--

Academic Year: 2024-25 **Semester: V Class /**
Branch: TE IT
Subject: DevOPs Lab (DL)
Subject Lab In-charge: Prof. Sujata Oak

EXPERIMENT NO. 08

Aim: To demonstrate container lifecycle using various docker commands.

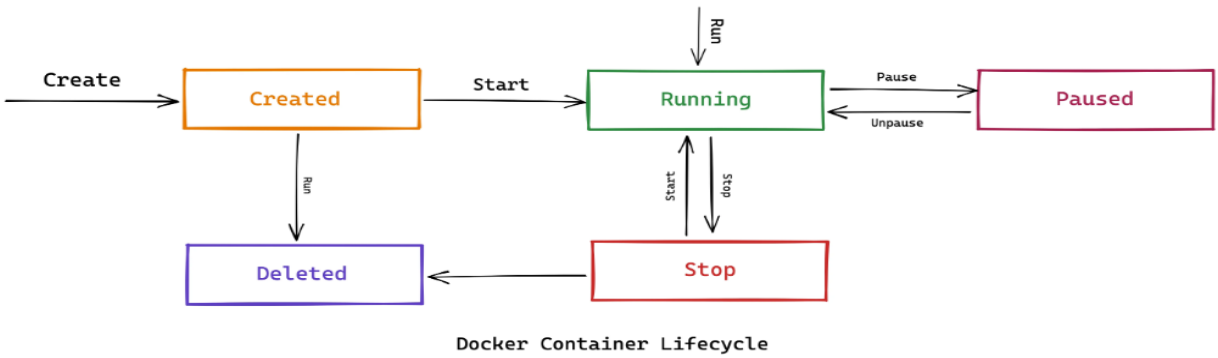
Theory:

Docker is a platform that allows us to package our applications into deployable executables called containers, with all its necessary OS libraries and dependencies.

A container is a process in OS. A process is an instance of a computer program that is being executed. But container processes are different. Container processes are fully-functional environments, and they have more isolation from the OS than the processes in OS.Just like processes, containers have different states throughout their lifecycle.

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

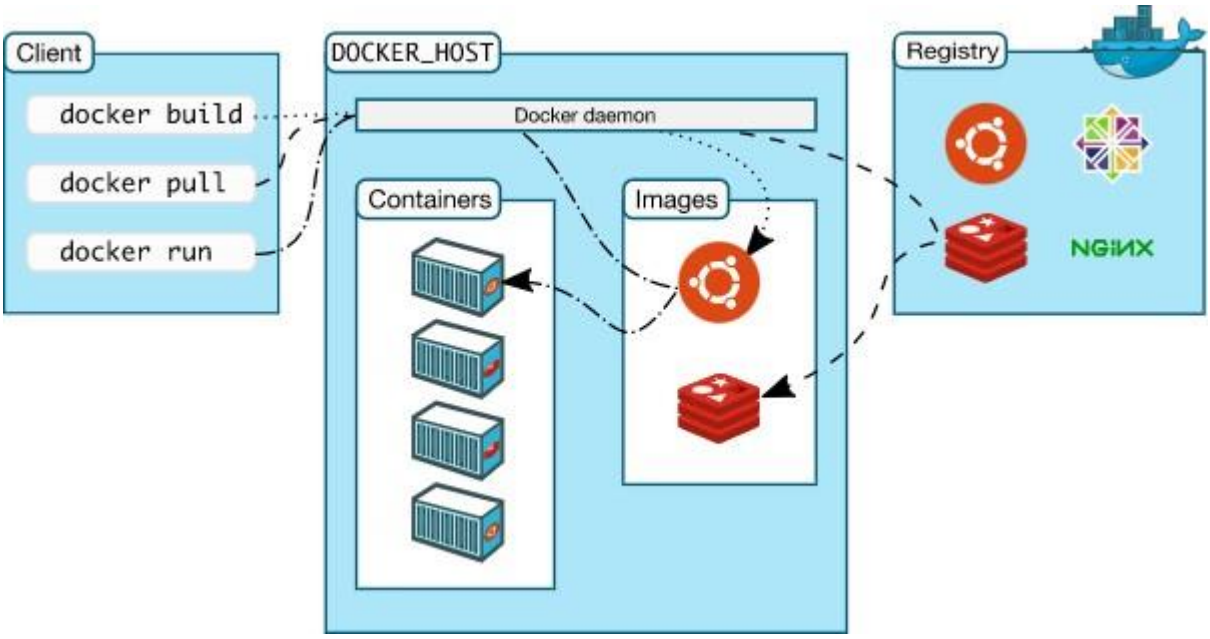


Fig. Architectural overview of Docker

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

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker --version
Docker version 27.1.2, build d01f264
```

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker -v
Docker version 27.1.2, build d01f264
```

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker version
Client: Docker Engine - Community
 Version:           27.1.2
 API version:       1.46
 Go version:        go1.21.13
 Git commit:        d01f264
 Built:             Mon Aug 12 11:51:03 2024
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:          27.1.2
```

DOCKER COMMANDS:

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

```
root@labvm:/home/devasc/Desktop/DOCKER_LAB# docker login
Log in with your Docker ID or email address to push and pull images from Docker
Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to crea
te one.
You can log in with your password or a Personal Access Token (PAT). Using a limi
ted-scope PAT grants better security and is required for organizations using SSO
. Learn more at https://docs.docker.com/go/access-tokens/

Username: 18061977
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/#credential-stores

Login Succeeded
```

2] #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 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
```

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 wil 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@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
```

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@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
```

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@labvm:/home/devasc/Desktop/DOCKER_LAB# docker run --name test-mysql -e MYSQ
L_ROOT_PASSWORD=sujata -d mysql
0e2b27de5979ae2bfa7c4e4648411577965004 58dfb194c18bd4716b63440de4
```

In New Terminal:

```
devasc@labvm:~/Desktop/DOCKER_LAB$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS
   PORTS                 NAMES
0e2b27de5979   mysql     "docker-entrypoint.s…"   35 seconds ago   Up 34 seconds
   3306/tcp, 33060/tcp   test-mysql
```

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@labvm:/home/devasc/Desktop/DOCKER_LAB# docker exec -it test-mysql bash
bash-5.1#
```

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@labvm:/home/devasc/Desktop/DOCKER_LAB# docker exec -it test-mysql bash
bash-5.1# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 9.0.1 MySQL Community Server - GPL

Copyright (c) 2000, 2024, 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> SELECT 'hello-world!!!';
+-----------------+
| hello-world!!! |
+-----------------+
| hello-world!!! |
+-----------------+
1 row in set (0.00 sec)
```

mysql> show databases;

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

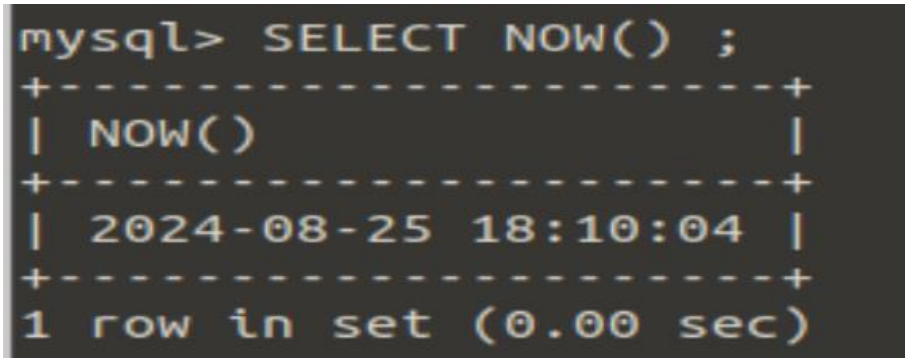
```
mysql> use mysql
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

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       |       |
```



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

https://www.datacamp.com/tutorial/set-up-and-configure-mysql-in-docker

https://www.geeksforgeeks.org/mysql-common-mysql-queries/

https://docs.docker.com/get-started/docker-concepts/the-basics/what-is-a-container/