1. **What is docker and why do we need it?**

Docker is a platform that enables developers to create, deploy, and run applications inside containers. Containers are lightweight, portable, and self-sufficient environments that include everything needed to run a piece of software, including the code, runtime, libraries, and dependencies. Docker provides a consistent environment to ensure that applications behave the same way regardless of where they are run.

We need Docker due to the following features.

* + **Portability**: Docker containers can run on any system that supports Docker, ensuring that applications behave consistently across different environments, from development to production.
* **Isolation:** Containers encapsulate an application and its dependencies, providing isolation from other applications running on the same host. This reduces conflicts between different applications and their dependencies.
* **Efficiency**: Containers are lightweight and share the host system's kernel, leading to faster startup times and lower resource consumption compared to traditional virtual machines.
* **Scalability**: Docker makes it easy to scale applications horizontally by adding more container instances. This is particularly useful in microservices architectures where different services can be scaled independently.
* **Consistency**: Docker ensures that the application runs the same way in different environments, reducing issues related to environment discrepancies. This consistency speeds up development, testing, and deployment processes.
* **Simplified Deployment**: Docker images can be built once and deployed anywhere, simplifying the deployment process and reducing the chances of deployment errors.

1. **What is the docker lifecycle?**
   * Pull Image:

Command: docker pull <image\_name>

Description: Before a container can be created, the required Docker image needs to be pulled from a Docker registry (such as Docker Hub).

* + Create Container:

Command: docker create <image\_name>

Description: A container is created from a Docker image. At this stage, the container is in a "created" state but not yet running.

* + Start Container:

Command: docker start <container\_id>

Description: The container transitions from the "created" state to the "running" state, and the application inside the container begins execution.

* + Run Container:

Command: docker run <image\_name>

Description: This command combines create and start into a single step. If the image is not already pulled, it will be pulled, and then a new container is created and started.

* + Stop Container:

Command: docker stop <container\_id>

Description: The running container is stopped. The container can be started again later. The container's state changes from "running" to "stopped."

* + Pause and Unpause Container:

Commands:

docker pause <container\_id>

docker unpause <container\_id>

* + Description: The pause command suspends all processes in the container, and the unpause command resumes them. This is useful for temporarily halting container operations without stopping them.
  + Restart Container:

Command: docker restart <container\_id>

Description: The container is stopped and then started again. This is equivalent to stopping and starting the container in one command.

* + Kill Container:

Command: docker kill <container\_id>

Description: The container is abruptly stopped by sending a SIGKILL signal to the main process inside the container. This is a forceful termination.

* + Remove Container:

Command: docker rm <container\_id>

Description: The container is removed from the Docker host. This can only be done for containers that are in a "stopped" state.

* + Remove Image:

Command: docker rmi <image\_name>

Description: The Docker image is removed from the host. This can only be done if there are no containers using the image.

1. **What is the difference between an image and a container?**

**Definition:**

**Docker Image:** An image is an immutable package that contains everything needed to run an application, including code, runtime, libraries, environment variables, and configuration files.

**Docker Container:** A container is a runtime instance of an image. It is a lightweight, portable, and isolated environment where the application runs.

**State:**

**Docker Image:** Images are read-only. They cannot be changed once created.

**Docker Container:** Containers are writable. Changes can be made to the filesystem, but these changes are not saved back to the original image.

**Creation:**

**Docker Image:** Images are created using the docker build command, typically from a Dockerfile.

**Docker Container:** Containers are created using the docker create or docker run commands.

**Storage:**

**Docker Image:** Images are stored in Docker registries (e.g., Docker Hub) and can be shared and distributed.

**Docker Container:** Containers are stored on the Docker host's storage.

**Persistence:**

**Docker Image:** Images are persistent and reusable. Multiple containers can be created from the same image.

**Docker Container:** Containers are ephemeral and disposable. They can be stopped, started, removed, and recreated without affecting the original image.

**Lifecycle:**

**Docker Image:** Images go through stages of being built, tagged, pushed to a registry, and pulled from a registry.

**Docker Container:** Containers go through stages of being created, started, stopped, paused, and removed.

**Example:**

**Docker Image:** An example of an image is nginx:latest.

**Docker Container:** An example of a container is a running instance created from the nginx:latest image.

1. **How to check docker container logs? Provide the command for the same?**

To check the logs of a Docker container, you can use the “**docker logs”** command.

**Command:**

docker logs [OPTIONS] CONTAINER

Common Options

* + -f or --follow: Follow log output (similar to tail -f).
  + --tail : Number of lines to show from the end of the logs (e.g., --tail 100).
  + --since : Show logs since a specific time (e.g., --since "2021-01-1T00:00:00").
  + --until : Show logs until a specific time.