**Docker**

**Introduction of Docker:**

* Docker is a containerization tool.
* It is a virtualization tool.
* It is written in GO language.
* It was released in 2013.
* Developed by Solomon Hykes and Sebastian phal.
* Makes developing and deploying applications much easier.
* It is free and open source.
* Docker is a lightweight, portable software tool that packages everything the application needs to run consistently across different environments.
* Docker is a powerful platform designed to help you develop, ship and run applications more efficiently by using containerization technology.
* Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies and ship it as one package.
* Developers develop the code and that code can be shipped them into container.
* **Features of Docker:**
* **Consistency**
* **Efficiency:** docker containers are lightweight.
* **Portability:** docker containers can transferred easily from one computer environment to another.
* **Isolation**: each container runs it in own isolated environment, which helps improve security and stability.
* **Version control:** Docker allows you to track and manages different versions of your application.
* **Containerization:**
* Containerization in Docker is a technology that allows developers to package applications and their dependencies into a standardized unit called a container.
* **Container:**
* A container in docker is a lightweight, portable, and self-sufficient unit that encapsulates an application and all its dependencies.
* Container is a way to package application with all the necessary dependencies and configuration.
* It is a virtual machine which does not have any operating system
* Containers provide a consistent environment for applications to run, regardless of where they are deployed.
* Containers nothing but operating system.
* Container is nothing but server.
* Why docker:
* Handover our application safely to the end user.
* Difference between docker file and docker image and docker container?

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Docker file | Docker image | Docker container |
| Definition | A docker file is a text file that contains a set of instructions for building a docker image. | A docker image is a read only template that contains the application code, libraries, dependencies, and runtime. | A docker container is a runnable instance of a docker image. it includes everything needed to run an application. |
| Purpose | Defines the environment and dependencies needed for an application. | Serves as blue print for creating containers. | Provides an isolated environment for running applications. |
| Immutability | Mutable (can be modified) | Immutable (once created it does not change) | Mutable (can be started, stopped, and modified) |
| Creation | Written manually by developers. | Built from a docker file. | Created from a docker image. |
| Storage | Not stored just a script. | Can be stored in a registry (docker hub). | Runs in memory and is ephemeral |
| Execution | Not executed directly. | Not executed directly. | Executed to run applications. |
| Layers | NA | Composed of multiple layers from docker file. | NA |

* Docker client and server architecture:
* Docker uses a client-server architecture.
* The docker client talks to the docker daemon, which does the heavy lifting of building, running, and distributing your docker containers.
* The docker client and daemon can run on same system.
* Docker client server architecture consists:

1. Docker client:

* A CLI that allows users to interact with the docker daemon.
* Sends commands to the docker daemon(server) to build, run and manage containers.
* Can communicate with multiple docker daemons(servers) over a network.

1. Docker daemon:

* It is the server-side component of docker.
* Listens from API requests from the docker client and manages docker objects such as images, containers, networks, and volumes.
* Handles the creation, execution, and management of containers.

1. Rest API:

* The communication between docker client and docker daemon is facilitated by a REST API.
* The API allows clients to send HTTP requests to perform operations on docker objects.

1. Docker registry/hub:

* A repository for storing and distributing docker images.
* Docker hub is the default public registry, but private registries can also be set up.
* The docker client can push images to and pull images from the registry.