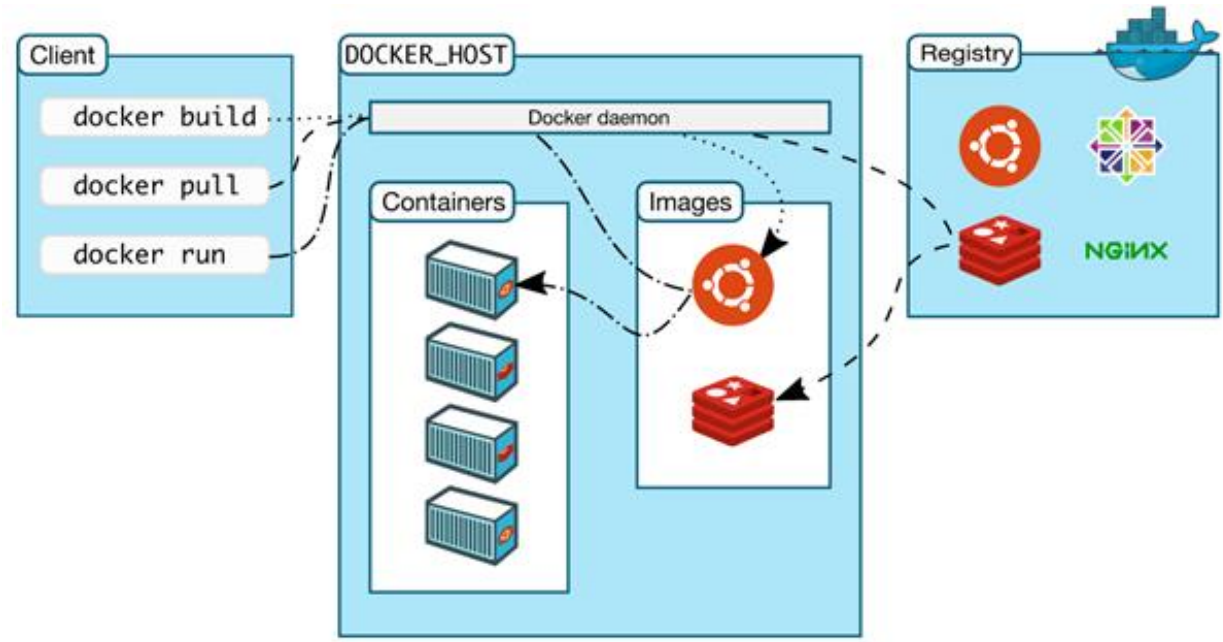
ASSIGNMENT 4

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

Docker is an open platform for developing, shipping, and running applications. Docker enables to separate applications from infrastructure so we can deliver software quickly. With Docker, we can manage your infrastructure in the same ways we manage our applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, we can significantly reduce the delay between writing code and running it in production.

2. Explain the Docker 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 the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate using a REST API, over UNIX sockets or a network interface. Another Docker client is Docker Compose, that lets you work with applications consisting of a set of containers.



Docker architecture components

The architecture components of Docker are the following:

Docker daemon (system service called “dockerd”) listens for requests and manages Docker objects such as images, containers, networks, and volumes alone or in coordination with other daemons that are installed at other machines.

Docker client (docker) a command line tool for connecting at Docker daemon. The docker command uses the Docker API.

Docker registry stores Docker images (anyone can setup their own registry in premises). An example is the “Docker Hub” which is a public registry that anyone can use (Docker is configured to look for images on Docker Hub by default). There are some pre-specified commands in order to access the registry i.e. the docker pull/push and docker run commands. The aforesaid commands are making the required images to be pulled from/pushed to your configured registry.

Docker objects are widely used in Docker. Docker is creating and using images, containers, networks, volumes, plugins, and other objects. The aforesaid components are handled as objects by Docker.

Image: is a read-only template with instructions for creating a Docker container. Maybe an image is based on another image. You might create your own images with the use of Dockerfile.

Container: is a runable instance of an image. You can create, start, stop, move, or delete a container using the Docker API or CLI with the use of Docker client. You can connect a container to one or more networks, attach storage to it, or even create a new image based on its current state.

3. What do you mean by a Dockerfile?

Docker can build images automatically by reading the instructions from a Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build users can create an automated build that executes several command-line instructions in succession.

4. What do you mean by Docker Images?

A Docker image is a read-only template that contains a set of instructions for creating a container that can run on the Docker platform. It provides a convenient way to package up applications and preconfigured server environments, which you can use for your own private use or share publicly with other Docker users.

5. What do you mean by Docker Hub?

Docker Hub is a hosted repository service provided by Docker for finding and sharing container images with your team. Key features include: Private Repositories: Push and pull container images. Automated Builds: Automatically build container images from GitHub and Bitbucket and push them to Docker Hub.

6. How to create a Docker container from an Image?

docker commit container_id imagename