Experiment 8

Devops lab
Chaudhary Tuba
231410
TE IT

Aim: to execute docker commands in docker desktop

1. docker –version : gives the version of the docker.
2. docker images – give the names and ID's of the already present images.

```
PS C:\Users\chaud> docker --version
Docker version 28.4.0, build d8eb465
PS C:\Users\chaud> docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

3. docker info – give all the information about the installed docker.

```
PS C:\Users\chaud> docker info
Client:
 Version:    28.4.0
 Context:    desktop-linux
 Debug Mode: false
 Plugins:
  ai: Docker AI Agent - Ask Gordon (Docker Inc.)
    Version:  v1.9.11
    Path:     C:\Program Files\Docker\cli-plugins\docker-ai.exe
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.28.0-desktop.1
    Path:     C:\Program Files\Docker\cli-plugins\docker-buildx.exe
  cloud: Docker Cloud (Docker Inc.)
    Version:  v0.4.29
    Path:     C:\Program Files\Docker\cli-plugins\docker-cloud.exe
  compose: Docker Compose (Docker Inc.)
    Version:  v2.39.4-desktop.1
    Path:     C:\Program Files\Docker\cli-plugins\docker-compose.exe
  debug: Get a shell into any image or container (Docker Inc.)
```

4. docker help – lists all the docker commands

```
PS C:\Users\chaud> docker help
Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
  run         Create and run a new container from an image
```

5. docker ps –- Lists all **running containers** on your system.

```
PS C:\Users\chaud> docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

6. docker run python -- Runs a new container from the Python image and executes its default command.

```
PS C:\Users\chaud> docker run python
Unable to find image 'python:latest' locally
latest: Pulling from library/python
401a98f7495b: Pull complete
5d32990caa16: Pull complete
15b1d8a5ff03: Pull complete
22718812f617: Pull complete
```

7. docker run -it python:3.12 -- Runs a new container from the Python 3.12 image in interactive mode with a terminal, letting you use the Python shell.

```
PS C:\Users\chaud> docker run -it python:3.12
Unable to find image 'python:3.12' locally
3.12: Pulling from library/python
280a8213ce16: Pull complete
3ea8285073a3: Pull complete
9263f10fdf97: Pull complete
Digest: sha256:1cb6108b64a4caf2a862499bf90dc65703a08101e8bfb346a18c9d12c0ed5b7e
Status: Downloaded newer image for python:3.12
Python 3.12.11 (main, Sep  8 2025, 22:53:21) [GCC 14.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
KeyboardInterrupt
>>> print("hello world")
hello world
>>>
```

8. docker ai ask “Explain containerization” -- Docker AI Agent (experimental, for AI-related tasks).

```
PS C:\Users\chaud> docker ai ask "Explain containerization"

  Containerization is the process of packaging an application and its dependencies into a container, which is a lightweight, isolated
  environment. Containers ensure that applications run consistently across different environments, whether it's a developer's laptop, a CI/CD
  system, or production servers.

  ### Key Features of Containers:

    1. Self-contained: Containers include everything needed to run the application, such as libraries, dependencies, and configuration files,
  without relying on the host system.
    2. Isolation: Containers run in isolated environments, minimizing conflicts between applications and improving security.
    3. Portability: Containers can run anywhere—on local machines, data centers, or cloud platforms—ensuring consistent behavior across
  environments.
    4. Efficiency: Unlike virtual machines, containers share the host system's kernel, making them lightweight and faster to start.

  ### Example Use Case:

  Imagine a web app with three components: a React frontend, a Python API, and a PostgreSQL database. Each component can be containerized
  separately, ensuring that:

    • Developers use the same versions of Node.js, Python, and PostgreSQL.
```

9. docker login -- Log in to a Docker registry (like Docker Hub)

```
PS C:\Users\chaud> docker login
Authenticating with existing credentials... [Username: tuba2102]

i Info →To login with a different account, run 'docker logout' followed by 'docker login'


Login Succeeded
PS C:\Users\chaud> # prompts for username/password
```

10.docker push – Upload a Docker image to a registry.

```
PS C:\Users\chaud> docker push python
Using default tag: latest
The push refers to repository [docker.io/library/python]
249a56c8e466: Layer already exists
22718812f617: Layer already exists
15b1d8a5ff03: Layer already exists
ad446e7df19a: Layer already exists
5d32990caa16: Layer already exists
a79d633abf9a: Layer already exists
401a98f7495b: Layer already exists
```

11.docker rmi php – deletes the image

```
PS C:\Users\chaud> docker rmi php
Untagged: php:latest
Deleted: sha256:9fe2d5c18d036c8e00658c0ecfbaece85694e82d65aeb680af57578119fc96ef
```

12. docker inspect python -- Display detailed information on one model.

```
PS C:\Users\chaud> docker inspect python
[
    {
        "Id": "sha256:2deb0891ec3f643b1d342f04cc22154e6b6a76b41044791b537093fae00b6884",
        "RepoTags": [
            "python:latest"
        ],
        "RepoDigests": [
            "python@sha256:2deb0891ec3f643b1d342f04cc22154e6b6a76b41044791b537093fae00b6884"
        ],
        "Parent": "",
        "Comment": "buildkit.dockerfile.v0",
        "Created": "2025-08-14T21:49:23Z",
        "DockerVersion": "",
        "Author": "",
        "Architecture": "amd64",
        "Os": "linux",
        "Size": 411942185,
```

13. docker builder ls – Lists all Docker builder instances and shows their status and details.

```
PS C:\Users\chaud> docker builder ls
NAME/NODE           DRIVER/ENDPOINT     STATUS    BUILDKIT   PLATFORMS
default             docker
 \_ default          \_ default         running   v0.24.0    linux/amd64 (+3), linux/arm64, linux/arm (+2), linux/ppc
desktop-linux*      docker
 \_ desktop-linux    \_ desktop-linux   running   v0.24.0    linux/amd64 (+3), linux/arm64, linux/arm (+2), linux/ppc
```

14. docker cloud -- Manages Docker Cloud services and resources from the command line.

```
PS C:\Users\chaud> docker cloud
Usage:  docker cloud COMMAND

Docker Cloud

Commands:
  accounts       Prints available accounts
  diagnose       Print diagnostic information
  start          Start a Docker Cloud session
  status         Show the status of the Docker Cloud connection
  stop           Stop a Docker Cloud session
  version        Prints the version

Run 'docker cloud COMMAND --help' for more information on a command.
PS C:\Users\chaud>
```

15. docker sbom python:3.12 -- Generates and shows a Software Bill of Materials (SBOM) for the python:3.12 image, listing all included packages.

```
PS C:\Users\chaud> docker sbom python:3.12
NAME                            VERSION                         TYPE
adduser                         3.152                           deb
apt                             3.0.3                           deb
autoconf                        2.72-3.1                        deb
automake                        1:1.17-4                        deb
autotools-dev                   20240727.1                      deb
base-files                      13.8+deb13u1                    deb
base-passwd                     3.6.7                           deb
bash                            5.2.37-2+b5                     deb
binutils                        2.44-3                          deb
binutils-common                 2.44-3                          deb
binutils-x86-64-linux-gnu       2.44-3                          deb
bsdutils                        1:2.41-5                        deb
bzip2                           1.0.8-6                         deb
ca-certificates                 20250419                        deb
comerr-dev                      2.1-1.47.2-3+b3                 deb
```

16. docker volume ls –- Lists all Docker volumes on the system.

```
PS C:\Users\chaud> docker volume ls
DRIVER    VOLUME NAME
PS C:\Users\chaud>
```