NAME : HARSHITA SINGH
CLASS: D10C
ROLL : 66

Experiment 5

Understanding Docker Architecture and Container Lifecycle

Aim: To understand the architecture of Docker, the lifecycle of containers, and to install Docker and execute basic Docker commands for managing images and interacting with containers.

Introduction: Docker is an open-source platform that enables developers to automate the deployment, scaling, and management of applications using containerization. A container is a lightweight, standalone, executable package that includes everything needed to run an application, including libraries, dependencies, and configuration files.

Docker provides a client-server architecture that consists of the following components:

1. **Docker Engine:** The core of Docker that runs and manages containers.
2. **Docker Daemon:** A background process that manages Docker containers and images.
3. **Docker CLI:** A command-line tool to interact with Docker.
4. **Docker Registry:** A repository for storing and sharing container images.

Docker Container Lifecycle:

1. **Create:** A container is created from an image but not yet running.
2. **Start:** The created container is started and begins execution.
3. **Running:** The container is actively running.
4. **Pause/Unpause:** The container can be paused and resumed.
5. **Stop:** The container is stopped gracefully.
6. **Kill:** The container is forcefully stopped.
7. **Restart:** The container is restarted.
8. **Remove:** The container is deleted from the system.

Basic Docker Commands:

- **Check Docker version:**
  docker --version

```
C:\Users\jswdolvi>docker --version
Docker version 27.5.1, build 9f9e405
```

- **Pull an image from Docker Hub:**
  docker pull ubuntu

```
C:\Users\jswdolvi>docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
5a7813e071bf: Pull complete
Digest: sha256:72297848456d5d37d1262630108ab308d3e9ec7ed1c3286a32fe
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

```
C:\Users\jswdolvi>docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
e6590344b1a5: Pull complete
Digest: sha256:bfbb0cc14f13f9ed1ae86abc2b9f11181dc50d779807ed3a3
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

- **List available images:**
  docker images

```
C:\Users\jswdolvi>docker images
REPOSITORY    TAG       IMAGE ID       CREATED       SIZE
ubuntu        latest    a04dc4851cbc   4 weeks ago   78.1MB
hello-world   latest    74cc54e27dc4   5 weeks ago   10.1kB
```

- **Run a container:**
  docker run -it ubuntu bash

```
C:\Users\jswdolvi>docker run -it ubuntu
root@45e178c7acde:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp
root@45e178c7acde:/# pwd
/
root@45e178c7acde:/# cd dev
root@45e178c7acde:/dev# ls
console  core  fd  full  mqueue  null  ptmx  pts  random  shm  stderr  stdin  stdout  tty  ur
root@45e178c7acde:/dev#
```

```
C:\Users\jswdolvi>docker run hello-world

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/
```

- **List running containers:**
  docker ps

```
C:\Users\jswdolvi>docker ps
CONTAINER ID   IMAGE     COMMAND       CREATED         STATUS         PORTS     NAMES
45e178c7acde   ubuntu    "/bin/bash"   3 minutes ago   Up 7 seconds             magical_lamar
```

- **List all containers (including stopped ones):**
  docker ps -a

```
C:\Users\jswdolvi>docker ps -a
CONTAINER ID   IMAGE         COMMAND      CREATED             STATUS                       PORTS     NAMES
9e79c9869d7a   hello-world   "/hello"     About a minute ago  Exited (0) About a minute ago          strange_carver
45e178c7acde   ubuntu        "/bin/bash"  4 minutes ago       Up 42 seconds                          magical_lamarr
```

- **Stop a running container:**
  docker stop <container_id>

```
C:\Users\jswdolvi>docker stop 45e178c7acdefe76535d5aa64cf10f8dddbfe289
45e178c7acdefe76535d5aa64cf10f8dddbfe289fbbf8cfeed5a235e00d6270b
```

- **Restart a container:**
  docker restart <container_id>

```
C:\Users\jswdolvi>docker restart 9e79c9869d7abb0e6ec3524650d9a13d4811de946f15a9e177f0
9e79c9869d7abb0e6ec3524650d9a13d4811de946f15a9e177f0b69b20ba60b6

C:\Users\jswdolvi>docker ps -a
CONTAINER ID   IMAGE         COMMAND       CREATED         STATUS
9e79c9869d7a   hello-world   "/hello"      6 minutes ago   Exited (0) 13 seconds ago
45e178c7acde   ubuntu        "/bin/bash"   9 minutes ago   Up 30 seconds
```

- **Remove a container:**
  docker rm <container_id>

```
C:\Users\jswdolvi>docker rm 9e79c9869d7abb0e6ec3524650d9a13
9e79c9869d7abb0e6ec3524650d9a13d4811de946f15a9e177f0b69b20b
```

- **Remove an image:**
  docker rmi <image_id>

```
C:\Users\jswdolvi>docker rmi sha256:74cc54e27dc41bb10dc4b2226072d469509f2f2
Untagged: hello-world:latest
Untagged: hello-world@sha256:bfbb0cc14f13f9ed1ae86abc2b9f11181dc50d779807ed
Deleted: sha256:74cc54e27dc41bb10dc4b2226072d469509f2f22f1a3ce74f4a59661a1d
Deleted: sha256:63a41026379f4391a306242eb0b9f26dc3550d863b7fdbb97d899f6eb89
```

Conclusion: Through this experiment, we explored Docker architecture, installed Docker, and executed various commands to interact with Docker images and containers. Understanding the container lifecycle is crucial for efficient container management and deployment in real-world applications.