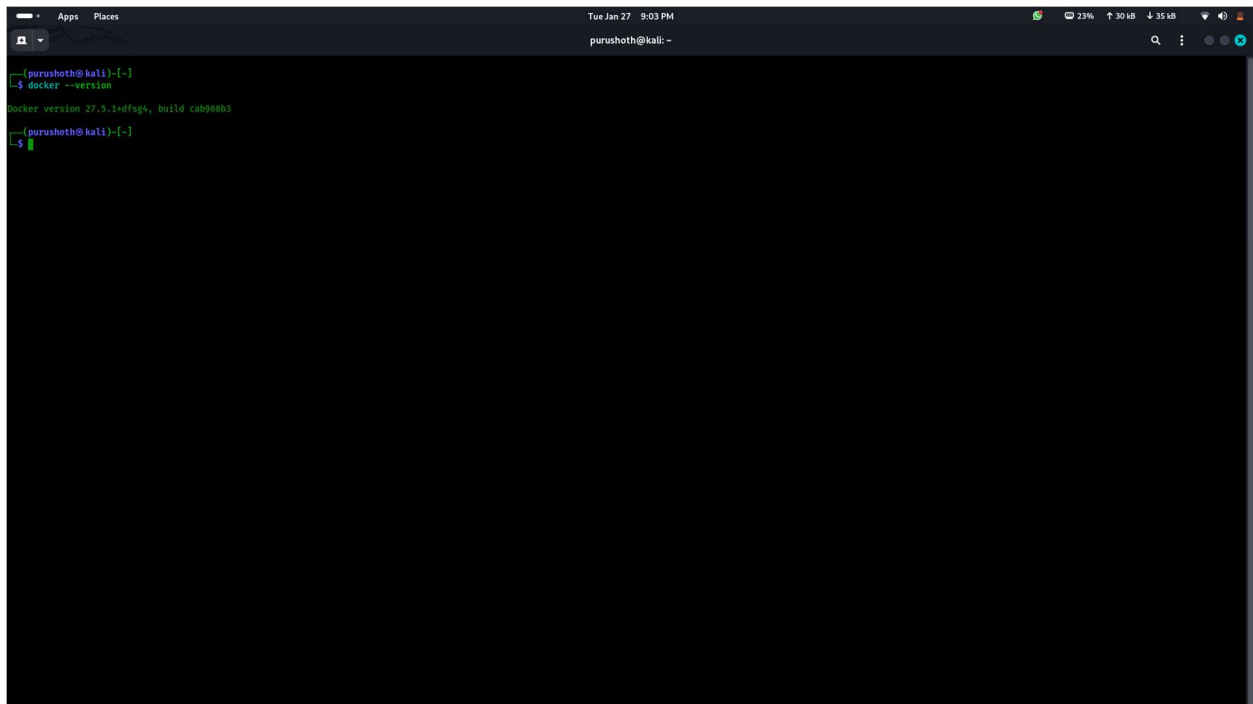
### Explanation:

This command confirms that Docker is installed correctly and displays the installed version.

### Observation:

Docker version is displayed without any error.

### Screenshot-2: Docker version confirmation



```
(purushoth@kali)~$ docker --version
Docker version 27.5.1-efags, build cab968b3
(purushoth@kali)~$
```

## Step 3: Create Ubuntu OS Container

```
docker run -it --name ubuntu_container ubuntu bash
```

### Explanation:

- `-it` → Interactive terminal
- `--name ubuntu_container` → Assigns a custom name
- `ubuntu` → Base OS image
- `bash` → Opens Ubuntu shell

### Commands executed inside the container:

```
ls
cat /etc/os-release
exit
```

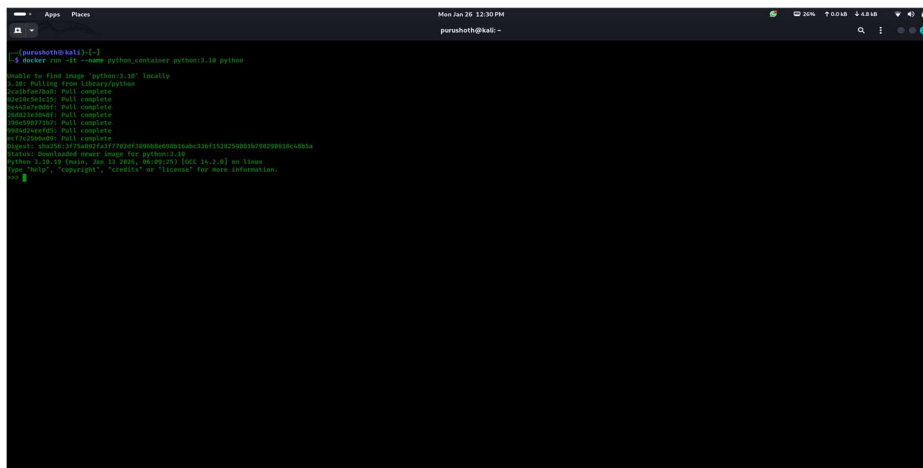
### Explanation of commands:

- `ls` → Lists directory contents
- `cat /etc/os-release` → Displays Ubuntu OS details
- `exit` → Exits container

### Observation:

Ubuntu shell opened successfully, and OS details were displayed.

### Screenshot-4: Ubuntu container terminal



```
purushothkali: ~  
└─$ docker run -it --name python_container python:3.10 python  
Unable to find image 'python:3.10' locally  
3.10 Pulling from library/python  
22328f4e7b4d: Pull complete  
043603a6125f: Pull complete  
5ee522760d8f: Pull complete  
00823a3a0a0f: Pull complete  
588c508772d7: Pull complete  
009403a40002: Pull complete  
e7f7c250a400: Pull complete  
Digest: sha256:1f7a607c9f78a9f89a4e081a4a3216f152259081679d7980161c4b1a  
Status: Downloaded more image for python:3.10  
python 3.10.10 (main, Jun 15 2024, 06:09:29) [GCC 14.2.0] on linux  
Type "help", "copyright", "credits" or "license()" for more information.  
>>> ls  
.  
..  
bin  
boot  
dev  
etc  
home  
lib  
lib64  
media  
mnt  
opt  
root  
run  
sbin  
srv  
tmp  
usr  
var  
>>> cat /etc/os-release  
PRETTY_NAME="Ubuntu 24.04 LTS"  
NAME="Ubuntu"  
VERSION="24.04 LTS (Noble Numbur)"  
VERSION_ID="24.04"  
BUILD_ID="24.04"  
UBUNTU_CODENAME=noble  
>>> exit  
└─$
```

```
Mon Jan 26 12:03 PM
purushoth@kali: ~
[~] purushoth@kali)~[~]
$ docker run -it --name ubuntu_container ubuntu bash
Unable to find image 'ubuntu:latest' locally

latest: Pulling from library/ubuntu
a3629ac5b9f4: Pull complete
Digest: sha256:cd1dbae51b3088c3680ecf4e3c4220f0260521fb76978881737d24f200828b2b
Status: Downloaded newer image for ubuntu:latest

root@3507d445776e:/#
root@3507d445776e:/#
root@3507d445776e:/# ls
cat /etc/os-release
exit
bin boot dev etc home lib lib64 media mnt opt proc root run shin srv sys usr var
PRETTY_NAME="Ubuntu 24.04.3 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.3 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logs
exit

[~] purushoth@kali)~[~]
```

## Step 4: Create Web Server Container (Nginx)

```
docker run -d --name nginx_container -p 8080:80 nginx
```

### Explanation:

- `-d` → Runs container in detached mode
- `--name nginx_container` → Container name
- `-p 8080:80` → Maps host port 8080 to container port 80
- `nginx` → Web server image

### Accessing the Web Server:

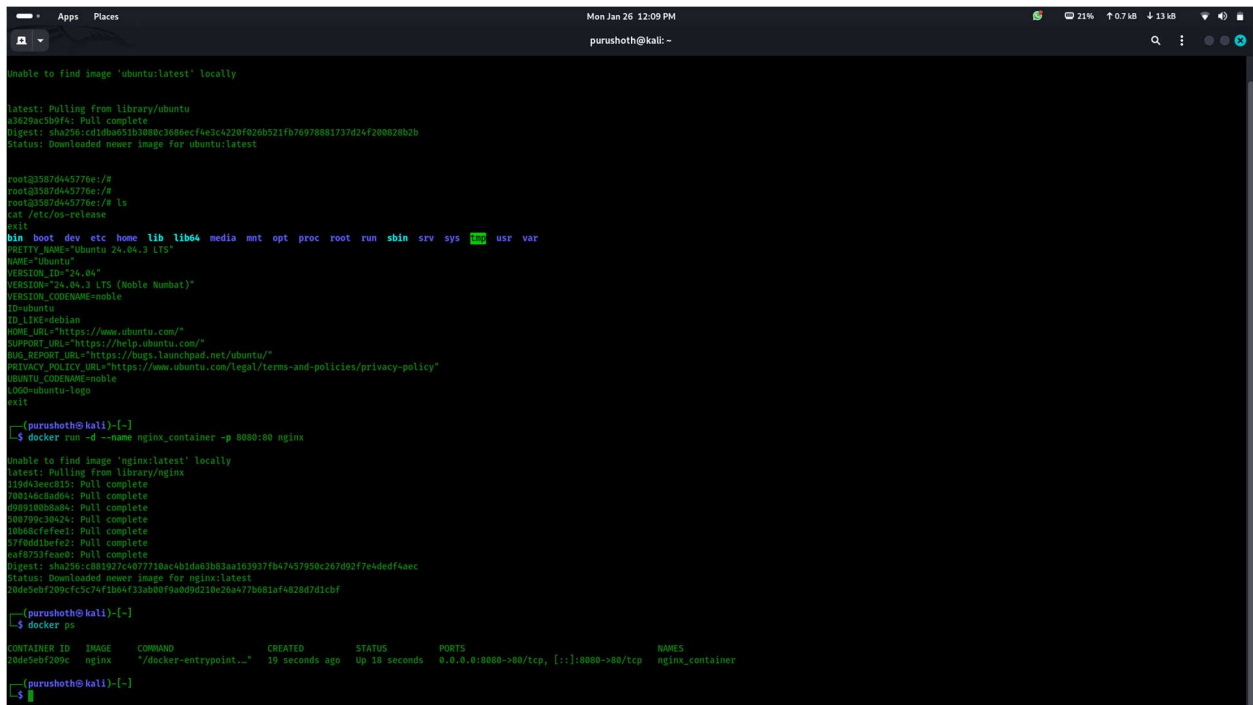
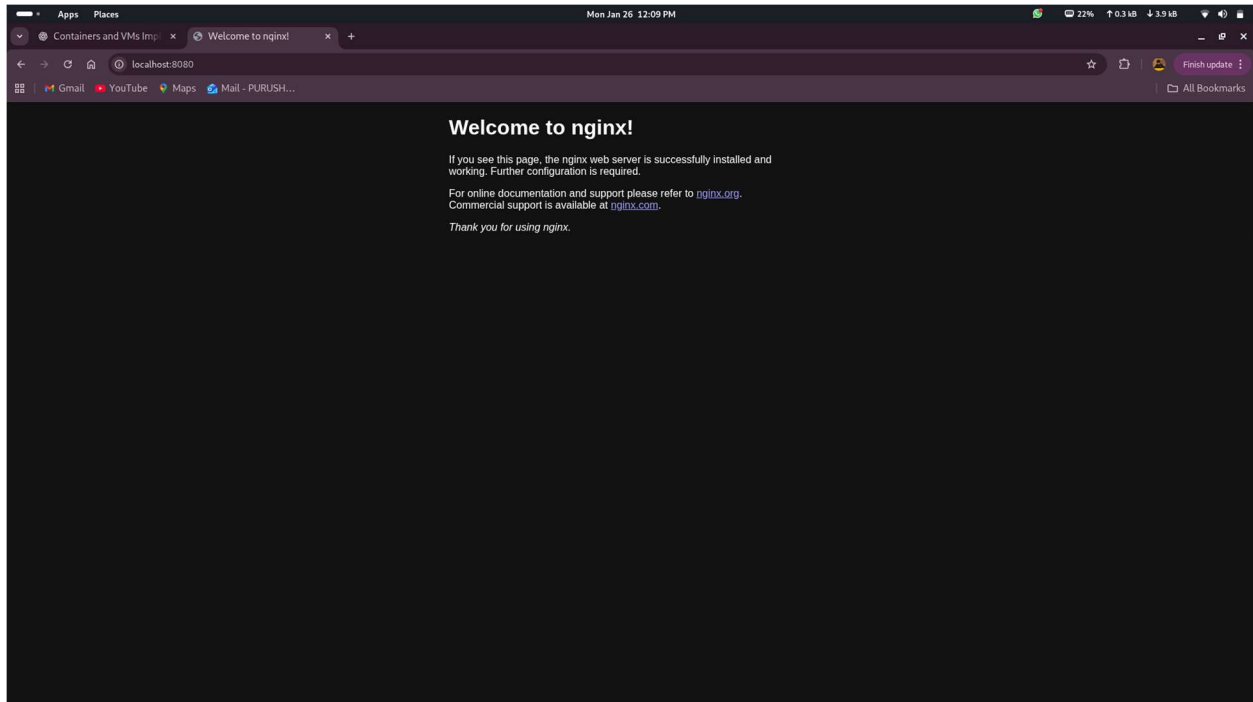
Open browser and visit:

<http://localhost:8080>

### Observation:

Nginx default welcome page loaded successfully in the browser.

## Screenshot-4: Nginx welcome page



## Step 5: Create Programming Container (Python)

```
docker run -it --name python_container python:3.10 python
```

### Explanation:

- Uses official Python 3.10 image
- Launches Python interpreter inside the container

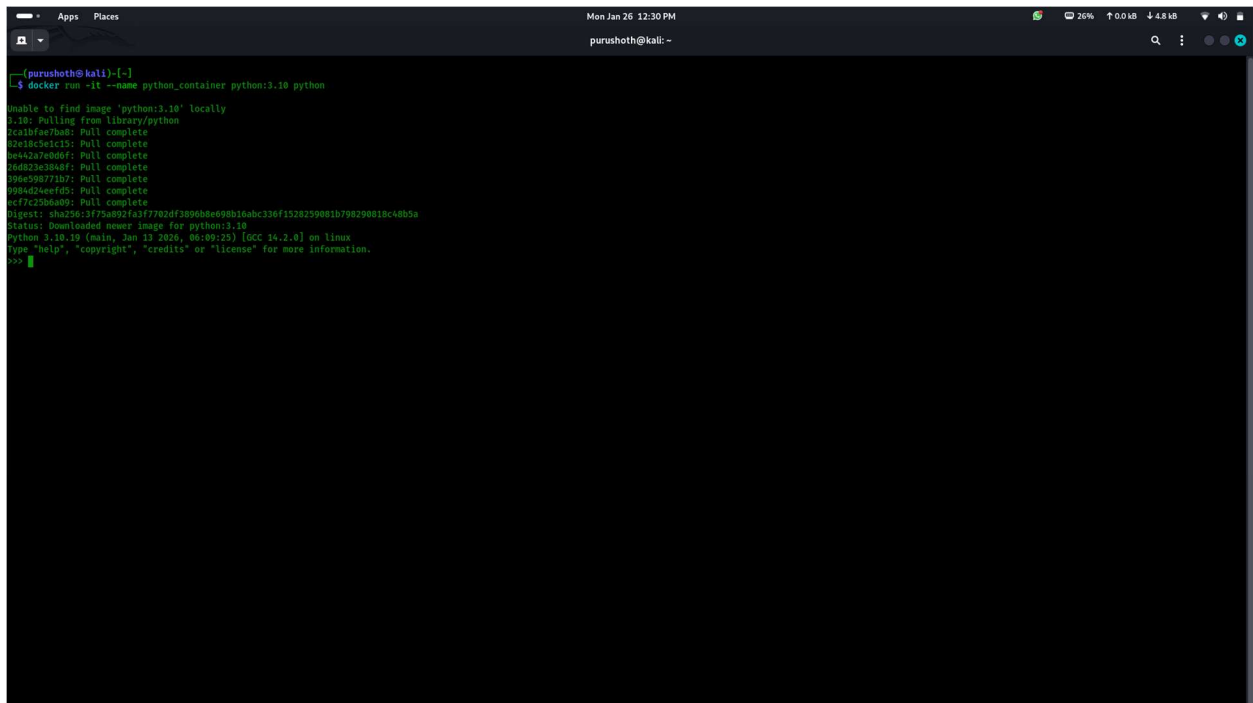
### Commands executed inside Python shell:

```
print("Hello from Python Container")  
exit()
```

### Observation:

Python code executed successfully inside container.

### Screenshot-5: Python container output



```
purushoth@kali:~$ docker run -it --name python_container python:3.10 python  
Unable to find image 'python:3.10' locally  
3.10: pulling from library/python  
3c1b1e7b0a8: Pull complete  
02e18c5e1c15: Pull complete  
be442a7e0d0f: Pull complete  
26d023e3840f: Pull complete  
596e590771b7: Pull complete  
0984d24eef05: Pull complete  
ecf7c25b0a09: Pull complete  
Digest: sha256:2f75a092f237792df2806b0e60b10ahc336f1328259081b798290818c4db9a  
Status: Downloaded newer image for python:3.10  
Python 3.10.19 (main, Jan 13 2026, 06:09:25) [GCC 14.2.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>>
```



```
Mon Jan 26 12:30 PM
purushoth@kali: ~
[~] (purushoth@kali)-[~]
└─$ docker run -it --name python_container python:3.10 python
Unable to find image 'python:3.10' locally
3.10: Pulling from library/python
7ca1bfae7ba8: Pull complete
82e18c5e1c15: Pull complete
be442a7e8d6f: Pull complete
26d82383848f: Pull complete
596e59b771b7: Pull complete
9984d2aef0d5: Pull complete
ecf7c25b6a09: Pull complete
Digest: sha256:2f77a0e27a377782df3896b8e698b1d9c336f1528259081b798290818c4d5a
Status: Downloaded newer image for python:3.10
Python 3.10.19 (main, Jan 12 2026, 06:09:25) [GCC 14.2.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Hello from Python Container")
Hello from Python Container
>>> exit()
[~] (purushoth@kali)-[~]
└─$
```

## Output

- Ubuntu OS container executed successfully
- Nginx web server container hosted a web page
- Python container executed a Python program
- All containers ran independently on the same host system

## Result

Successfully implemented and executed **different Docker containers** including:

- Hello-World
- Ubuntu OS
- Nginx Web Server
- Python Programming Environment

## Conclusion

Docker containers provide a **lightweight, fast, and efficient** method of application isolation.

Multiple containers can run on a single host without requiring separate operating systems, making Docker ideal for **development, deployment, and cloud environments**.