IMPLEMENTATION OF DOCKER CONTAINERS

NAME: Navinesharan S

ROLL NO: 2023115015

ASSIGNMENT -3

Aim

This experiment aims to study and practically implement Docker containerization by installing Docker Engine and creating multiple containers such as Ubuntu OS, Nginx Web Server, and Python Programming Environment. The objective is to clearly understand how containers work, how they differ from virtual machines, and how applications can be deployed efficiently using Docker.

Introduction

Docker is an open-source containerization platform used to package applications along with their dependencies into isolated environments called containers. Unlike traditional virtual machines, Docker containers do not require a complete guest operating system. Instead, they share the host operating system kernel, making them lightweight, faster, and more efficient.

Docker is widely used in software development, cloud computing, DevOps pipelines, and microservices architecture because it ensures consistency across development, testing, and production environments.

Software Requirements

• Operating System: Kali Linux

• Container Platform: Docker Engine

• Hardware: Minimum 4 GB RAM

• Internet Connection: Required for downloading Docker images

• Terminal Access

Step 1: Docker Installation

To begin the experiment, Docker Engine was installed on the Kali Linux system using the Linux package manager. The following commands were executed sequentially:

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

The installation process ensures that Docker is available system-wide and automatically starts during system boot. After installation, the Docker daemon was verified to be running successfully.

Step 2: Docker Version Verification

After installation, Docker was verified using the command:

```
docker –version
```

This command confirms the successful installation of Docker and displays the currently installed Docker version. The version output verifies that the Docker Engine is properly configured and ready for use

Step 3: Ubuntu Container Creation

In this step, an Ubuntu operating system container was created using the official Ubuntu image from Docker Hub.

Command used:

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

This command launches an interactive Ubuntu shell within the container. Inside the container, basic Linux commands such as ls and cat /etc/os-release were executed to verify the operating system details. This demonstrates that containers behave like independent operating environments.

Step 4: Nginx Web Server Container

A web server container was created using the Nginx image. Nginx is a high performance web server widely used in real-world applications.

Command used:

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

This command runs the container in detached mode and maps the container port to the host system. When accessed through the browser using http://localhost:8080, the Nginx default welcome page was successfully displayed.

Step 5: Python Programming Container

In this step, a Python programming environment was created using the official Python 3.10 image.

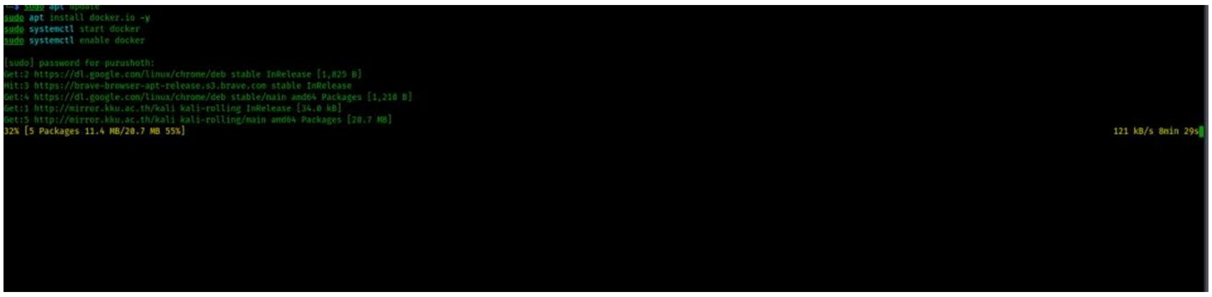
Command used:

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

Inside the Python shell, a simple program was executed to verify functionality. This demonstrates how Docker can be used to run programming environments without installing software directly on the host system.

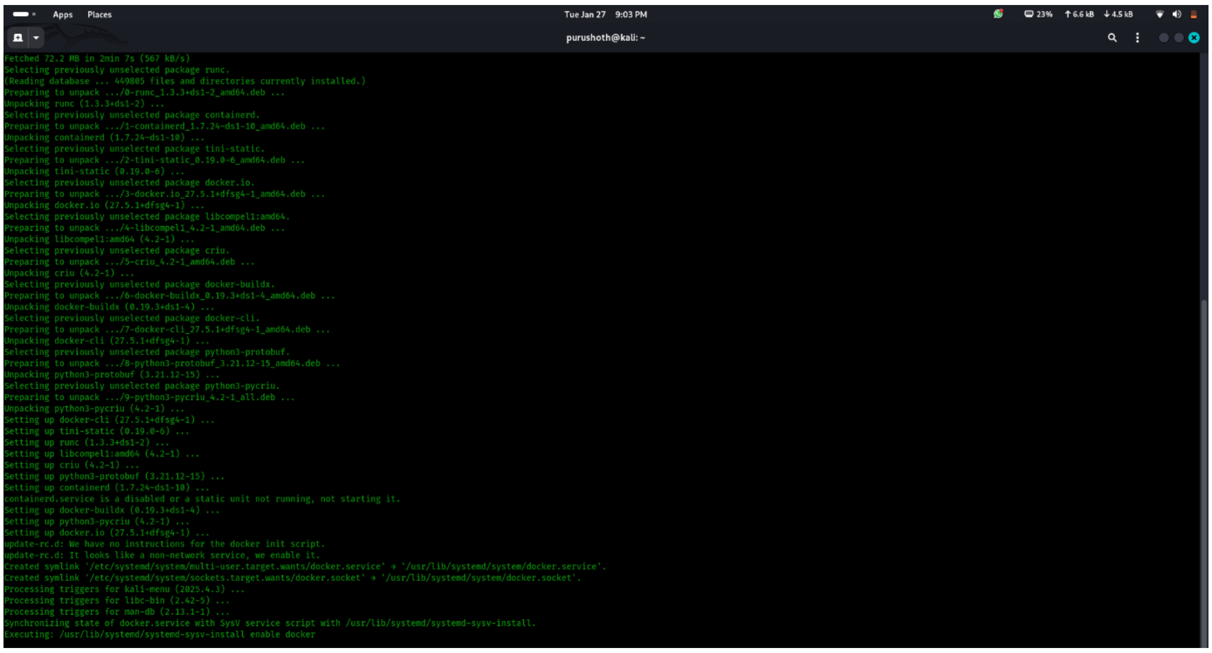
Screenshots and Output

Docker Installation Process



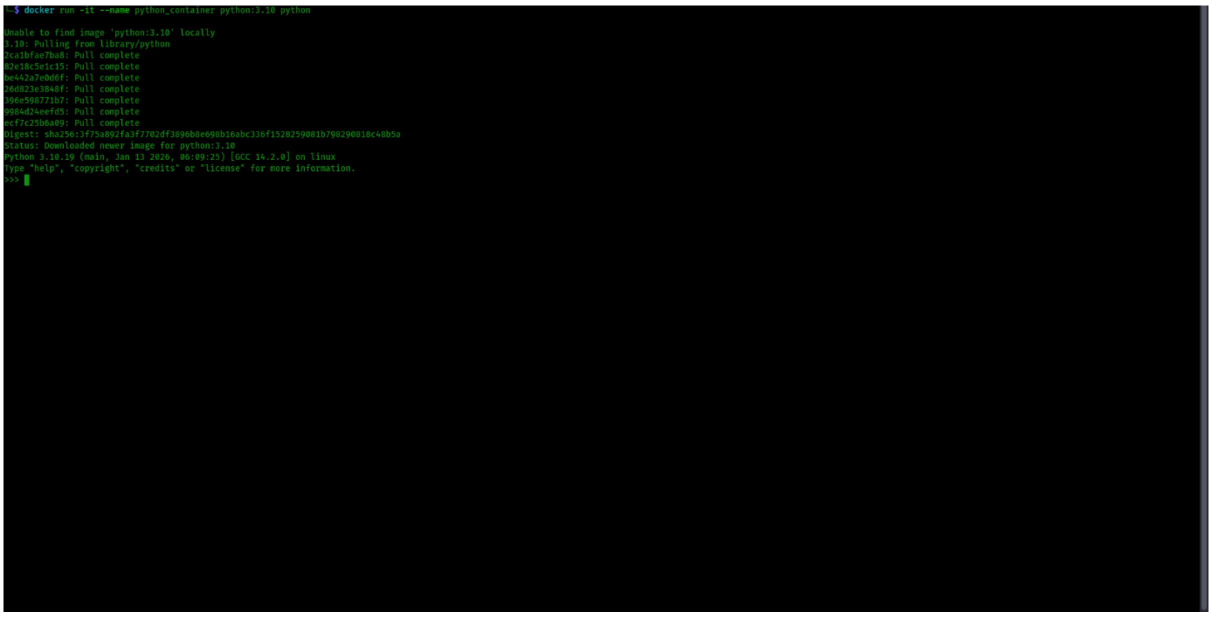
```
└─$ sudo apt update
sudo apt install docker.io -y
sudo systemctl start docker
sudo systemctl enable docker

[sudo] password for purushoth:
Get:2 https://dl.google.com/linux/chrome/deb stable InRelease [1,825 B]
Hit: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:1 http://mirror.kku.ac.th/kali kali-rolling InRelease [34.0 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
```

Docker Installation Completed



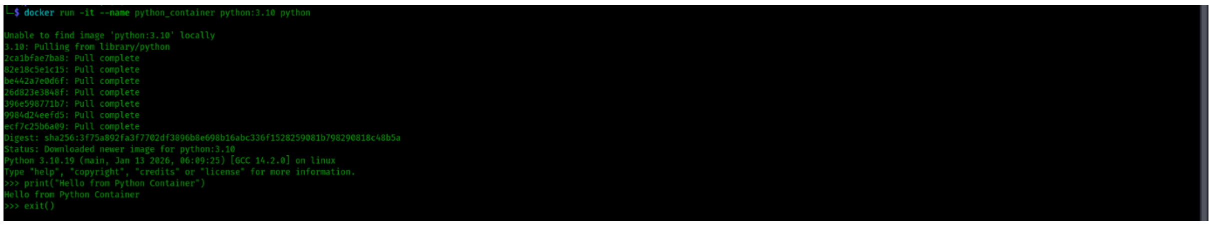
```
Fetched 72.2 MB in 2min 7s (567 kB/s)
Selecting previously unselected package runc.
(Reading database ... 449805 files and directories currently installed.)
Preparing to unpack .../0-runc_1.3.3+ds1-2_amd64.deb ...
Unpacking runc (1.3.3+ds1-2) ...
Selecting previously unselected package containerd.
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 libcompel1:amd64.
Preparing to unpack .../4-libcompel1_4.2-1_amd64.deb ...
Unpacking libcompel1: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 libcompel1: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 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/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

Python Container Creation



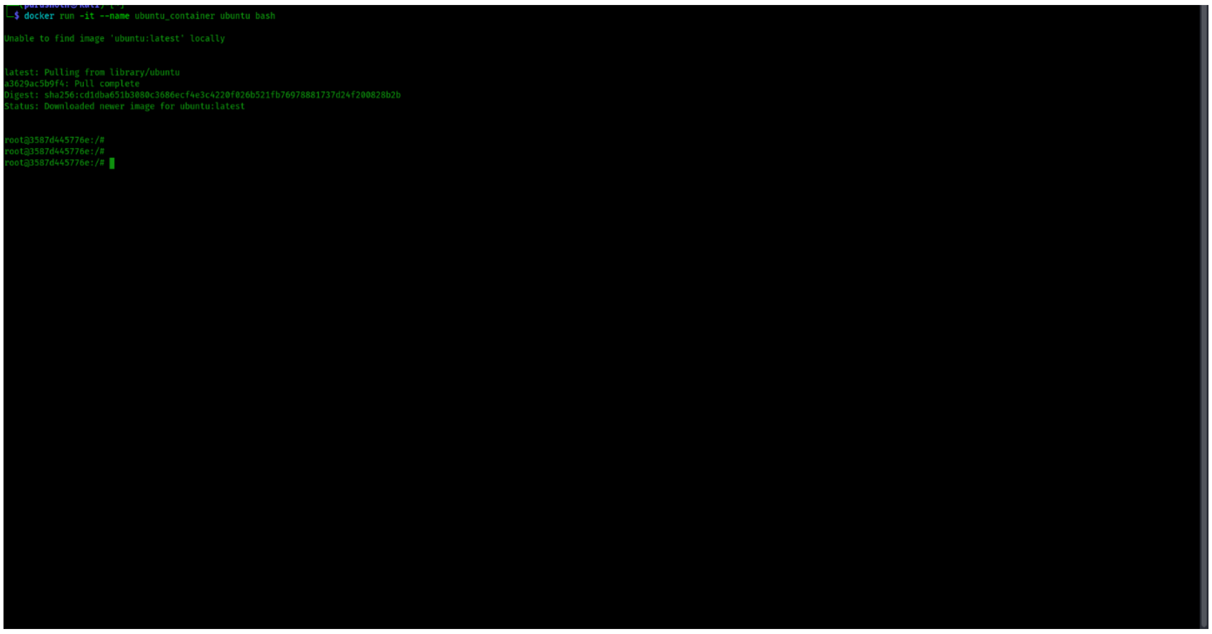
```
└─$ docker run -it --name python_container python:3.10 python

Unable to find image 'python:3.10' locally
3.10: Pulling from library/python
2ca1bfae7ba8: Pull complete
82e18c5e1c15: Pull complete
be442a7e0d6f: Pull complete
26d823e3848f: Pull complete
396e598771b7: Pull complete
9984d24eefd5: Pull complete
ecf7c25b6a09: Pull complete
Digest: sha256:3f75a892fa3f7702df3896b8e698b16abc336f1528259081b798290818c48b5a
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.
>>> 
```

Python Program Output



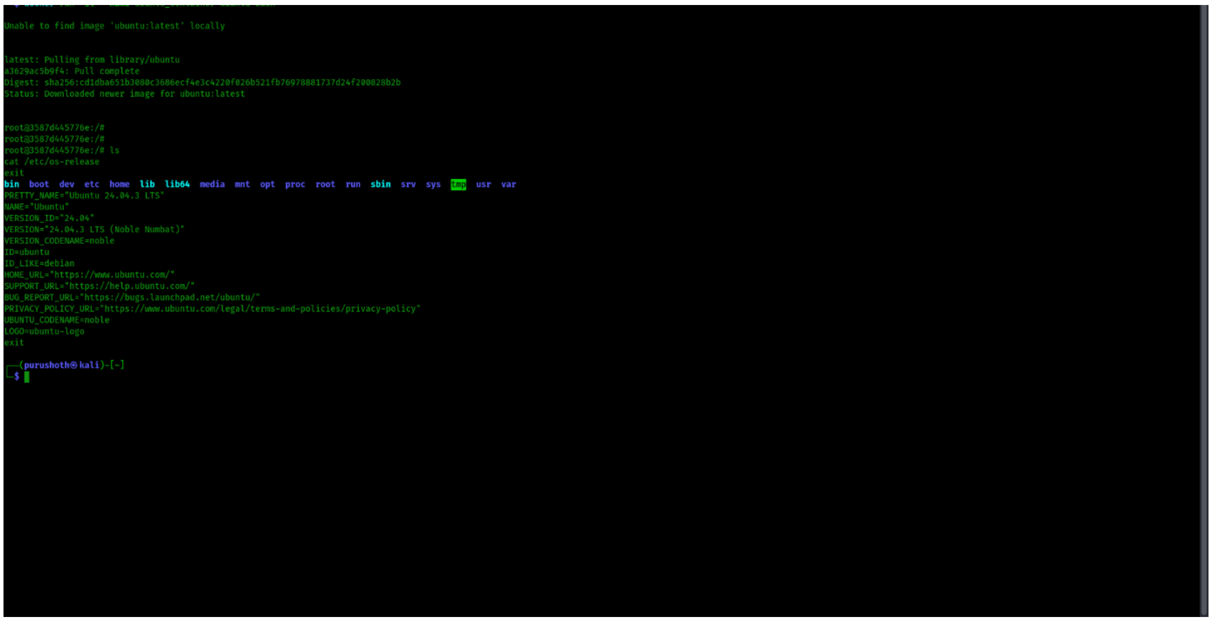
```
└─$ docker run -it --name python_container python:3.10 python

Unable to find image 'python:3.10' locally
3.10: Pulling from library/python
2ca1bfae7ba8: Pull complete
82e18c5e1c15: Pull complete
be442a7e0d6f: Pull complete
26d823e3848f: Pull complete
396e598771b7: Pull complete
9984d24eefd5: Pull complete
ecf7c25b6a09: Pull complete
Digest: sha256:3f75a892fa3f7702df3896b8e698b16abc336f1528259081b798290818c48b5a
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.
>>> print("Hello from Python Container")
Hello from Python Container
>>> exit()
```

Ubuntu Container Creation



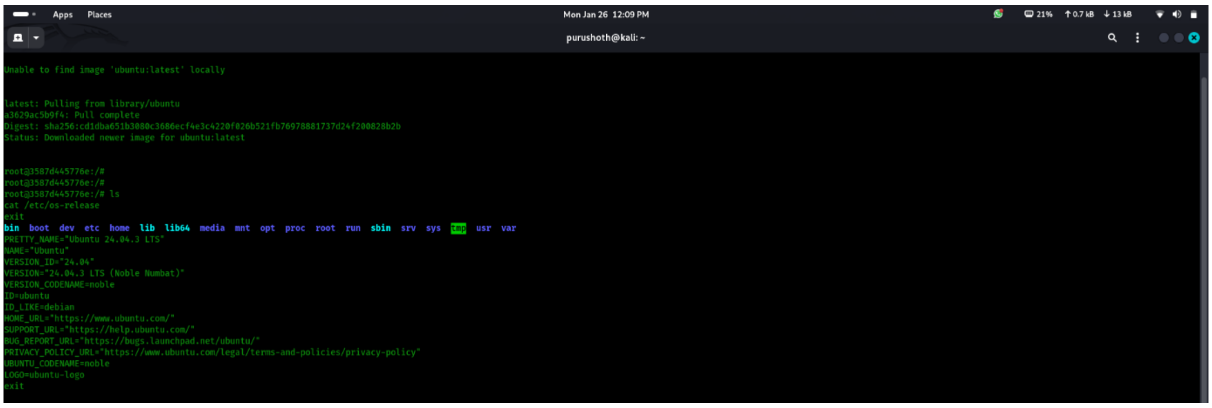
```
└─$ docker run -it --name ubuntu_container ubuntu bash

Unable to find image 'ubuntu:latest' locally


latest: Pulling from library/ubuntu
a3629ac5b9f4: Pull complete
Digest: sha256:cd1dba651b3080c3686ecf4e3c4220f026b521fb76978881737d24f200828b2b
Status: Downloaded newer image for ubuntu:latest


root@3587d445776e:/#
root@3587d445776e:/#
root@3587d445776e:/# 
```

Ubuntu OS Details



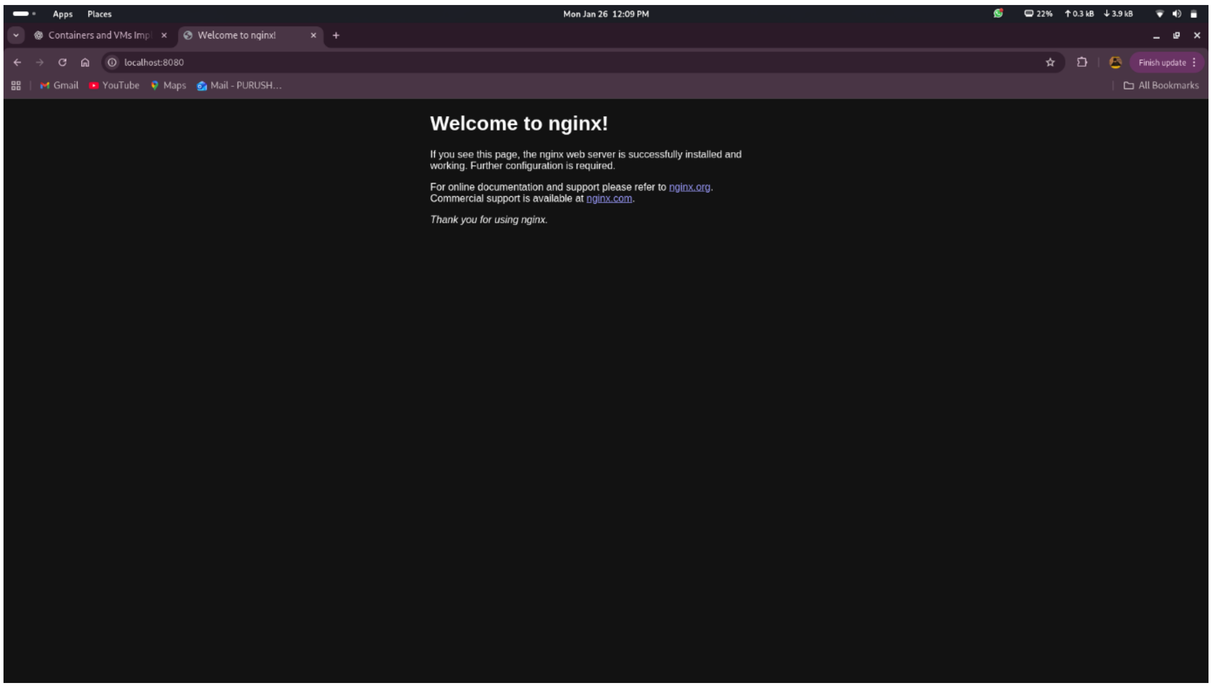
```
Unable to find image 'ubuntu:latest' locally


latest: Pulling from library/ubuntu
a3629ac5b9f4: Pull complete
Digest: sha256:cd1dba651b3080c3686ecf4e3c4220f026b521fb76978881737d24f200828b2b
Status: Downloaded newer image for ubuntu:latest


root@3587d445776e:/#
root@3587d445776e:/#
root@3587d445776e:/# ls
cat /etc/os-release
exit
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  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-logo
exit

┌──(purushoth㉿kali)-[~]
└─$ 
```

Nginx Web Server Container



```
Unable to find image 'ubuntu:latest' locally


latest: Pulling from library/ubuntu
a3629ac5b9f4: Pull complete
Digest: sha256:cd1dba651b3080c3686ecf4e3c4220f026b521fb76978881737d24f200828b2b
Status: Downloaded newer image for ubuntu:latest


root@3587d445776e:/#
root@3587d445776e:/#
root@3587d445776e:/# ls
cat /etc/os-release
exit
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  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-logo
exit
```

Nginx Welcome Page Output



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Output

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

Result

The experiment successfully demonstrated the implementation of Docker containers. Different container types such as operating system containers, web server containers, and programming containers were executed without conflicts, proving Docker's isolation capability.

Conclusion

Docker provides an efficient, scalable, and lightweight solution for application deployment. Containers reduce system overhead, improve portability, and ensure environment consistency. Through this experiment, the advantages of Docker over traditional virtual machines were clearly understood.