

IMPLEMENTATION OF CONTAINERS

Name: Vijay Krishna S

RollNo: 2023115075

Aim

To install Docker on Windows Subsystem for Linux (WSL), create and manage containers, and execute a Python environment inside a Docker container.

Objective

- Install Docker Engine in WSL.
- Verify Docker functionality.
- Run lightweight containers.
- Deploy a web server container.
- Execute Python inside a containerized environment.

System Requirements

- Windows 10/11 with WSL enabled
- Ubuntu (WSL distribution)
- Docker Engine
- Internet connectivity

Procedure

Step 1 – Install Docker

Docker packages and dependencies were installed using the apt package manager.

Command used:

```
sudo apt update
```

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

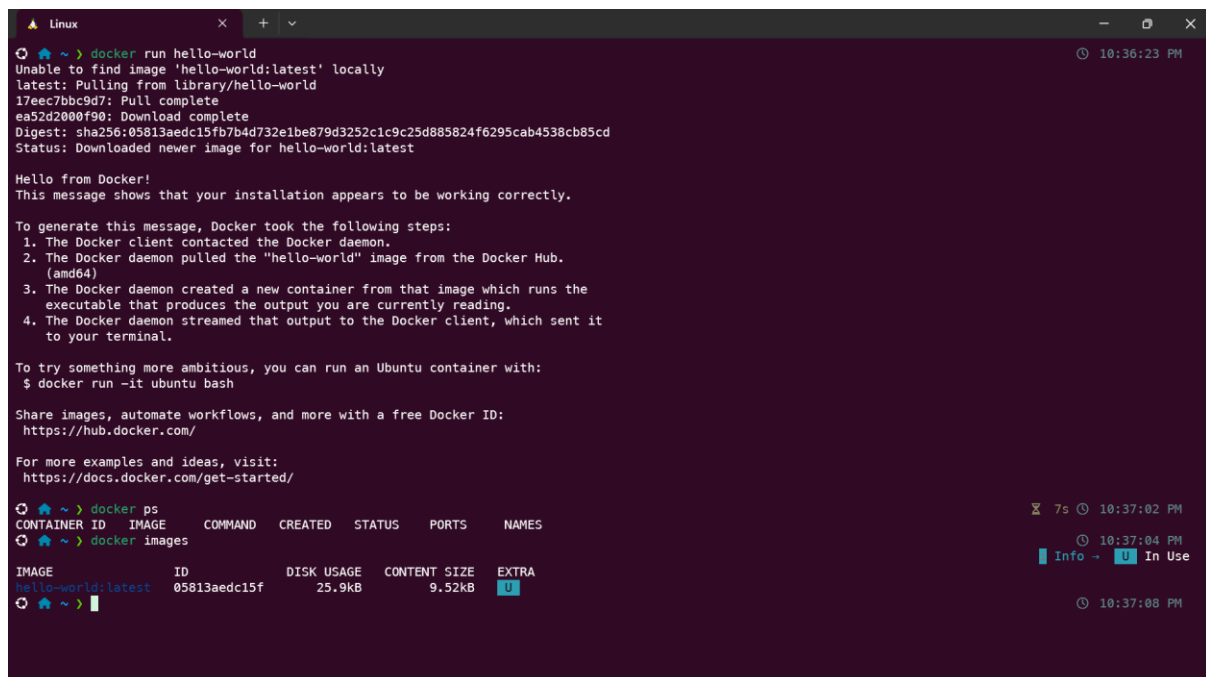
Step 2 – Start Docker Service

The Docker service was started and verified.

```
sudo systemctl start docker
```

```
docker --version
```

Step 3 – Create Containers



```
Linux
~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
17aec7bb9d7: Pull complete
ea52d2000f90: Download complete
Digest: sha256:05813aedc15fb7b4d732e1be879d3252c1c9c25d885824f6295cab4538cb85cd
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

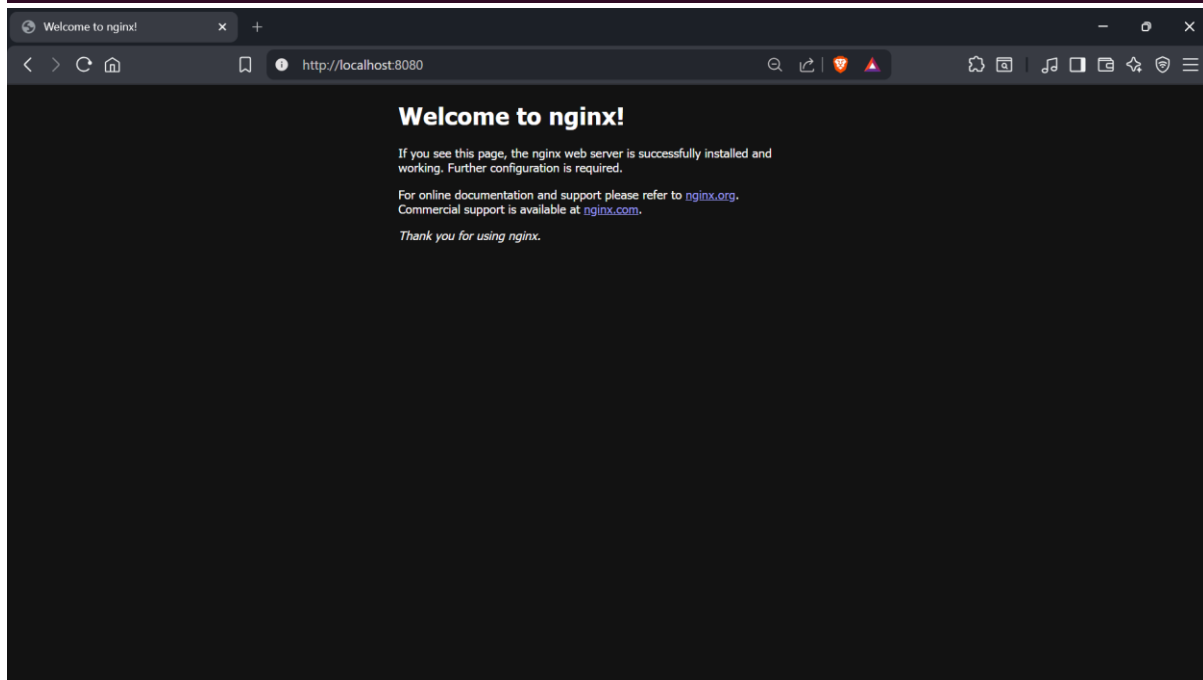
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

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

~$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
~$ docker images
IMAGE                ID              DISK USAGE   CONTENT SIZE   EXTRA
hello-world:latest   05813aedc15f    25.9kB       9.52kB         U
```

```
root@9c13f25464b3:~# docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
root@9c13f25464b3:~# docker images
IMAGE          ID            DISK USAGE   CONTENT SIZE  EXTRA
hello-world:latest  05813aedc15f  25.9kB      9.52kB        U
python:latest     1c4c033d6660  1.63GB      432MB         U
root@9c13f25464b3:~# docker run -it ubuntu bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
a3629ac5b9f4: Pull complete
1ba705536e37: Download complete
Digest: sha256:cd1dba651b3080c3686ecf4e3c4220f026b521fb76978881737d24f200828b2b
Status: Downloaded newer image for ubuntu:latest
root@9c13f25464b3:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
root@9c13f25464b3:/#
```

```
root@9c13f25464b3:~# docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
9c13f25464b3  ubuntu    "bash"                  4 minutes ago  Up 4 minutes  -            wizardly_antonelli
root@9c13f25464b3:~# docker images
IMAGE          ID            DISK USAGE   CONTENT SIZE  EXTRA
hello-world:latest  05813aedc15f  25.9kB      9.52kB        U
python:latest     1c4c033d6660  1.63GB      432MB         U
ubuntu:latest     cd1dba651b30  119MB       31.7MB        U
root@9c13f25464b3:~# docker run -d -p 8080:80 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
0c8d55a45c0d: Pull complete
47cd406a84ef: Pull complete
4f4efe02d542: Pull complete
7b6cb8ccac7b: Pull complete
46bf3a120c8e: Pull complete
bae5a1799a80: Pull complete
f73400a233fd: Pull complete
a5d78d617315: Download complete
2e02dba24409: Download complete
Digest: sha256:b17697e86d0c02378716277d09f45b946f8709aaa12c708e30fdd4736f536af1
Status: Downloaded newer image for nginx:latest
9f4a4258f259d868c081a9c2e922568c6f1ab524cd177368aa31e59832829620
root@9c13f25464b3:~#
```



Step 4 – View Docker Containers

All containers were listed to verify their status.

`docker ps -a`

This displayed running and stopped containers.

The screenshot shows the Docker Desktop application window. The left sidebar contains navigation options: Ask Gordon, Containers, Images (selected), Volumes, Kubernetes, Builds, Models, MCP Toolkit, Docker Hub, Docker Scout, and Extensions. The main panel displays the 'Images' tab, showing a list of local images. The status bar at the bottom indicates the engine is running with 3.96 GB RAM and 3.01% CPU usage.

Name	Tag	Image ID	Created	Size	Actions
hello-world	latest	05813aedc15f	6 months ago	25.9 KB	[Play] [More] [Delete]
ubuntu	latest	cd1dba651b30	23 days ago	119.25 MB	[Play] [More] [Delete]
nginx	latest	b17697e86d0c	17 hours ago	239.91 MB	[Play] [More] [Delete]
python	latest	1c4c033d6660	21 hours ago	1.62 GB	[Play] [More] [Delete]

Showing 4 items

Engine running | RAM 3.96 GB CPU 3.01% | Disk: 32.38 GB used (limit 1006.85 GB) | Update available

```
Linux x Linux + v
10:41:47 PM
root@wiz:~# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
9c13f25464b3   ubuntu   "bash"    4 minutes ago   Up 4 minutes   wizardly_antonelli

root@wiz:~# docker images
IMAGE                ID              DISK USAGE  CONTENT SIZE  EXTRA
hello-world:latest   05813aedc15f    25.9KB      9.52KB        U
python:latest        1c4c033d6660    1.63GB      432MB         U
ubuntu:latest        cd1dba651b30    119MB       31.7MB        U
10:45:05 PM
root@wiz:~#
```

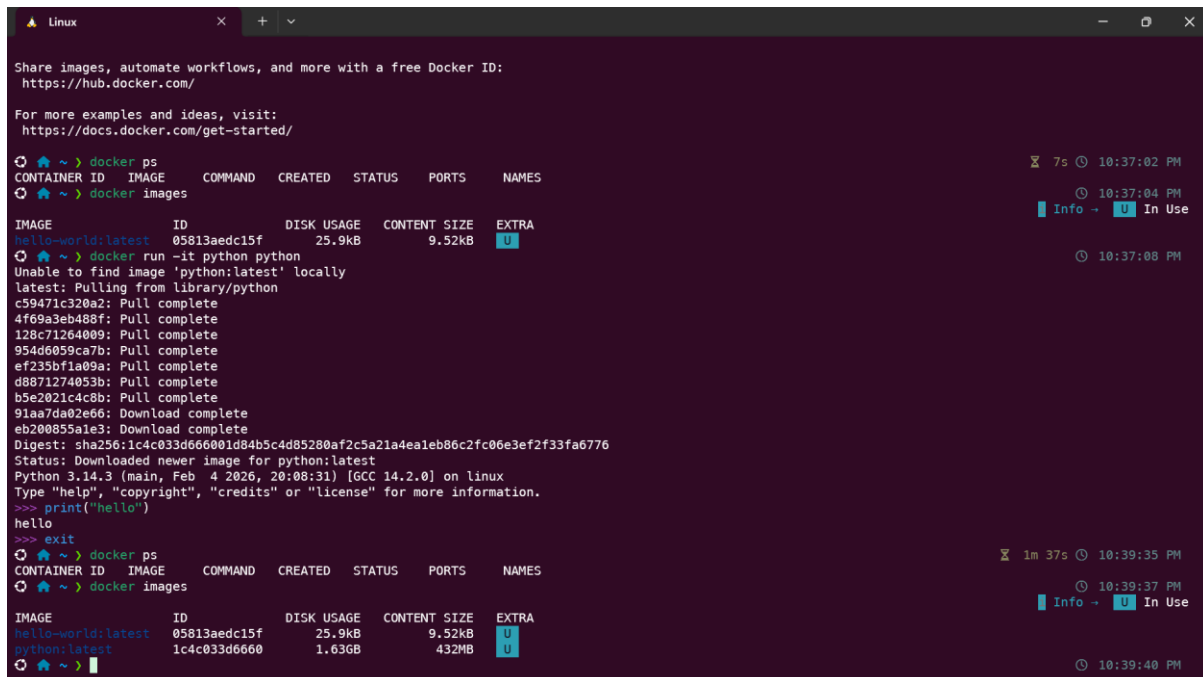
Step 5 – Run Python Container

A Python development container was launched.

```
docker run -it python:3.11-slim bash
```

```
python3
```

Python executed successfully inside the container.



The screenshot shows a Linux terminal window with a dark background. It displays the output of several Docker commands. First, `docker ps` shows a single container named `hello-world` running. Then, `docker images` lists the `hello-world:latest` image. Next, `docker run -it python python` is executed, showing the process of pulling the `python:latest` image from Docker Hub. The terminal output includes the image layers being pulled and the final Python prompt. The user enters `print('hello')`, and the output `hello` is displayed. Finally, the user enters `exit`, and the terminal shows the container has exited. The terminal also displays system information like disk usage and content size for the images.

```
Linux
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/

~$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
~$ docker images
IMAGE          ID          DISK USAGE  CONTENT SIZE  EXTRA
hello-world:latest  05813aedc15f  25.9kB      9.52kB        U
~$ docker run -it python python
Unable to find image 'python:latest' locally
latest: Pulling from library/python
c59471c320a2: Pull complete
4f69a3eb488f: Pull complete
128c71264009: Pull complete
954d6059ca7b: Pull complete
ef235bf1a09a: Pull complete
d8871274053b: Pull complete
b5e2021c4c8b: Pull complete
91aa7da02e66: Download complete
eb200855a1e3: Download complete
Digest: sha256:1c4c033d66001d84b5c4d05200af2c5a21a4ea1eb06c2fc06e3ef2f33fa6776
Status: Downloaded newer image for python:latest
Python 3.14.3 (main, Feb  4 2026, 20:08:31) [GCC 14.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print('hello')
hello
>>> exit
~$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
~$ docker images
IMAGE          ID          DISK USAGE  CONTENT SIZE  EXTRA
hello-world:latest  05813aedc15f  25.9kB      9.52kB        U
python:latest    1c4c033d6600  1.63GB      432MB         U
~$
```

Output

- Docker Engine installed successfully.
- Containers were created and executed without errors.
- Python environment functioned correctly inside Docker.

Result

Docker was successfully installed and configured in WSL. Multiple containers were created and managed, and a Python runtime environment was executed inside a Docker container, demonstrating effective containerization.