**DOCKER USEFUL COMMANDS**

* In Linux(Ubuntu)

1. **First Method**
   * sudo apt update (Auto update the environment)
   * sudo apt install docker.io -y (Docker Installation)
   * docker --version (Show Version)
   * sudo apt remove docker.io -y (Docker Uninstallation)
   * docker --version
2. **Second Method**
   * *Set Up The Docker Repository*
     1. sudo apt-get update (Auto update the environment)
     2. sudo apt-get install apt-transport-https ca-certificates curl software-properties-common
     3. curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –
        1. sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"
   * *Install Docker CE*
     1. sudo apt install docker-ce (Docker Community Edition Installation)
     2. docker –version
3. apt update && apt install nano
4. sudo docker images (Show images)
5. sudo su (Root Access)
6. docker pull ubuntu (Retrive Images)
7. docker run -it -d ubuntu (Create Container)
8. docker ps -a (Show all container including stopped)
9. docker run -it -d --name DataEngineer ubuntu (Create container with name “DataEngineer” in “ubuntu” image)
10. docker run -it -d --name DataScience -p 80:80 ubuntu (Run imgaes)
11. sudo docker exec -ti DataScience bash (Access inside container)
12. ls (show inside container directory)
13. apt update (update container)
14. apt install nginx -y (install nginx)
15. service nginx status (check the status of nginx)
16. service nginx start (Start the nginx)
17. service nginx stop (Stop the nginx)
18. docker stop “containerID” (Stop the running container)
19. docker start “containerID or Name” (start the container)
20. docker kill “containerID” (Forcefully stop the container)
21. docker restart “containerID” (Restart the container)
22. docker rm “containerID” (Remove stopped container)
23. docker rm -f “containerID” (Forcefully Remove running container)
24. docker commit “containername” filename (Save image of docker setting)
25. docker login (SignIn in your docker hub accoount)
26. docker tag ubuntu amitkothiyalutknd/ubuntude (Rename ubuntu image as ubuntude)
27. docker rename usaws usanz (Rename usaws container as usanz)
28. docker push amitkothiyalutknd/ubuntude (Upload image in dockerhub account)
29. docker rmi amitkothiyalutknd/ubuntude (Remove Images)
30. docker run -it -d -p 80:80 amitkothiyalutknd/ubuntude
31. mkdir dockerstudy
32. cd dockerstudy
33. ls (list all the files & directories of current directory)
34. nano dockerfile
35. docker build -t pythonlanguage .
36. docker file format (For creation of images in docker)
    * *First Format*
      1. FROM ubuntu
      2. LABEL learner=”AmitKothiyal”
      3. RUN apt update && apt install nginx -y
      4. WORKDIR /apps
      5. COPY ./index.html .
      6. CMD ["service", "nginx", "start"]
    * *Second Format*
      1. From nginx:alpine
      2. LABEL maintainer=”AmitKothiyal”
      3. COPY index.html /user/share/nginx/html
      4. WORKDIR /apps
37. sudo docker build -t ubuntuimagenginx (Build the images with name “ubuntuimagenginx”)
38. sudo docker run -it -d -p 80:80 --name aplinecont ae795f991f30
39. apt update && apt install git -y (Github installation)
40. git init (git initialization)
41. git add . (Addition of directory)
42. git commit -m "Added DockerFile" (Upload Dockerfile)
43. git remote add origin <https://github.com/amitkothiyalutknd/DockerStudyMaterial.git>
44. git push origin master
45. sudo docker tag aac67a14f3d4 amitkothiyalutknd/dockerimg (Rename images to “amitkothiyalutknd/dockerimg”)
46. sudo docker login (Sign in docker hub account)
47. sudo docker push amitkothiyalutknd/dockerimg (Upload images on dockerhub account)
48. sudo docker logout (Sign out docker hub account)
49. **Creation of docker volume**
    1. sudo docker volume create dockstores *(Create the docker volume having name dockstores)*
    2. sudo docker volume create dockstoresRO *(Create the docker volume in mode of Read Only)*
50. sudo docker volume ls (Show all the listed docker’s volume with driver)
51. sudo docker volume inspect dockstores (Show the configuration of dockstores volume as JSON format)
52. **Deletion of docker volume**
    1. sudo docker volume rm dockstores *(Remove the mentioned docker volume)*
    2. sudo docker volume prune *(Remove all the exist dockers volume simultaneously)*
53. **Run the container with attached volume**
    1. sudo docker run -it -d --name usaws --mount source=dockstores target=/apps ubuntu *(Create or open exist the docker volume having name “dockstores” on location “/apps” with attached container “usaws” in image “ubuntu” )*
    2. sudo docker run -it -d –name usaws --volume dockstores:/apps ubuntu *(Generally declared the docker volume having name “dockstores” on location “/apps” with attached container “usaws” in image “ubuntu”)*
    3. sudo docker run -it -d --name usaws --mount source=dockstores, target=/apps, readonly, ubuntu *(Create or open exist container with attached docker volume in mode of readonly )*
    4. sudo docker run -it -d --name usaws --mount type=bind, source=$(pwd), target=/apps ubuntu *(Create or open exist container with attached docker volume with mentioned type “bind”*)
    5. sudo docker run -it -d --name usaws --mount type=bind, source=$(pwd)/fun, target=/apps, readonly ubuntu *(Create or open exist container with attached docker volume with mentioned type “bind”*)
    6. sudo docker run -it -d --name usaws --mount type=tmpfs, target=/apps ubuntu *(Create or open exist container with attached docker volume with mentioned type “tmpfs”, no require to mention source*)
54. sudo docker container inspect usaws (show the configuration of “usaws" volume as JSON format)
55. touch cont.txt (Create file in current directory)
56. nano cont.txt (create new one or open if file already exist)
57. vi cont.txt (Open file)
58. cat cont.txt (Show the content of file)
59. **Creation of docker networks**
    1. sudo docker network create --driver bridge brinet (Create docker network of bridge type having name “brinet”)
    2. sudo docker network create --driver overlay2 ovlynet (Create docker network of overlay2 type having name “ovlynet”)
60. sudo docker network ls (Show the all the listed docker networks)
61. sudo docker network inspect brinet (Show the configuration of mentioned docker network)
62. apt update && apt install iputils-ping -y (update & install the ping inside the current container)
63. sudo docker run -it -d --name usaws --network dataengineer ubuntu (Run the container with mentioned attached network)
64. sudo docker run -it -d --name usaws --network host nginx:latest (Run the container with attached host type network)
65. curl localhost (Run the container application inside the host network)
66. sudo docker network rm dataengineer (Remove network having name ‘dataengineer’)
67. sudo docker network prune (Remove all unused networks)
68. sudo docker swarm init (It initialize the swarm, made current running node into a manager)
69. sudo docker service create --name tsla --network ntwrk --replicast 3 nginx:latest (Start service having name “tsla” in network “ntwrk”& make 3 copies of nginx engine)
70. sudo docker service ps tsla (Show all the running container inside the “tsla” service)
71. sudo docker run -it -d --name usaws --network none ubuntu
72. DOCKER\_CONFIG=${DOCKER\_CONFIG:-$HOME/.docker} (installs Compose for the active user under $HOME directory)
73. mkdir -p $DOCKER\_CONFIG/cli-plugins (Docker Compose for all users on your system, replace ~/.docker/cli-plugins with /usr/local/lib/docker/cli-plugins)
74. Installation of docker compose
    1. curl -SL https://github.com/docker/compose/releases/download/v2.17.3/docker-compose-linux-x86\_64 -o $DOCKER\_CONFIG/cli-plugins/docker-compose (downloads the latest release of docker Compose)
    2. sudo apt-get update && sudo apt-get install docker-compose-plugin
75. chmod +x $DOCKER\_CONFIG/cli-plugins/docker-compose (Apply executable permissions to the binary)

or

sudo chmod +x /usr/local/lib/docker/cli-plugins/docker-compose (if you chose to install Compose for all users)

1. docker compose version (Show the docker compose version)
2. nano dockerfile (Create dockerfile)
   * FROM ubuntu (file script to build an ubuntu images on run)
3. sudo docker build -t ubuntu1 .(To run dockerfile script for creation of ubuntu image as a name ubuntu1)
4. nano docker-compose.yml
   * version: '3'
   * networks:
   * virginia:
   * driver: bridge
   * services:
   * web:
   * image: "nginx:latest"
   * ports:
   * - "5000:5000"
   * networks:
   * - virginia
   * database:
   * image: "mysql"
   * networks:
   * - virginia
   * volumes:
   * - db\_data:/var/lib/mysql
   * volumes:
   * db\_data: {}
5. sudo docker compose build (Build the docker-compose file)
6. sudo docker-compose up (Execute the written scripts of docker-compose.yml file)
7. sudo docker-compose down (Stop the running services of docker-compose.yml file)
8. nano docker-compose.yml
   * version: '3.3'
   * services:
   * db:
   * image: mysql:latest
   * volumes:
   * - db\_data:/var/lib/mysql
   * restart: always
   * environment:
   * MYSQL\_ROOT\_PASSWORD: admin786
   * MYSQL\_DATABASE: wordpress
   * MYSQL\_USER: wordpress
   * MYSQL\_PASSWORD: wordpress
   * wordpress:
   * depends\_on:
   * - db
   * image: wordpress:latest
   * ports:
   * - "8000:80"
   * restart: always
   * environment:
   * WORDPRESS\_DB\_HOST: db:3306
   * WORDPRESS\_DB\_USER: wordpress
   * WORDPRESS\_DB\_PASSWORD: wordpress
   * WORDPRESS\_DB\_NAME: wordpress
   * volumes:
   * db\_data: {}
9. sudo docker compose build (Build the docker-compose file)
10. sudo docker-compose up -d (Execute the written scripts of docker-compose.yml file)
11. sudo docker-compose down (Stop the running services of docker-compose.yml file)
12. sudo docker-compose ps (Show all the running services by docker-compose)
13. sudo docker-compose run db env (Show all the running database)
14. sudo docker-compose stop
15. sudo docker swarm init --advertise-addr 35.154.33.200 (Made current node as a manager node)
16. sudo docker swarm join --token SWMTKN-1-3vrq9dgv5enf4xbfpdc04r2q1nq3jenf2cxmmst6zk1e8vwk8s-cjpg6544hpqxca8aa2t9icn8v 35.154.33.200:2377 (This token provide by manager node. To make other running node as a worker node of current manager, have to run this token command on the remaining node)
17. sudo docker node ls (Show the members node of current swarm including manager node & worker nodes)
18. sudo docker info
19. sudo docker swarm leave (Remove current worker node from the swarm group)
20. sudo docker swarm join-token worker (Provide join token to add again other node as worker node of swarm group)
21. sudo docker node rm so53rhoxhuqd1wke0mx6gingg (Remove worker node from docker swarm)
22. sudo docker swarm leave --force (Remove current manager node to leave docker swarm also stops the docker swarm)
23. sudo docker swarm join-token manager (It gives us to join-token command to made other node as a manager node)
24. sudo docker service create --name apache\_serv --replicas 4 -p 80:80 httpd (Run & replica apache service in docker swarm)
25. sudo docker service ls (Show all the running services)
26. sudo sudo docker service ps apache\_serv (Show all the running services under apache\_serve)
27. sudo docker rm -f apache\_serv.2.kpkkt9v1ul8r4lap85ta72dvz (Forcefully crash the current running service having id “apache\_serv.2.kpkkt9v1ul8r4lap85ta72dvz”)
28. sudo docker service inspect apache\_serv (Show all the configuration of service having name “apache\_serv”)
29. sudo docker service rm apache\_serv (Remove docker service having name “apache\_serv”)
30. sudo docker stack deploy -c docker-compose.yml dwarm (Rum yml file on docker swarm)
31. sudo docker service ps dwarm\_wordpress (Listed all service having name “dwarm\_wordpress”)
32. sudo docker service scale dwarm\_wordpress=4 (4 Replication of service “dwarm\_wordpress”)
33. sudo docker stack services dwarm (Listed docker services having name combination of “dwarm”)
34. sudo docker node update --availability drain m7sbj6eujkh2ym6zaiwfv4f5o (Stop all the running container on current manager node)
35. sudo docker service update --image mysql:latest b5ohy5jpixk8 (Update image of particular service “b5ohy5jpixk8”)
36. sudo docker stack rm dwarm (Removed docker services having name combination of “dwarm”)
37. **AWS ECR & ECS**
    1. nano index.html
    2. nano dockerfile
    3. sudo docker build -t htmlwebsite .
    4. sudo docker run -it -d -p 80:80 htmlwebsite
    5. sudo aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 799345200116.dkr.ecr.ap-south-1.amazonaws.com
    6. docker build -t htmlwebsite .
    7. sudo docker tag htmlwebsite 799345200116.dkr.ecr.ap-south-1.amazonaws.com/htmlwebsite
38. **Prometheus & Grafana (Monitoring Tools)**
    1. sudo apt install git (git installation)
    2. git –version (Show git version)
    3. git clone https://github.com/amitkothiyalutknd/swarmprom.git (Copy repository from github account)
    4. cd swarmprom
    5. ls
    6. nano docker-compose.yml
    7. cd prometheus
    8. ls
    9. cat dockerfile
    10. cd config
    11. nano prometheus.yml
    12. cd grafana
    13. cd dashboards
    14. cat swarmprom-nodes-dash.json
    15. nano swarmprom\_dashboards.yml
    16. cd datasources
    17. nano prometheus.yaml
    18. sudo docker stack deploy -c docker-compose.yml dockmonitoring
    19. sudo docker service ls
    20. sudo docker service create --name docmon --replicas 3 nginx:latest