## DOCKER Commands

1. Sudo apt install docker.io
2. Sudo systemctl enable docker.service
3. Sudo systemctl start docker.service
4. Sudo systemctl status docker.service

*docker info*
*docker version*
*docker help*
*docker images*
*docker ps*
*docker ps -a*
*docker inspect container_name docker stop container_name docker logs container_name*

- List available images
  ***docker images***

- Find more details about docker images
  ***docker image inspect imageName***

- Working with a real world imageName
  ***docker pull nginx***

- Pull specific version of the image
  ***docker pull nginx:1.12-alpine-perl***

- Try to run image
  ***docker run -p 80:80 nginx***

- Let's try to make a curl request to see if the nginx webserver is running.
  ***curl http://localhost:80***

- List out all the running containers
  ***docker ps***

- Let's stop running containers
  ***docker stop container_id docker kill container_d***

- Docker container start
  ***docker start container_id***

- List out the available containers and check whether it is running or not
  ***docker ps***

- Try following command
  ***docker ps -a***

- Remove container in your docker environment
  ***docker rm container_id***

- List out all available images
  ***docker images***

- Remove any image from docker
  ***docker rmi image_id***

- Get login with docker hub
  ***docker login***

- Tag docker file with docker hub repository
  ***docker tag docker_id/image_name***

- Upload docker image to docker hub for everyone use
  ***docker push docker_id/image_name***

- Access the working container bash terminal
  ***docker exec -it container_id bash***

- Create a new image with existing image
  ***docker commit container_id new_name***

- Export a container to the PC
  ***docker export --output="any_name" contianer_id***

- Export an image to the PC
  ***docker save -o image_name <repo>:<tag>***

- Import an image into the docker from PC
  ***docker load -i image_name***

**docker rm $(docker ps -aq)**

create web using nginx

cretae two files

index.html
Dockerfile

**only nginx**

```
FROM ubuntu
MAINTAINER dimuthu
Run apt-get update
Run apt-get install -y nginx
ENTRYPOINT ["/usr/sbin/nginx","-g","daemon off;"]
EXPOSE 80
```

```
FROM ubuntu
MAINTAINER dimuthu
Run apt-get update
Run apt-get install -y nginx
ADD index.html /var/www/html/index.nginx-debian.html
ENTRYPOINT ["/usr/sbin/nginx","-g","daemon off;"]
EXPOSE 80
```

```
docker build -t web_1:1.0 .
docker run -p 80:80 web_1:1.0
docker run -p 8001:80 web_1:1.0
```

***OTHER WAY (This is easy way)***

```
FROM nginx
MAINTAINER dimuthu
RUN apt-get update
RUN apt-get install -y nginx
COPY . /usr/share/nginx/html
```

```
FROM nginx
COPY . /usr/share/nginx/html
```

```
docker build -t web_3:1.0 .
docker run -p 8001:80 web_3:1.0
```

**Wordpress**

```
docker pull wordpress
docker pull mysql
```

```
touch docker-compose.yml
vim docker-compose
```

```
version: '3'
```

```
services:
  wordpress:
    image: wordpress
    ports:
      - "8880:80"
    depends_on:
      - db
    volumes:
      - wordpress_files:/var/www/html
    environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: my_wordpress
    restart: always

  db:
    image: mysql
    volumes:
      - db_data:/var/lib/mysql
    environment:
      MYSQL_ROOT_PASSWORD: mydb
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: my_wordpress
    restart: always

volumes:
  wordpress_files:
  db_data:


run - docker-compose up -d
```