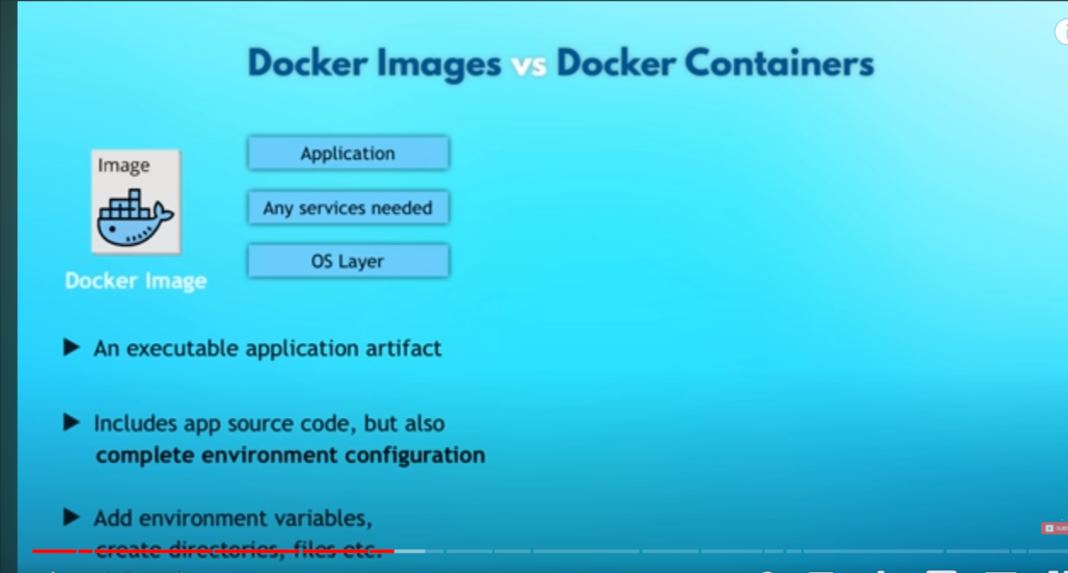
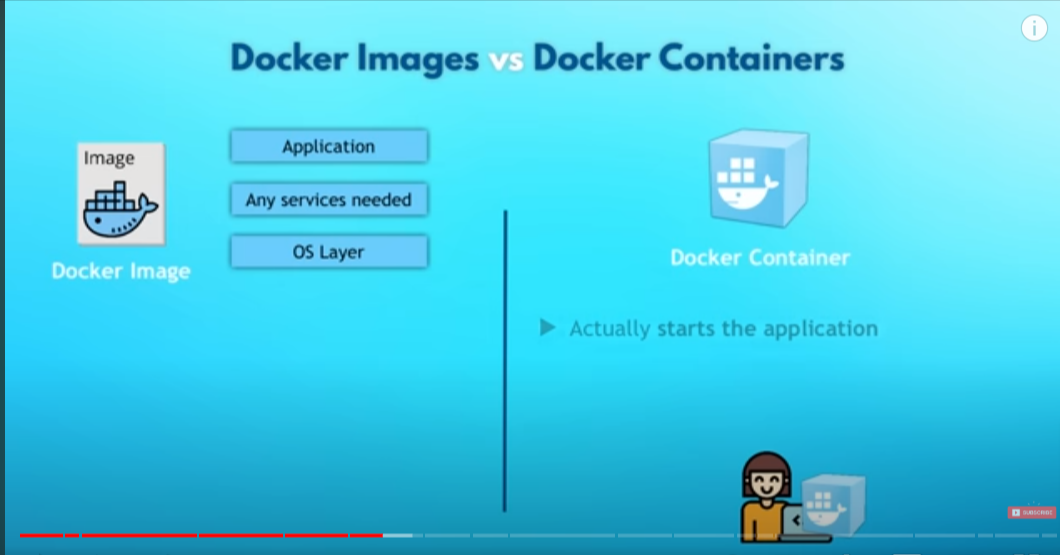


Docker images vs docker container

s



note : container is basically running instant of docker images

1.command to know the docker images :

in cmd run docker images

2.command to check docker container

in cmd run – docker ps

3.command to pull docker images:

docker pull name\_of\_image eg :docker pull nginx:1.23

4.command to run docker images(create container for that paticular docker image )

docker run image\_name:if\_any\_specific\_version

docker run nginx:1.23

5.to run docker images dtech from terminal(rnning in background) :

docker run -d image\_name:version

6.to know the docker logs of dtech docker conatiner

docker logs container\_id

7. to stop the running container in dtechmode :

docker stop container\_id

8.port binding to run the container in localhost:

docker run -d -p localhost:imagehost/containerhost image\_name:version\_if\_any

9. to know the all container i started

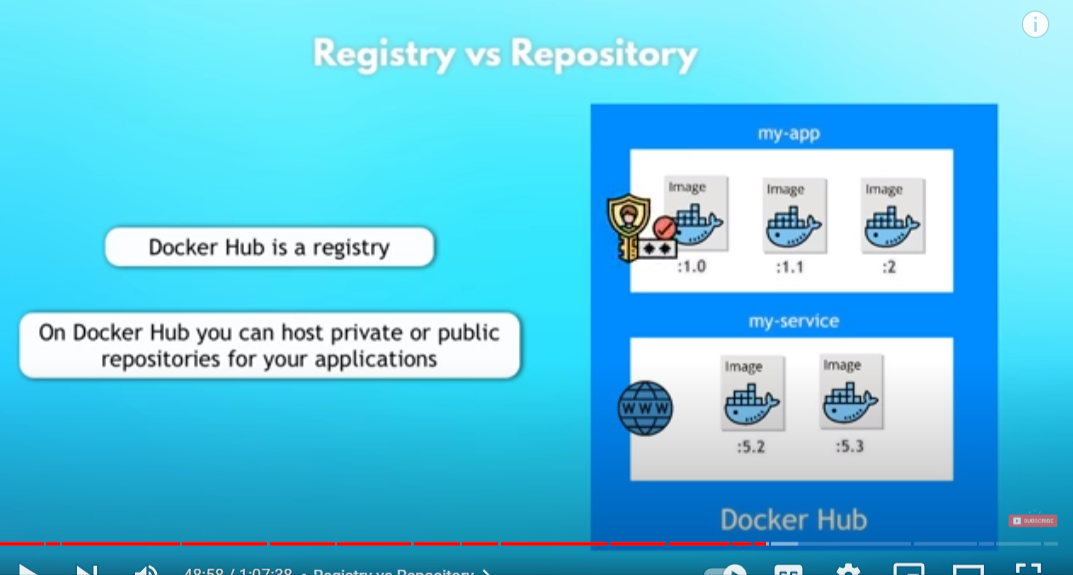
docker ps -a (all)

10. to start the all ready crated container

docker start container\_id (get from docker ps -a )

11. to give the name to conatiner

docker run --name give\_name -d -p localhost:host image\_name:version



12 . to delete docker images

docker rmi -f image\_id

13 . to delete conatainer

docker rm constainer\_id

14.

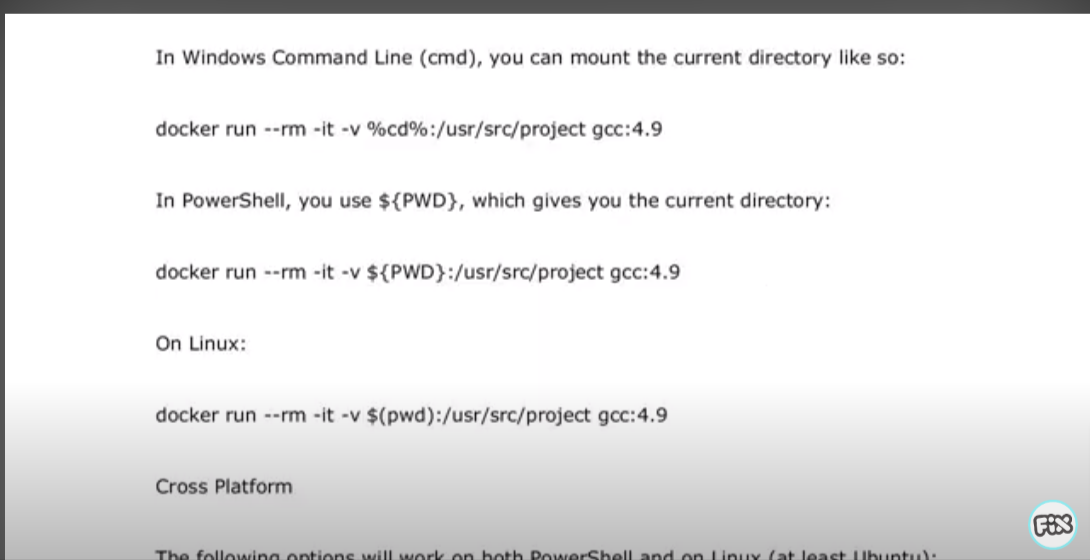
docker run --rm -ti --name conainer\_name -d -p 3000:3000 image\_name:version

--rm : remove the container as the container is stop

15. docker container changes as source container chages like we see in react for that called bind mount (add -v means )

docker run --rm -d -p 3000:3000 -v /$(pwd):/myapp reactapp

creating our own docker images:



16.Passing enviromental variable when during opening of images

docker run --rm -ti --name container\_name -d -p 3000:3000 -e enviometal\_variable\_name=value image\_name

for passing envfile for more varible

docker run --rm -ti --name container\_name -d -p 3000:3000 --env-file path(./config.env) image\_name

1. for devlopment purpose

step 1 : create docker file

1 . create a file in root directory of application as : Dockerfile.dev

2. create .dockerignore file

3 . open folder in cmd and run : docker build -f Dockerfile.dev -t reactapp .

docker build -f Dockerfile.dev -t reactapp .

note : t : tag

4 . after build we can run with the command

docker run -it --name container\_name -p 3000:3000 reactapp

note : it : intractive mode

2. Creating docker images for production purpose

3. docker-compose.yml

after bulid : run -> docker-compose up -d

to stop : docker-compose down

pushing docker images to docker repo :

1 .creating resp on docker hub

2. creating tag for images

docker tag image\_name:image\_tag ajajalam1399/docker\_first\_repo:tag(latest)

3. pushing tag to that resp