

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
```

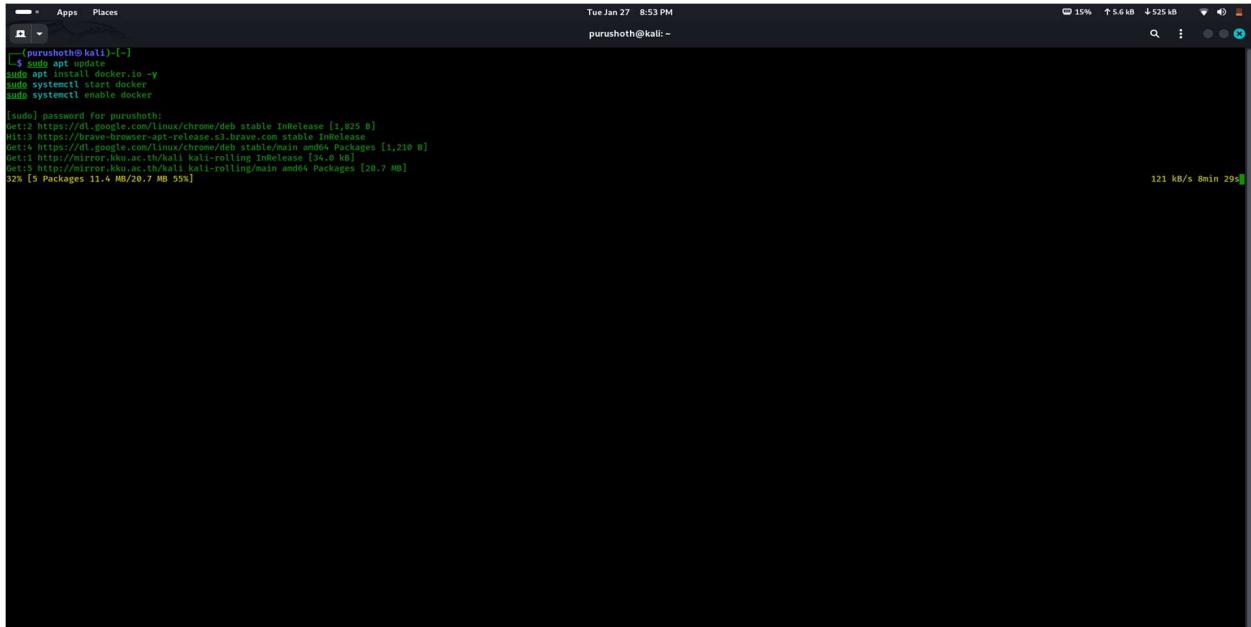
Explanation:

- `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



The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal window title is '(purushoth@kali)'. The command history shows the following sequence of commands:

```
(purushoth@kali) ~
$ sudo apt update
[sudo] password for purushoth:
[...]
get 2 https://dl.google.com/linux/chrome/deb stable InRelease [1,025 B]
get 3 https://brave-browser-apt-release.s3.brave.com stable InRelease
get 4 https://dl.google.com/linux/chrome/deb/stable/main amd64 Packages [1,210 B]
get 5 http://mirror.kku.ac.th/kali kali-rolling InRelease [34.0 kB]
get 6 http://mirror.kku.ac.th/kali kali-rolling/main amd64 Packages [20.7 MB]
32% [5 Packages 11.4 MB/20.7 MB 55%]
```

The terminal window also displays system status information at the top right, including battery level (15%), signal strength, and network speed (121 kB/s).

```
  Apps  Places      Tue Jan 27  9:03 PM
purushoth@kali:~
```

Fetched 72.2 MB in 2min 7s (567 kB/s)

Selecting previously unselected package runc.

Reading database ... 449889 files and directories currently installed.

Preparing to unpack .../runc_1.3.3-0+deb9u2_amd64.deb ...

Unpacking runc (1.3.3-0+deb9u2) ...

Selecting previously unselected package containerd.

Preparing to unpack .../containerd_1.7.24-0+deb10_amd64.deb ...

Unpacking containerd (1.7.24-0+deb10) ...

Selecting previously unselected package tini-static.

Preparing to unpack .../tini_0.19.0-0_amd64.deb ...

Unpacking tini-static (0.19.0-0) ...

Selecting previously unselected package docker.io.

Preparing to unpack .../3-docker.io_27.5.1+dfsg-1_amd64.deb ...

Unpacking docker.io (27.5.1+dfsg-1) ...

Selecting previously unselected package libcompose1:amd64.

Preparing to unpack .../libcompose1_4.2-1_amd64.deb ...

Unpacking libcompose1: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+deb1-0_amd64.deb ...

Unpacking docker-buildx (0.19.3+deb1-0) ...

Selecting previously unselected package docker-cli.

Preparing to unpack .../7-docker-cl_27.5.1+dfsg-1_amd64.deb ...

Unpacking docker-cl (27.5.1+dfsg-1) ...

Selecting previously unselected package python3-protobuf.

Preparing to unpack .../8-python3-protobuf_3.21.12-19_amd64.deb ...

Unpacking python3-protobuf (3.21.12-19) ...

Selecting previously unselected package python3-pycrui.

Preparing to unpack .../9-python3-pycrui_4.2-1_all.deb ...

Unpacking python3-pycrui (4.2-1) ...

Setting up docker-cl (27.5.1+dfsg-1) ...

Setting up docker-buildx (0.19.3+deb1-0) ...

Setting up runc (1.3.3-0+deb2-0) ...

Setting up libcompose1:amd64 (4.2-1) ...

Setting up criu (4.2-1) ...

Setting up python3-protobuf (3.21.12-19) ...

Setting up containerd (1.7.24-0+deb10) ...

Containerd is already enabled on a static unit not running, not starting it.

Setting up docker-buildx (0.19.3+deb1-0) ...

Setting up python3-pycrui (4.2-1) ...

Setting up docker.io (27.5.1+dfsg-1) ...

update-rc.d: We have no instructions for the docker init script.

update-rc.d: It looks like a non-network multi-service, we probably don't want to start it.

Creating 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 libc-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/system-sysv-install.

Executing: /usr/lib/systemd/systemd-sysv-install enable docker

```
[purushoth@kali:~]
```

Step 2: Verify Docker Installation

```
docker --version
```

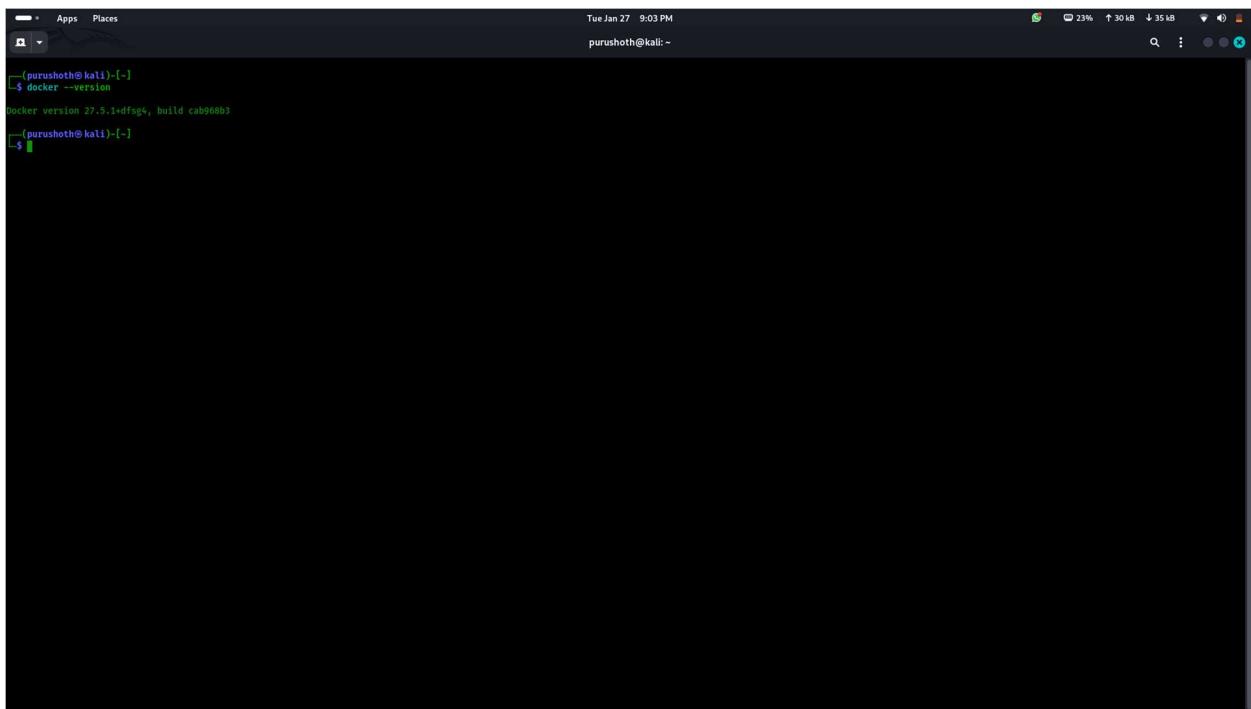
Explanation:

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



The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal window has a dark background and white text. At the top, it shows the date and time: "Tue Jan 27 9:03 PM". Below that, it shows the user's name: "purushoth@kali: ~". The terminal window contains the following text:
```(purushoth@kali) [-]  
-\$ docker --version  
Docker version 27.5.1+dfsg1, build cab968b3  
(-[purushoth@kali) [-]  
-\$```  
The terminal window is surrounded by a black border, which is part of the desktop environment's window frame.

## 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
```

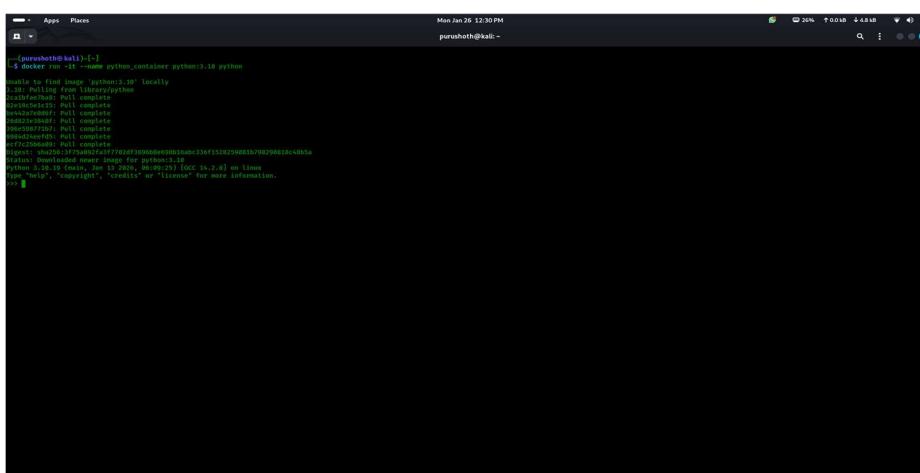
## **Explanation of commands:**

- `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



```
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
3629ac50bf4: Pull complete
Digest: sha256:cddba651b308c3a00ecf4e3c4220f026b521fb7697888173fd24f20082852b
Status: Downloaded newer image for ubuntu:latest

root@3587d445776e:/#
root@3587d445776e:#
root@3587d445776e:~# ls
root@3587d445776e:~# /etc/os-release
exit
bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys [sys] usr var
PRETTY_NAME='Ubuntu 20.04.3 LTS'
NAME='Ubuntu'
VERSION_ID='20.04.3'
VERSION='20.04.3 LTS (Noble Numbat)'
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL='https://www.ubuntu.com/'
SUPPORT_URL='https://help.ubuntu.com/'
BUG_URL='https://bugs.launchpad.net/ubuntu/'
PRIVACY_POLICY_URL='https://www.ubuntu.com/legal/terms-and-policies/privacy-policy'
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
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:

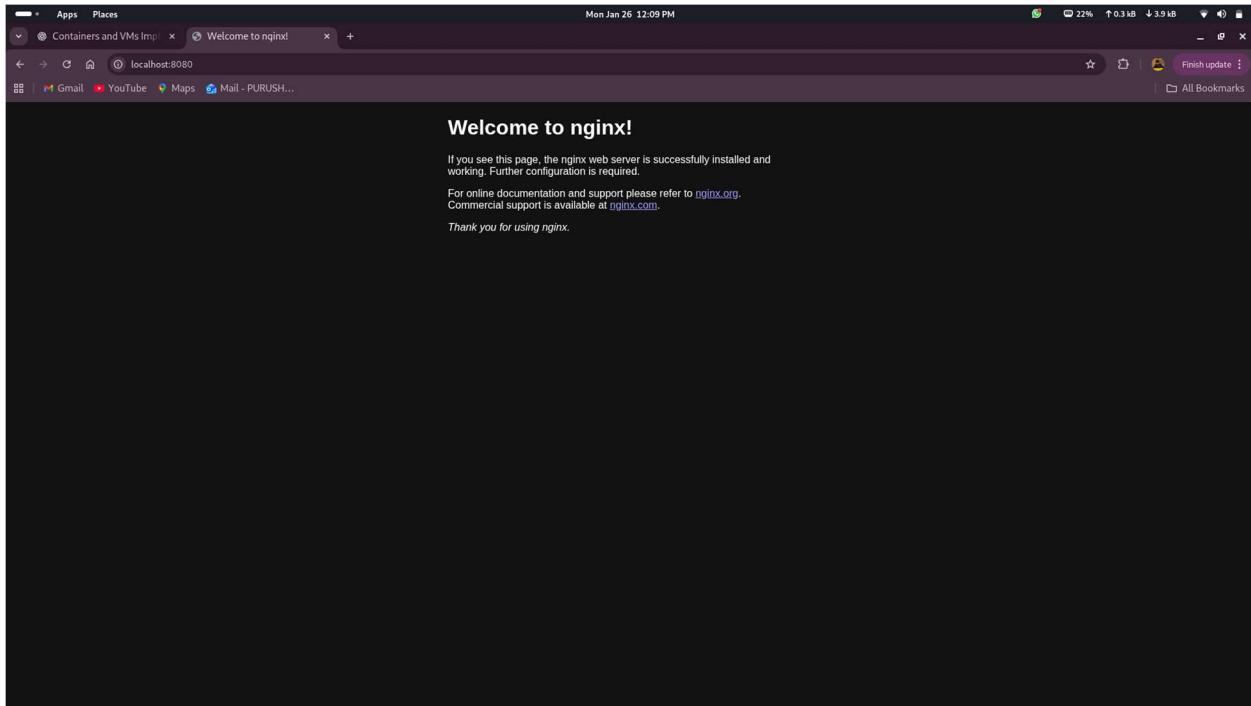
Open browser and visit:

<http://localhost:8080>

### Observation:

Nginx default welcome page loaded successfully in the browser.

#### Screenshot-4: Nginx welcome page



```
Mon Jan 26 12:09 PM
purushoth@kali:~>

Unable to find image 'ubuntu:latest' locally

latest: Pulling from library/ubuntu
c3629a5cb9f4: Pull complete
Digest: sha256:cdd1ba651b3080c3686ecf4e3c4220f020b521fb76978881737d24f200828b2b
Status: Downloaded newer image for ubuntu:latest

00013307d4a5577e6:/#
root@3307d4a5577e6:/#
root@3307d4a5577e6:/# ls
cat /etc/os-release
exit
bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys [REDACTED] usr var
[REDACTED]_NAME='Ubuntu'
[REDACTED]'=24.04.3 LTS (Noble Numbat)'
VERSION_ID='24.04.3 LTS (Noble Numbat)'
VERSION_CODENAME=noble
ID=ubuntu
ID=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
[REDACTED]=ubuntu-logo
exit
[REDACTED]purushoth@kali:[-]
[REDACTED]$ docker run -d --name nginx_container -p 8080:80 nginx

Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
119043eecd55: Pull complete
700146ca0d04: Pull complete
00981008a0a6: Pull complete
00079520a424: Pull complete
000000000000: Pull complete
07ff0dd1be02: Pull complete
0af8753feaa0: Pull complete
Digest: sha256:c8019197e4077710ac4b1da63b83aa16997fb47457950c267092f7e4dedf4aec
Status: Downloaded newer image for nginx:latest
00de5ef209fcfc7xf1bd4f33ab00f9ab9d0218e28a477b601af4828d7d1cbf

[REDACTED]purushoth@kali:[-]
[REDACTED]$ docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
00de5ef209fc nginx "/docker-entrypoint..." 19 seconds ago Up 18 seconds 0.0.0.0:8080->80/tcp, [::]:8080->80/tcp nginx_container

[REDACTED]purushoth@kali:[-]
[REDACTED]$
```

## 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

The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal title is 'purushoth@kali:~' and the date and time are 'Mon Jan 26 12:30 PM'. The command entered is 'docker run -it --name python\_container python:3.10 python'. The output shows the Python interpreter starting and executing the print statement 'Hello from Python Container' before exiting.

```
[purushoth@kali:~]
$ docker run -it --name python_container python:3.10 python
Hello from Python Container
>>> exit()
```

The screenshot shows a terminal window on a Kali Linux host system. The terminal session starts with the command `docker run -it --name python_container python:3.10 python`. The output shows the Docker daemon pulling the image from the library and then executing a Python script within the container. The script prints "Hello from Python Container" and then exits. The terminal prompt at the end is `[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
5ca1bfae7ba8: Pull complete
02e8c8c5e151: Pull complete
0e442a7e0ecf: Pull complete
0a6d9a849: Pull complete
09e859371b7: Pull complete
090ad2eeefdf5: Pull complete
0cf7c2506a09: Pull complete
Digest: sha256:3f75a892fa3ff702df3896bbe698b16abc336f1528259081b798290818c48b9a
status: Downloaded newer image for python:3.10
python: Python 3.10.0 (main, Dec 20 2022, 20:13) [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()
[~]$
```

## 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**.