**Assignment**

1. What is Docker, and why is it used?

* Docker is a containerization tool.
* Docker is a platform enables to build, package and run the applications in containers.
* It is lightweight, portable and ensure consistency across different environments.

**Usage of Docker:**

* **Portability:** Containers can run on any system with docker installed.
* **Efficiency:** Containers share the host OS kernel, making them lightweight has compared to virtual machines (VMs).
* **Consistency:** Works same way in development, testing and production.
* **Isolation:** Applications run in isolated environments, avoiding dependency conflicts.

1. How is Docker different from a virtual machine (VM)?

|  |  |  |
| --- | --- | --- |
| **Feature** | **Docker** | **Virtual** |
| **Architecture** | Shares host OS kernel | Each VM has separate OS |
| **Performance** | Lightweight, fast startup | Heavier, slower startup |
| **Resource Usage** | Low | High |
| **Isolation** | Process-level isolation | Full OS-level isolation |
| **Portability** | High portable across environments. | Less-portable, OS dependent. |

1. What are the main components of Docker?

* **Docker Engine:** The core service that runs and manages containers.
* **Docker Images:** Read-only templates used to create containers.
* **Docker Containers:** Running instances of docker images.
* **Docker Registry:** Stores and distributes docker images.
* **Docker Compose:** Tool for managing multi-container applications using a YAML file.
* **Docker CLI and API:** Command-Line Interface and API to interact with docker.

1. Explain the difference between Docker images and Docker containers.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Docker Image** | **Docker Container** |
| **Definition** | A read-only blueprint used for creating containers. | Running instances of docker images. |
| **State** | Static | Dynamic |
| **Persistence** | Remains unchanged | Changes a lost unless committed to a new image. |
| **Storage** | Stored in Docker Registry | Exists in runtime and removed when stopped. |
| **Creation** | Built using a Dockerfile | Created from an Image. |

1. What is a Dockerfile?

* Docker file is a text file that contains set of instructions to automate the creation of Docker Image.
* It defines how an image is built, including the base image, dependencies, environment variables and commands.

1. What command is used to build a Docker image?

* To build the image we use the command:

**docker build -t image\_name .**

1. How do you run a container from an image?

* To run the container:

**docker run -itd –name <container\_name> -p <port\_number> <image\_id>**

1. How do you list all running containers?

* To list all the running containers:

**docker ps**

1. What command is used to stop a running container?

* To stop the container:

**docker stop container\_id**

1. How do you remove a Docker container?

* To remove the container:
* **docker rm container\_id**

1. What is the difference between CMD and ENTRYPOINT in a Dockerfile?

* **CMD** is a run-time command. It sets default arguments but can be overridden in the run-time.
* **ENTRYPOINT** is a run-time command. It is the main command that runs, and arguments are appended.

1. What is a Docker volume, and why is it used?

* Docker volume is a storage mechanism that allows data persistence in Docker containers. Volumes are managed by Docker and exist outside the container’s filesystem, preventing data loss when a container stops or is removed.

**Why it is used?**

* **Data Persistence:** Prevents data loss when containers restart.
* **Container Sharing:** Multiple containers can access the same data.

**Creating a volume:**

**docker volume create\_my\_volume**

**docker run -v my\_volume:/app/data my\_image**

**docker volume ls**

1. How do you persist data in Docker containers?

* **By using Docker volumes:**
* Managed by Docker, stored outside the container’s filesystem
* Data remains even if the container is removed.
* Data volumes are recommended for better performance, security and protability.

1. What is a Docker Compose file? How is it used?

* A docker compose file is YAML configuration file used to define and manage multi-container applications in Docker.
* It allows you to run multiple containers with a single command.
* It simplifies the orchestration, networking and scaling for multi-container applications.

**How it is used?**

* **Define services** in --docker-compose.yml
* **Run** docker-compose up –to start all containers.

**Common commands:**

* Start services: **docker-compose up -d**
* Stop services: **docker-compose down**
* Check logs: **docker-compose logs**

1. How do you scale services using Docker Compose?

* Docker compose allows you to scale services by running multiple instances of a container.
* This is useful for **load balancing and high availability.**

**docker-compose up –scale service\_name=n -d**

1. How do you check the logs of a running container?

* To check the logs of the running container:

**docker logs container\_name**

1. What is the purpose of the .dockerignore file?

* The .dockerignore file is used to exclude specific files and directories from being copied into the docker image during the docker build process.
* When using COPY or ADD in a Dockerfile, Docker will **ignore** the files specified in .dockerignore, preventing them from being added to the image.

**Why it is important?**

* **Reduces image size:** Prevents unnecessary files from bloating the image.
* **Improves build performance:** Speed up the build process by skipping unnecessary files.
* **Enhances security:** Excludes sensitive files.

1. What are the different networking modes in Docker?

* **Bridge:** The default network driver. Containers communicate internally via a virtual network.
* **Host:** Remove the network isolation between the container and the docker host. Used when low-latency networking is needed.
* **None:** Completely isolate a container from the host and other containers. Used for security or testing.
* **Overlay:** Enables multi-host networking across Docker swarm nodes.
* **Macvlan:** Assigns a real MAC address to the container. Used for direct communication with the local network.

1. How do you expose ports in a Docker container?
2. **Using EXPOSE command:**

EXPOSE 8080

1. **Using -p flag in docker run:**

docker run -p 8080:80 my\_image

1. **Using docker compose:**

Docker-compose up -d

1. What is the difference between docker stop and docker kill?

|  |  |  |
| --- | --- | --- |
| **Command** | docker stop | docker kill |
| **Behavior** | **Gracefully stops** a container by sending **SIGTERM**. | **Immediately terminates** a container by sending **SIGKILL**. |
| **Process Handling** | Gives the process time to save state and shutdown. | Does not allow cleanup, forcefully stops execution. |
| **Use-case** | Preferred when you want a safe shutdown. | Used when a container is unresponsive or needs to be stopped immediately. |
| **Used for** | Clean shutdown | Force-stop |
| **Command Example** | docker stop container\_id | docker kill container\_id |