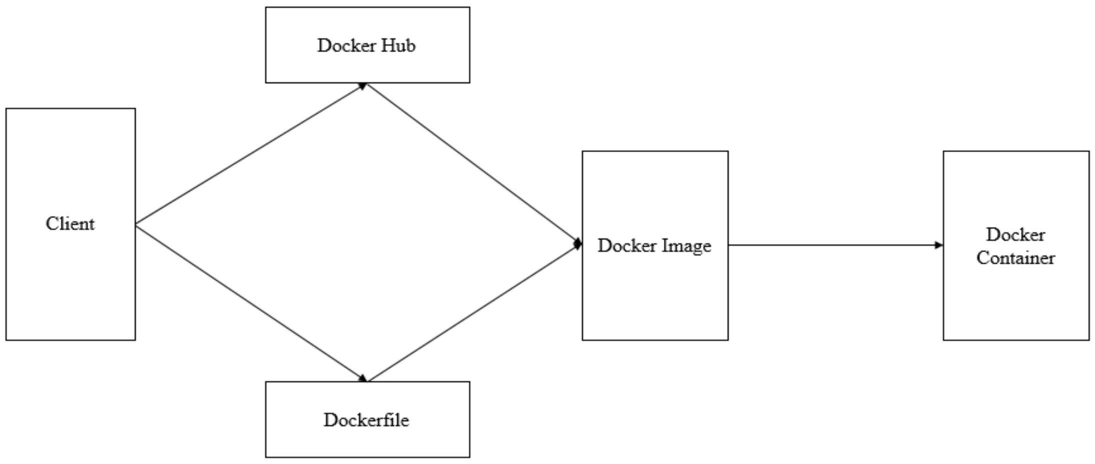
What is Docker? And Benefits of Docker:

The founder of Docker is Solomon Hykes. He released the Docker in June 2014. Docker is the leading open source software container platform which is making container easy to use. It is a tool designed to create, deploy and run applications by using container. Docker allows faster and more efficient deployments without worrying about running user app on different platforms. Container is providing a developer all kind of libraries and other dependencies to run an application as a package. Docker is like a virtual machine but not like a virtual machine to create a whole virtual operating system. The best part of Docker is, it's providing small footprint and lower overhead. Docker is not helping only a developer also system administration. Like a developer will package all the software and its components in a container and Docker will take care of those container. Every application works on its own container and does not interfaces with others. After remove a container, it will not affect the others file on the system.

Docker Architecture:



A user can pull an image from docker hub or his/her written docker file. from image user can create a docker container or instance.

Dockerfile: A developer will define all the applications and dependencies and requirements keeps in file which known as docker file. docker file is used to create docker images.

Docker image and Docker container: Docker image is the template used to create docker container. In a docker image have all the applications, dependencies and requirements. when run a docker images, it's created a docker containers. Image contains a union of layered of filesystems. Docker containers are the run time instance of docker image. User can first pull an image from docker hub and store the image in docker daemon then create a docker container or user can use run command to pull an image from docker hub. Run command pull the image in the docker daemon first if it is not present in locally and start the container. User can use docker build command to create an image from dockerfile and store in docker daemon. This image also stored in online cloud repository called docker hub. User can create his / her own repository also. Images are pulled to create containers.

In Docker, command line interface is the client and Docker Daemon is the server. Docker Daemon or Server has the all containers and images. Docker server receive the commands from docker client in the form of command or REST API request. All the docker clients and docker servers together forms docker engine. Docker client and daemon can be existing on the same machine or different machine.

Docker keyword: Dockerfile, Docker Images, Docker Containers, Docker Hub/Registry, Docker client, Docker Server, Docker Engine

Install Docker:

To get the Information about operating system: uname -r

```
1. sudo apt-get update

2. sudo apt install apt-transport-https ca-certificates curl software-properties-common

3. curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –

4. sudo apt-key fingerprint 0EBFCD88

5. sudo add-apt-repository “deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable”

6. sudo apt-get update

7. sudo apt-get install docker-ce
```

Important Command for docker:

docker --version

docker version // to know about client and server

docker info

sudo service docker start // to start docker

sudo service docker stop // to stop docker

sudo remove docker // to remove docker

export PS1=”\u$” // enter in root mood

// to check the user group: id -nG

Add a user name for not writing sudo for docker: sudo usermod -a -G docker user-group-name

Command for Image:

docker images // to check the all images in the system

docker pull image-name // pull an image from docker hub

docker run --name test -it image-name // to run a container from image

docker run -it image-name // create a container from image without name

docker rmi image-name // to remove an image docker inspects image-name // to know details about an image

docker rmi -f image-ID // to remove an image with force

Command for Container:

docker ps / for all running container

docker ps -a // for all container

docker exec -it container-name bash // to insert in container

docker attach container-name // main difference between exec and attach is: attach stop the container when exit command is used in container.

docker start container-id // start a container

docker stop container-id // stop a container

docker kill container-name/id // to kill a container

docker inspect container-id | grep "IPAddress" // to get the container ip address

docker rename container-name new-container-name // to rename a container name

Exit command from exit from container

docker system df

docker system prune –help // be careful about this command because it will delete all containers

-a remove all unused

// to create an image from **Docker file**

mkdir Dockerfile // to create a directory

touch Dockerfile // create a docker file

Write in Dockerfile-- >

FROM ubuntu or scratch // scratch is the empty image and ubuntu is the image with all functionality for ubuntu

MAINTAINER name of the user <email address of user>

RUN apt-get update // run execute the during the building the images

CMD ["echo", "hello this is my first image for docker"] // cmd execute when user create container

docker build . // if user in Docker file directory run this command. Which will create an image without name. don't forget to give the (.)

docker build -t image-name:1.0 . // build an image with name and 1.0 is the tag name for image

After created an image from Dockerfile:

```
docker run –name container-name -it image-id
```

Docker compose:

Docker compose is the tool for defining and running multi container docker application from image.

It's written in yml file to configure application services. For example, docker-compose.yml. docker compose can scale up services when user need.

Docker compose install:

```
apt install docker-compose
```

To check the version:

```
docker-compose -v
```

To create docker compose file:

```
mkdir DockerComposeFile // it's an optional part to keep all the docker compose file in one folder
Touch docker-compose.yml // to create a file
```

Write the following text for testing and keep eyes on indentation. It's a sample for docker compose file:

```
session: '3'
services:
        web:
         image: nginx
        ports:
         -    "8080:80"   // 80 for webserver and 8080 for host machine
        database:
         image: redis
```

To check the validity of docker compose file, run the following command before executing the docker compose file:

```
docker-compose config
```

If there have any error, it will show the error. Version is very important and indentation also.

To start all services, run the following command:

```
docker-compose up -d // -d use to start in detach mood and use the terminal for writing the others command
docker-compose up -d –scale database=4 // scale up and create 4 database containers
```

To check the docker image and container run the as usual command for docker.

To stop all services, execute the following command:

docker-compose down // will delete all containers created from docker compose

Resource for Docker:

https://opensource.com/resources/what-docker

https://www.upguard.com/articles/docker-vs-lxc

https://docker.com

12 months free VM: aws free tier