**Docker Complete Guide**

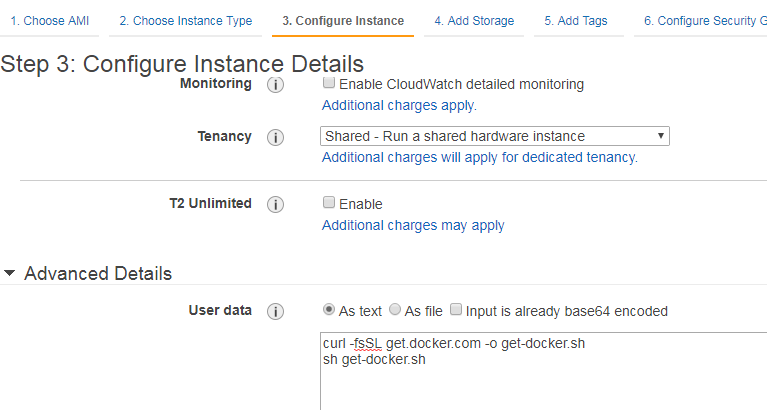
**Docker hub user:sra123456**

**Pwd :sra123456**

**========================================================================**

**Step 1 : How to install the docker**

**To create the docker first launch the ubuntu instance. If you want to install docker while launching the instance you have to give the commannds in the Configure Instance tab**

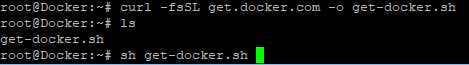
****

**Docker Installation URL**

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

****

# $ curl -fsSL get.docker.com -o get-docker.sh  
# $ sh get-docker.sh



**To see the version:-**

docker -v

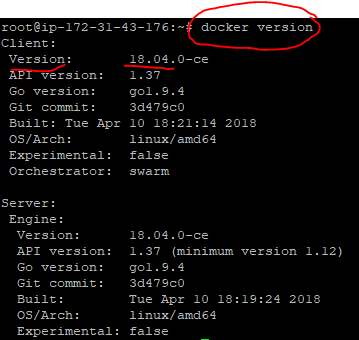
Docker version 18.01.0-ce, build 03596f5

Or

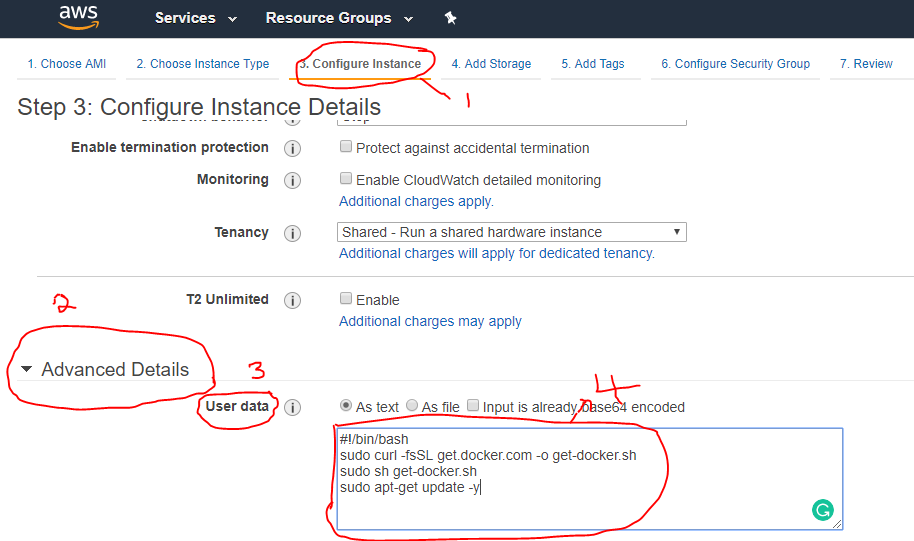
Docker --version

For detailed

# docker version



Step-1A...How to install the docker while launching the instance



**Enter the shell script into the user data**

#!/bin/bash

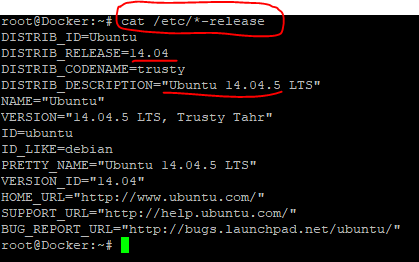
sudo curl -fsSL get.docker.com -o get-docker.sh

sudo sh get-docker.sh

sudo apt-get update -y

How to see the version of os by

#cat /etc/\*-release



Following commands uses in Docker

**## How to search the Images in docker**

$ docker search ubuntu

$ docker search centos

**How to pull the images**

$ docker pull centos:6.9

$ docker pull ubuntu:14.04

**## How to check the docker images**

$ docker images

**## Remove Docker images**

Docker rmi < Repositry name /image id>

**## remove the docker containner**

Docker rm -f <container name / container id>

**## The above command is divided as follows:**

**D---daemon**

**T-- terminal**

**I--- interactive**

Create the docker Container:

* Docker run –tdi - -name web -p 80:80 ubuntu /bin/bash

**To see the container running in the background**

$ docker ps

$ docker ps -a -----> command to see all the running containers and stopped ones

**To access the shell of container that runs in the background mode**

$ docker exec web apt-get update –y

To see the hostname of the container:

$ docker run –tdi - - name webserver –p 33:22 –p 81:80 ubuntu /bin/bash

$docker exec –it webserver apt-get update –y

$ docker exec –it webserver apt-get install apache2 –y

$docker exec –it webserver service apache2 start

To know the hostname of a container

docker exec <webserver/container name> hostname –i

Open the web browser by using <instanceIP:81>

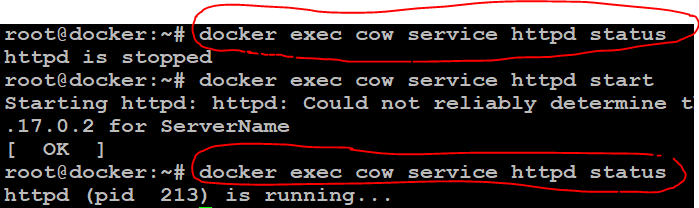
Change the index.html file in apache2

$ docker attach webserver

$ cd /var/www/html/

$apt-get install vim

$ vi index.html



**Start/Stop Container**

# docker stop <container id>

# docker start <container id >

**To view logs for docker container**

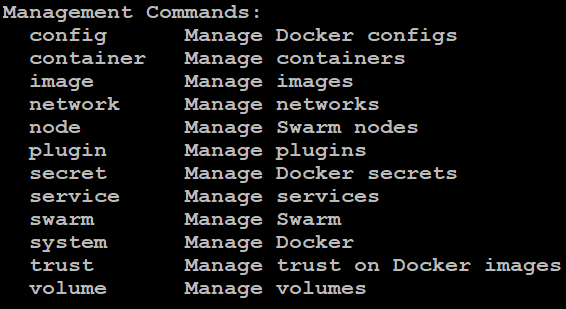
$ docker logs <container id >

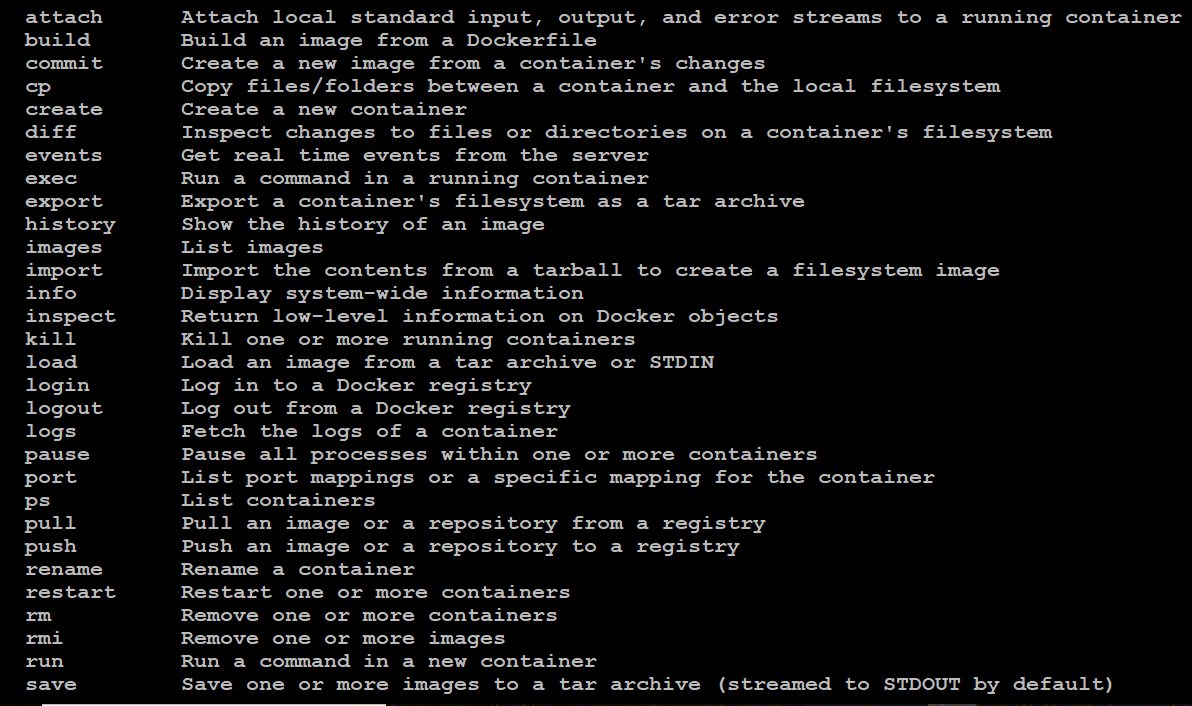
**To rename docker container**

$ docker rename <old\_name> <new\_name>

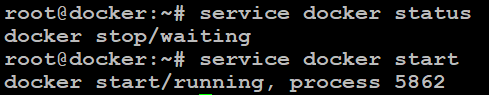
**How to see the only stopped containers**

$ docker ps -f "status=exited"





**How to stop the docker engine and start the docker engine**



How to start more than one container in docker engine.



How many ways we can create docker images

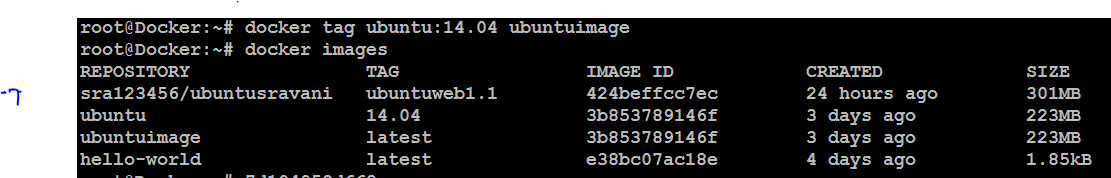
1. Docker container
2. Docker file

Step -1 **Docker commit**

**To create an image of container(The container has to be run while creating the image)**

**# docker commit <existing-containerid> <Newimagename>**

**#docker commit ubuntu:14.04 ubuntuimage**

****

**# docker tag ubuntu;14.04 ubuntuimage**

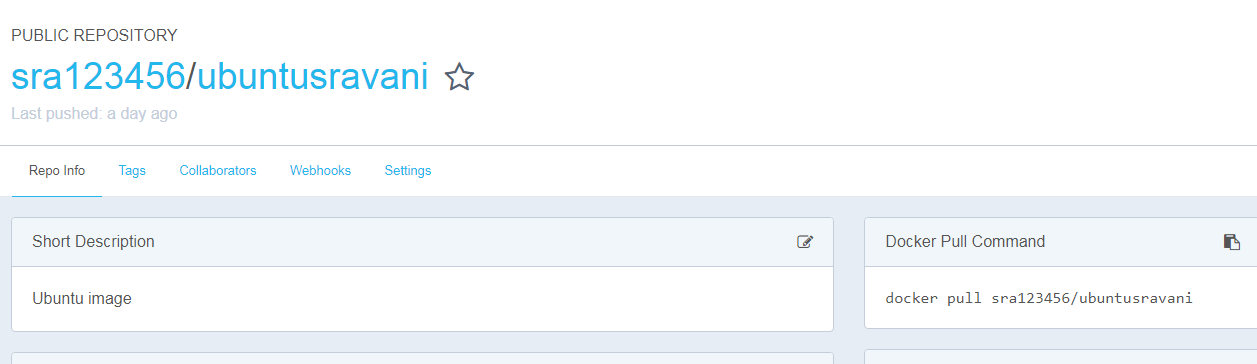
**# docker login (establish the connection between the docker client and docker hub)**

**Username :sra123456**

**Passwd: sra123456**

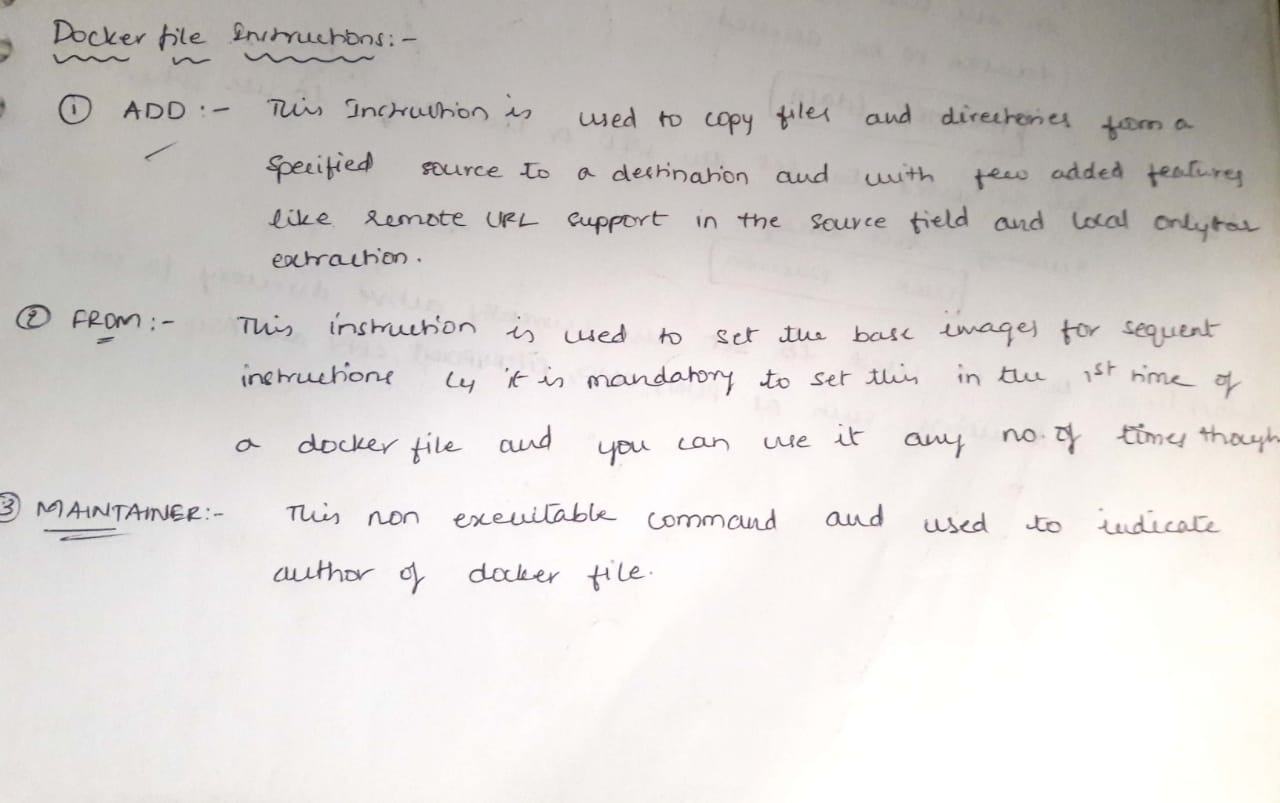
**# docker pull sra123456/ubuntusravani:ubuntuweb1.1**

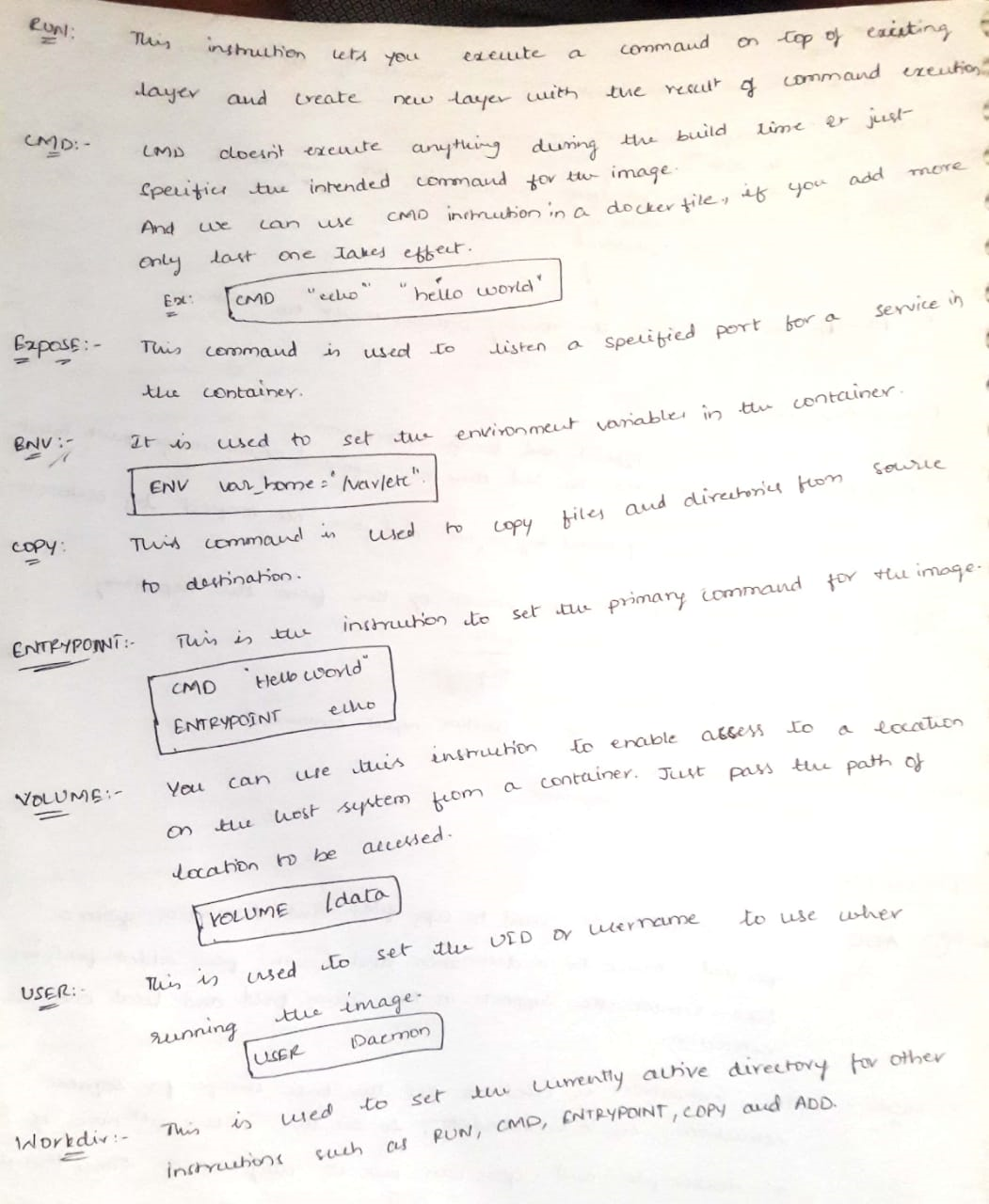
**Hub repository name: tag name**

****

**Step -2 Docker file**

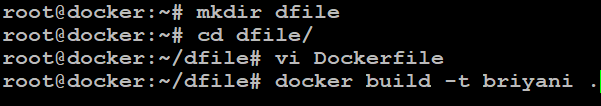
**Create one directory**

****

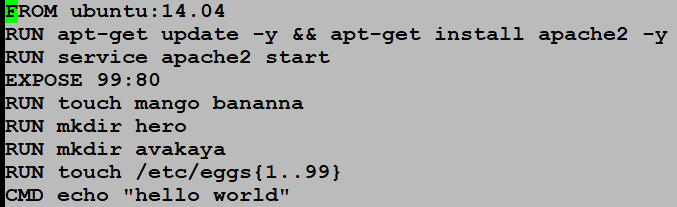


Create the file

# vi Dockerfile



Vi Dockerfile ( add the commands inside the file)



The command to run the dockerfile

# docker build –t testimg .

#docker images

Docker testimag is going to created. By using this image we can create the containers.

# docker run

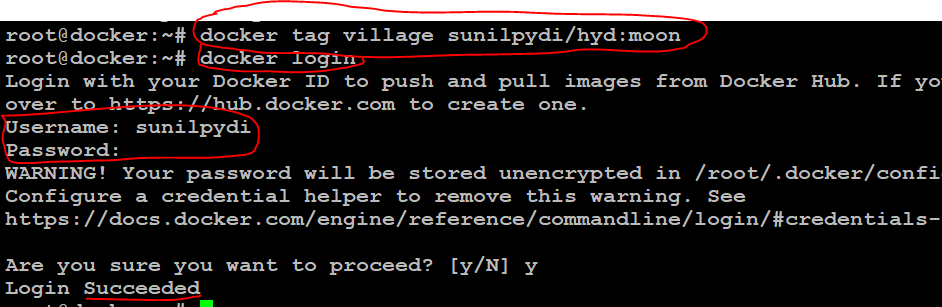
**Taging the docker hub and pushing the image to docker hub**

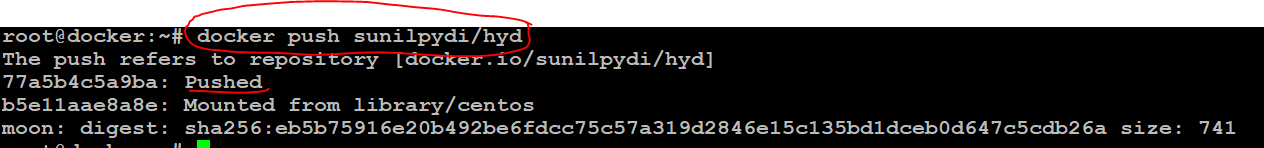
**Village -- existing <imagename/id>**

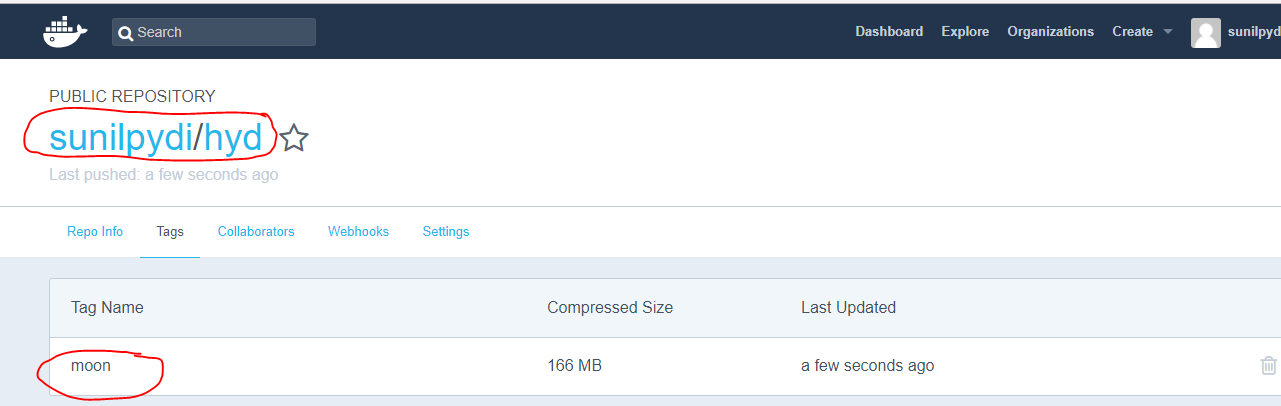
**sunilpydi/hyd ---- repository name**

**moon <tag/image name>**

**(we renamed the village to moon by tagging)**







\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Installing the apache2 in container



# docker run -dti --name dal -p 33:22 -p 81:80 ubuntu:14.04 /bin/bash

#docker exec dal apt-get update

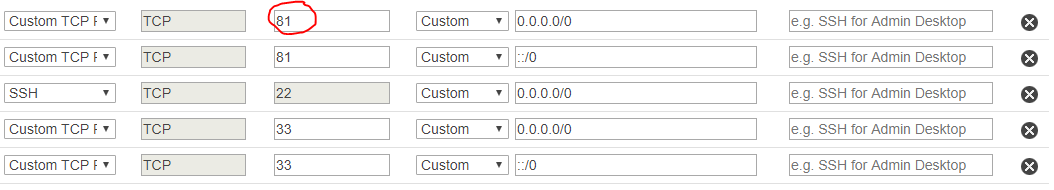
# docker exec dal apt-get update -y

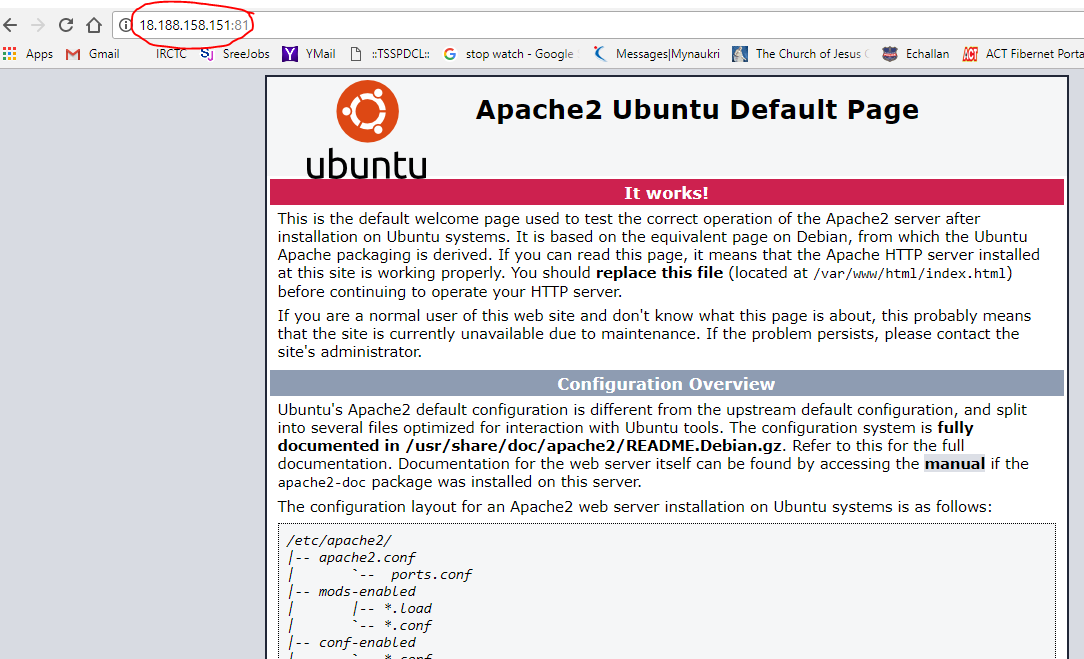
# docker exec dal apt-get install apache2 -y

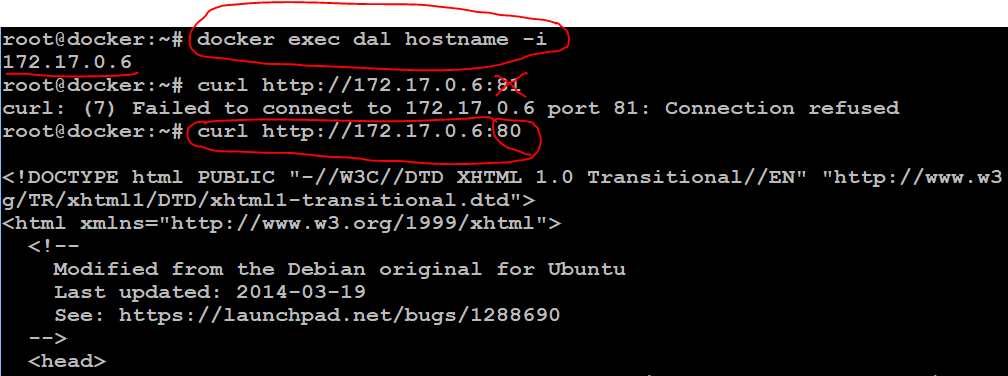
# docker exec dal service apache2 status

# docker exec dal service apache2 status

**Enable the port number in the EC2 security groups**









**How to give docker container access to client/other user**

**Docker giving access to other user/client**

Work on ubuntu

# docker run -dti --name access -p 32:22 -p 82:80 ubuntu:16.04 /bin/bash

33..exportal port

22..importal port

# docker attach dal

# apt-get update -y

# apt-get install ssh -y

# apt-get install vim -y

apt-get install openssh-server

yum install openssh-server...centos only

# vim /etc/ssh/sshd\_config

Open the file..serial number

...28 PermitRootLogin without-password

PermitRootLogin without-password yes

...52 #PasswordAuthentication yes

>service ssh restart

You can even add the user in the docker container

Useradd test

Passwd test

**To login to the container by two ways**

1. **By putty with ipaddress and port no.**
2. **Ssh -p 33 root@18.188.254.152…..docker engine ip address.**

**#########**

**To see the docker logs.**

**# docker logs containerid/name**

**&&&&&&&**

**Docker network**

**# docker network ls**

**# docker network create --subnet=172.16.0.0/16 act**

**# docker network ls**

**# ifconfig**

**# docker network ls**

**# docker inspect act… to see the act ipaddress range**

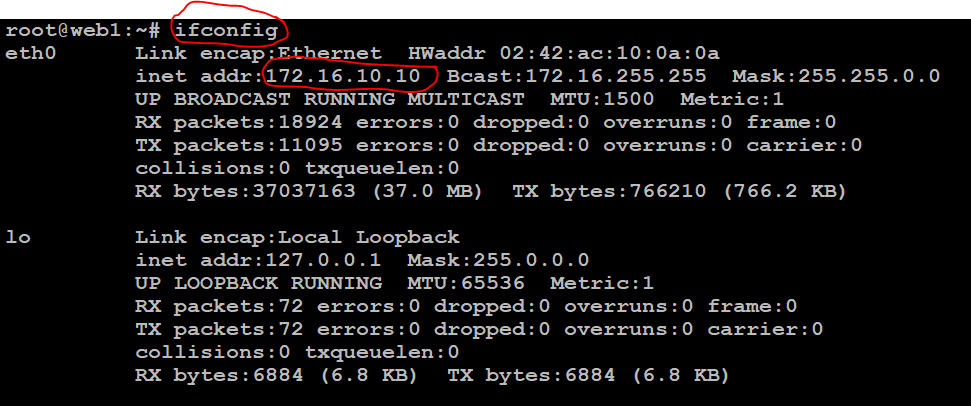
**# docker inspect beem**

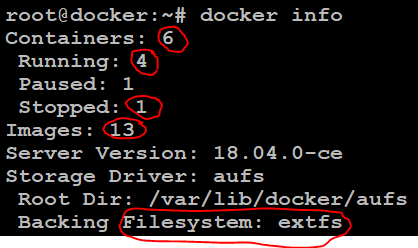
**# docker inspect act**

**# docker inspect beem**

**# docker run -dti --name bill --net act --ip 172.16.10.10 -h web1.example.com -p 34:22 -p 85:80 ubuntu:14.04 /bin/bash**

**# docker attach bill**

****

****

****

****

**Docker overlay**

[**http://blog.nigelpoulton.com/demystifying-docker-overlay-networking/**](http://blog.nigelpoulton.com/demystifying-docker-overlay-networking/)

**open these ports :**

**4789**

**7946**

**4243**

**2377**

**How to remove Docker Containers images using script file**

**========================================**

**docker-destroy-all.sh**

**#!/bin/bash**

**# Stop all containers**

**docker stop $(docker ps -a -q)**

**# Delete all containers**

**docker rm -f $(docker ps -a -q)**

**# Delete all images**

**docker rmi $(docker images -q)**

**./docker-destroy-all.sh**

**Docker compose**

[**http://markheath.net/post/install-wordpress-docker-compose**](http://markheath.net/post/install-wordpress-docker-compose)

**Single command you can manage all your application related services you can even define order.**

**Step-1**

**Installing docker compose:** [**https://docs.docker.com/compose/install/#install-compose**](https://docs.docker.com/compose/install/#install-compose)

**sudo curl -L https://github.com/docker/compose/releases/download/1.21.0/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose**

**chmod +x /usr/local/bin/docker-compose**

**docker-compose --version**

**Step-2**

**# mkdir rice**

**#cd rice**

**rice#vim docker-compose.yml**

**version: '2'**

**services:**

**wordpress:**

**image: wordpress**

**ports:**

**- 8080:80**

**environment:**

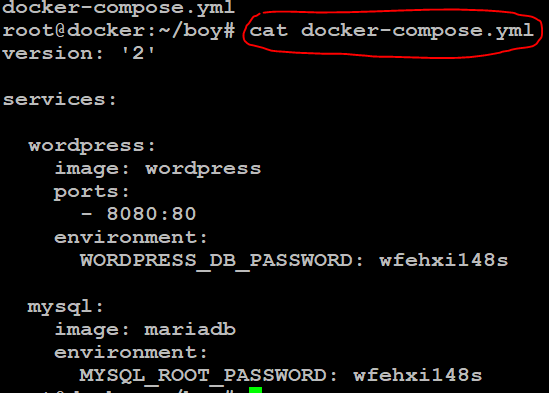
**WORDPRESS\_DB\_PASSWORD: wfehxi148s**

**mysql:**

**image: mariadb**

**environment:**

**MYSQL\_ROOT\_PASSWORD: wfehxi148s**

****

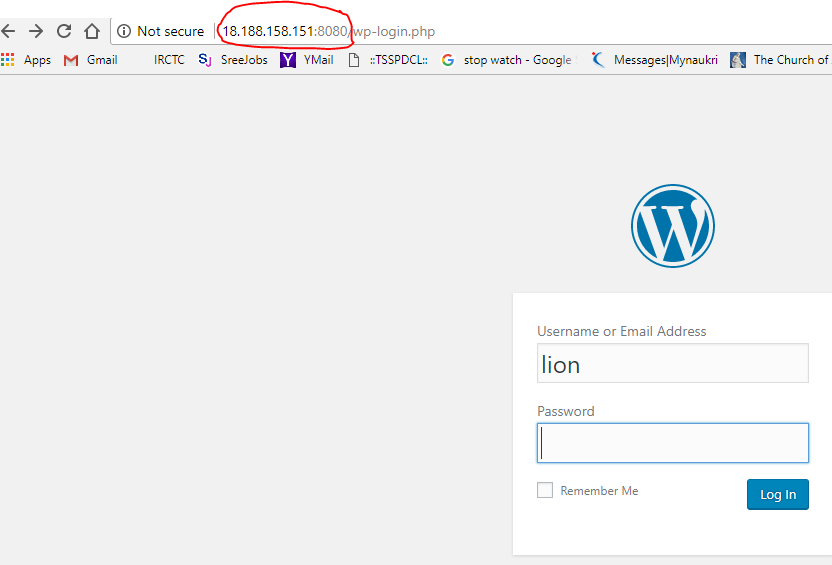
**rice#docker-compose up -d**

**Go to the ec2 instance enable the port :8080**

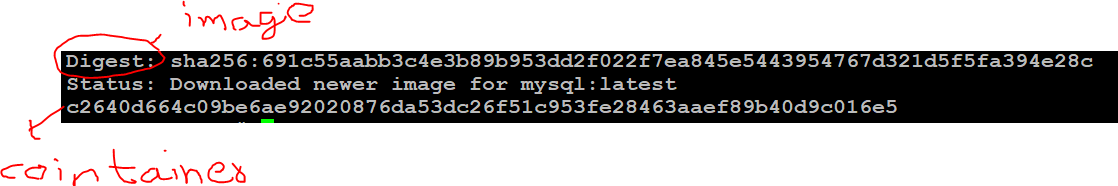
**Go to the browse the ipaddress:8080**

**\*\*\*\* Digest: it will only show when image created**

**Digest: sha256:201d004f55669dd2c0884f00fc44145f**

****

**\*\*\*\* Digest: it will only show when image created**

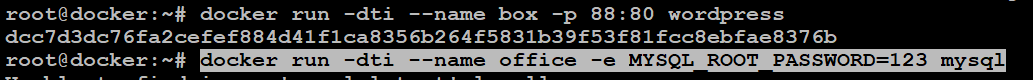
**Digest: sha256:201d004f55669dd2c0884f00fc44145f**

**Docker link**

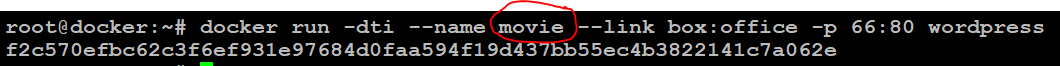
**Docker link provides ..docker with a reference point that allows it to reffer other containers for example,we can specify to link the container web to container db**

**# docker run -dti --name box -p 88:80 wordpress**

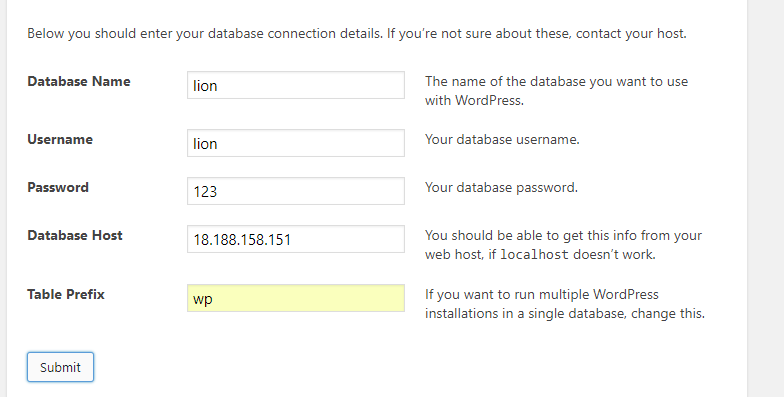
**#docker run -dti --name office -e MYSQL\_ROOT\_PASSWORD=123 mysql**

****

**Created two containers, now we need to establish link between these two containers**

****

**# docker run -dti --name movie --link box:office -p 66:80 wordpress**

****

**Try this….**

**Can we communicate with two containers**

**1..docker link ( manually)**

**2. Docker compose ( by using script file)**

**docker pull wordpress**

**Docker pull mysql**

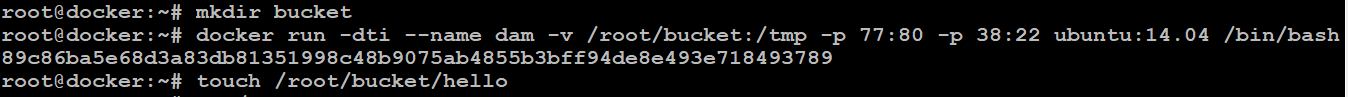
**Docker run -dti --name wp -p 8000:80 wordpress /bin/bash…..**

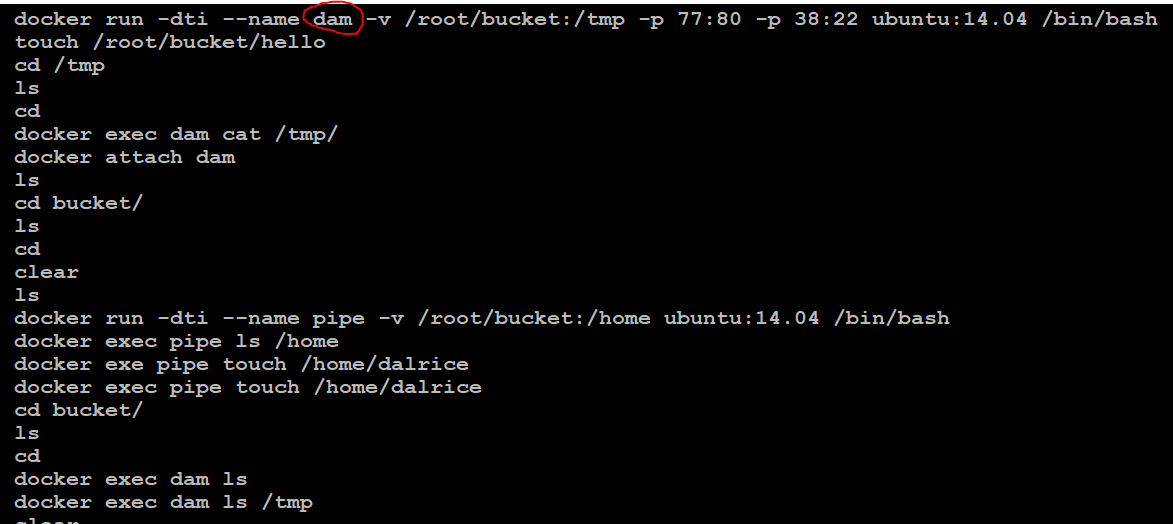
**docker run -dti --name web -e MYSQL\_ROOT\_PASSWORD=654321 mysql**

**docker run -dti --name wp99 --link web:mysql -p 99:80 wordpress**

**Docker volumes**

**docker run -dti --name dam -v /root/bucket:/tmp -p 77:80 -p 38:22 ubuntu:14.04 /bin/bash**

****

****

**Docker volume**

**Docker Volume**

**Step: 1**

**Create the instance**

**First see the available zone of the instance. Ex:- us east:2a**

**Step:2**

**Create the volume group ec2---volume--create**

Size can be 2 gb or 100 gb

Action--Attach volume--- name of the instance..

**Step:3**

**Login to the instance through putty**

**# lsblk**

**Xvda1**

**fdisk /dev/xvda1**

**Press m**

**Primary--2g**

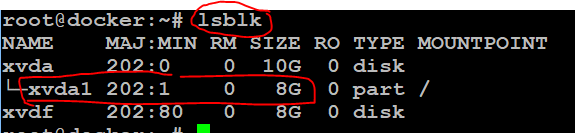
**W---save**

**Partprobe**

**Mkfs.ext4 /dev/xvda1**

**Mount /dev/xvda docvol**

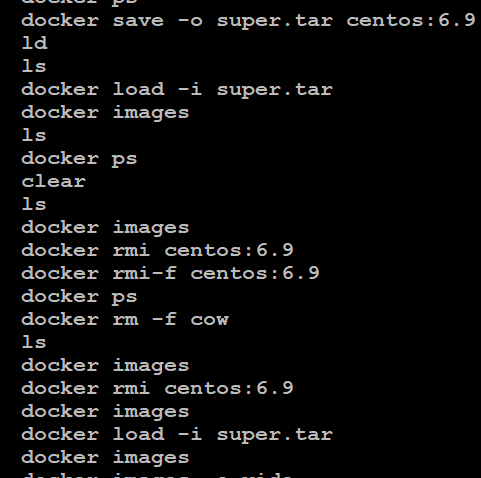
**Before that “mkdir docvol” to create in the docker instance.**

****

**docker run -dti --name docaws -v /root/docvol:/tmp -p 83:80 centos:6.9 /bin/bash**

**if you want to see the volume information of this container**

**#docker inspect docaws**

****