Docker is a container management service

Want to share projects: Push code on **Docker Hub**with all dependencies. Firstly, create the **Images** and push the Image on Docker Hub. After that, pull the same image shared with all dependencies.

**Important Terminologies in Docker**

**1. Docker Image**

* It is a file, comprised of multiple layers, used to execute code in a Docker container.
* They are a set of instructions used to create docker containers.

**2. Docker Container**

* It is a runtime instance of an image.
* Allows developers to package applications with all parts needed such as libraries and other dependencies.

**3. Docker file**

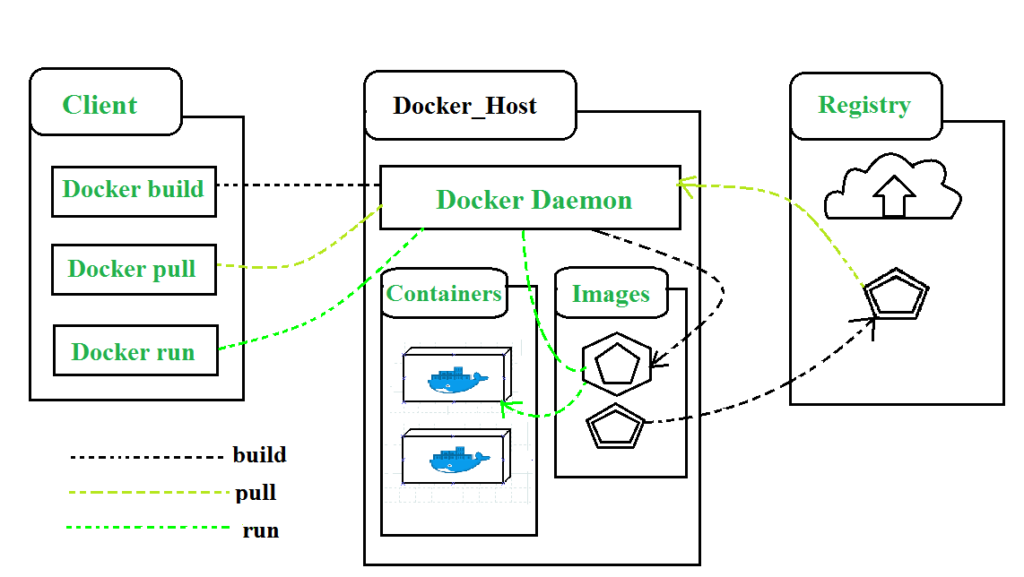
* It is a text document that contains necessary commands which on execution helps assemble a Docker Image.
* Docker image is created using a Docker file.

**4. Docker Engine**

* The software that hosts the containers is named Docker Engine.
* Docker Engine is a client-server based application
* The docker engine has **3 main** components:
  + **Server**: It is responsible for creating and managing Docker images, containers, networks, and volumes on the Docker. It is referred to as a daemon process.
  + **REST API**: It specifies how the applications can interact with the Server and instructs it what to do.
  + **Client**: The Client is a docker command-line interface (CLI), that allows us to interact with Docker using the docker commands.

**5. Docker Hub**

* Docker Hub is the official online repository where you can find other Docker Images that are available for use.
* It makes it easy to find, manage, and share container images with others.



*Architecture of Docker*

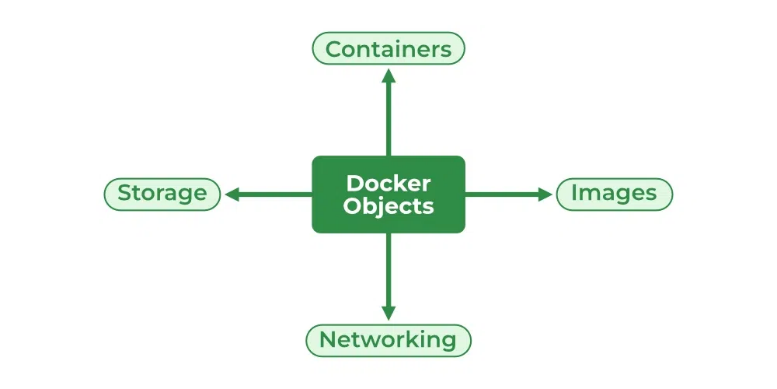
## **What is Docker Daemon?**

Docker daemon manages all the services by communicating with other daemons. It manages docker objects such as images, containers, networks, and volumes with the help of the API requests of Docker.

The common commands which are used by clients are **docker build, docker pull,**and **docker run**

## **Docker Objects**

Whenever we are using a docker, we are creating and use images, containers, volumes, networks, and other objects.



## **Docker Images**

An image contains instructions for creating a docker container. It is just a **read-only template**. It is used to store and ship applications. Images are an important part of the docker experience as they enable collaboration between developers in any way which is not possible earlier.

## **Docker Containers**

Containers are created from docker images as they are ready applications. With the help of Docker API or CLI, we can start, stop, delete, or move a container. A container can access only those resources which are defined in the image unless additional access is defined during the building of an image in the container.

## **Docker Storage**

We can store data within the writable layer of the container but it requires a storage driver. [Storage driver](https://www.geeksforgeeks.org/data-storage-in-docker/)controls and manages the images and containers on our docker host.

## Types of Docker Storage

1. **Data Volumes:**Data Volumes can be mounted directly into the filesystem of the container and are essentially directories or files on the Docker Host filesystem.
2. **Volume Container**: In order to maintain the state of the containers (data) produced by the running container, Docker volumes file systems are mounted on Docker containers. independent container life cycle, the volumes are stored on the host. This makes it simple for users to exchange file systems among containers and backup data.
3. **Directory Mounts:** A host directory that is mounted as a volume in your container might be specified.
4. **Storage Plugins:**Docker volume plugins enable us to integrate the Docker containers with external volumes like Amazon EBS by this we can maintain the state of the container.

## **Docker Networking**

[Docker networking](https://www.geeksforgeeks.org/basics-of-docker-networking/) provides complete isolation for docker containers. It means a user can link a docker container to many networks. It requires very less OS instances to run the workload.

Types of Docker Network

1. **Bridge:** It is the default network driver. We can use this when different containers communicate with the same docker host.
2. **Host:**When you don’t need any isolation between the container and host then it is used.
3. **Overlay:**For communication with each other, it will enable the swarm services.
4. **None:**It disables all networking.
5. **macvlan:**This network assigns MAC (Media Access control) address to the containers which look like a physical address.

# **Docker Hub**

**Docker Hub:** cloud-based service, push/pull Images