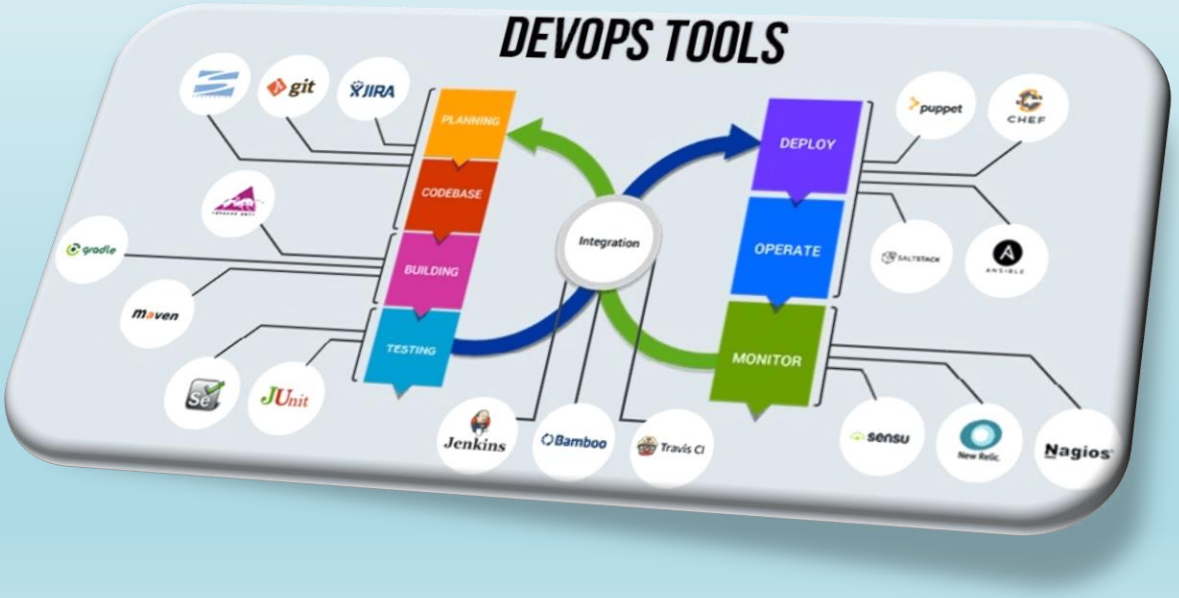
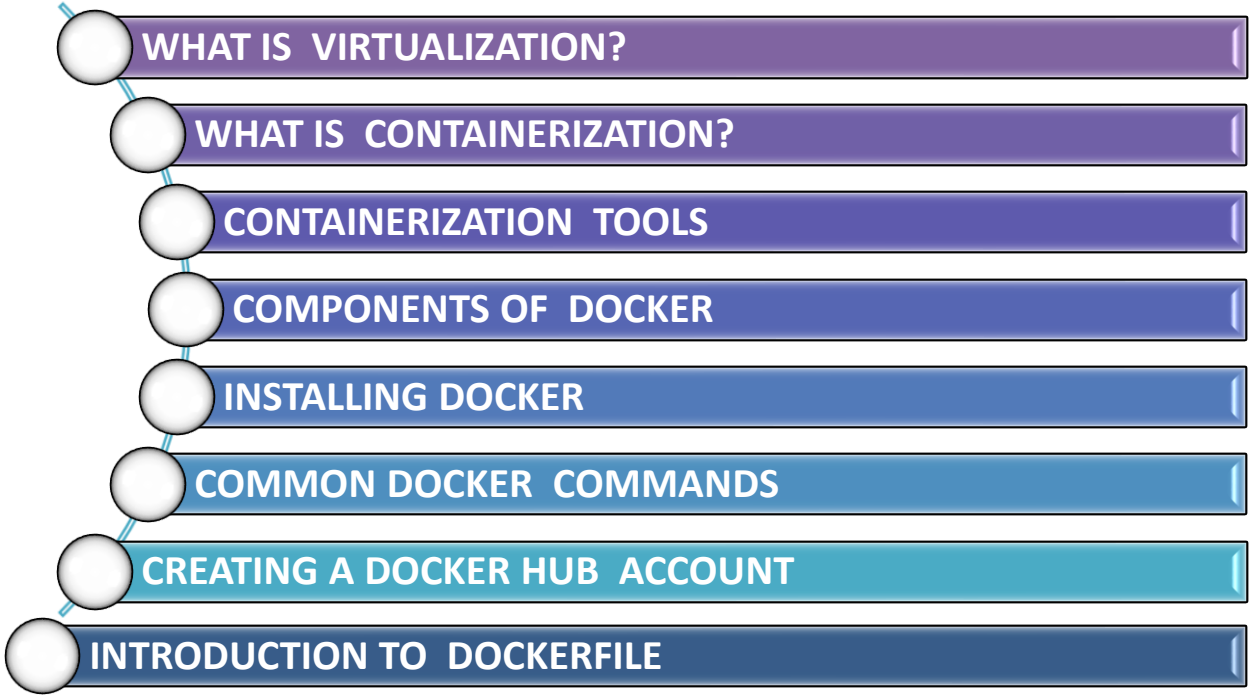
CLOUD TRAIN
ACCELERATE YOUR GROWTH

# Containerization [ Docker (I)]



CLOUD TRAIN
ACCELERATE YOUR GROWTH

# Agenda

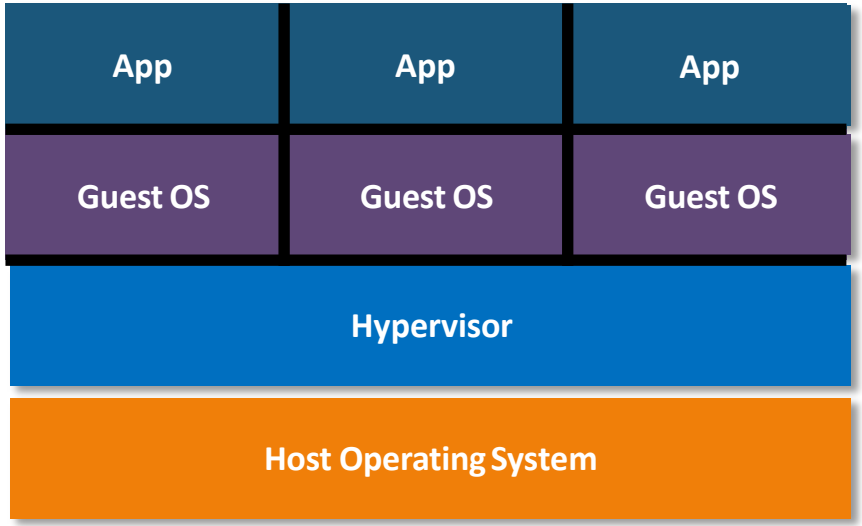


# *What is Virtualization?*

# What is Virtualization?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Virtualization is the process of running multiple virtual systems or resources on top of a single physical machine. These resources could be a storage device, network or even an operating system!



# Problems before Virtualization



Software A

Server A running on Ubuntu

CPU 10%

Imagine Software A running on Server A which has Ubuntu running on it. This software can only run in the Ubuntu environment.

# Problems before Virtualization

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Software A

Server A running on Ubuntu

CPU 10%

Software B

Server B running on Windows

CPU 10%

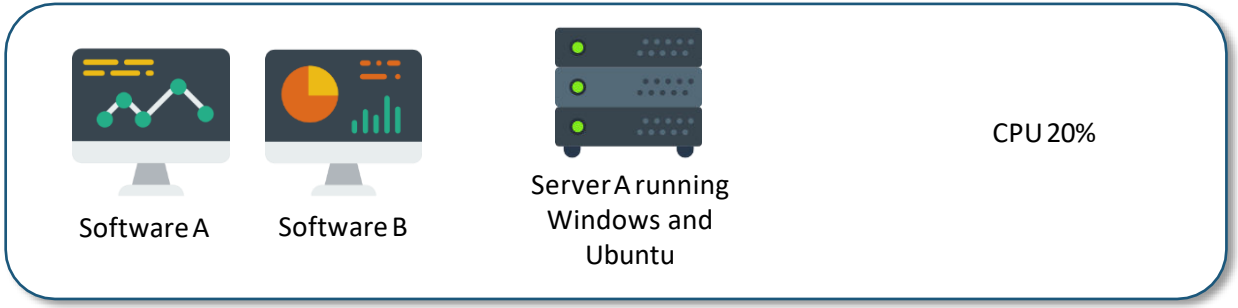
Some time later, we needed Software B which can only run on Windows. Therefore, we had to buy and run a Server B which had windows running on it. The software took only 10% of the CPU resources.

# Problems before Virtualization



- Buying servers was expensive.
- Resources were not being utilized at their full potential.
- The process of getting any software up and running was time consuming.
- Disaster recovery was difficult.

# After Virtualization




Windows and Ubuntu OS now are running on the same server in parallel using the Virtualization technology. This accounts for better CPU utilization and costsavings!

# Advantages of Virtualization



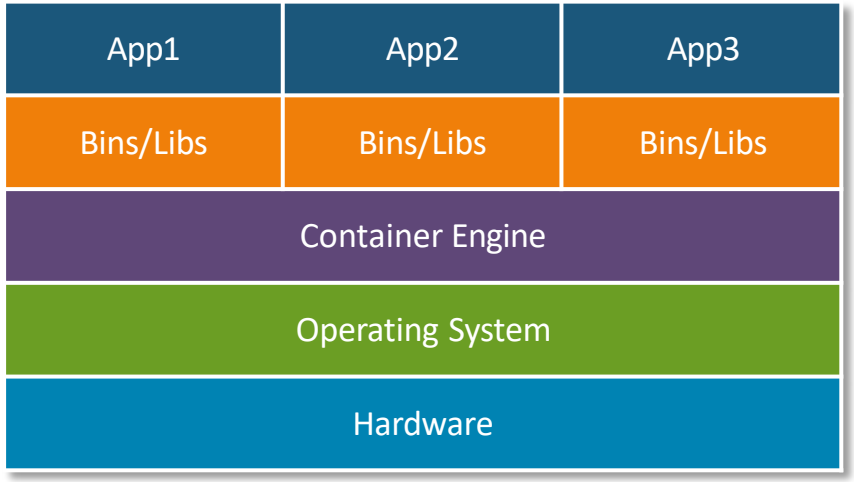
- It results in reduced spending.
- Resources are utilized more efficiently.
- Process of getting software up and running is shorter.
- Easier backup and disaster recovery is available.

# *What is Containerization?*

# What is Containerization?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Application **containerization** is an OS-level virtualization method used to deploy and run distributed applications without launching an entire virtual machine (VM) for each app.



# Problems before Containerization

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Developers when run the code on their system, it would run perfectly. But the same code would not run on the operations team's system.

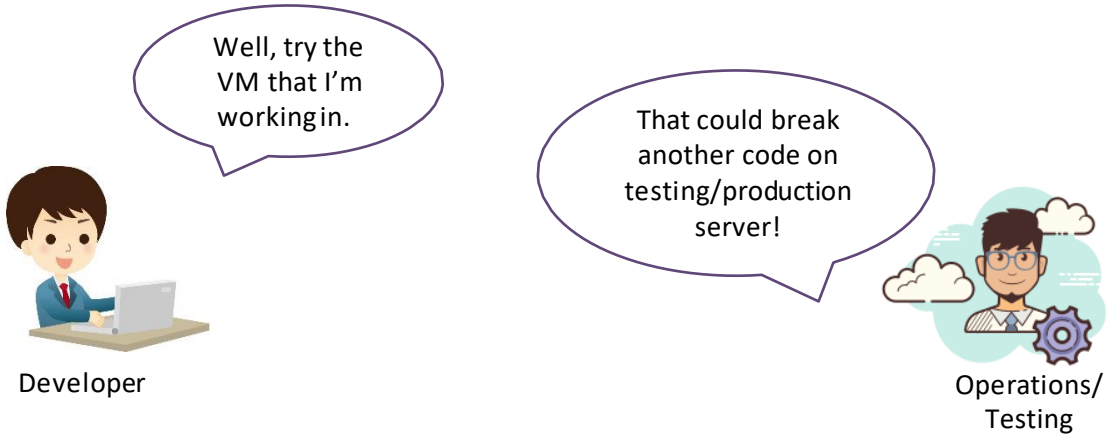

Developer

Operations/ Testing

# Problems before Containerization

CLOUD TRAIN
ACCELERATE YOUR GROWTH

The problem was with the environment the code was being run in. Well, a simple answer could be, why not give the same VM to the operations/testing team to run the code.



# Problems before Containerization

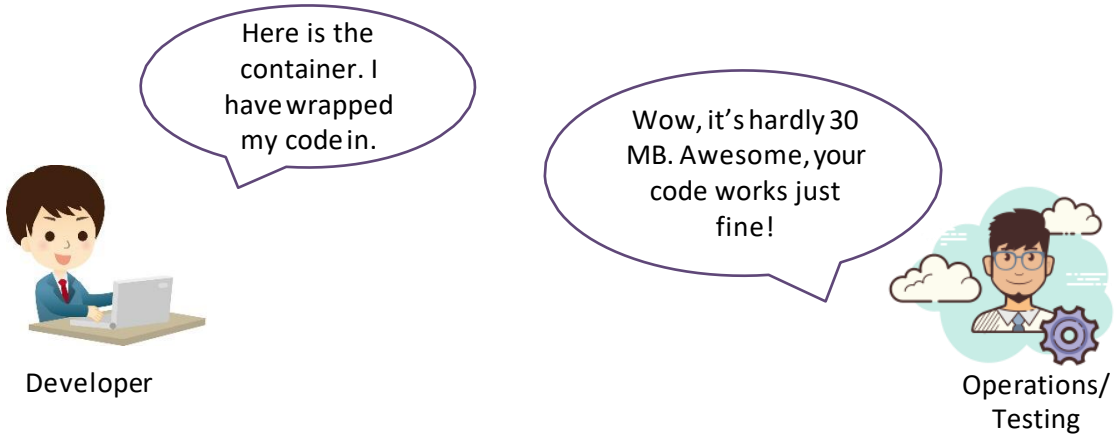
CLOUD TRAIN
ACCELERATE YOUR GROWTH

- VMs took too many resources to run.
- VMs were too big in size to be portable.
- VMs were not developer friendly.

# How did containers solve the problems?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

With containers, all the environment issues were solved. The developer could easily wrap their code in a lightweight container and pass it on to the operationsteam.



# Advantages of Containers



- Containers are not resource hungry.
- They are lightweight and hence portable.
- They are developer friendly and can be configured through the code.

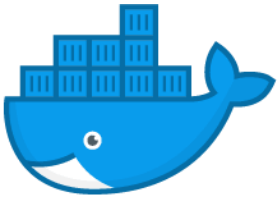
# Containerization Tools

# Containerization Tools

CLOUD TRAIN
ACCELERATE YOUR GROWTH

MESOS

rkt

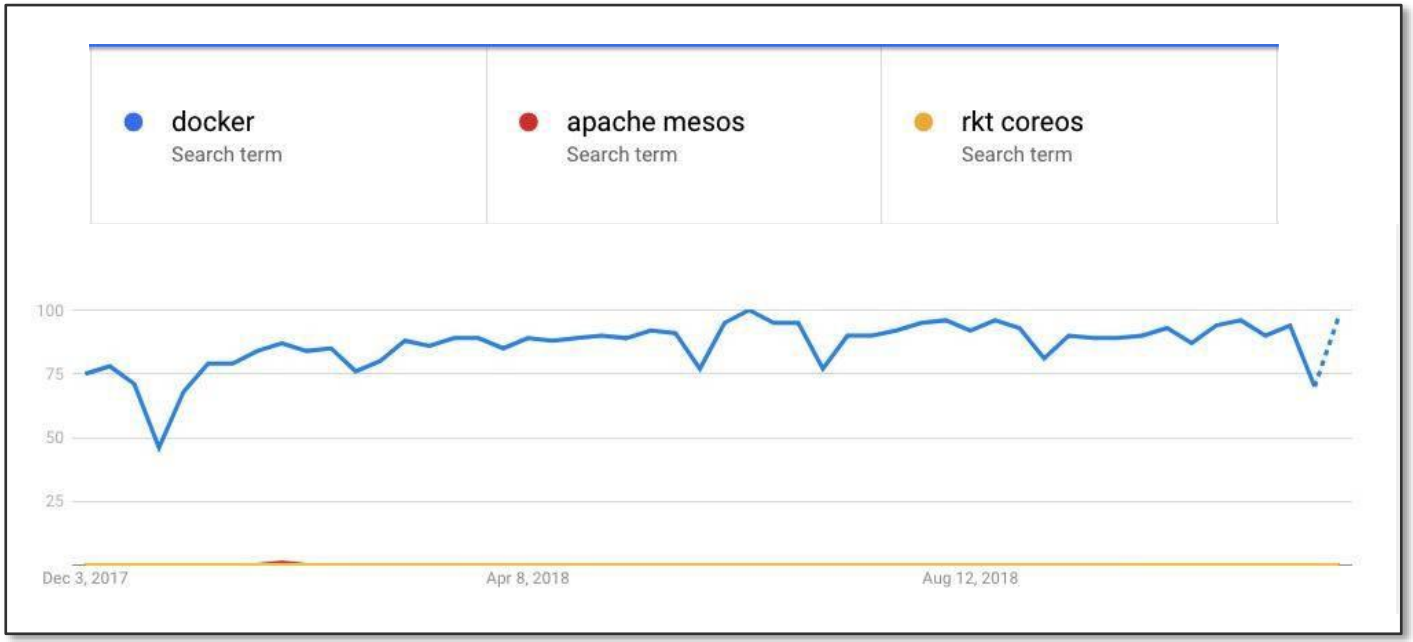


docker

# Containerization Tools

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Docker is clearly the most famous among them all!

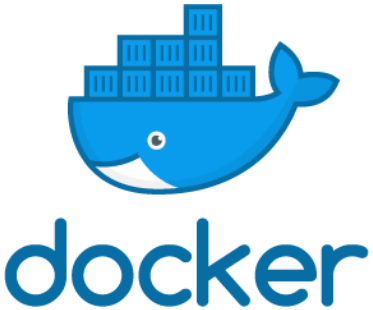


# *What is Docker?*

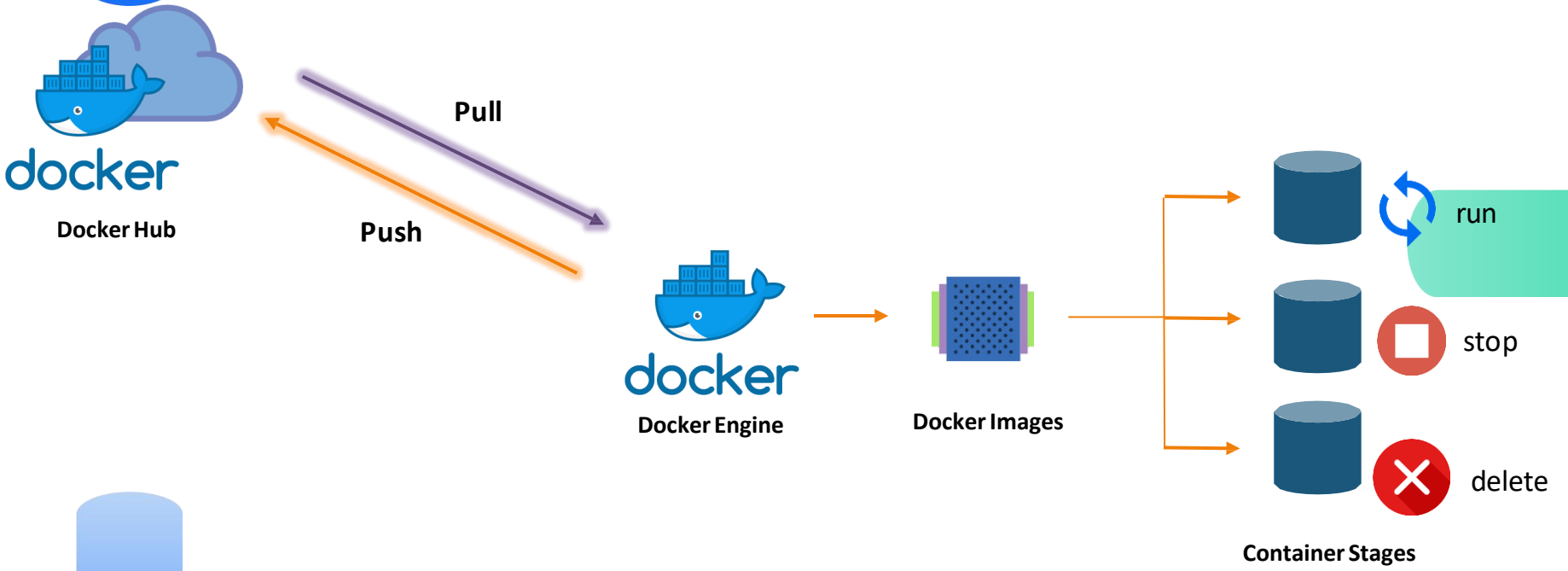
# What is Docker?



Docker is a computer program that performs operating-system-level virtualization, also known as "containerization". It was first released in 2013 and is developed by Docker, Inc. Docker is used to run software packages called "containers".



# Docker Container Life Cycle



# Components of Docker Ecosystem

# Components of Docker Ecosystem

CLOUD TRAIN
ACCELERATE YOUR GROWTH


Docker Hub


Docker Engine

Docker Images

Containers

Docker Volumes

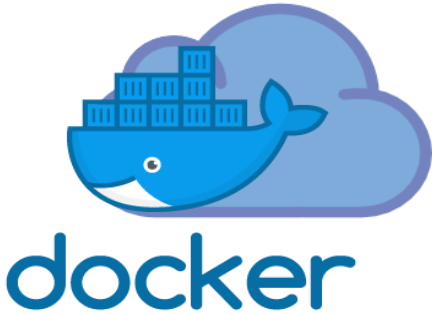
Docker File

# Components of Docker Ecosystem

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

- Docker Hub is a central public docker registry.
- Itcan store custom docker images.
- The service is free, but your images would be public.
- It requires username/password.

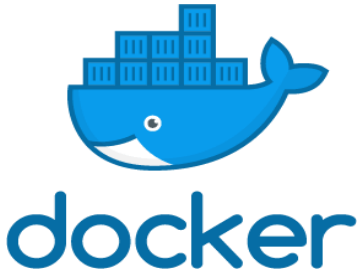


# Components of Docker Ecosystem



- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

- Docker Engine is the heart of the docker ecosystem.
- It is responsible for managing your container runtimes.
- It works on top of operating system level.
- It utilizes the kernel of the underlying OS.

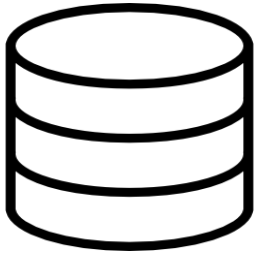


# Components of Docker Ecosystem

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

- Docker Image is like the template of a container.
- It is created in layers.
- Any new changes in the image results in creating a new layer.
- One can launch multiple containers from a single docker image.

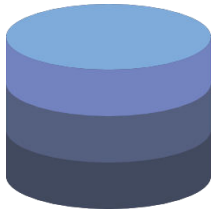


# Components of Docker Ecosystem

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

- A Docker Container is a lightweight software environment.
- It works on top of the underlying OS kernel.
- It is small in size and therefore is highly portable.
- It is created using the docker image.

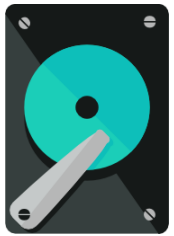


# Components of Docker Ecosystem



- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

- Docker Containers cannot persist data.
- To persist data in containers, we can use Docker Volume.
- A Docker Volume can connect to multiple containers simultaneously.
- If not created explicitly, a volume is automatically created when we create a container.



# Components of Docker Ecosystem



- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Dockerfile

- Dockerfile is a YAML file, which is used to create custom containers
- It can include commands that have to be run on the command line
- This Dockerfile can be used to build custom container images

# *Installing Docker*

# Common Docker Commands

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

`docker --version`

```
ubuntu@instance-1:~$ docker --version
Docker version 20.10.3, build 48d30b5
ubuntu@instance-1:~$
```

This command helps you know the installed version of the docker software on your system.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

```
docker pull <image-name>
```

```
ubuntu@instance-1:~$ docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
83ee3a23efb7: Pull complete
db98fc6f11f0: Pull complete
f611acd52c6c: Pull complete
Digest: sha256:703218c0465075f4425e58fac086e09e1de5c340b12976ab9eb8ad26615c3715
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

This command helps you pull images from the central docker repository.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**docker images**

```
ubuntu@instance-1:~$ docker images
REPOSITORY                                      TAG       IMAGE ID       CREATED         SIZE
ubuntu                                          latest    f63181f19b2f   3 weeks ago     72.9MB
```

This command helps you in listing all the docker images downloaded on your system.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

`docker run <image-name>`

```
ubuntu@instance-1:~$ docker run ubuntu
ubuntu@instance-1:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@instance-1:~$ docker run -ti -d ubuntu
1ebd5b33f06333f6ea9e95a7b6e0505fde6c3a19c5ea7041358846d16be43879
```

This command helps in running containers from their image name.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**docker ps**

```
ubuntu@instance-1:~$ docker ps
CONTAINER ID   IMAGE     COMMAND       CREATED         STATUS         PORTS     NAMES
1ebd5b33f063   ubuntu    "/bin/bash"   3 seconds ago   Up 2 seconds             sweet_kirch
```

This command helps in listing all the containers which are **running** in the system.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**docker ps -a**

```
ubuntu@instance-1:~$ docker ps -a
CONTAINER ID   IMAGE                                                COMMAND             CREATED             STATUS
                  PORTS     NAMES
1ebd5b33f063   ubuntu                                               "/bin/bash"         About a minute ago  Up About a mi
nute                       sweet_kirch
1990f3f37a78   ubuntu                                               "/bin/bash"         About a minute ago  Exited (0) Ab
out a minute ago           zen_lichterman
2e5be0570216   hello-world                                          "/hello"            3 minutes ago       Exited (0) 3
minutes ago                ecstatic_albattani
```

If there are any stopped containers, they can be seen by adding the **-a** flag in this command.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

```
docker exec <container-id>
```

```
ubuntu@instance-1:~$ docker exec -ti 1ebd5b33f063  bash
root@1ebd5b33f063:/# pwd
/
root@1ebd5b33f063:/# whoami
root
root@1ebd5b33f063:/#
```

For logging into/accessing the container, one can use the **exec** command.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

```
docker stop <container-id>
```

```
ubuntu@instance-1:~$ docker stop 1ebd5b33f063
1ebd5b33f063
```

For stopping a running container, we use the **stop** command.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

```
docker kill <container-id>
```

```
ubuntu@instance-1:~$ docker kill 57dd1c160a62
57dd1c160a62
ubuntu@instance-1:~$
```

This command kills the container by stopping its execution immediately.
The difference between **docker kill** and **docker stop**: 'docker stop' gives the container time to shutdown gracefully; whereas, in situations when it is taking too much time for getting the container to stop, one can opt to kill it.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

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

```
ubuntu@instance-1:~$ docker rm 57dd1c160a62
57dd1c160a62
ubuntu@instance-1:~$ docker ps -a
CONTAINER ID   IMAGE                                                                COMMAND             CREATED         STATUS
             PORTS     NAMES
1ebd5b33f063   ubuntu                                                               "/bin/bash"         4 minutes ago   Exited (0) About a
 minute ago            sweet_kirch
1990f3f37a78   ubuntu                                                               "/bin/bash"         5 minutes ago   Exited (0) 5 minut
es ago                 zen_lichterman
2e5be0570216   hello-world                                                          "/hello"            7 minutes ago   Exited (0) 7 minut
es ago                 ecstatic_albattani
```

To remove a stopped container from the system, we use the **rm** command.

# Common Docker Commands

CLOUD TRAIN
ACCELERATE YOUR GROWTH

```
docker rmi <image-id>
```

```
ubuntu@instance-1:~$ docker rmi ubuntu
Untagged: ubuntu:latest
Untagged: ubuntu@sha256:703218c0465075f4425e58fac086e09e1de5c340b12976ab9eb8ad26615c3715
Deleted: sha256:f63181f19b2fe819156dcb068b3b5bc036820bec7014c5f77277cfa341d4cb5e
Deleted: sha256:0770b7f116f8627ec336a62e65a1f79e344df7ae721eb3e06e11edca85d3d1e7
Deleted: sha256:476e931831a5b24b95ff7587cca09bde9d1d7c0329fbc44ac64793b28fb809d0
Deleted: sha256:9f32931c9d28f10104a8eb1330954ba90e76d92b02c5256521ba864feec14009
```

To remove an image from the system, we use the **rmi** command.

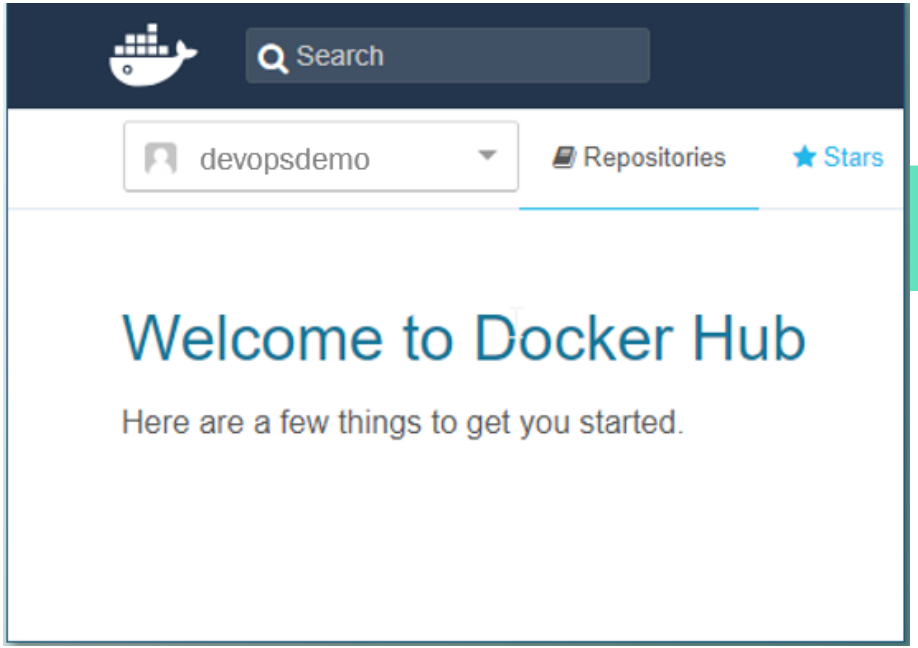
CLOUD TRAIN
ACCELERATE YOUR GROWTH

# Creating a Docker Hub Account

# Creating a Docker Hub Account

CLOUD TRAIN
ACCELERATE YOUR GROWTH

1. Navigate to https://hub.docker.com
2. Sign up on the website
3. Agree to the terms and conditions
4. Click on Sign up
5. Check your email, and verify your email by clicking on the link
6. Finally, login using the credentials you provided on the sign up page



# *Committing Changes to a Container*

# Committing Changes to a Docker Container



Let's try to accomplish the following example with a container and see how we can commit this container into an image.


Ubuntu Container

Install Apache server on this container

Ubuntu Container with Apache installed

# Committing Changes to a Docker Container



**1.** Pull the Docker Container using the command:

```
docker pull ubuntu
```

```
ubuntu@instance-1:~$ docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
83ee3a23efb7: Pull complete
db98fc6f11f0: Pull complete
f611acd52c6c: Pull complete
Digest: sha256:703218c0465075f4425e58fac086e09e1de5c340b12976ab9eb8ad26615c3715
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

**In our case, the image name is “ubuntu”.**

# Committing Changes to a Docker Container

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**2.** Run the container using the command:

```
docker run -it -d ubuntu
```

```
ubuntu@instance-1:~$ docker run -it -d ubuntu
c19e8f37f44b935804e5c06b3a2a59b8cdfbf7d73ac16e87efa792b24d4d542a
```

# Committing Changes to a Docker Container

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**3.** Access the container using the command:

```
docker exec -it <container-id> bash
```

```
ubuntu@instance-1:~$ docker run -it -d ubuntu
c19e8f37f44b935804e5c06b3a2a59b8cdfbf7d73ac16e87efa792b24d4d542a
```

# Committing Changes to a Docker Container

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**4.** Install Apache2 on this container using the following commands:

```
apt-get update
apt-get install apache2
```

```
root@c19e8f37f44b:/# apt-get install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils ca-certificates file krb5-locales libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap libasn1-8-heimdal libbrotli1 libcurl4 libexpat1
  libgdbm-compat4 libgdbm6 libgssapi-krb5-2 libgssapi3-heimdal libhcrypto4-heimdal libheimbase1-heimdal libheimntlm0-heimdal libhx509-5-heimdal libicu66 libjansson4 libk5crypto3
  libkeyutils1 libkrb5-26-heimdal libkrb5-3 libkrb5support0 libldap-2.4-2 libldap-common liblua5.2-0 libmagic-mgc libmagic1 libnghttp2-14 libperl5.30 libps15 libroken18-heimdal librtmp1
  libsasl2-2 libsasl2-modules libsasl2-modules-db libsqlite3-0 libssh-4 libssl1.1 libwind0-heimdal libxml2 mime-support netbase openssl perl perl-modules-5.30 publicsuffix ssl-cert tzdata
  xz-utils
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser ufw gdbm-l10n krb5-doc krb5-user libsasl2-modules-gssapi-mit | libsasl2-modules-gssapi-heimdal
  libsasl2-modules-ldap libsasl2-modules-otp libsasl2-modules-sql perl-doc libterm-readline-gnu-perl | libterm-readline-perl-perl make libb-debug-perl liblocale-codes-perl
  openssl-blacklist
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils ca-certificates file krb5-locales libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap libasn1-8-heimdal libbrotli1 libcurl4
  libexpat1 libgdbm-compat4 libgdbm6 libgssapi-krb5-2 libgssapi3-heimdal libhcrypto4-heimdal libheimbase1-heimdal libheimntlm0-heimdal libhx509-5-heimdal libicu66 libjansson4 libk5crypto3
  libkeyutils1 libkrb5-26-heimdal libkrb5-3 libkrb5support0 libldap-2.4-2 libldap-common liblua5.2-0 libmagic-mgc libmagic1 libnghttp2-14 libperl5.30 libps15 libroken18-heimdal librtmp1
  libsasl2-2 libsasl2-modules libsasl2-modules-db libsqlite3-0 libssh-4 libssl1.1 libwind0-heimdal libxml2 mime-support netbase openssl perl perl-modules-5.30 publicsuffix ssl-cert tzdata
  xz-utils
0 upgraded, 57 newly installed, 0 to remove and 5 not upgraded.
Need to get 24.1 MB of archives.
After this operation, 117 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

# Committing Changes to a Docker Container

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**5.** Exit the container and save it using this command. The saved container will be converted into an image with the name specified.

```
docker commit <container-id> <username>/<container-name>
```

```
ubuntu@instance-1:~$ docker commit c19e8f37f44b [redacted]/ubuntu
sha256:ee09cca4f1d161d3e255b1bea28739fe2771951e4dee195d9a8a753bf6336c8e
ubuntu@instance-1:~$ docker images
REPOSITORY                TAG       IMAGE ID       CREATED          SIZE
[redacted]/ubuntu         latest    ee09cca4f1d1   5 seconds ago    213MB
demo/ubuntu               latest    532be5fb53c7   2 minutes ago    213MB
ubuntu                    latest    f63181f19b2f   3 weeks ago      72.9MB
```

The **username** has to match with the username you created on DockerHub.
The **container-name** can be anything.

CLOUD TRAIN
ACCELERATE YOUR GROWTH

# *Pushing the Container on DockerHub*

# Pushing the Container on DockerHub

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**1.** The first step is to login. It can be done using the followingcommand:

```
docker login
```

```
ubuntu@instance-1:~$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: [redacted]
Password:
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

# Pushing the Container on DockerHub



**2.** For pushing your container on DockerHub, use the following command:

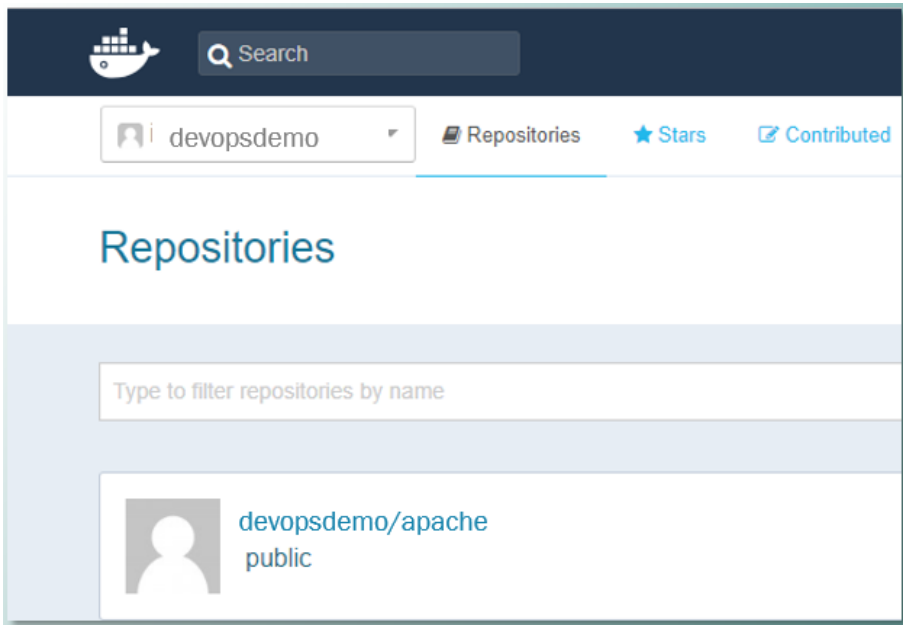
```
docker push <username>/<container-id>
```

```
ubuntu@ip-172-31-26-120:~$ docker push devopsdemo/apache
The push refers to repository [docker.io/devopsdemo/apache]
7a1d3c7d7a50: Pushed
b9b7103af585: Mounted from library/ubuntu
ca2991e4676c: Mounted from library/ubuntu
a768c3f3878e: Mounted from library/ubuntu
bc7f4b25d0ae: Mounted from library/ubuntu
latest: digest: sha256:4c21181c6db3695dd2c509fb778e8d851a51e26afe1b6f9cc2b434ea4
81b7263 size: 1362
ubuntu@ip-172-31-26-120:~$
```

# Pushing the Container on DockerHub

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**3.** You can verify the push on DockerHub.




Now anyone, who wants to download this container, can simply pass the following command:

**docker pull demo/apache**

# Private Registry for Docker

# Private Registry for Docker

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- DockerHub is a publicly available Docker Registry
- You may want to create a Private Registry for your company or personal use
- The registry is available on DockerHub, as a container named ‘registry’

CLOUD TRAIN
ACCELERATE YOUR GROWTH

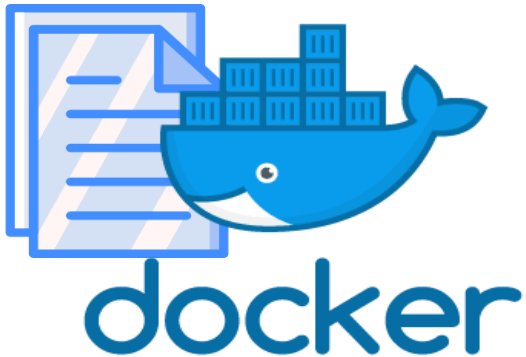
# *Hands-on: Creating a Private Registry in Docker*

# *Introduction to Dockerfile*

# Introduction To Dockerfile



A **Dockerfile** is a text document that contains all the commands a user could call on the command line to assemble an image. Using the **docker** build, users can create an automated build that executes several command-line instructions in succession.



# Various Commands in Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- **FROM**
- ADD
- RUN
- CMD
- ENTRYPOINT
- ENV

The **FROM** keyword is used to define the base image, on which we will be building.

Example

```
FROM ubuntu
```

Dockerfile

# Various Commands in Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- FROM
- ADD
- RUN
- CMD
- ENTRYPOINT
- ENV

The **ADD** keyword is used to add files to the container being built. The syntax used is:
**ADD <source> <destination in container>**

Example

```
FROM ubuntu
ADD .  /var/www/html
```

Dockerfile

# Various Commands in Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- FROM
- ADD
- **RUN**
- CMD
- ENTRYPOINT
- ENV

The **RUN** keyword is used to add layers to the base image, by installing components. Each RUN statement adds a new layer to the docker image.

Example

```
FROM ubuntu
RUN apt-get update
RUN apt-get -y install apache2
ADD . /var/www/html
```

Dockerfile

# Various Commands in Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- FROM
- ADD
- RUN
- **CMD**
- ENTRYPOINT
- ENV

The **CMD** keyword is used to run commands on the start of the container. These commands run only when there is no argument specified while running the container.

Example

```
FROM ubuntu
RUN apt-get update
RUN apt-get -y install apache2
ADD . /var/www/html
CMD apachectl –D FOREGROUND
```

Dockerfile

# Various Commands in Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- FROM
- ADD
- RUN
- CMD
- **ENTRYPOINT**
- ENV

The **ENTRYPOINT** keyword is used strictly to run commands the moment the container initializes. The difference between CMD and ENTRYPOINT: ENTRYPOINT will run irrespective of the fact whether the argument is specified or not.

Example

```
FROM ubuntu
RUN apt-get update
RUN apt-get -y install apache2
ADD . /var/www/html
ENTRYPOINT apachectl –D FOREGROUND
```

Dockerfile

# Various Commands in Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- FROM
- ADD
- RUN
- CMD
- ENTRYPOINT
- ENV

The **ENV** keyword is used to define environment variables in the container runtime.

Example

```
FROM ubuntu
RUN apt-get update
RUN apt-get -y install apache2
ADD . /var/www/html
ENTRYPOINT apachectl –D FOREGROUND
ENV name Devops Tutorial
```

Dockerfile

# *Running the Sample Dockerfile*

# Running the Sample Dockerfile



Let’s see how we can run this sample Dockerfile now.

Example

```
FROM ubuntu
RUN apt-get update
RUN apt-get -y install apache2
ADD . /var/www/html
ENTRYPOINT apachectl -DFOREGROUND
ENV name Devops Tutorial
```

Dockerfile

# Running the Sample Dockerfile



**1.** First, create a folder docker in the home directory.

```
ubuntu@instance-1:~$ mkdir docker
ubuntu@instance-1:~$ cd docker
ubuntu@instance-1:~/docker$
```

# Running the Sample Dockerfile



**2.** Enter into this directory and create a file called ‘Dockerfile’, with the same contents as the sample Dockerfile.

```
FROM ubuntu
ENV TZ=Asia/Kolkata
RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone
RUN apt-get update
RUN apt-get -y install apache2
ADD .  /var/www/html
ENTRYPOINT apachectl -D FOREGROUND
ENV name Devops Tutorial
```

# Running the Sample Dockerfile



**3.** Create one more file called ‘index.html’ with the following contents.

```
<html>
<title> Sample Website </title>
<body>
Hello World
</body>
</html>
```

# Running the Sample Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**4.** Now, pass the following command:

**docker build <directory-of-dockerfile> -t <name of container>**

```
ubuntu@instance-1:~/docker$ docker build . -t ubuntu_apache
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM ubuntu
 ---> f63181f19b2f
Step 2/6 : RUN apt-get update
 ---> Running in fb6247b73aa5
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [109 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [664 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [612 kB]
```

# Running the Sample Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**5.** Finally, run this built image, using the following command:

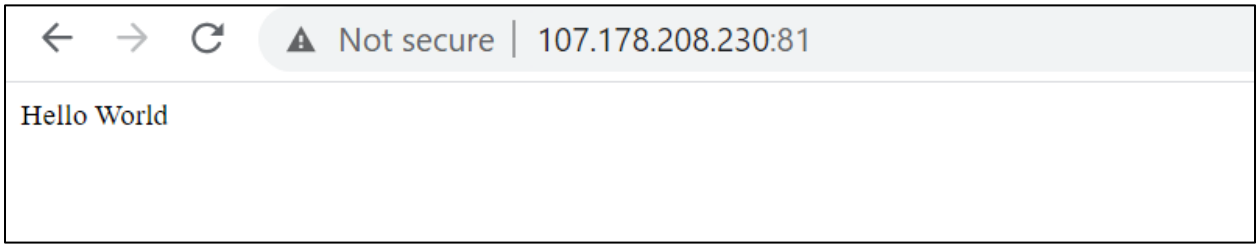
**docker run –it –p 81:80 –d demo/custom**

```
ubuntu@instance-1:~/docker$ docker run -it -p 81:80 -d ubuntu_apache
d086862179964e1d07b5ccfa34010766da740285241fc7528f00bad5791b6c79
ubuntu@instance-1:~/docker$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                  NAMES
d08686217996   ubuntu_apache  "/bin/sh -c 'apachec…"   6 seconds ago    Up 4 seconds    0.0.0.0:81->80/tcp     unruffled_wozniak
```

# Running the Sample Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**6.** Now, navigate to the server IP address on port 81.



# Running the Sample Dockerfile

CLOUD TRAIN
ACCELERATE YOUR GROWTH

**7.** Finally, login into the container and check the variable $name. It  will have the same value as given in the Dockerfile.

```
ubuntu@instance-1:~/docker$ docker exec -ti d08686217996   bash
root@d08686217996:/# echo $name
Devops Tutorial
root@d08686217996:/#
```

CLOUD TRAIN
ACCELERATE YOUR GROWTH

# Got queries or need more info?

## Contact us

TO ACCELERATE YOUR CAREER GROWTH

For questions and more details:

please call @ +91 98712 72900 or

visit https://www.thecloudtrain.com/ or

email at join@thecloudtrain.com or

WhatsApp us >>