**Docker Installation**

sudo apt update -y

sudo apt install docker.io -y

sudo systemctl start docker

sudo systemctl status docker

sudo systemctl enable docker

sudo usermod ubuntu -g docker

exit

Docker Installation - <https://docs.docker.com/engine/install>

docker pull ubuntu

docker image ls

docker search centos

docker pull centos

docker image ls

docker run -it  --name mycontainer ubuntu bash

apt update -y

apt install apache2 git -y

cd /var/www/html

git clone https://github.com/themes12/Real-Estate-Landing-Page-v2

cd /etc/init.d

./apache2 start

apt install curl -y

curl localhost

docker run -it -p 80:80 --name container ubuntu bash

apt update -y

apt install apache2 git -y

cd /var/www/html

git clone https://github.com/themes12/Real-Estate-Landing-Page-v2

cd /etc/init.d

./apache2 start

apt install curl -y

curl localhost

docker run -d -p 8080:80 nginx

**To login to running container**

docker exec -it conatiner\_name bash

**To Logout from container**

exit

Ctrl + A + D

To stop a running container

docker stop container\_name

To start a stopped container

docker start container\_name

**Day -3**

—-------

docker run -d -p 80:80 --name webserver-01 nginx

**To list running container**

docker ps

**To list all containers (running/ stopped)**

docker ps -a

**To check the size of container**

docker ps -s

**To Stop a running container**

docker stop container\_name/container\_id

docker stop webserver-01

**To Start a stopped container**

docker start webserver-01

**To Restart a container**

docker restart webserver-01

**To Rename a Container**

Docker rename old\_container\_name new\_container\_name

docker rename webserver-01 nginx-server

**To Check Logs of running container**

docker logs container\_name/container\_Id

docker logs -f container\_name/container\_Id

docker logs -f webserver-01

**To Check resource usage (CPU/Memory/ Network/ Disk)**

docker stats

**Get Detailed info of container**

docker inspect container\_name/container\_id

**Get Detailed info of image**

docker image inspect container\_image/image\_id

**To deleted all stopped containers and images not in use:**

docker system prune -fa

**Get Details of containers/images in the host**

docker info

**Deploy Wordpress Application**

**Deploying database mysql - https://hub.docker.com/search?q=mysql**

docker run -d --name database -p 3306:3306 -e MYSQL\_DATABASE=wordpress -e MYSQL\_USER=admin -e MYSQL\_PASSWORD=redhat mysql

**Deploying wordpress - https://hub.docker.com/\_/wordpress**

docker run -d --name=wordpress -p 8080:80 wordpress

**Homework**

https://hub.docker.com/\_/drupal

‘

**Day -4**

**Automate Docker Installation in AWS EC2**

#Launch Instance > Advanced Details > UserData

#!/bin/bash

sudo apt update -y

sudo apt install docker.io -y

sudo systemctl start docker

sudo systemctl enable docker

sudo usermod ubuntu -g docker

Docker hub - <https://hub.docker.com/>

Building Image

============

1. **Create docker image and clone it from git code /project from git inside the nginx/html folde**

docker run -d -p 80:80 --name web nginx

docker exec -it web bash

cd /usr/share/nginx/html

rm -rf \*

apt update -y

apt install git -y

git clone <https://github.com/mdn/beginner-html-site-styled>

mv [beginner-html-site-styled](https://github.com/mdn/beginner-html-site-styled)/\* .

Exit or Ctrl + A + D

1. **commit the docker image & to dockerhub**

**To create a image from container.**

docker commit container\_name image\_name

docker commit web myfirstimage

1. **Create & setup profile on dockerHub**

**Create a docker hub account:**

Docker hub - <https://hub.docker.com/>

1. **Login to Docker hub**

**To Login to Dockerhub account using cli**

docker login

Username - docker hub account username

Password - dockerhub account password

Login Success

1. **To tag and push the image**

docker tag image\_name dockerhub\_username/repo\_name

docker tag myfirstimage gauravdemo06/myfirstimage

docker push gauravdemo06/myfirstimage

1. **To prune  the docker**

docker system prune -fa

1. **For pull the docker image**

docker pull gauravdemo06/myfirstimage

docker run -d -p 80:80 gauravdemo06/myfirstimage

**Docker file**

1. **Create docker file**

touch Dockerfile

1. Open & Edit the docker file

vim Dockerfile

1. Docker file command

—-------------------------------------------

FROM nginx

RUN apt update -y && apt install git -y

WORKDIR /usr/share/nginx/html

RUN rm -rf \*

RUN git clone https://github.com/mdn/beginner-html-site-styled.git && mv beginner-html-site-styled/\*

EXPOSE 80

—--------------------------------------------

1. Build the image from docker file

**To build the image from Dockerfile**

docker build -t image\_name .

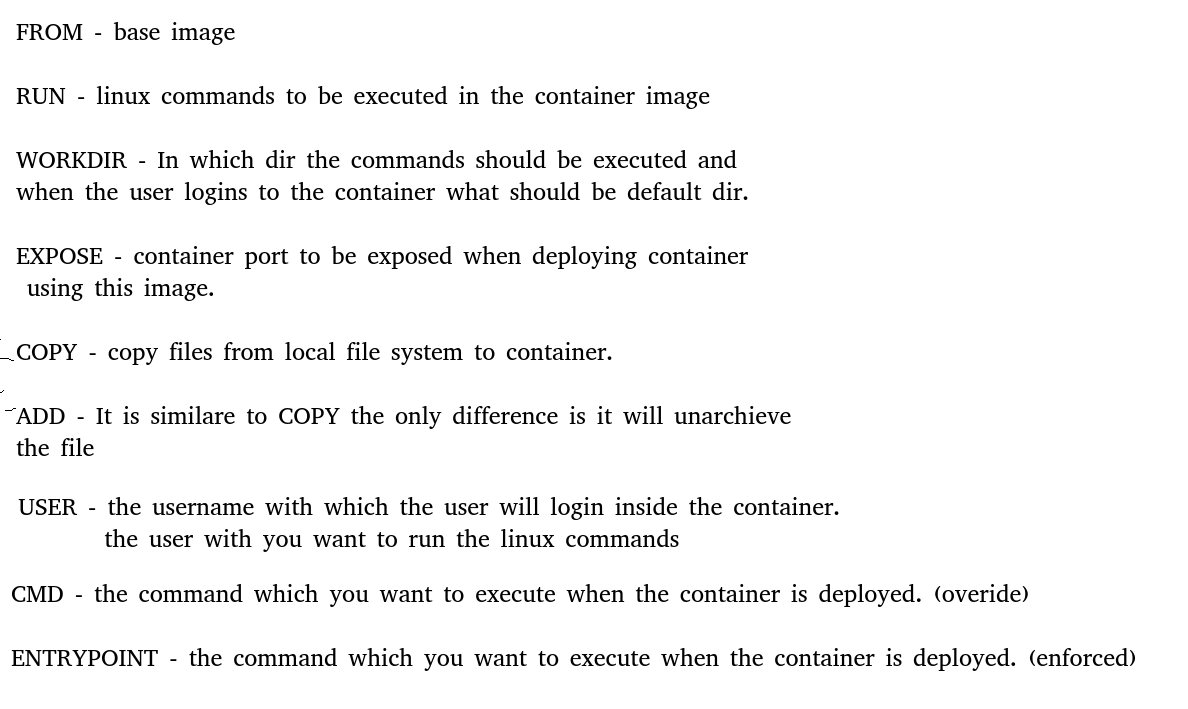
1. To buld the docker image from different dicker file

**To build the image from different file name other than Dockerfile**

docker build -f filename -t image\_name .

1. To run the docker image

docker run -d -p 80:80 --name webapp image\_name



**Home work-**

Build a docker image for a php application

<https://github.com/fajar7xx/olshopfajar>

**Day - 5**

**Docker file**

LABEL - (Metadata of Image)

ENV - (OS Env variable, which will be in the container when the container is deployed)

**Docker Networking**

1. Bridge
2. Host
3. None (also known as null)

**To List Network**

docker network ls

**To create a custom network**

docker network create --driver bridge mynetwork

**To use the network created**

docker run -d --network mynetwork --name web nginx

docker run -d --network mynetwork --name db nginx

docker exec -it web bash

apt update -y

apt install iputils-ping -y

ping db

**Homework -**

Mysql - <https://github.com/docker-library/mysql/tree/e0d43b2a29867c5b7d5c01a8fea30a086861df2b/8.0>

Wordpress - <https://github.com/docker-library/wordpress/tree/97f75b51f909fbd9894d128ea6893120cfd23979/latest/php7.4/apache>

Setup own Image Registry -

<https://hub.docker.com/_/registry>

**Day 6**

**—-----**

**Docker Storage**

**—-------------------**

1. **Bind Mount**

mkdir /home/ubuntu/data

docker run -d -p 80:80 --name webserver -v /home/ubuntu/data:/usr/share/nginx/html nginx

1. **Docker volumes**

**To create docker volume**

docker volume create mydata

**To list docker volumes**

docker volume ls

**To delete docker volume**

docker volume rm volume\_name

docker run -d -p 80:80 --name webserver -v mydata:/usr/share/nginx/html nginx

**Docker Compose**

sudo apt install docker-compose -y

**Docker-compose.yml**

version: "3"

services:

 db:

   image: mysql

   ports:

     - "3306:3306"

   environment:

     - MYSQL\_ROOT\_PASSWORD="redhat"

     - MYSQL\_DATABASE="wordpress"

     - MYSQL\_USER="wordpress"

     - MYSQL\_PASSWORD="redhat"

   volumes:

     - "mydbvol:/var/lib/mysql"

 wordpress:

   image: wordpress

   ports:

     - "80:80"

   volumes:

     - "mywordpress:/var/www/html"

volumes:

  mydbvol:

  Mywordpress:

Troubleshooting:

1. Docker exec -it wordpress bash

* Ping (connectivity)
* Connect to DB

apt update -y

apt install iputils-ping -y

ping db

apt install mysql -y

mysql -h db -u wordpress -p

password:

Access Denied

Hint:

Mysql remote connection = localhost

—----------------------------------------------------

**Day -7**

**Docker-compose integration with Dockerfile**

1. **//Create folder**

mkdir project

1. **//create docker file**

touch Dockerfile

1. **//docker file command**

FROM nginx

COPY index.html /usr/share/nginx/html

EXPOSE 80

1. **//create index.html**

echo “<h1> Hello World </h1>” >> index.html

1. **//create docker composes file**

touch docker-compose.yaml

1. **//Docker composes file command**

version: "3"

services:

 web:

  build:

    context: .

  ports:

   - "80:80"

1. **To apply the docker-compose.yml file**

docker-compose up

1. **To run the containers in background**

docker-compose up -d

1. **To stop & remove the containers deployed through docker-compose**

docker-compose down

1. **To list running containers**

docker-compose ps

1. **To check logs**

           docker-compose logs

1. **To validate the docker-compose file**

docker-compose config

1. **To apply docker-compose with a different file name other than docker-compose.yaml**

            docker-compose -f filename.yaml up

1. **To build the image and deploy the containers**

docker-compose build --no-cache

            docker-compose up