--

Academic Year: 2025-26 **Semester: V Class / Branch: TE IT**
Subject: DevOPs Lab (DL)
Subject Lab In-charge: Prof. Sujata Oak **Name : Siddhi Tangsali**
Student Id : 23104069 **Date of submission: 30/7/25**

EXPERIMENT NO. 08

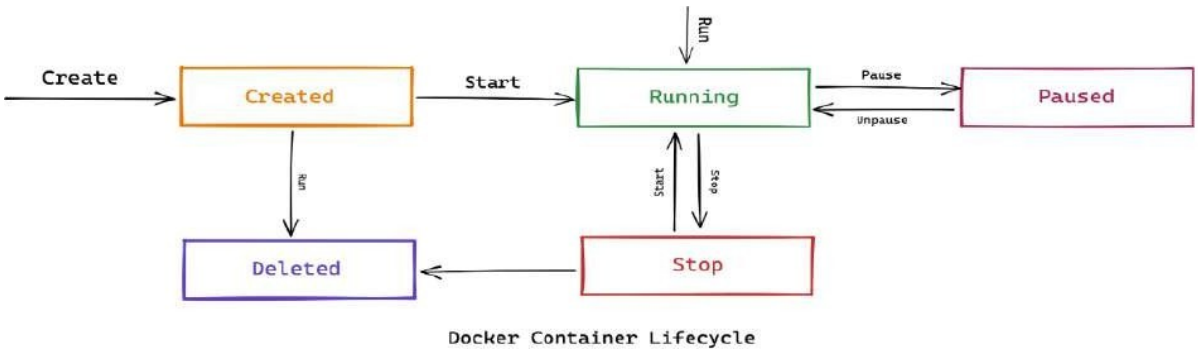
Aim: To demonstrate container lifecycle using various docker commands.

Theory:

Docker is an open-source platform that helps developers **build**, **ship**, and **run** applications in **containers**. A **container** is a lightweight, standalone executable package that includes everything needed to run a piece of software — **code, runtime, libraries, and dependencies**.

There are mainly five states that a container can be in during its lifecycle -

- Created state
- Running state
- Paused state/ Unpaused state
- Stopped state
- Killed/Deleted state



Docker Container Lifecycle

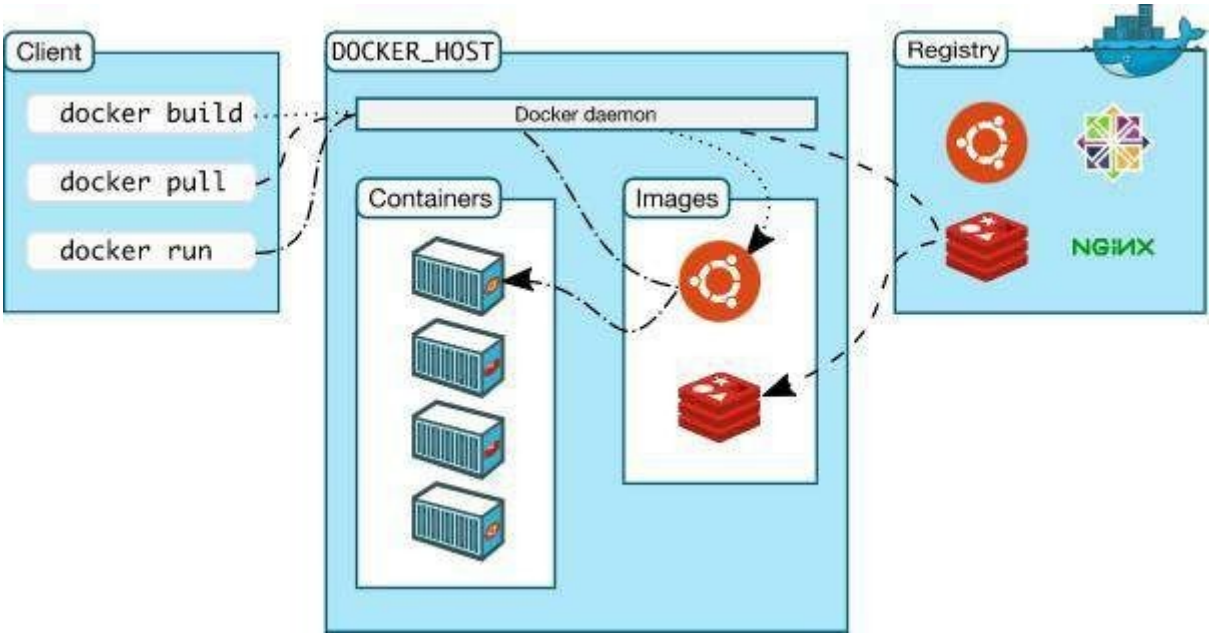


Fig. Architectural overview of Docker

Command	Description
docker --version	Check Docker version installed
docker info	Shows detailed info about Docker system
docker login	Login to Docker Hub registry
docker pull <image>	Download an image from Docker Hub (e.g., docker pull nginx)
docker images	List all locally available Docker images
docker rmi <image>	Remove a Docker image from local system
docker build -t <name> .	Build an image from a Dockerfile (-t tags the image)
docker run <image>	Create and start a container from an image
docker run -it <image>	Run container in interactive mode with terminal
docker run -d <image>	Run container in detached mode (background)
docker run -p 8080:80 <image>	Map port 80 in container to port 8080 on host
docker ps	List running containers
docker ps -a	List all containers (running + stopped)
docker stop <container>	Stop a running container
docker start <container>	Start a stopped container
docker restart <container>	Restart a container
docker rm <container>	Remove a stopped
container docker logs <container>	View logs of a container
docker exec -it <container> bash	Execute command inside a running container (open terminal)

docker inspect <container/image> Get low-level info (JSON) about a container or image
docker top <container> Display processes running inside a container

Check the version of OS: # cat /etc/os-release

Step1: Install Docker using the convenience script
Take sudo privileges.

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

```
devasc@labvm:~/Desktop/DOCKER_LAB$ sudo su
root@labvm:/home/devasc/Desktop/DOCKER_LAB# curl -fsSL https://get.docker.com -o get-do
cker.sh
```

 # sudo sh get-docker.sh

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# sudo sh get-docker.sh
# Executing docker install script, commit: 0d6f72e671ba87f7aa4c6991646a1a5b9f9dae84
Warning: the "docker" command appears to already exist on this system.

If you already have Docker installed, this script can cause trouble, which is
why we're displaying this warning and provide the opportunity to cancel the
installation.

If you installed the current Docker package using this script and are using it
again to update Docker, you can safely ignore this message.

You may press Ctrl+C now to abort this script.
```

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# ls
get-docker.sh
```

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# cat get-docker.sh
```

Step2: To verify docker is installed or not:

 # docker version

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Documents/git-SiddhiT$ sudo su
[sudo] password for apsit:
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker --
version
Docker version 26.1.3, build 26.1.3-0ubuntu1~22.04.1
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT#
```

DOCKER COMMANDS:

1] #docker info: Give details about docker

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker i
Client:
 Version:    26.1.3
 Context:    default
 Debug Mode: false

Server:
 Containers: 1
  Running: 0
  Paused: 0
  Stopped: 1
 Images: 3
 Server Version: 26.1.3
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Using metacopy: false
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: systemd
 Cgroup Version: 2
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local splunk syslog
 Swarm: inactive
 Runtimes: io.containerd.runc.v2 runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version:
 runc version:
 init version:
 Security Options:
  apparmor
  seccomp
   Profile: builtin
  cgroupns
```

2] #systemctl status docker: To Check docker engine is active or not
#systemctl start docker: To start the docker engine
#systemctl stop docker : To stop the docker engine

```
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset>
     Active: active (running) since Wed 2025-07-30 10:09:50 IST; 46min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2092 (dockerd)
      Tasks: 17
     Memory: 117.1M
        CPU: 684ms
     CGroup: /system.slice/docker.service
             └─2092 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/cont>

Jul 30 10:09:40 apsit-HP-ProDesk-600-G4-PCI-MT dockerd[2092]: time="2025-07-30T>
Jul 30 10:09:40 apsit-HP-ProDesk-600-G4-PCI-MT dockerd[2092]: time="2025-07-30T>
Jul 30 10:09:44 apsit-HP-ProDesk-600-G4-PCI-MT dockerd[2092]: time="2025-07-30T>
Jul 30 10:09:45 apsit-HP-ProDesk-600-G4-PCI-MT dockerd[2092]: time="2025-07-30T>
Jul 30 10:09:47 apsit-HP-ProDesk-600-G4-PCI-MT dockerd[2092]: time="2025-07-30T>
Jul 30 10:09:48 apsit-HP-ProDesk-600-G4-PCI-MT dockerd[2092]: time="2025-07-30T>
Jul 30 10:09:50 apsit-HP-ProDesk-600-G4-PCI-MT dockerd[2092]: time="2025-07-30T>
Jul 30 10:09:50 apsit-HP-ProDesk-600-G4-PCI-MT dockerd[2092]: time="2025-07-30T>
Jul 30 10:09:50 apsit-HP-ProDesk-600-G4-PCI-MT systemd[1]: Started Docker Appli>
Jul 30 10:09:50 apsit-HP-ProDesk-600-G4-PCI-MT dockerd[2092]: time="2025-07-30T>
~
lines 1-22/22 (END)
```

3] How you login into your Docker Hub Account from CLI? #docker login

4]

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker login -u siddheee
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT#
```

docker run: It helps you to run a container on top of your docker engine, but the ingredients that it needs is image name.

 # docker container run ubuntu cat /etc/os-release

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker co
ntainer run ubuntu
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker co
ntainer run ubuntu cat /etc/os-release
PRETTY_NAME="Ubuntu 24.04.1 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.1 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-poli
cy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT#
```

5] #docker images : Lists all images

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker im
ages
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
nginx        latest    53a18edff809   5 months ago   192MB
```

6] # docker container run ubuntu cat /etc/os-release #Again Hit the same command.

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker container run ubuntu cat /etc/os-release
PRETTY_NAME="Ubuntu 24.04.1 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.1 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
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker images
```

7] To list docker images: #docker images

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker images
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
nginx        latest    53a18edff809   5 months ago   192MB
ubuntu       latest    a04dc4851cbc   6 months ago   78.1MB
mongo        3.4       f76f959b2a49   5 years ago    431MB
```

8] To List Container: #docker container ls

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

Here the container started and exited at same time.

9] To check container have exited or NOT

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS                     PORTS     NAMES
36eb967be397   ubuntu    "cat /etc/os-release"    2 minutes ago   Exited (0) 2 minutes ago             cranky_galois
8320e56c7771   ubuntu    "/bin/bash"              4 minutes ago   Exited (0) 4 minutes ago             nervous_nash
d3a7bf9ebe10   ubuntu    "top"                    3 months ago    Exited (0) 3 months ago              keen_vaughan
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT#
```

Open another terminal:

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# ^C
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker container ls -a
CONTAINER ID   IMAGE     COMMAND       CREATED          STATUS                      PORTS     NAMES
64d6a04b1ebd   ubuntu    "sleep 60"    25 minutes ago   Exited (0) 24 minutes ago             jovial_bell
8320e56c7771   ubuntu    "/bin/bash"   30 minutes ago   Exited (0) 30 minutes ago             nervous_nash
d3a7bf9ebe10   ubuntu    "top"         3 months ago     Exited (0) 3 months ago               keen_vaughan
```

#docker run hello-world

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:53cc4d415d839c98be39331c948609b659ed725170ad2ca8eb36951288f81b75
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.
```

The command used to access the running container Is:
You Can run a ubuntu container with following command:
#docker run -it ubuntu bash

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker run -it ubuntu bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
31e907dcc94a: Pull complete
Digest: sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee
Status: Downloaded newer image for ubuntu:latest
root@1d1a286d11ea:/#
```

3] How to see the image that I just downloaded whether it is available on my machine or not?

docker images: This command is used to show all the pulled images from docker

 # docker images

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
ubuntu        latest    edbfe74c41f8   3 weeks ago      78.1MB
mysql         latest    a82a8f162e18   4 weeks ago      586MB
hello-world   latest    d2c94e258dcb   16 months ago    13.3kB
```

Try to launch a docker image for testing purpose, you can find the images in docker public repository at https://hub.docker.com

4] docker pull: This command is used to pull images from the docker repository(hub.docker.com)
Usage: docker pull <image name>

#docker pull mysql

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
6e839ac3722d: Pull complete
ad912193ad5f: Pull complete
25d13d87fd8d: Pull complete
004d383c75ef: Pull complete
6d9bbc82a0b8: Pull complete
81fec07ea550: Pull complete
83357cb2d3a5: Pull complete
8ffe968b82c1: Pull complete
30dfd9a7ed57: Pull complete
35844ae33cbe: Pull complete
Digest: sha256:86cdfe832c81e39a89cfb63c3fde1683c41cc00ef91e67653c9c1df0ba80f454
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
```

OR

#docker container run nginx

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker container run nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
59e22667830b: Pull complete
140da4f89dcb: Pull complete
96e47e70491e: Pull complete
2ef442a3816e: Pull complete
4b1e45a9989f: Pull complete
1d9f51194194: Pull complete
f30ffbee4c54: Pull complete
Digest: sha256:84ec966e61a8c7846f509da7eb081c55c1d56817448728924a87ab32f12a72fb
Status: Downloaded newer image for nginx:latest
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform conf
iguration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/defaul
t.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/defa
ult.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
```

Open another terminal:
#docker container run nginx

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker container ls -a
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS
                  PORTS                                     NAMES
3739e1fd10a8   nginx          "/docker-entrypoint.…"   34 seconds ago   Up 31 se
conds             80/tcp                                    pedantic_kirch
baad61dfa5c7   ubuntu         "sleep 60"               8 minutes ago    Exited (
0) 7 minutes ago                                         stupefied_kepler
101cc947a3e8   ubuntu         "sleep 60"               12 minutes ago   Exited (
0) 11 minutes ago                                        fervent_pike
a1a3a06f8b5e   ubuntu         "cat /etc/os-release…"   13 minutes ago   Exited (
1) 12 minutes ago                                        awesome_jones
530ef56a0a90   ubuntu         "cat /etc/os-release"    26 minutes ago   Exited (
0) 26 minutes ago                                        sharp_perlman
499fb530aa8e   ubuntu         "cat /etc/os-release"    29 minutes ago   Exited (
0) 29 minutes ago                                        boring_herschel
81cf4991575a   59ab366372d5   "cat /etc/os-release"    38 minutes ago   Exited (
0) 38 minutes ago                                        silly_hodgkin
41aa6f73a6bd   f1e530d55abc   "nginx -g 'daemon of…"   9 months ago     Exited (
255) 6 months ago   0.0.0.0:4032->80/tcp, :::4032->80/tcp   ssjcoeserver
659dd47038a0   d2c94e258dcb   "/hello"                 9 months ago     Exited (
0) 9 months ago                                          gifted_visvesvaraya
bc8a1f0b194f   d2c94e258dcb   "/hello"                 11 months ago    Exited (
0) 11 months ago                                         peaceful_aryabhata
fbd148039aee   d2c94e258dcb   "/hello"                 11 months ago    Exited (
```

Ctrl C to come out of the current screen and again check whether nginx image container is exited.

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker container ls -a
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS
                  PORTS                                     NAMES
3739e1fd10a8   nginx          "/docker-entrypoint.…"   4 minutes ago    Exited (
0) 13 seconds ago                                        pedantic_kirch
baad61dfa5c7   ubuntu         "sleep 60"               12 minutes ago   Exited (
```

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker images
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
nginx        latest    2cd1d97f893f   13 days ago    192MB
ubuntu       latest    65ae7a6f3544   2 weeks ago    78.1MB
root@labvm:/home/devasc/Desktop/DOCKER_LAB#
```

5] **docker ps : This** command lists the running containers on my system

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

6] docker ps -a : This command list all containers running or exited from the system.

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker ps -a
CONTAINER ID   IMAGE          COMMAND     CREATED          STATUS
     PORTS     NAMES
1d1a286d11ea   ubuntu         "bash"      12 minutes ago   Exited (0) 8 minutes ag
o              sleepy_cerf
fbd148039aee   hello-world    "/hello"    22 minutes ago   Exited (0) 22 minutes a
go             infallible_cerf
```

Now to get the container running : #docker run -it ubuntu bash

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker run -it ubuntu bash
root@a0baee026da8:/#
```

In New terminal: docker ps -a

```
devasc@labvm:~/Desktop/DOCKER_LAB$ docker ps -a
CONTAINER ID   IMAGE          COMMAND     CREATED          STATUS
     PORTS      NAMES
a0baee026da8   ubuntu         "bash"      39 seconds ago   Up 39 seconds
                friendly_easley
1d1a286d11ea   ubuntu         "bash"      18 minutes ago   Exited (0) 13 minutes a
go             sleepy_cerf
fbd148039aee   hello-world    "/hello"    27 minutes ago   Exited (0) 27 minutes a
go             infallible_cerf
```

In First Terminal: ls

You will see the lists of directories available in ubuntu container

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker run -it ubuntu bash
root@19151f20756a:/#
root@19151f20756a:/# ls
bin   dev   home   lib64  mnt   proc  run   srv  tmp  var
boot  etc   lib    media  opt   root  sbin  sys  usr
```

Now get exit from ubuntu container

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker run -it ubuntu bash
root@a0baee026da8:/# exit
exit
```

In New terminal: docker ps -a

So You see the container has exited 3 minutes ago.

```
devasc@labvm:~/Desktop/DOCKER_LAB$ docker ps -a
CONTAINER ID   IMAGE          COMMAND     CREATED          STATUS
     PORTS      NAMES
a0baee026da8   ubuntu         "bash"      3 minutes ago    Exited (0) 6 seconds ag
o              friendly_easley
```

NOTE: Every container created has a unique container id. That is, from a single image multiple containers can be created. Also, every container will be independent of itself, will be isolated from other container

NOTE: If you don't provide name to your container , the docker-engine gives fancy name to your container.

7] docker exec: This command is used to executes the container.

Usage: docker exec -it <container id> bash

8] To delete the container: #docker rm <container-name/container-id>

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker rm 191
191
```

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker rm a0b 1d1
a0b
1d1
```

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED          STATUS
    PORTS     NAMES
fbd148039aee   hello-world   "/hello"   48 minutes ago   Exited (0) 48 minutes a
go            infallible_cerf
```

9] To delete the image: #docker rmi <image-name/image-id>

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
ubuntu        latest    edbfe74c41f8   3 weeks ago      78.1MB
mysql         latest    a82a8f162e18   4 weeks ago      586MB
hello-world   latest    d2c94e258dcb   16 months ago    13.3kB
```

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker rmi edb
Untagged: ubuntu:latest
Untagged: ubuntu@sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34
a9ab63ee
Deleted: sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a
Deleted: sha256:f36fd4bb7334b7ae3321e3229d103c4a3e7c10a263379cc6a058b977edfb46de
```

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
mysql         latest    a82a8f162e18   4 weeks ago      586MB
hello-world   latest    d2c94e258dcb   16 months ago    13.3kB
```

10] Commands related to containers:

docker start <container-id> : Start the stop container
docker stop <container-id> : Stop the running container
docker pause <container-id> : Pause the processes in running container
docker kill <container id> : Kill the container.

Task 2: HOW TO SETUP AND CONFIGURE MYSQL DATABASE INSIDE DOCKER CONTAINER?

MySQL is the single most popular relational database tool.
MySQL is popular because it is simple yet powerful. Here are its best features:

- **Relational: follows the relational model and uses SQL to manage databases.**
- **Open-source (GNU license): the community loves it. Companies love it.**
- **Scalable: can handle applications from small-sized to enterprise-level.**
- **Secure: offers user authentication, access management, and encryption.**
- **High-performance: known for its speed and efficiency in handling complex queries and large volumes of data.**
- **Replication and backup: it has options for data replication and backup, allowing for disaster recovery strategies.**

Step 1: To pull the image of mysql from docker hub

#docker pull mysql:latest

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
3ea90c3a51a3: Pull complete
afaae28bbee0: Pull complete
2ba13c6c03c3: Pull complete
31c5b5c14efc: Pull complete
938fdecc6487: Pull complete
19f77cf53703: Pull complete
6c4bb10a796d: Pull complete
468bb678f509: Pull complete
90fd4b47993e: Pull complete
6dfd9f336bec: Pull complete
Digest: sha256:082063dca94535c76b91c6ef9b9f6748810e546887af5e8574a8cfa11eace184
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
```

EXPLANATION: The code is a Docker command, not SQL.
- It's used to download the latest version of the MySQL Docker image.
- "docker pull" is a command that tells Docker to download an image from Docker Hub.
- "mysql:latest" specifies the image to download.
- "mysql" is the name of the image and "latest" is the tag.

Step 2: List the mysql images

#docker images

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker images
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
mysql        latest    fe7f726d39a6   7 days ago     921MB
nginx        latest    53a18edff809   5 months ago   192MB
ubuntu       latest    a04dc4851cbc   6 months ago   78.1MB
mongo        3.4       f76f959b2a49   5 years ago    431MB
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT#
```

Docker images are blueprints for building containers. Just like a blueprint allows you to build a house, a Docker image contains all the necessary instructions and components to create a running instance of an application or service.

STEP 3: Running and Managing a MySQL Server Container

Now, let's create our first container from the mysql image. Here is the command we will use:

$ docker run --name test-mysql -e MYSQL_ROOT_PASSWORD=strong_password -d mysql

EXPLANATION:

- ✓ **run: creates a new container or starts an existing one**
- ✓ **--name CONTAINER_NAME: gives the container a name. The name should be readable and short. In our case, the name is test-mysql.**
- ✓ **-e ENV_VARIABLE=value: the -e tag creates an environment variable that will be accessible within the container. It is crucial to set MYSQL_ROOT_PASSWORD so that we can run SQL commands later**

from the container. Make sure to store your strong password somewhere safe (not your brain).

- ✓ -d: short for detached, the -d tag makes the container run in the background. If you remove this tag, the command will keep printing logs until the container stops.
- ✓ image_name: the final argument is the image name the container will be built from. In this case, our image is mysql.
- ✓

If the command returns a long string of gibberish (the container ID), it means the container has started. You can check its status with docker ps:

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker run --name test-siddhi -e MYSQL_ROOT_PASSWORD=siddhi@05 -d mysql
e184b46df0a6ec04304b0061f4f21ef7a8029053b9e0027cf70d66a739bdfa3a
```

In New Terminal:

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                      PORTS                 NAMES
e184b46df0a6   mysql     "docker-entrypoint.s…"   17 seconds ago   Up 15 seconds               3306/tcp, 33060/tcp   test-siddhi
e70db86213b7   ubuntu    "/bin/bash"              10 minutes ago   Exited (0) 9 minutes ago                          exciting_wing
93de985e318b   ubuntu    "sleep 30"               12 minutes ago   Exited (0) 12 minutes ago                         cool_meitner
64d6a04b1ebd   ubuntu    "sleep 60"               40 minutes ago   Exited (0) 39 minutes ago                         jovial_bell
8320e56c7771   ubuntu    "/bin/bash"              46 minutes ago   Exited (0) 46 minutes ago                         nervous_nash
d3a7bf9ebe10   ubuntu    "top"                    3 months ago     Exited (0) 3 months ago                           keen_vaughan
```

Step 4: To access the terminal inside your container, you can use the following command:

$ docker exec -it container_name bash This

will launch a bash session.

Connecting to the MySQL Server Container Locally :

All MySQL containers launch a MySQL server that includes everything to create and manage databases using SQL. To connect to the server, containers also come with a MySQL client that lets us run SQL queries. The client is just a fancy name for the mysql terminal command. Let's use it inside test-mysql's terminal:

1. Open the bash terminal of test-mysql:

$ docker exec -it test-mysql bash

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker exec -it test-mysql bash
Error response from daemon: No such container: test-mysql
```

2. Connect to the client as a root user:

We are using the -u tag to specify the username (root) and adding the -p tag to enter the password when prompted.

$ mysql -u root -p Enter password: ... mysql>

```
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# docker exec -it test-mysql bash
Error response from daemon: No such container: test-mysql
root@apsit-HP-ProDesk-600-G4-PCI-MT:/home/apsit/Documents/git-SiddhiT# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.42-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

mysql> SELECT 'hello-world!!!';

mysql> show databases;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| CollegeDB          |
| collegeDB          |
| information_schema |
| mysql              |
| performance_schema |
| studentDB          |
| studentrecordDB    |
| students           |
| studentsDB         |
| sys                |
+--------------------+
10 rows in set (0.02 sec)

mysql> create database hello;
Query OK, 1 row affected (0.11 sec)

mysql>
```

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)
```

```
mysql> use mysql
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

```
mysql> SELECT DATABASE();
+------------+
| DATABASE() |
+------------+
| mysql      |
+------------+
1 row in set (0.00 sec)
```

```
mysql> SHOW TABLES;
+---------------------------------------------------+
| Tables_in_mysql
+---------------------------------------------------+
| columns_priv
| component
| db
| default_roles
| engine_cost
| func
| general_log
| global_grants
| gtid_executed
```

```
mysql> DESCRIBE db;
+-----------------------+---------------+------+-----+---------+-------+
| Field                 | Type          | Null | Key | Default | Extra |
+-----------------------+---------------+------+-----+---------+-------+
| Host                  | char(255)     | NO   | PRI |         |       |
| Db                    | char(64)      | NO   | PRI |         |       |
| User                  | char(32)      | NO   | PRI |         |       |
| Select_priv           | enum('N','Y') | NO   |     | N       |       |
| Insert_priv           | enum('N','Y') | NO   |     | N       |       |
| Update_priv           | enum('N','Y') | NO   |     | N       |       |
| Delete_priv           | enum('N','Y') | NO   |     | N       |       |
| Create_priv           | enum('N','Y') | NO   |     | N       |       |
| Drop_priv             | enum('N','Y') | NO   |     | N       |       |
| Grant_priv            | enum('N','Y') | NO   |     | N       |       |
| References_priv       | enum('N','Y') | NO   |     | N       |       |
| Index_priv            | enum('N','Y') | NO   |     | N       |       |
```

```
mysql> SELECT NOW() ;
+---------------------+
| NOW()               |
+---------------------+
| 2024-08-25 18:10:04 |
+---------------------+
1 row in set (0.00 sec)
```

Conclusion: In this experiments student have learnt how to deal with containerization technology using various docker commands.