**Basic Docker Commands:**

\* Docker run <image name>

-> To run a container and it will pull the image

\* Docker ps

-> To list all running containers

\* Docker ps -a

-> To list all running and stopped containers

\* Docker stop <container name or ID >

-> To stop a running container

\* Docker rm <container name or ID>

-> To remove stopped or exited container permanently.

\* Docker rm <con1> < con2>

\* Docker images

-> To see the list of images we have

\* Docker rmi <image name or id >

-> To delete an image and we need to ensure that all containers are stopped.

\* Docker pull <image name>

-> to download the image

\* Docker run Ubuntu

\* Docker run Ubuntu sleep 5

\* Docker run <image name>

-> It will run a container in attached mode. ( All the running process will be showing) Press ctrl+c to quit from that (by pressing ctrl +c that container will stop automatically)

\* Docker run -d <image name>

-> This will run the container in detached mode (In background)

\* Docker attach <container name or ID)

-> It is used to attach back to the container which is running in detached mode

\* Docker run – it <image name>

-> It is used to open an interactive session within that container

\* Docker run - - name <my-container name> < image name or id>

-> It is used to create a container with specified name

\*Docker run <Image name : version>

-> To run a container on specified version, If you don’t specify default tag will be latest.s

\* docker run -p <host-port>:<container-port> <image-name>

-> you can map ports between the container and the host.

\* docker run -v <host-path>:<container-path> <image-name>

Docker run -v /opt/datadir:/var/lib/mysql mysql

-> suppose if you created a MySql container and stored some data. Accidentally or intentionally that container was stopped and deleted. So the data will be gone. So if we store the data outside container then that data will be safe. To store the data outside container we use volumes.

\* docker inspect <container name or ID>

-> To see the more details of container in json format.

\* docker logs <container-id or container-name>

-> used to display the logs generated by a running or stopped container

\* docker run Ubuntu cat /etc/\*release\*

-> it will run a container on Ubuntu and shows the release version of Ubuntu

\*

Docker is a set of platforms as a service (PaaS) products that use Operating system-level virtualization to deliver software in packages called containers

Docker is an open-source containerization platform by which you can pack your application and all its dependencies into a standardized unit called a container

WHAT IS DOCKERFILE????

The Dockerfile uses DSL (Domain Specific Language) and contains instructions for generating a Docker image. Dockerfile will define the processes to quickly produce an image.

While creating your application, you should create a Dockerfile in order since the Docker daemon runs all of the instructions from top to bottom.

WHAT IS DOCKER IMAGES???

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.

Docker Image is an executable package of software that includes everything needed to run an application. This image informs how a container should instantiate,

determining which software components will run and how. Docker Container is a virtual environment

that bundles application code with all the dependencies required to run the application. The application runs quickly and reliably from one computing environment to another.

WHAT IS DOCKER CONTAINERS??

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

are runtime instances of Docker images

INSTALLATING A DOCKER ON UBUNTU -

sudo apt-get update

sudo apt install docker.io

1. Remove old version of Docker -- $ sudo apt-get remove docker docker-engine docker.io containerd runc

DOCKER COMMANDS

1.docker run <image-name> -- to run a container from an image

2.docker run --name <container\_name> <image-name> -- to give name of cotainer

3.docker pull <image-name> -- to pull any image which is present in offical docker hub

4.docker ps -- list all running container

5.docker ps -a -- listing all stopped and running containers

6.docker ps -l -- list the latest container

7.docker ps -q -- shows the id of the container

8.docker stop <container-id/container-name> -- to stop a container

9.docker start <container-id/container-name> -- to start a container

10.docker rm {options} <container-id/container-name> - to remove the container

-f flag: remove the container forcefully.

-v flag: remove the volumes.

-l flag: remove the specific link mentioned.

11.docker rmi <image ID/ image name> -- to delete image in docker

12.docker images -- list of images

13.docker exec {options} --This command allows us to run new commands in a running container.

-d flag: for running the commands in the background.

-i flag: it will keep STDIN open even when not attached.

-e flag: sets the environment variables

ex: docker exec <cont-id> cat /etc/\*release\*

14.docker login -- login to docker hub

15.docker push <imagename/image-id> -- to push inti hub

15.docker build -t image-name:tag . -- to build docker image