**DOCKER**

**Docker Setup**

* **Launch EC2 instance**
  1. **Give any name like 🡪 Docker Server**
  2. **Use Ubuntu 18.0 Image**
  3. **Use Instance type** 🡪 **t2.micro**
  4. **Create security group**
  5. **Allow all traffic from anywhere**
  6. **Create a key pair**
  7. **Launch EC2 instance**
* **Go to** [**https://get.docker.com/**](https://get.docker.com/)
* **Use below commands to quick install Docker for Ubuntu machines**
* $ curl -fsSL https://get.docker.com -o get-docker.sh
* $ sh get-docker.sh
* **Give permissions for user “Ubuntu” to execute Docker commands**
  + You can check the Docker group 🡪 sudo vim /etc/group
  + You can give user “ubuntu” for docker group or you can use usermod command 🡪 **sudo usermod -aG docker ubuntu**
  + Check it 🡪 $ **id ubuntu**
  + Now you can run the docker commands

**Docker Image**

**A Docker image is a combination of bin/libs that are necessary for a software application to work. Initially all the software’s of docker are available in the form of docker images.**

**You can refer the Docker repository where you will find all kinds of docker images**

[**https://hub.docker.com/**](https://hub.docker.com/)

**Docker Container**

**A running instance of an image is called as a docker container**

**Docker Host**

**The server where docker is installed is called docker host**

**Docker Client**

**This is the CLI of docker where the user can execute the docker commands. The docker client accepts these commands and passes them to a background process called "docker daemon"**

**Docker daemon**

**This process accepts the commands coming from the docker client and routes them to work on docker images or containers or the docker registry**

**Docker Registries**

* **Storage for Docker Images**
* **Dockerhub is a default Registry**
* **Cloud based Registries**
  + - **DockerHub**
    - **GCR (Google Container Registry)**
    - **ACR (Azure Container Registry)**
    - **Amazon ECR (Elastic Container Registry)**
* **InHouse or Local Registries**
  + - **Nexus 3+**
    - **Jfrog Artifactory**
    - **DTR (Docker Trusted Registry)**

**Docker Importance Commands**

1. **$ docker images 🡪 List all the images locally**
2. **$ docker run 🡪 Creates new container**
3. **$ docker ps 🡪 List running containers**
4. **$ docker ps -a 🡪 List all the containers**
5. **$ docker exec 🡪 Execute commands on containers**
6. **$ docker start/stop/restart container\_id / container\_name**
7. **$ docker rmi image\_name / image\_id 🡪 Remove docker images locally**
8. **$ docker rmi -f image\_name / image\_id 🡪 Forcibly remove the docker image associated with a running container**
9. **$ docker inspect (docker image inspect image\_name) 🡪 Detail of Container & Image**
10. **$ docker pull 🡪 Pull the docker images from Docker Repository**
11. **$ docker system prune -af 🡪 To delete all images**
12. **$ docker rm container\_name / container\_id 🡪 To remove the container**
13. **$ docker rm -f container\_name / container\_id 🡪 To remove forcibly a running container**
14. **$ docker stop $(docker ps -aq) 🡪 To stop all the running containers**
15. **$ docker rm $(docker ps -aq) 🡪 To delete all the stopped containers**
16. **$ docker rm -f $(docker ps -aq) 🡪 To delete all the running containers**
17. **ctrl+p,ctrl+q 🡪 To come outside of the container safely**

**Examples:**

1. $ docker run –name myweb nginx
2. $ docker run –name myweb -d nginx
3. $ docker run –name myweb -d -p 8080:80
   1. Here 8080 🡪 is host port
   2. And 80 🡪 is container port
4. $ docker stop container\_name / container\_id
5. $ docker start containr\_name / container\_id
6. $ docker exec container\_name ls /
7. $ docker exec -it container\_name /bin/bash
   1. We can run commands inside container
   2. Run “ps” command
   3. You can install it “apt update” 🡪 “apt install procps -y”
   4. Run “ps -ef”
8. $ docker rmi nginx:tag\_name
9. $ docker run -it ubuntu /bin/bash

**Bind Mount:**

**Bind mount is mostly used inject data from host machine to container. But for preserving the data the best option is docker volumes.**

**Example:**

* 1. $mkdir mysql\_backup
  2. $ docker run –name mydb -d -e MYSQL\_ROOT\_PASSWORD=Password123 -p 3030:3306 -v /home/ubuntu/mysql\_backup:/var/lib/mysql mysql
  3. verify it 🡪 $ docker exec -it mydb /bin/bash
  4. $ cd /var/lib/mysql
  5. $ ls

**Docker Volumes**

**Containers are ephemeral or you can say like containers are volatile in nature. Means if the container crashes or exited it will lose the data inside it. In order to save the data inside the container we will use docker volumes.**

**Command:**

* + 1. **Docker volume**
    2. **Docker volume create mydbdata**
    3. **Docker run –name mydb -d -p 3030:3306 -e MYSQL\_ROOT\_PASSWORD=password -v mydbdata:/var/lib/mysql mysql**
    4. **Sudo su –**
    5. **Ls /var/lib/docker/volumes/ 🡪 you should able to see your docker volume there**
    6. **You can check the volume bonded to this container using docker inspect command**

**$ docker inspect mydb**

**Check the information regarding “Binds” & “Mounts”**

* + 1. **You can login to mysql container using this command**

**$ mysql -h 172.17.0.0 -u root -ppassword**

**$ show databases;**