Docker

```
##############################################################
                    Docker Installation
##############################################################
```

---** Remove old packages **---

```
$ for pkg in docker.io docker-doc docker-compose podman-docker containerd runc; do sudo apt-get
remove $pkg; done
```

---** Set up Repository **---

```
$ sudo apt-get update

$ sudo apt-get install ca-certificates curl gnupg -y

$ sudo install -m 0755 -d /etc/apt/keyrings

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg

$ sudo chmod a+r /etc/apt/keyrings/docker.gpg

$ echo \
  "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

---** Install Docker Engine CE **---

```
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y
```

---** Verify that the Docker Engine installation **---

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

##

---** Install using the convenience script **---

>>> Dry run

```
$ curl -fsSL https://get.docker.com -o get-docker.sh
$ sudo sh ./get-docker.sh --dry-run
```

>> Actual

```
$ curl -fsSL https://get.docker.com -o get-docker.sh

$ sudo sh get-docker.sh
```

##

----*** Running docker commands ***----

```
$ sudo docker version

$ sudo docker info

$ sudo docker

$ sudodocker container run --publish 80:80 nginx
                              [host:container]

$ sudo docker container run --publish 80:80 --detach nginx

$ sudo docker container ls

$ sudo docker container stop <container-id>
$ sudo docker container ls

$ sudo docker container ls -a

$ sudo docker container run --publish 80:80 --detach --name my-container nginx

$ sudo docker container logs my-container

$ sudo docker container top my-container

$ sudo docker container --help

$ sudo docker container stop my-container

$ sudo docker container rm my-container

$ sudo docker container rm <container-id> <container-id>
```

>> Check process on OS

```
$ sudo docker run --name mongo -d mongo

$ sudo docker container ls

$ ps aux

$ ps aux | grep mongo

$ sudo docker top mongo
```

>> Inspect and stat command

```
$ sudo docker container run --name webserver -d -p 8080:80 httpd

$ sudo docker container inspect webserver

$ sudo docker container stats webserver

$ $ sudo docker container stats
```

>> Start new container interactively

```
$ sudo docker run --help

$ sudo docker container run -it --name abc nginx bash

$ exit

$ sudo docker container ls -a
```

>> Start existing container

```
$ sudo docker container start -ai abc
        # cat > abc
          test
        #exit

$ sudo docker container start -ai abc
        # ls

$ sudo docker container run -it --name ubuntu ubuntu
```

>> exec command - Run additional process inside a container

```
$ sudo docker container run --name mynginx -d nginx

$ sudo docker container ls -a

$ sudo docker container exec -it mynginx bash

        apt update
        apt install procps -y
        ps aux
```

>> Networking in Docker

```
$ sudo docker container run -p 80:80 --name web1 -d nginx

$ sudo docker container port web1

$ sudo docker container ls
$ sudo docker container inspect web1

$ sudo docker container inspect --format '{{.NetworkSettings.IPAddress}}' web1

$ ifconfig

$ sudo docker network ls

$ sudo docker network inspect bridge

$ sudo docker network create my-net-1

$ sudo docker network ls

$ sudo docker network inspect my-net-1

$ sudo docker container run -d --name web2 --network my-net-1 nginx
```

(Start container with custom network interface)

```
$ sudo docker network inspect my-net-1

$ sudo docker network connect my-net-1 web1
```

(Connect existing container to custom network interface)

```
$ sudo docker network inspect my-net-1

$ sudo docker container inspect web1
```

(Check the network interfaces container is connected to)

```
$ sudo docker network disconnect my-net-1 web1
```

>> Docker Images

```
$ sudo docker pull apache/kafka

$ sudo docker image ls

$ sudo docker run -d --name broker apache/kafka:latest

$ sudo docker container ls

$ sudo docker exec --workdir /opt/kafka/bin/ -it broker sh
$ ./kafka-topics.sh --bootstrap-server localhost:9092 --create --topic test-topic

$ ./kafka-console-producer.sh --bootstrap-server localhost:9092 --topic test-topic

$ ./kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic test-topic --from-beginning

$ exit
```

>> Building and pushing Images

```
$ mkdir myproject

$ cd myproject

$ nano hello.py
from flask import Flask
app = Flask(__name__)
@app.route("/")
def hello():
   return "Hello World!"


$ nano dockerfile
# syntax=docker/dockerfile:1
FROM ubuntu:22.04

# install app dependencies
RUN apt-get update && apt-get install -y python3 python3-pip
RUN pip install flask==3.0.*

# install app
COPY hello.py /

# final configuration
ENV FLASK_APP=hello
EXPOSE 8000
CMD ["flask", "run", "--host", "0.0.0.0", "--port", "8000"]
```

```
$ sudo docker build -t test:latest .

$ sudo docker run -p 8000:8000 test:latest

$ sudo docker image ls

$ sudo docker image tag test quaziaasem/hello-python

$ sudo docker image ls

$ sudo docker image push quaziaasem/hello-python

$ sudo docker login

$ sudo docker image push quaziaasem/hello-python
```