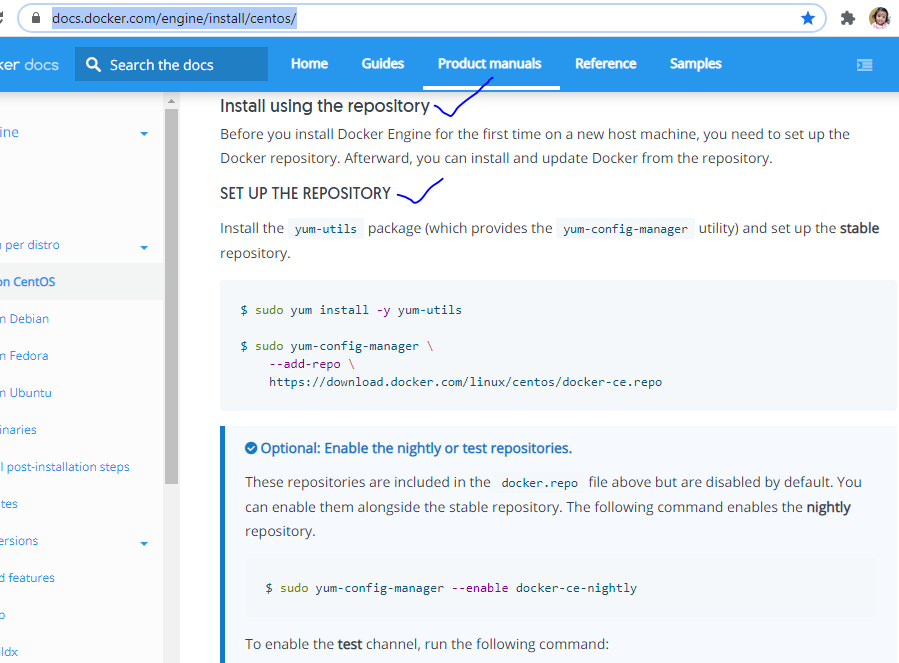
**Docker Tutorial**

**Install Docker through repo’s:- Scenario1 2**

[**https://docs.docker.com/engine/install/centos/**](https://docs.docker.com/engine/install/centos/)

****

**sudo yum install -y yum-utils -> for get all util packages**

**sudo yum-config-manager --add-repo** [**https://download.docker.com/linux/centos/docker-ce.repo**](https://download.docker.com/linux/centos/docker-ce.repo) **-> download docker repo**

**sudo yum-config-manager --enable docker-ce-nightly -> enabled docker-ce-nightly package**

**sudo yum-config-manager --enable docker-ce-test -> enabled docker-ce-test**

**sudo yum-config-manager --disable docker-ce-nightly -> after install disable the docker-ce-nightly package**

**sudo yum install docker-ce docker-ce-cli containerd.io ->install docker-ce-cli**

**Note: If you got any error reg broke or allowerasing or nobest...etc use this cmnd :**

**sudo dnf install docker-ce --allowerasing**

**yum list docker-ce --showduplicates | sort –r -> shows installed docker packages**

**sudo systemctl start docker -> start docker in cent os**

**sudo docker --version**

**sudo docker run hello-world -> run by default docker image**

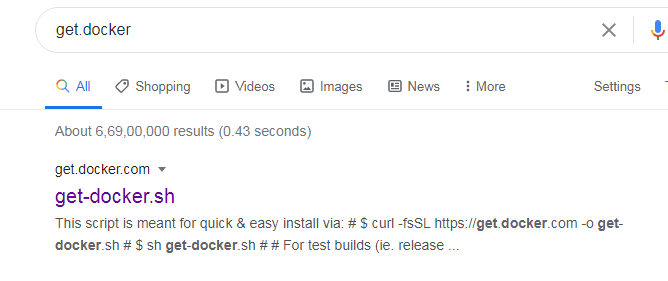
**docker images -> shows all downloaded docker images**

**docker ps –a -> shows all running docker services and exited services**

**docker ps -> shows only running docker image services**

**Un-Install docker: - sudo dnf remove docker-ce\* (remove the all packages )**

**Install Docker through script:- Scenario2**

****

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

**Open Linux machine:-**

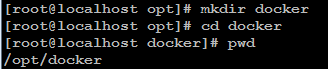
**Create one docker dir.**

**[root@localhost opt]# mkdir docker**

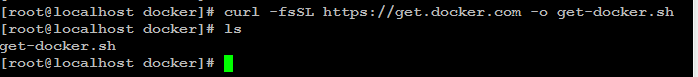
**[root@localhost opt]# cd docker**

**[root@localhost docker]# pwd**

**/opt/docker**

****

**curl -fsSL https://get.docker.com -o get-docker.sh** -> used to download the get-docker.sh file (same like wget cmnd).



sh get-docker.sh -> running script



**Scenario2** Installation end.

**How to uninstall the docker for through script installed :-**

[root@localhost ~]# yum list installed|grep docker -> find docker



[root@localhost ~]# yum -y remove docker-io.x86\_64 -> Remove that showed specific version.

OR

yum -y remove docker\*

OR

yum dnf -y remove docker\* (for remove packages)



OR

 remove docker dir.

Check again.



If you cannot find any lists like step 1 or step 2, that's mean you successfully uninstall  docker.

Refer: <https://africalocals.com/?/article/29>

Note: to verify check docker –version cmnd.

# Docker installation 3: Install Docker Desktop on Windows : <https://docs.docker.com/docker-for-windows/install/>

Download setup file and install it.

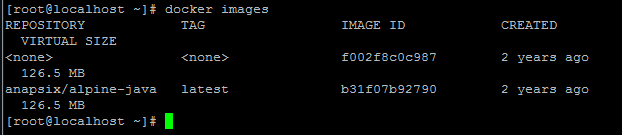
Show docker version



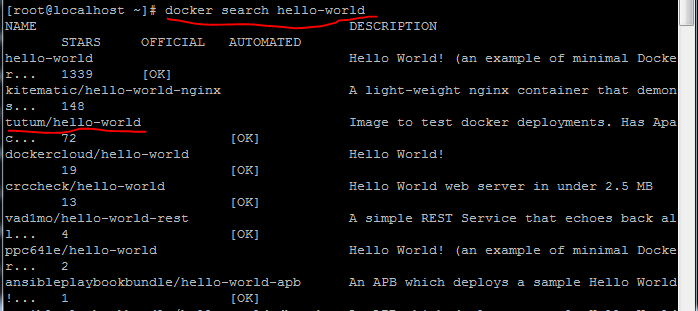
Start docker



Show images



Search images

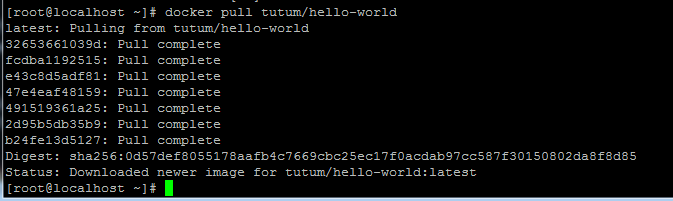


Pull image from centralize docker hub.

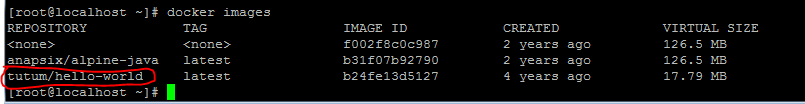
If you want to remove image: docker rmi bf756fb1ae65 (img id)

docker rmi **-f** 31e17b0746e4 ->img id

docker rmi -f **jenkins:2.60.3** ->img name:img tag



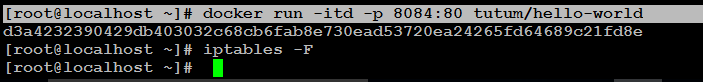
Search images



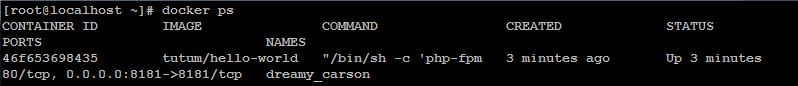
Run the image ->now image will run as containerized. (itd -> interactive terminal detached mode , p -port)

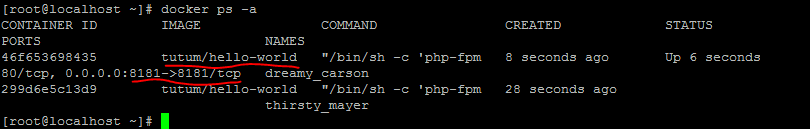
Note : provide image name or ID also.

[root@localhost ~]# **docker run -itd -p 8181:80 tutum/hello-world**



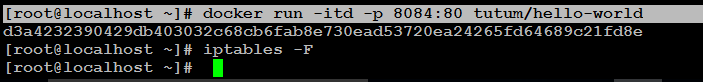
Running docker image.





**How to access project image in browser:-**

[root@localhost ~]# **docker run -itd -p 8181:80 tutum/hello-world**

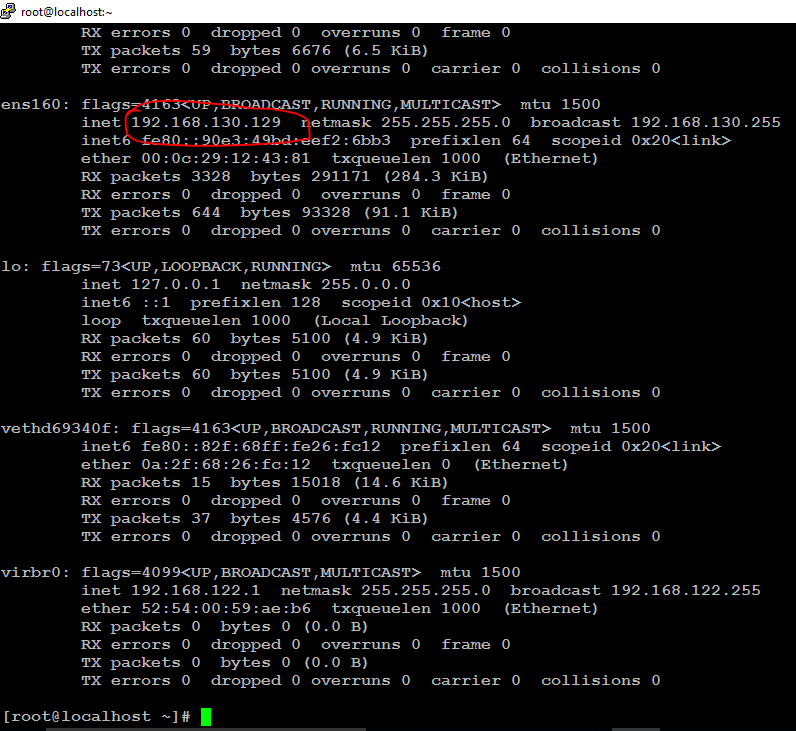


Map IP address in outside, enable the Firewalls .

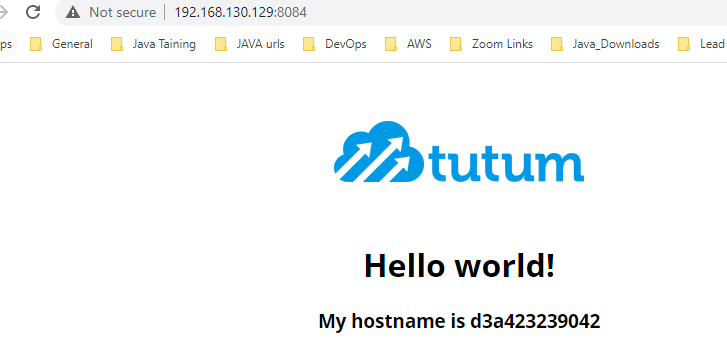




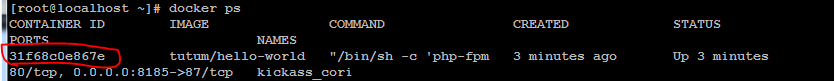
Get ip address : ifconfig or hostname -I



Access web appln in browser : <http://192.168.130.129:8181/>

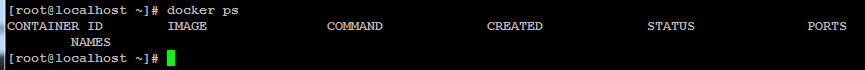


Stop docker image -> provide docker image id

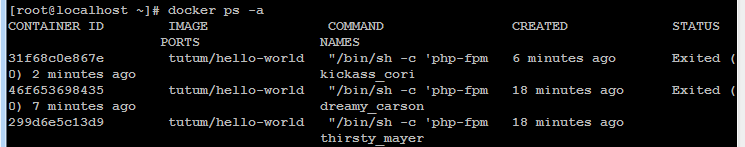




Verify -> nothing processing (ps -> only show running containers )



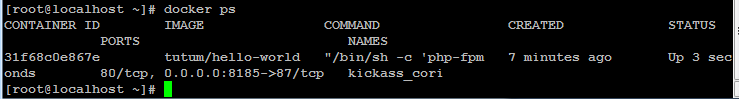
Start docker image (ps -a -> it will show running containers and exited containers also )

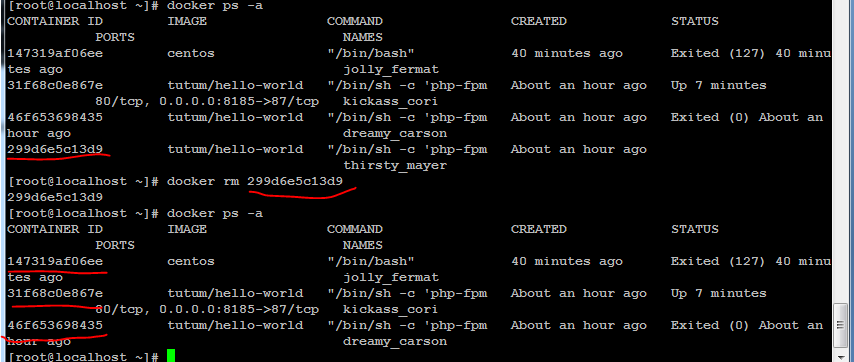


Provide cid to start

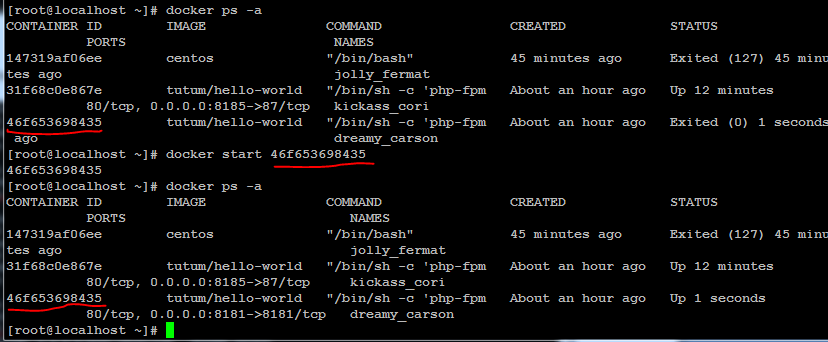


Verify running docker services

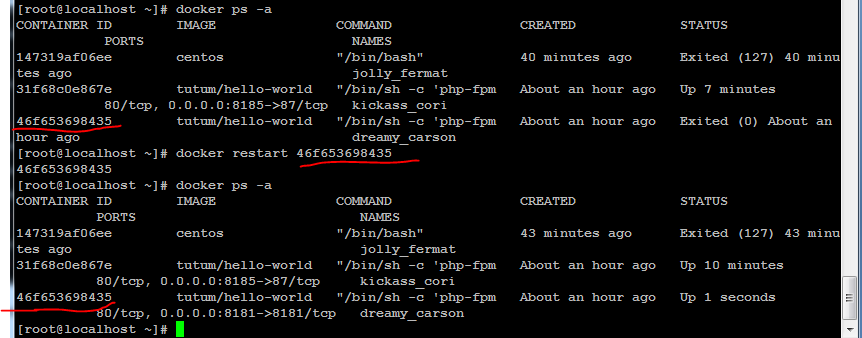


Remove container (rm -> remove stopped container )

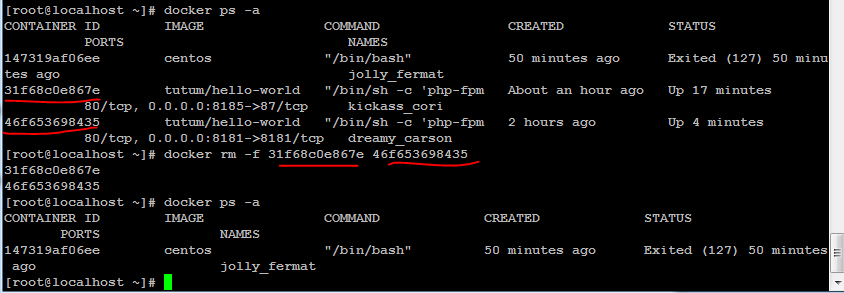
Start container



Restart container

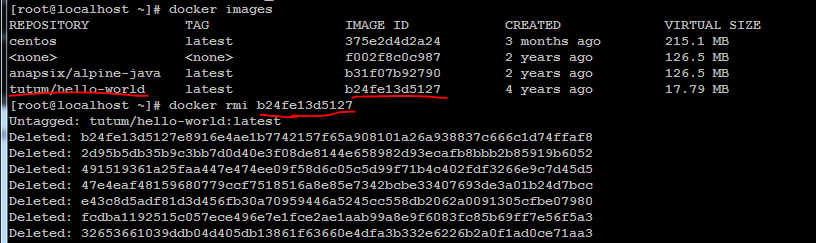


Remove multiple containers rm –f -> forcefully stop and remove the container)

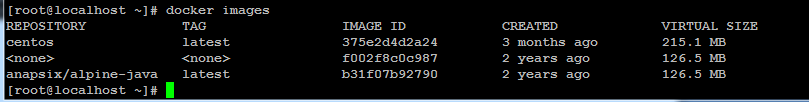


Remove images -> provide image id or image name (that image should be stopped)

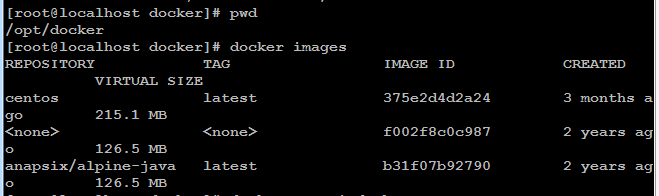
( rmi -> remove image)



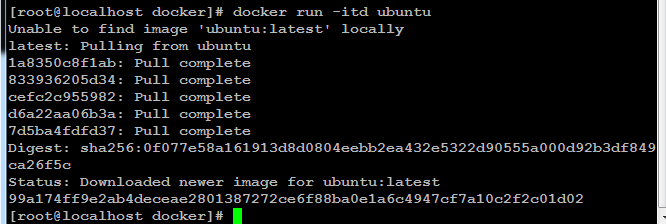
Verify



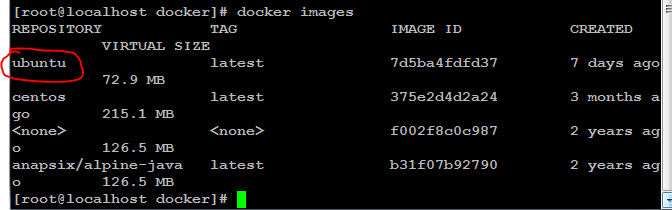
**Ubuntu OS image download without pull:-**

Ubuntu image is not pulled. 

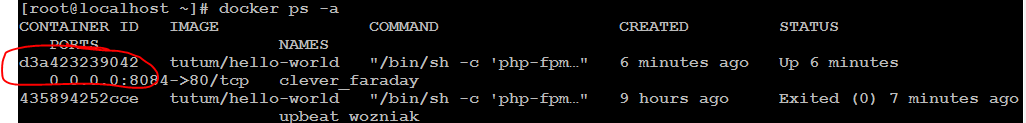
Without pull cmd, ubuntu image in local machine ,we can run directly it will download if not available image in local.

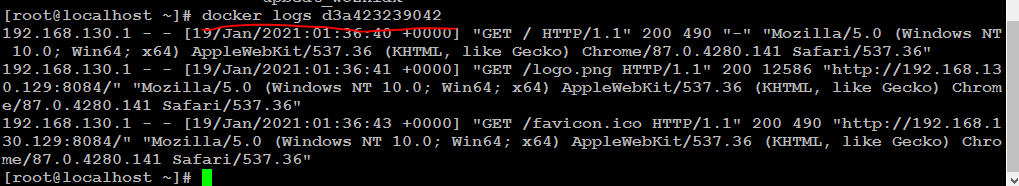


Ubuntu image downloaded.



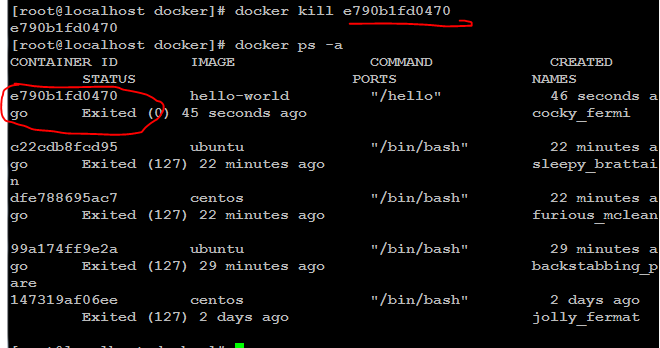
**Check container logs:- docker logs cid**



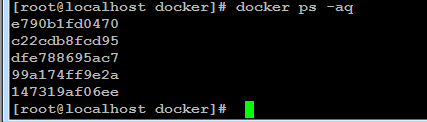


Note :- **docker stop cid** 🡪it will take 10 seconds to stop , but if we use kill cmnd immediately it will stop container.

Kill cid :- docker kill cid



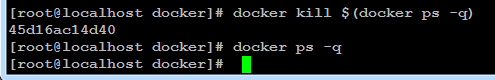
[root@localhost docker]# docker ps –aq -> Show all container ids .(running and not running cid’s)



[root@localhost docker]# docker ps –q -> it will show only running containers ids.



[root@localhost docker]# docker kill $(docker ps -q) ->Stop multiple running containers .



docker rm $(docker ps -aq) ->remove all containers

docker images –q 🡪shows only docker images ids.



docker attach cid -> goto inside of this cid.

Cd /opt

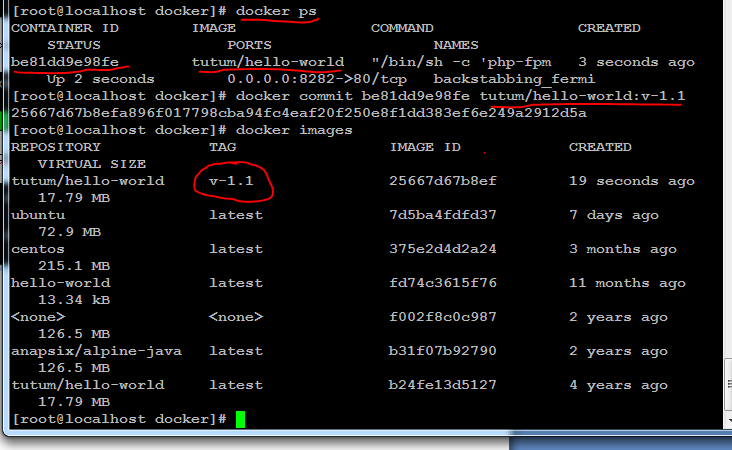
# touch f{1..20} -> creating 20 files

# mkdir f{1..20} -> creating 20 dir’s

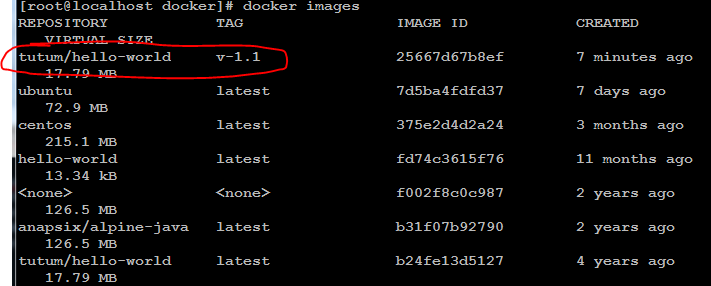
# yum install wget -> installing package

# ls -lt

Create tag version for image/cid with using container id. (that img should be in running mode)



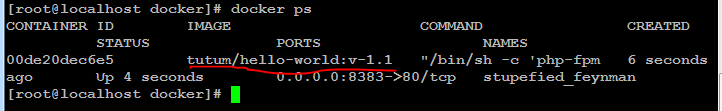
Only Run specific tag version image.



Run image name with tag version OR run with image id (docker run –itd img\_id)

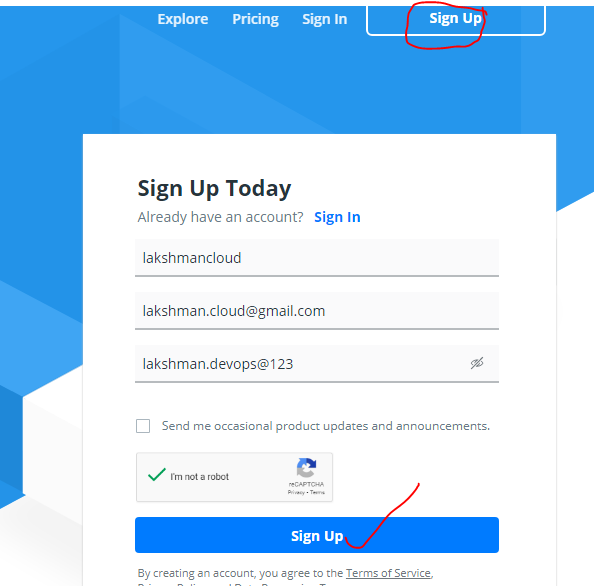


Verify

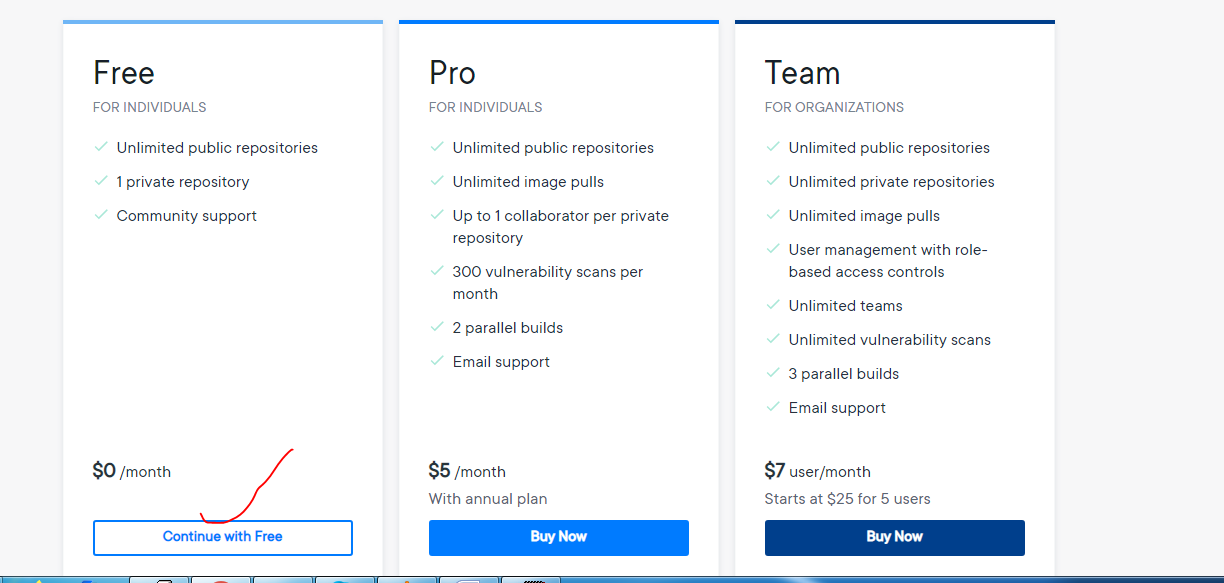


**Docker Hub Registry**:- create one account

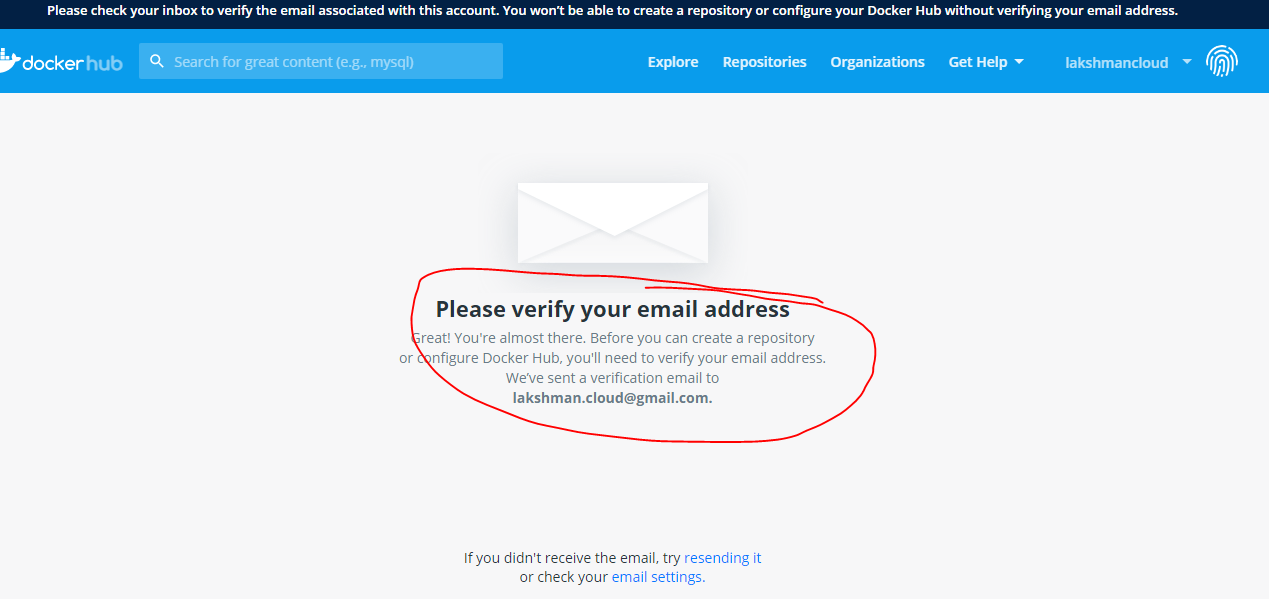
<https://hub.docker.com/>



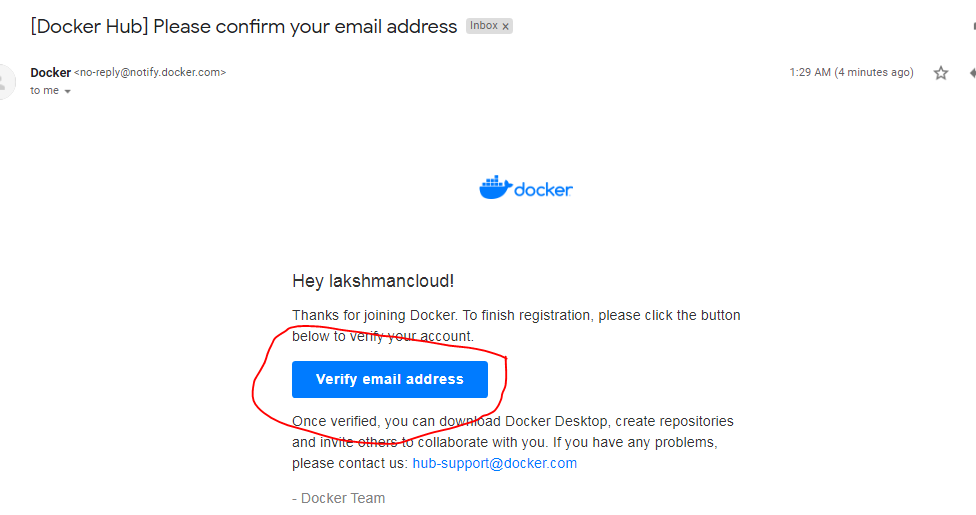
Choose free

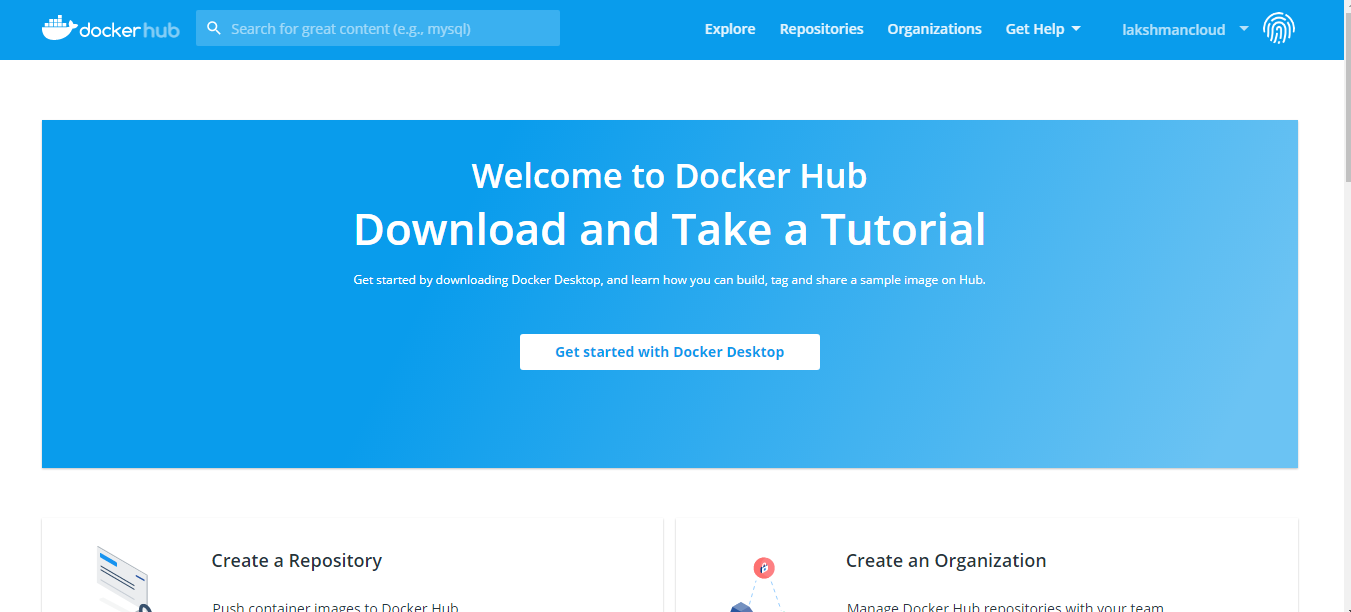


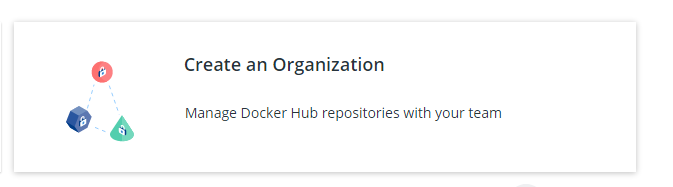
Verify email.

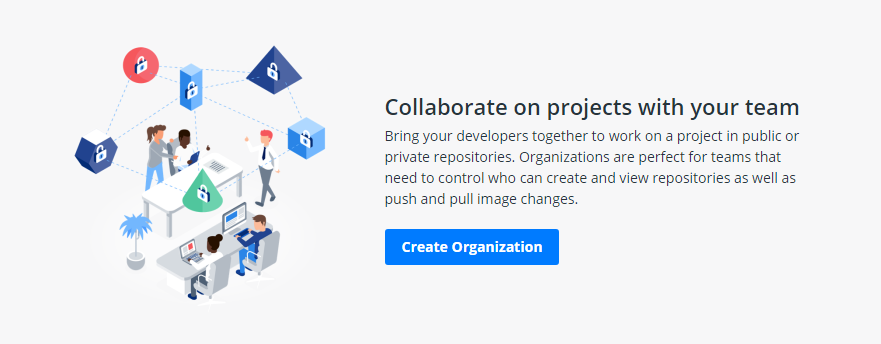


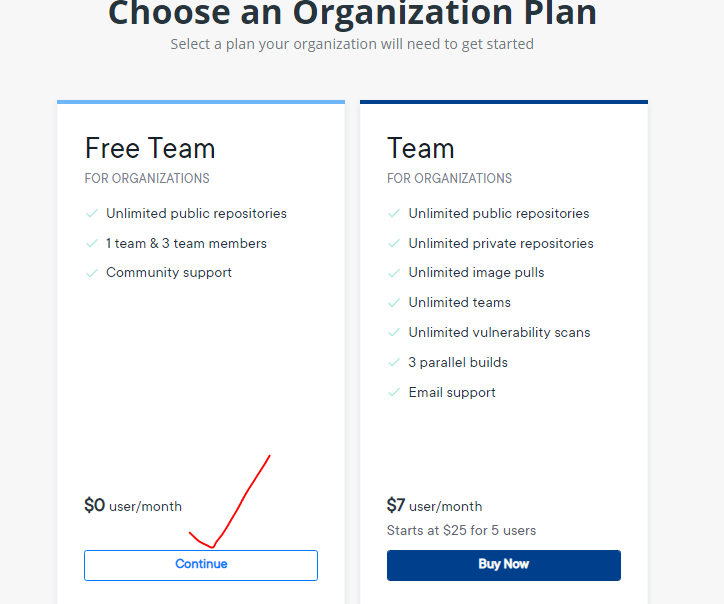
Complete email activation (check gmail inbox for activate email versification)



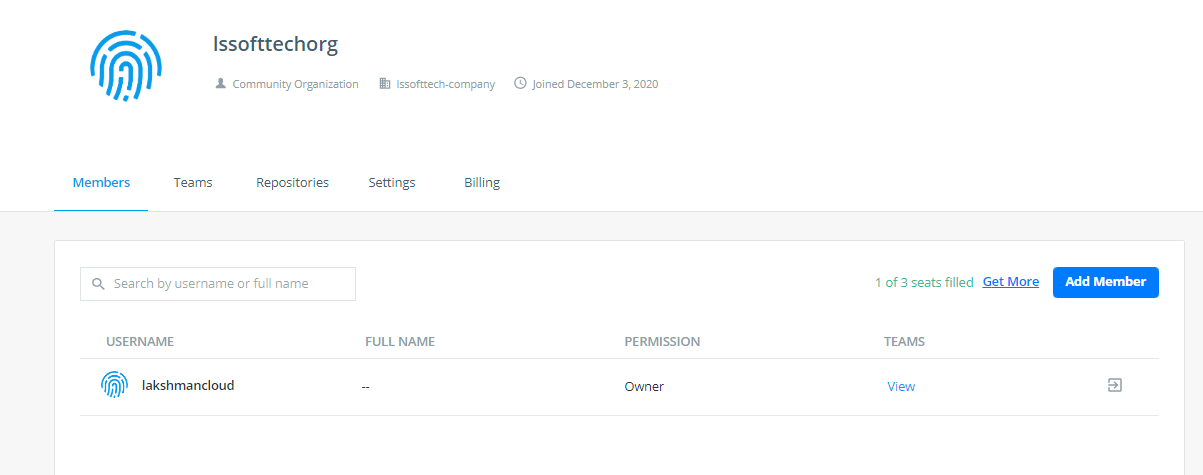




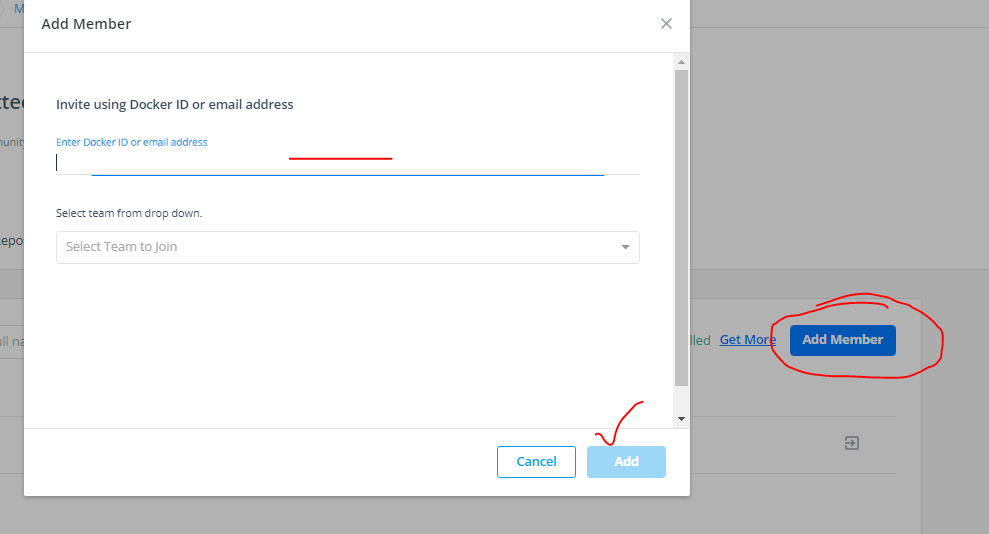


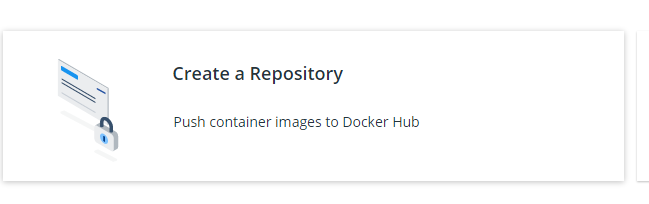


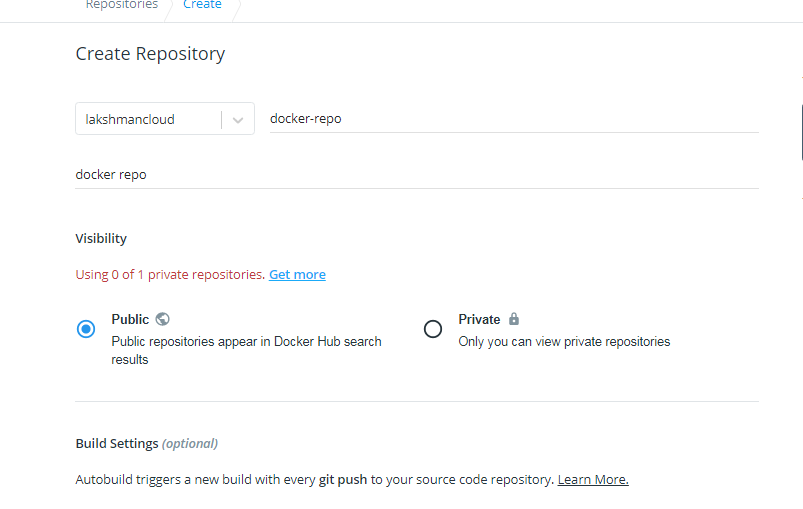


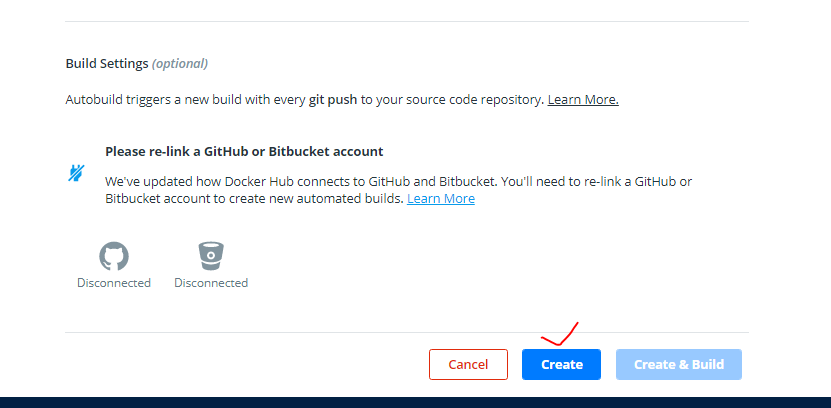


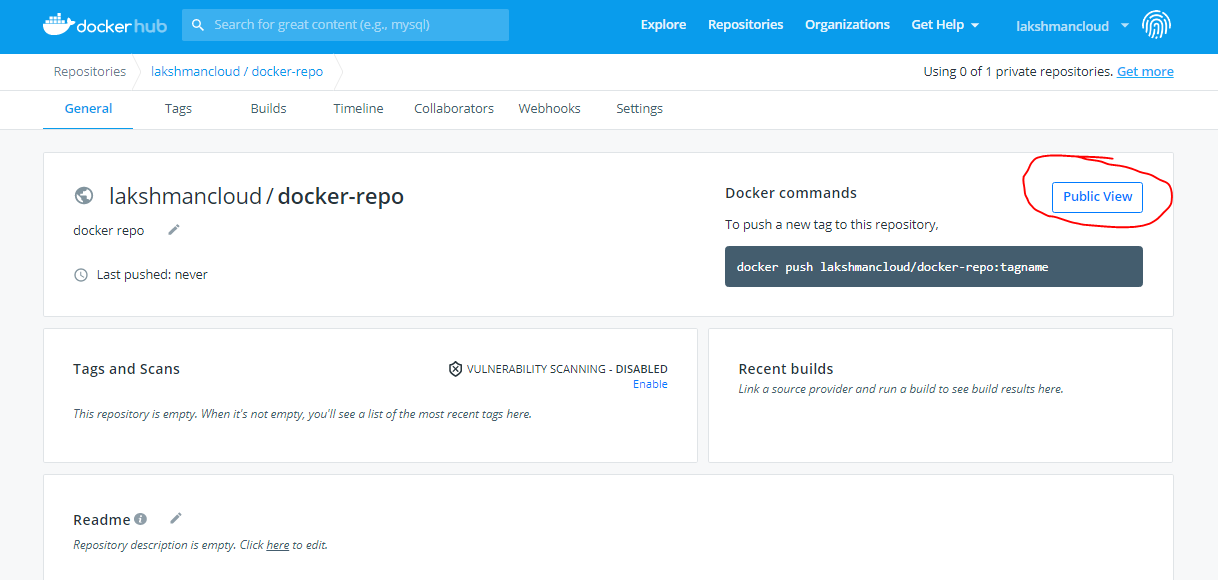
Add already registered users

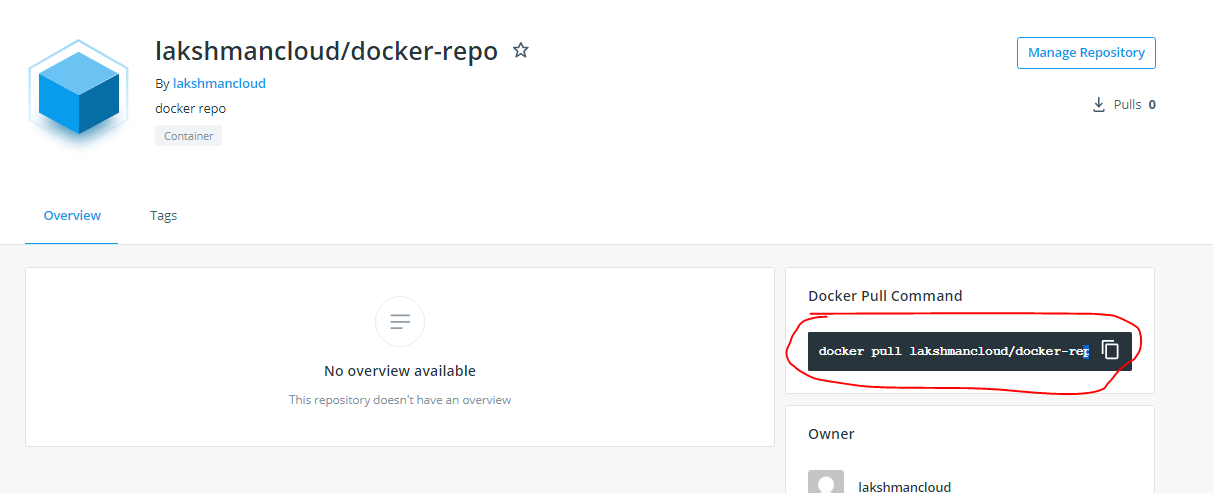






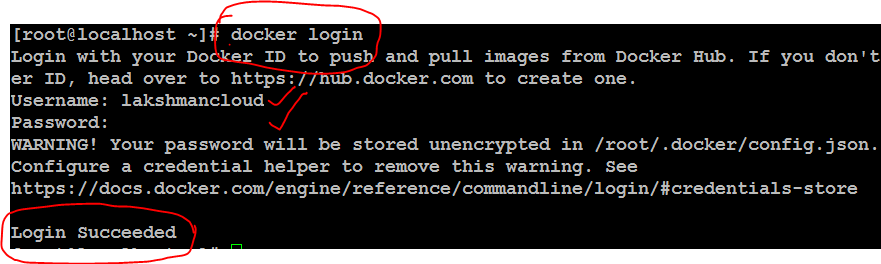




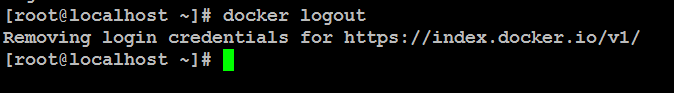


**Push image into docker hub:-**

# docker login 🡪enter un/pwd of docker hub registry ([lakshmancloud/lakshman.devops@123](mailto:lakshmancloud/lakshman.devops@123))

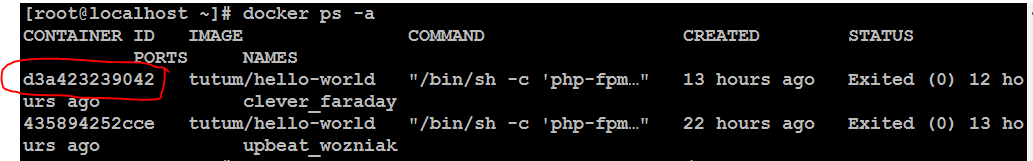


# docker logout 🡪 logout the account.



Again login the docker hub registry.

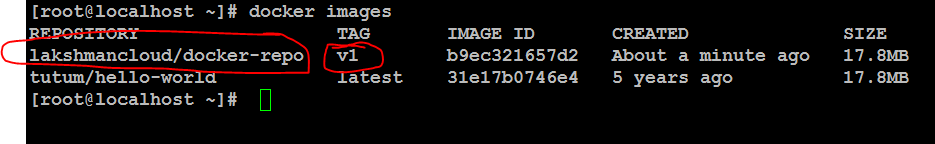
# docker ps –a



# docker commit **d3a423239042** lakshmancloud/docker-repo:v1 🡪 lakshmancloud - docker hub name, docker-repo -> docker repo name , v1 -> any tag version name.

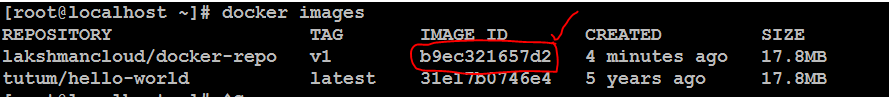


[root@localhost ~]# docker images



**How to change existing image tag name using img\_id :-**

[root@localhost ~]# docker images

****

# docker tag **b9ec321657d2** lakshmancloud/docker-repo:v2

# docker images 🡪 verify (image name and id will be same, only img tag is changed)



# docker push img\_name -> here img\_name related all tag versions(v1&v2) will pushed.

OR

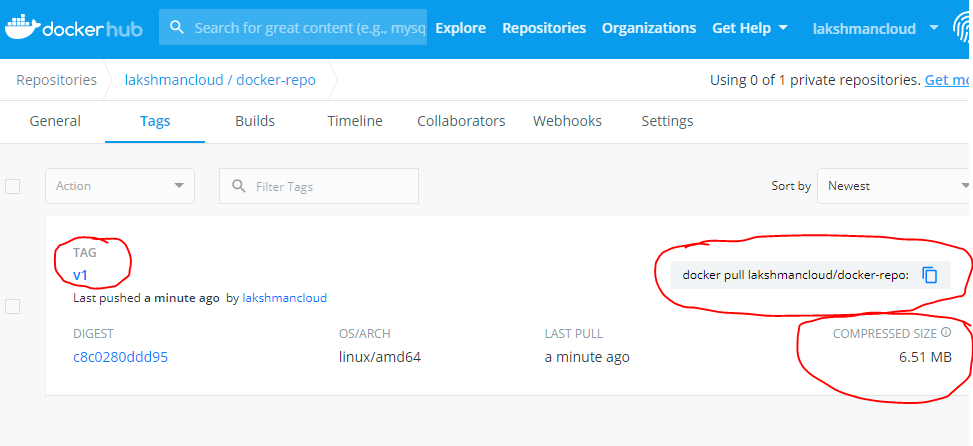
#docker push img\_name:tag\_version -> here only specific tag version will pushed.

**[root@localhost ~]# docker push lakshmancloud/docker-repo:v1**

Note: it will do archive(zip) and push the image into docker hub registry, when we pull again it will extract the img zip file and download it.

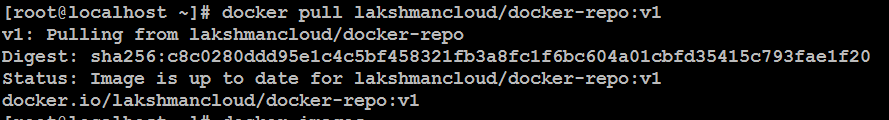


Image v1 is pushed to docker hub registry.(refresh docker hub registry url)

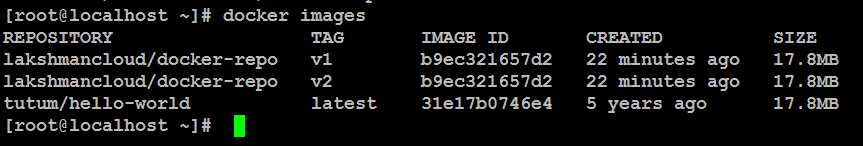


[root@localhost ~]# docker pull lakshmancloud/docker-repo:v1 🡪 pull the v1 img to local docker machine

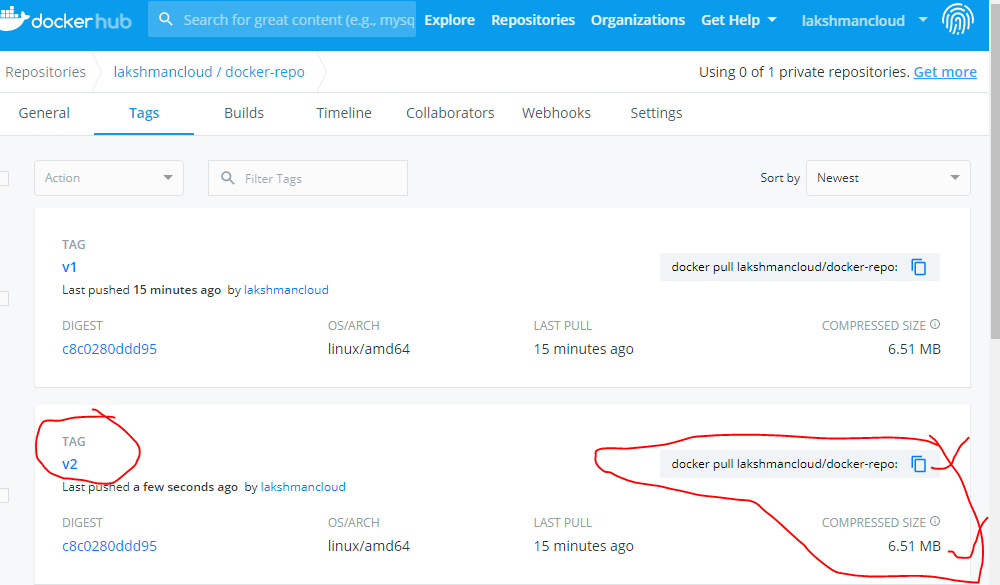
Note: if not given any tag name(v1) , it will pick **latest** tag name if it will available.

****

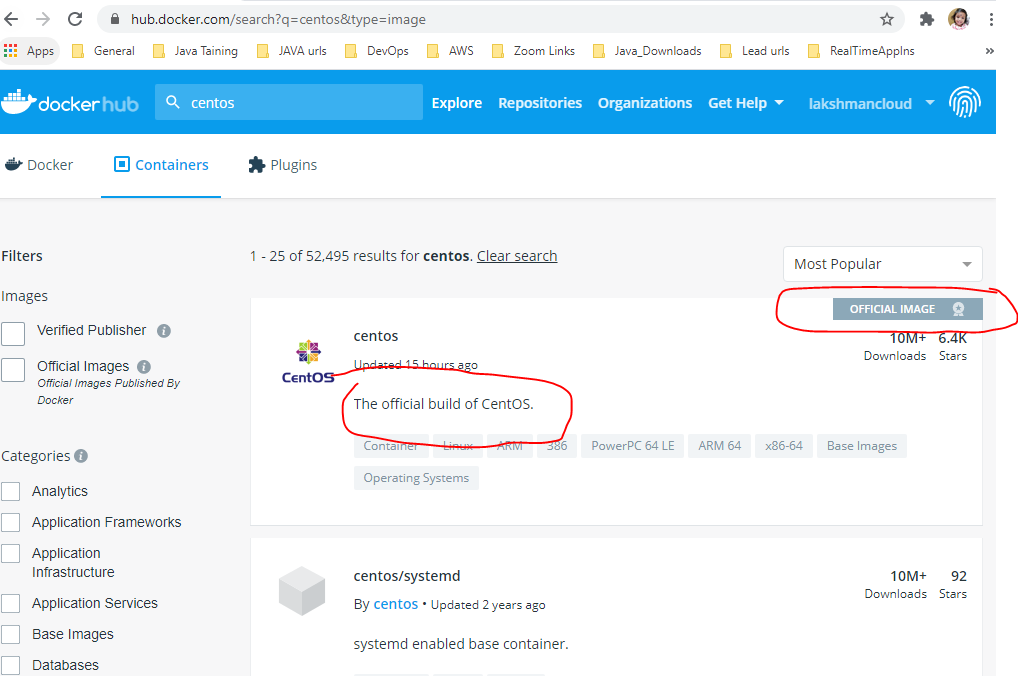
Already available v1 so not pulled again.

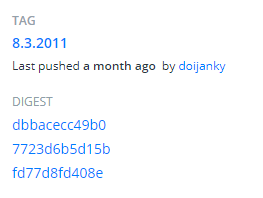
****

**[root@localhost ~]# docker push lakshmancloud/docker-repo:v2 -> v2 pushing to central repo .**

****

**hub.docker.com** 🡪 it will displays public/users images and official images. (search : centos)





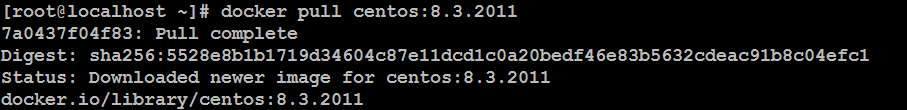
**Pull the centos image :-**

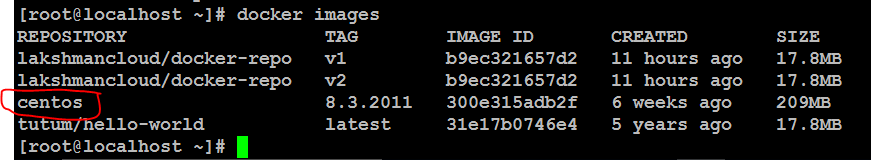
Goto inside tags option and take any specific tag version and pull the centos image.

[root@localhost ~]# **docker pull centos:8.3.2011** (it will take specific version centos image)

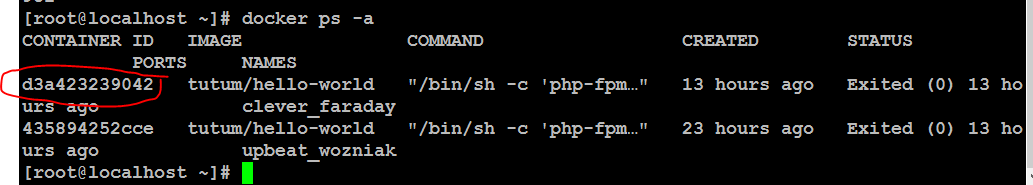
OR

[root@localhost ~]# **docker pull centos** (it will take **latest** tag centos image)

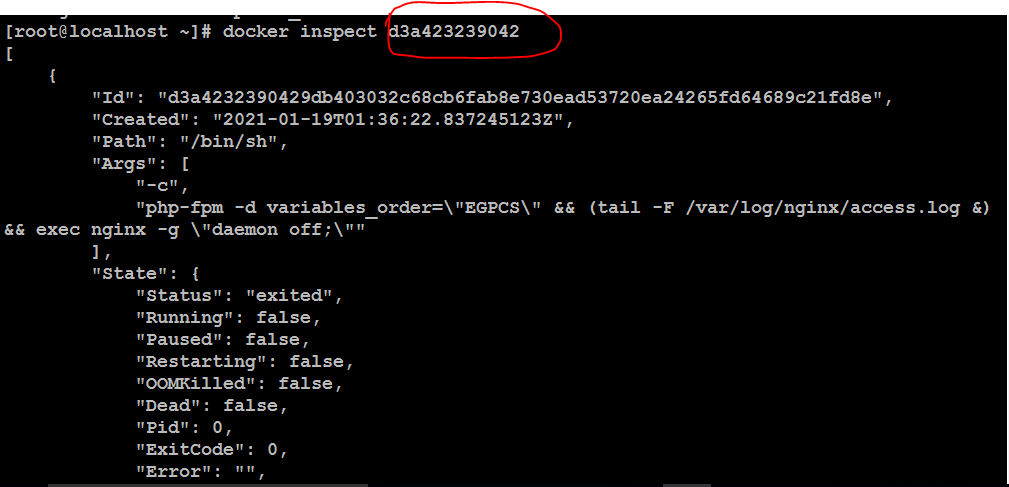




# docker ps -a

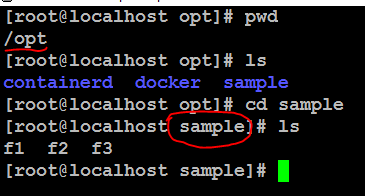


# docker **inspect** cid -> it will show full container information.

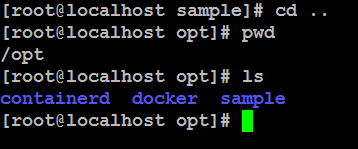


**Share the data from host machine to container:-** sharing some data(any directory) from host machine to container.

Here taking one dir with files.

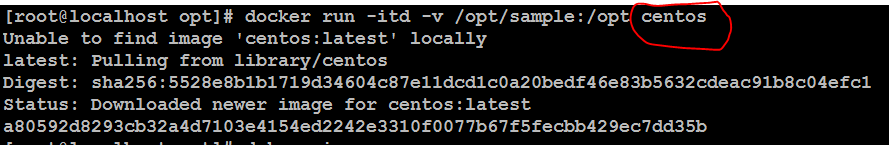


Come out from sample dir.

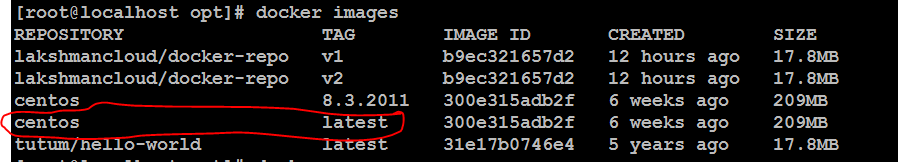


#docker run –itd -v /opt/test:/opt centos 🡪 test ->dir , centos -> img\_name

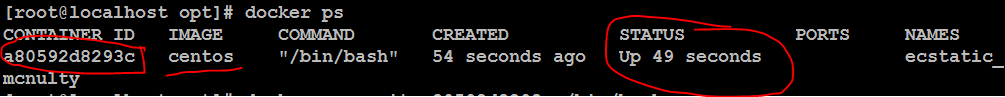
[root@localhost opt]# **docker run -itd -v /opt/sample:/opt centos:7** (centos- if not given any version it will download the latest tag img )



[root@localhost opt]# docker images

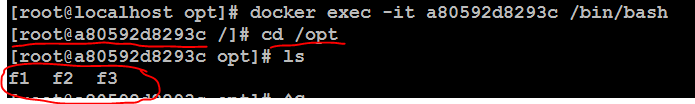


# docker ps



#docker exec –it cid /bin/bash -> went to cid inside, /bin/bash - default path to entry.

[root@localhost opt]# docker exec -it **a80592d8293c** /bin/bash -> goto inside container

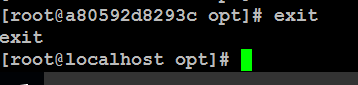


# cd /opt

#ls –lt -> directory moved to here.

F1 f2 f3 files are mapped into provided destination container path

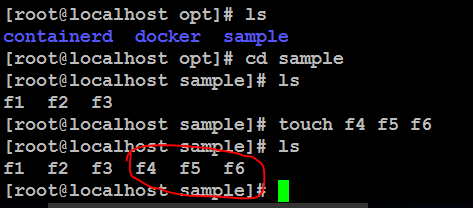
# exit -> exit from container.



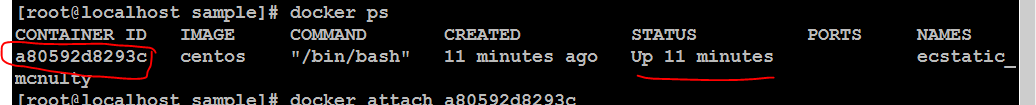
Note :- Now create some new files in **sample** directory, those new files will be mapped automatically into container destination path.

# cd sample -> goto dir

# touch f4 f5 f6 -> some new files got created.



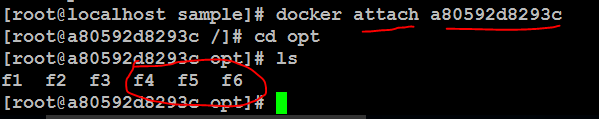
#docker ps



# docker attach cid -> here take same cid

# cd /opt

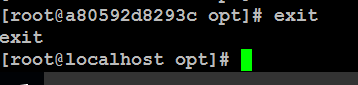
# ls –lt 🡪 here update all new files available in host machine



f4 f5 f6 are mapped here.

Note: **exec** or **attach** both are same cmnds to go inside container.

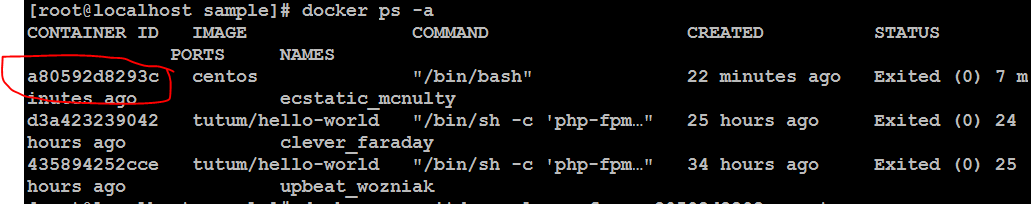
# exit -> exit from container.



**How to share/map/link data from container to container:-**

#docker run –itd --volumes-from cid img\_name (provide already data available cid)

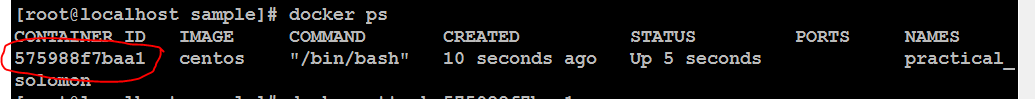
#docker ps –a



[root@localhost sample]# docker run -itd --volumes-from **a80592d8293c** centos (cid and img name)



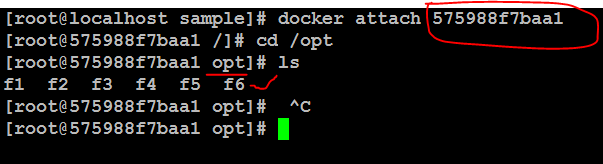
# docker attach cid -> goto inside of cid



[root@localhost sample]# docker attach **575988f7baa1**

# cd /opt

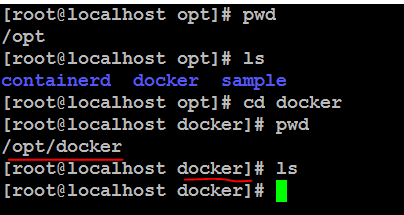
# ls –lt -> test dir is available.



**How to create Docker file:-**

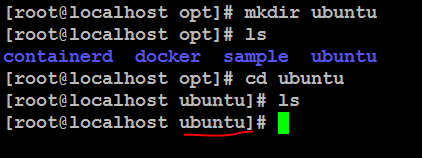
**#** mkdir docker **🡪**Create one dir

**#**  cd docker



**#** mkdir ubuntu 🡪 one more dir create

**#cd** ubuntu

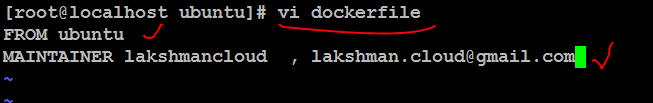


# vi dockerfile (press i) -> creating one dockerfile

FROM ubuntu ->get ubuntu latest version image from public dockerhub.

MAINTAINERlakshmancloud , [lakshman.cloud@gmail.com](mailto:lakshman.cloud@gmail.com) -> provide docker hub details.

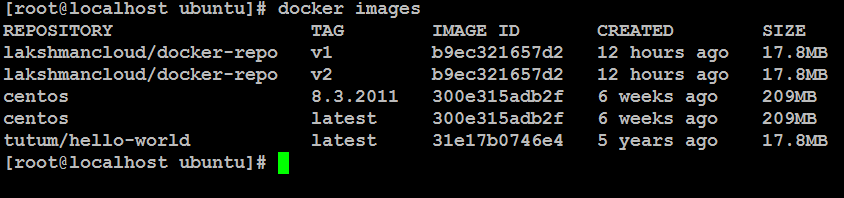
Esc:wq! -> save and quite file.



# ls –lt -> docker file is created.



#docker images -> ubuntu img is not available here



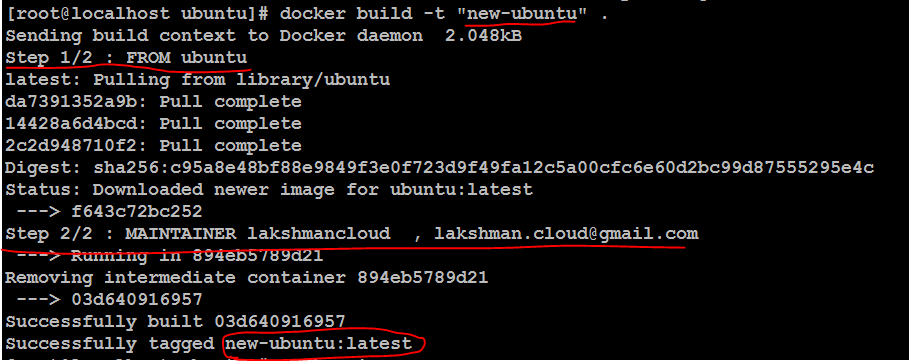
# docker **build** –t “tag\_name” . -> build image , -t tag name(any) , dot is current dir of docker file.

OR

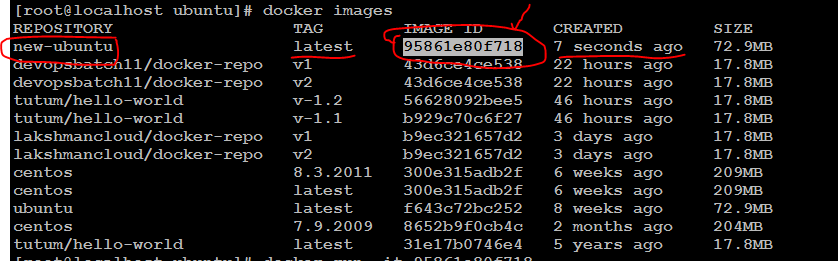
# docker build –t “**tag\_name**” **path\_of\_dockerfile** -> build the image , -t tag name(any)

Note: ubantu latest tag image will download while build the docker file.

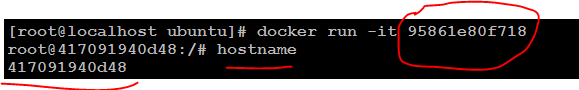
[root@localhost ubuntu]# **docker build -t "new-ubuntu" .**



[root@localhost ubuntu]# docker images -> latest new-ubuntu img created .



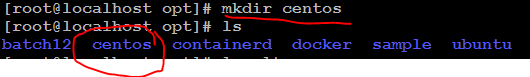
[root@localhost ubuntu]# docker run -it 95861e80f718 -> run the img



Exit from container



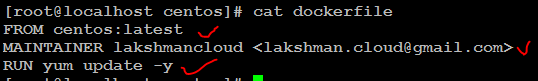
Create dir



Create dockerfile

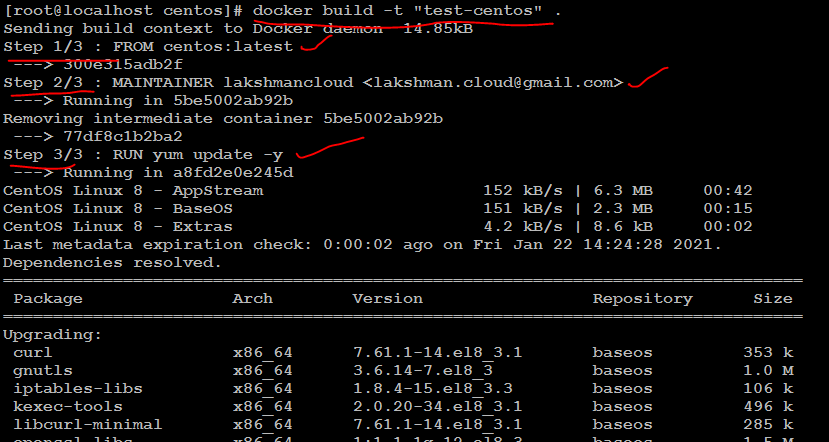


Given 3 steps in dockerfile

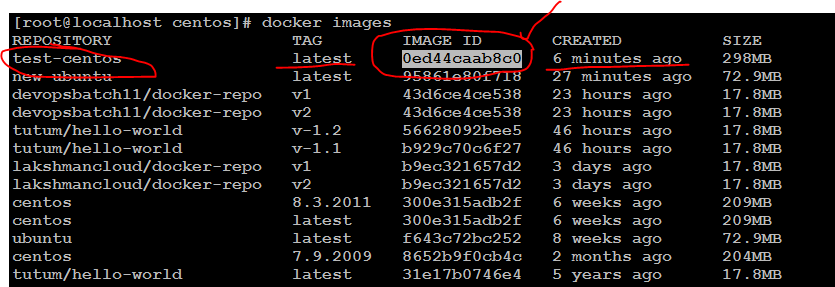


Note:- after created centos latest image , inside container will run “yum update -y” cmnd for updates.

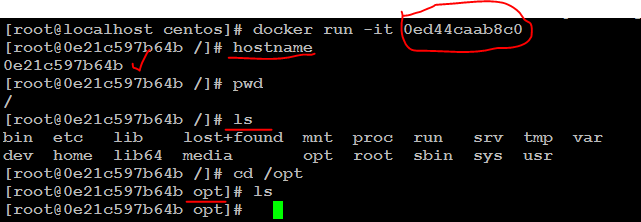
[root@localhost centos]# docker build -t "test-centos" . (dot for dockerfile in current dir)



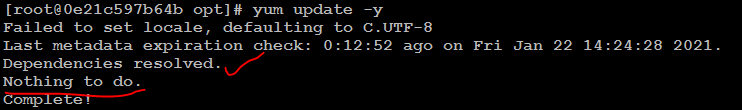
[root@localhost centos]# docker images



[root@localhost centos]# docker run -it **0ed44caab8c0**



[root@0e21c597b64b opt]# yum update –y ->verify – already installed ntg to do.



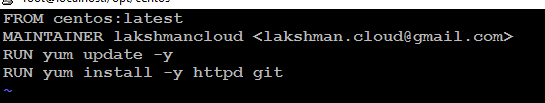
**Dockerfile:-**

FROM centos:latest

MAINTAINER lakshmancloud <lakshman.cloud@gmail.com>

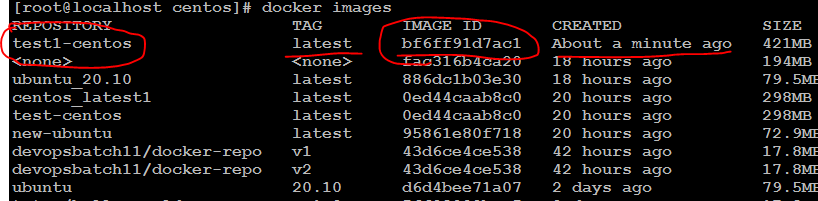
RUN yum update -y

RUN yum install -y httpd git



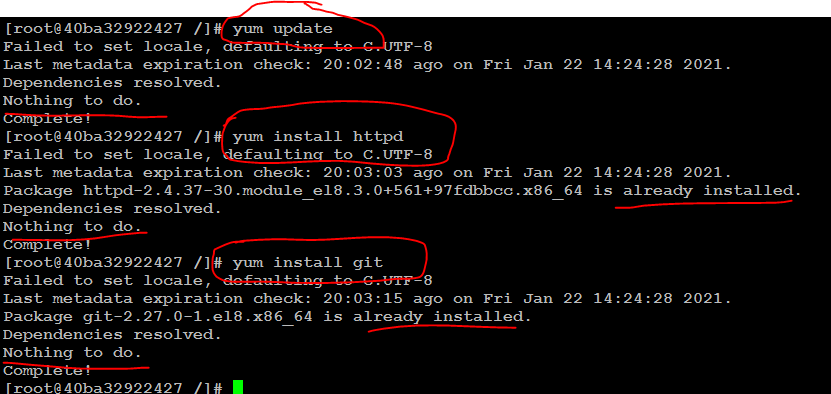
[root@localhost centos]# docker build -t "test1-centos" .



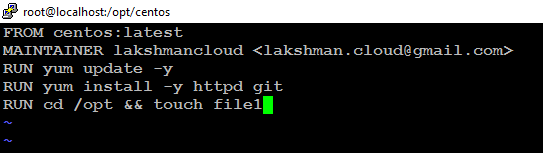


[root@localhost ubuntu]# docker run -it **fa82732af7c1** (img\_id)

Verify all cmnds:-

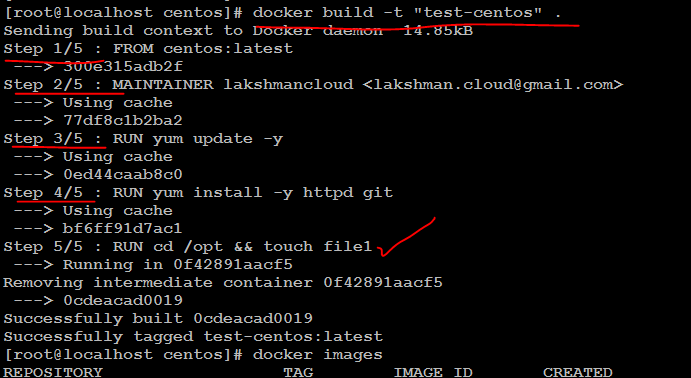


Added one more step(At a time 2 cmnds will run): RUN cd /opt && touch file1

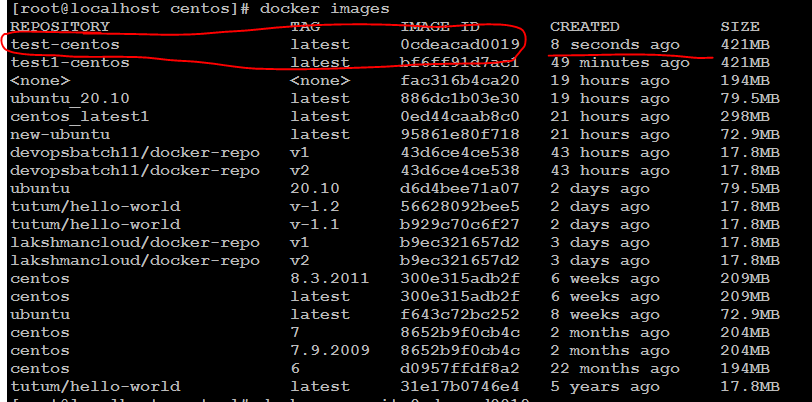


Again build same image(already 4 steps complted )

[root@localhost centos]# docker build -t "test-centos" .

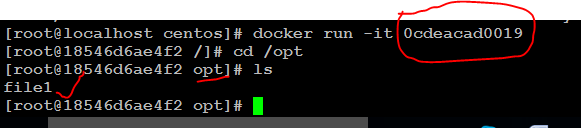


[root@localhost centos]# docker images



[root@localhost centos]# docker run -it **0cdeacad0019**

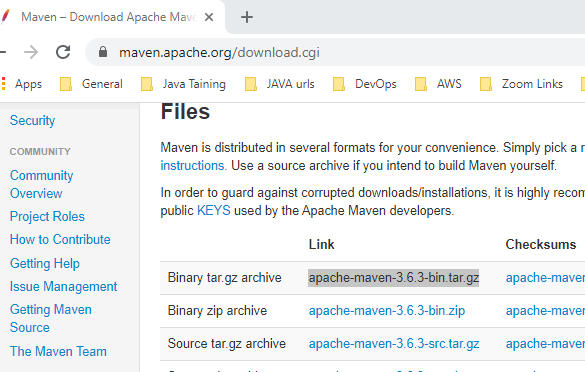
File1 created under opt dir inside new container.



Mvn dir will create and maven software will donwload on below cmnds



Download mvn software

<https://maven.apache.org/download.cgi> 

Mvn Software link : <https://mirrors.estointernet.in/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz>



RUN yum update -y

RUN yum install -y httpd git **wget**

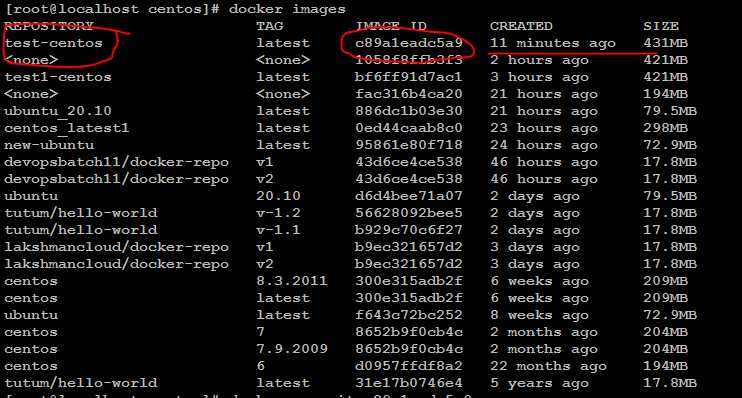
**RUN cd /opt && touch file1**

**RUN mkdir /opt/mvn**

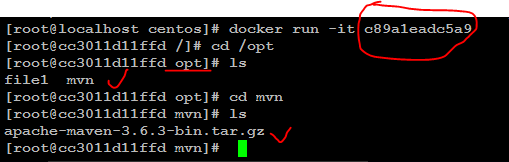
**RUN cd /opt/mvn && wget https://mirrors.estointernet.in/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz**

[root@localhost centos]# docker build -t "test-centos" .

[root@localhost centos]# docker images



Downloaded maven software



FROM centos:latest

MAINTAINER lakshmancloud <lakshman.cloud@gmail.com>

RUN yum update -y

RUN yum install -y httpd git wget

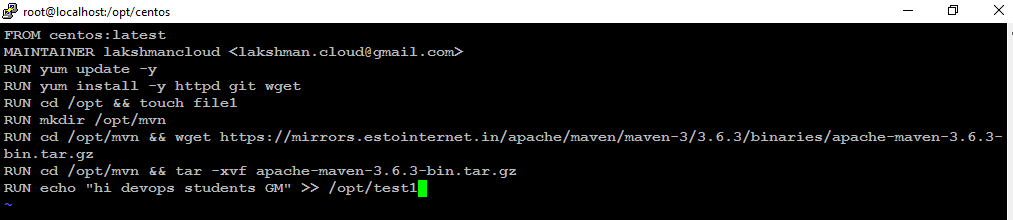
RUN cd /opt && touch file1

RUN mkdir /opt/mvn

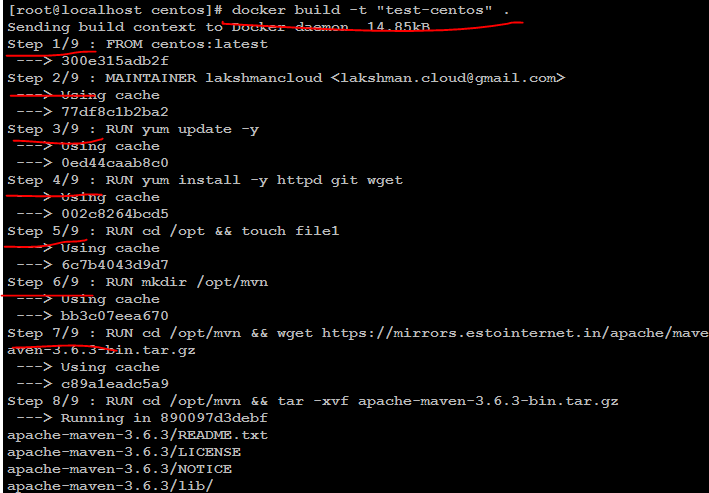
RUN cd /opt/mvn && wget https://mirrors.estointernet.in/apache/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz

**RUN cd /opt/mvn && tar -xvf** apache-maven-3.6.3-bin.tar.gz

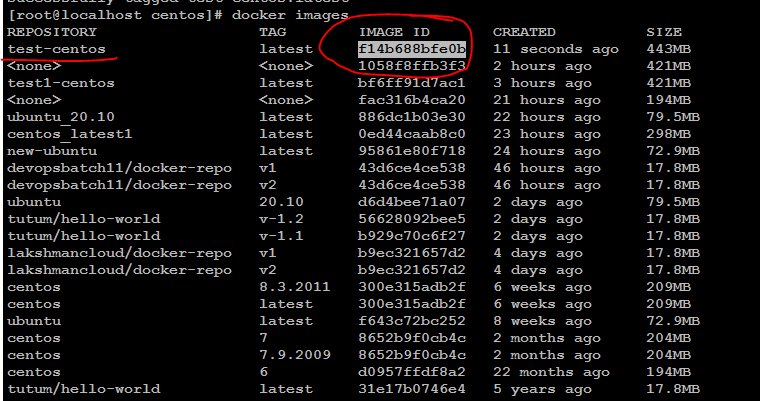
**RUN echo "hi devops students GE" >> /opt/test1**



[root@localhost centos]# docker build -t "test-centos" .



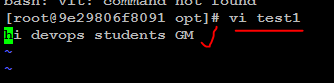
[root@localhost centos]# docker images



[root@localhost centos]# docker run -it **f14b688bfe0b**



Created **test1** file



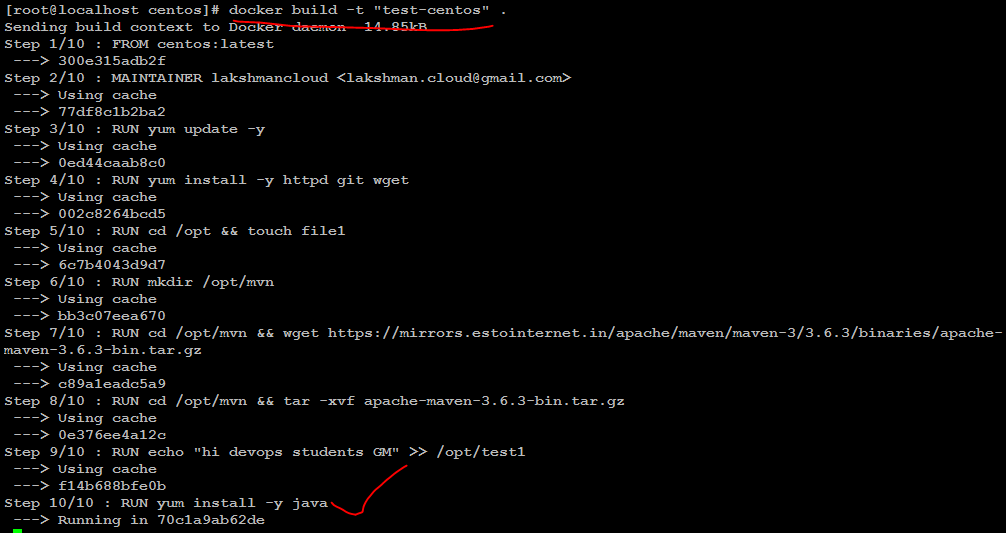
Extrated maven tar file



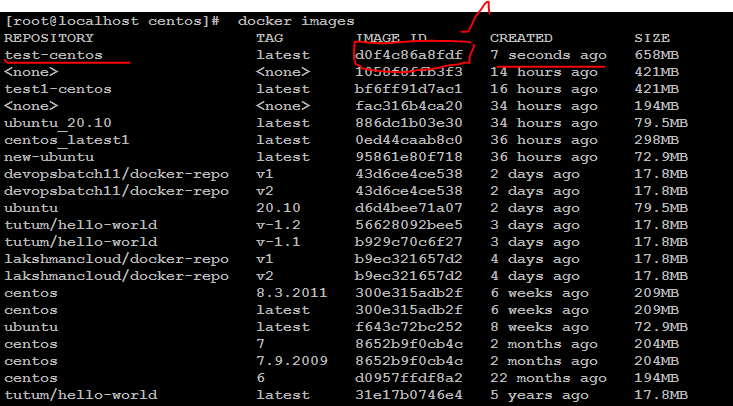
Added new step : RUN yum install -y java



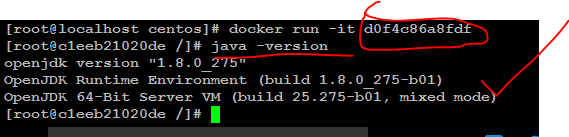
[root@localhost centos]# docker build -t "test-centos" .



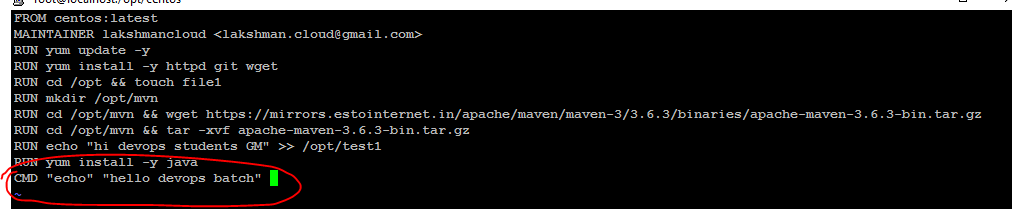
[root@localhost centos]# docker images



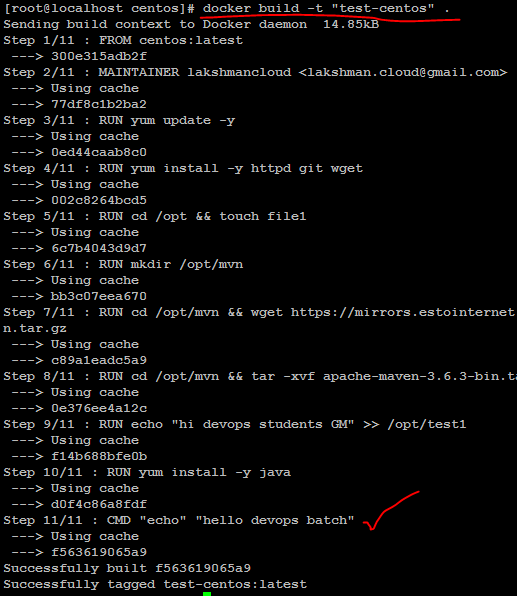
[root@localhost centos]# docker run -it **d0f4c86a8fdf**



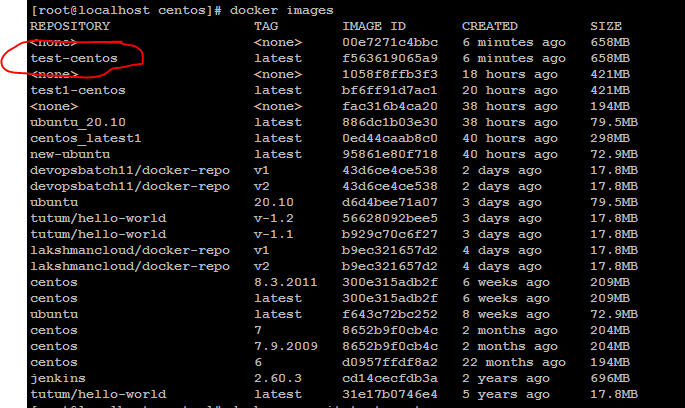
CMD step:- CMD "echo" "hello devops batch"



[root@localhost centos]# docker build -t "test-centos" .



[root@localhost centos]# docker images



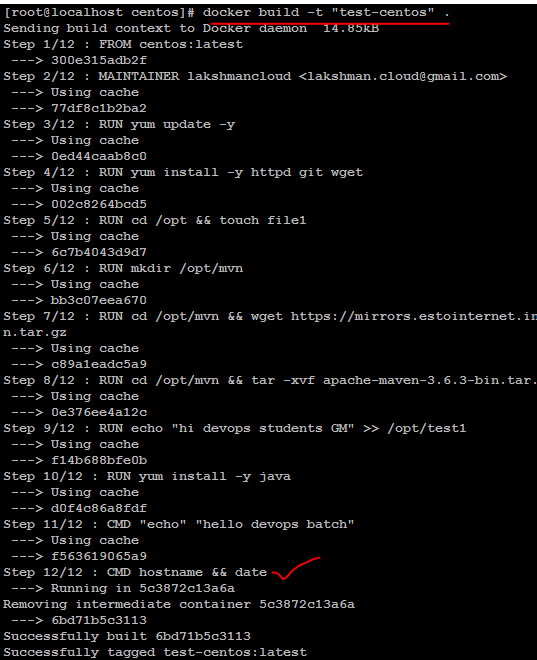
[root@localhost centos]# docker run -it **test-centos**



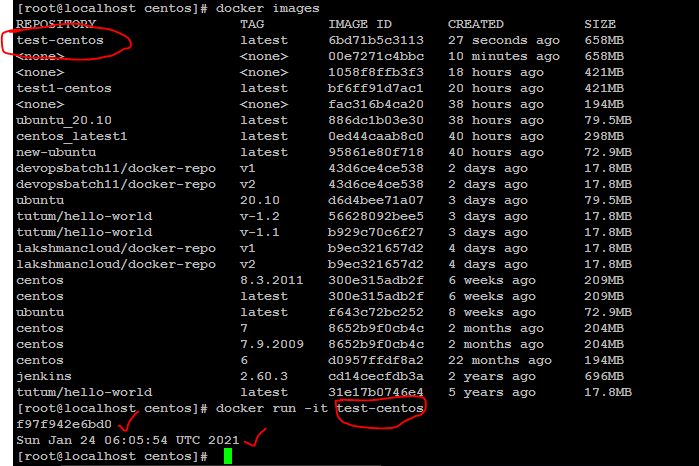
Add step: CMD hostname && date



[root@localhost centos]# docker build -t "test-centos" .

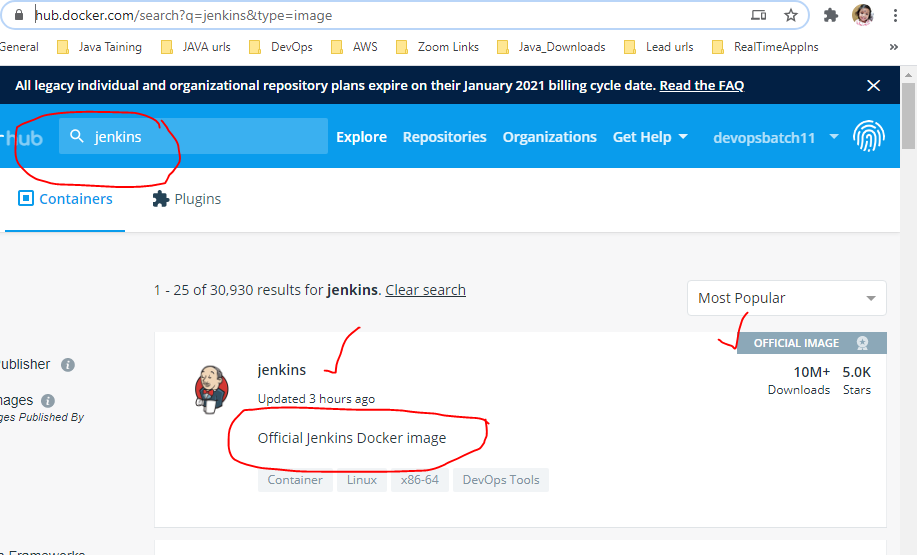


Run with img\_name.

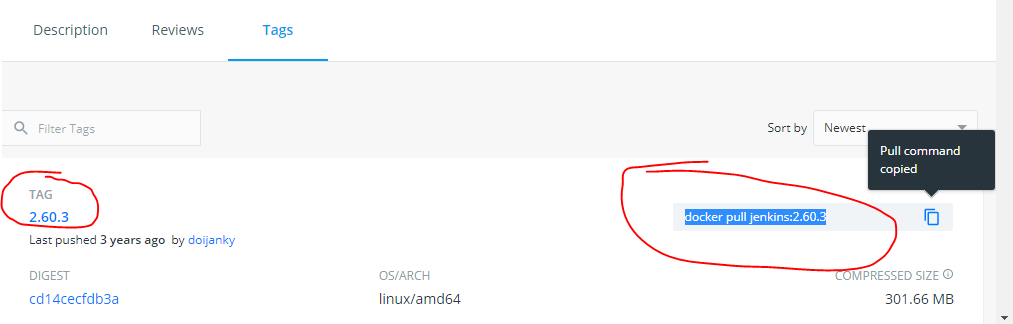


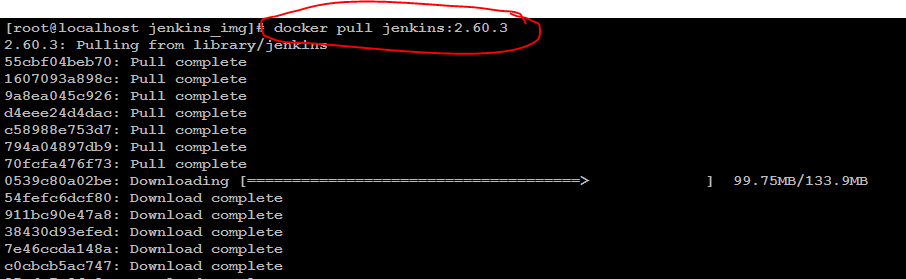
**Download jenkins official img and run container:-**

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

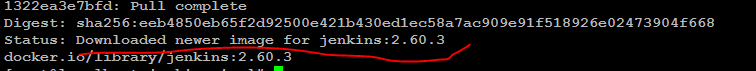


docker pull jenkins:2.60.3

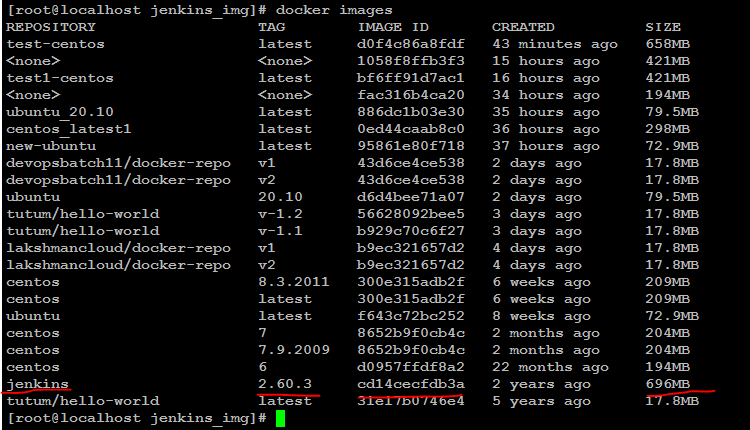


****

**Pulling jenkins specific version img.**



[root@localhost jenkins\_img]# docker images

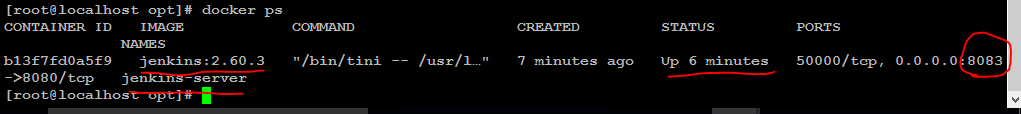


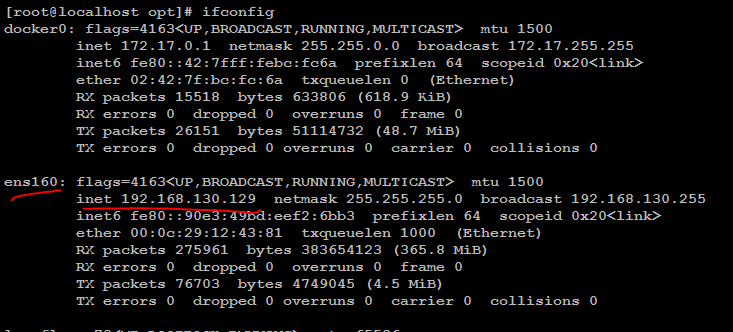
[root@localhost jenkins\_img]# docker run -it --name **jenkins-server** -p **8083**:8080 **jenkins:2.60.3**

(8080 –default port, 8083 –new port, jenkins:2.60.3 –img name, jenkins-server –jenkins service name)

Take new session.

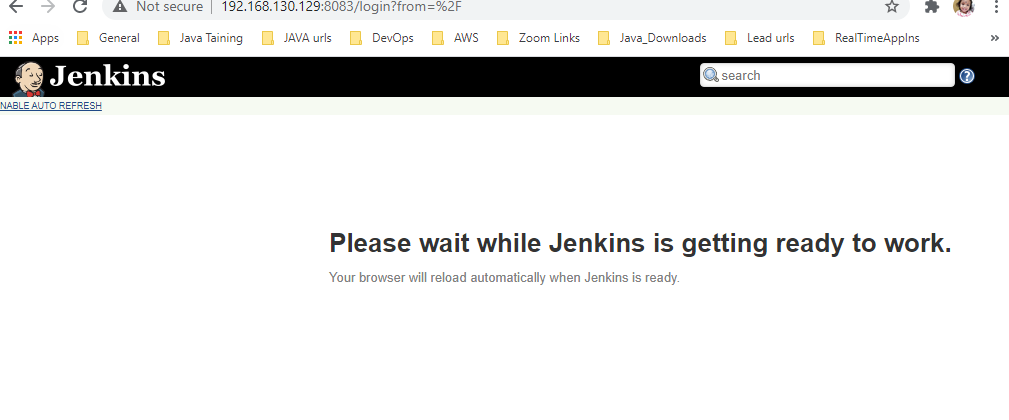
Check ps:



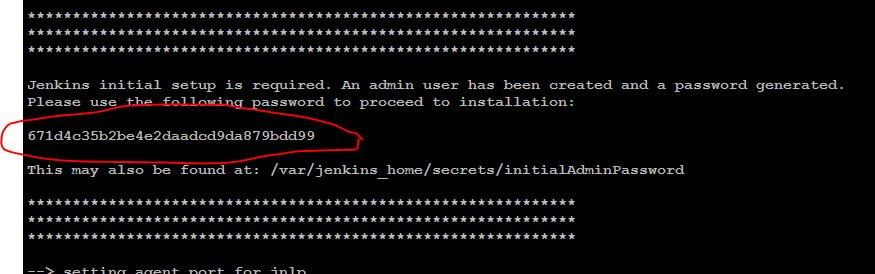
****

****

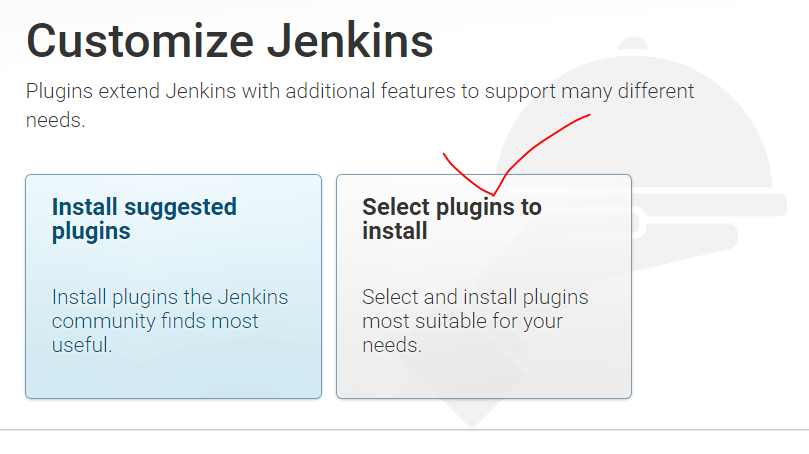
[**http://192.168.130.129:8083/**](http://192.168.130.129:8083/)

****

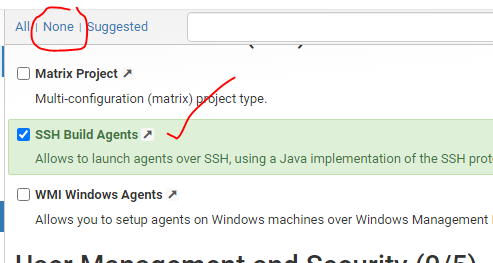
**Take jenkins pwd**

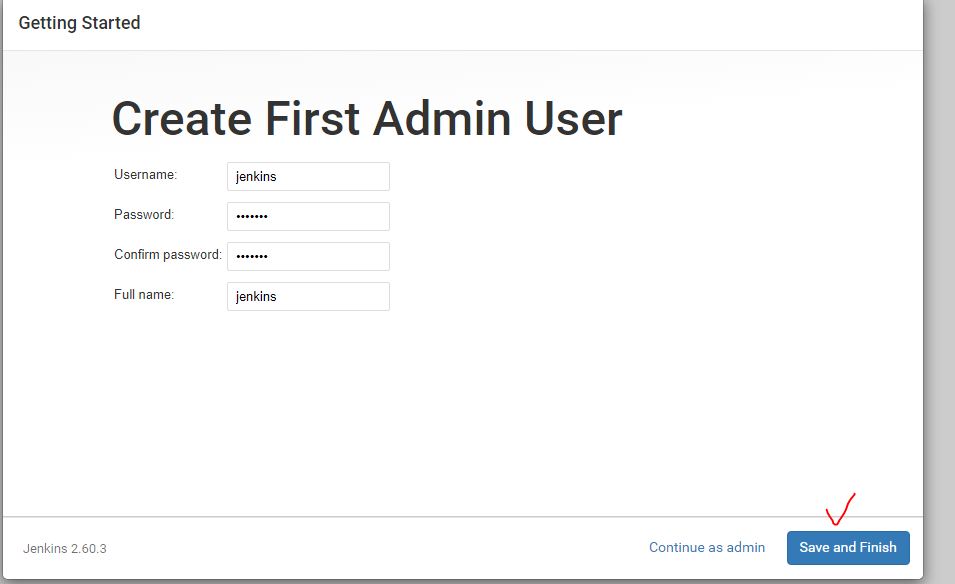
****

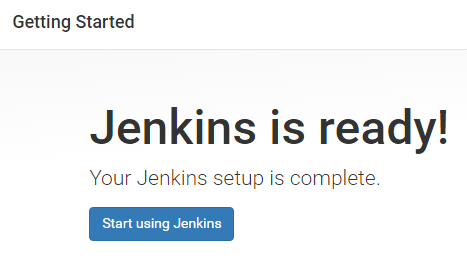
****

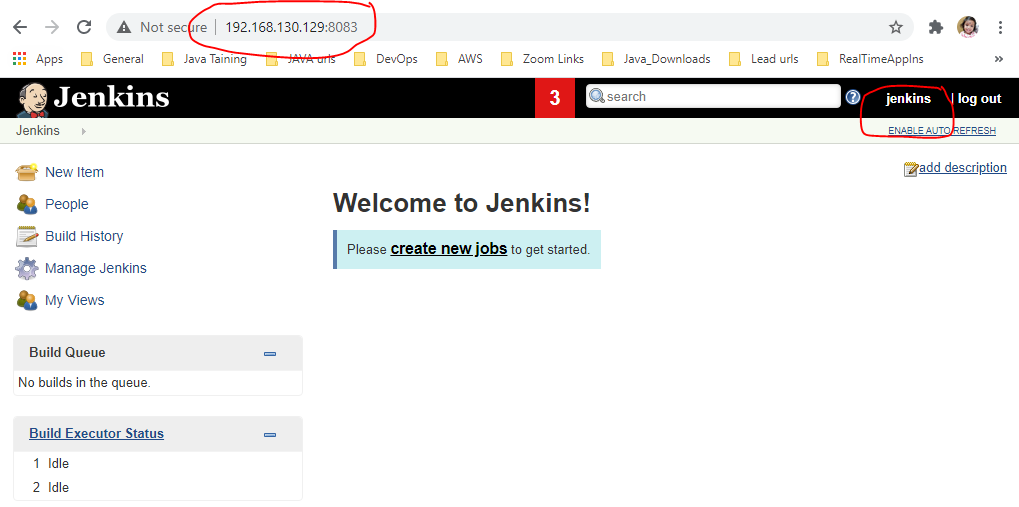
****

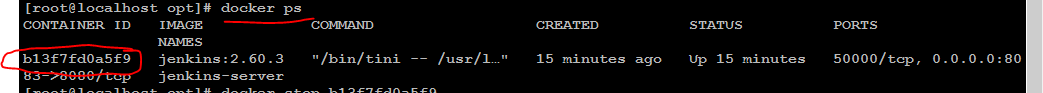
**For time reduce select one plugin**







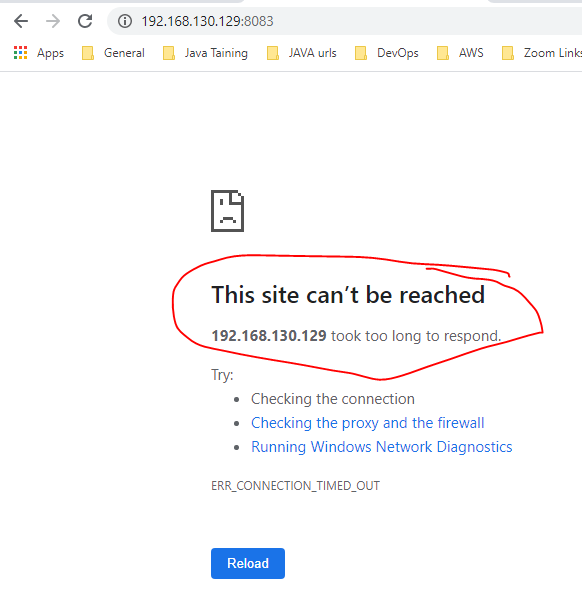


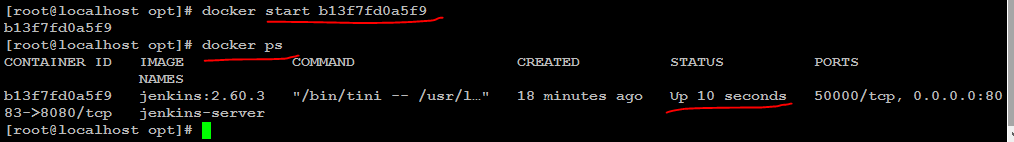


[root@localhost opt]# docker **stop** b13f7fd0a5f9

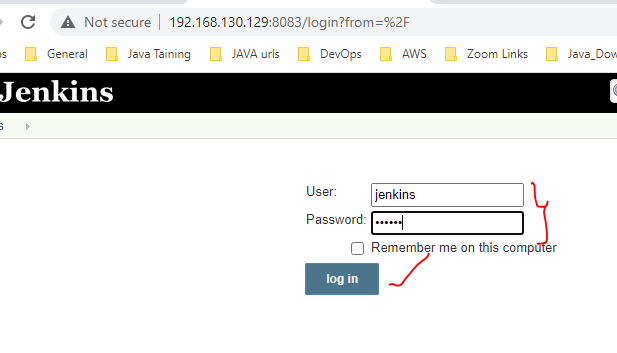


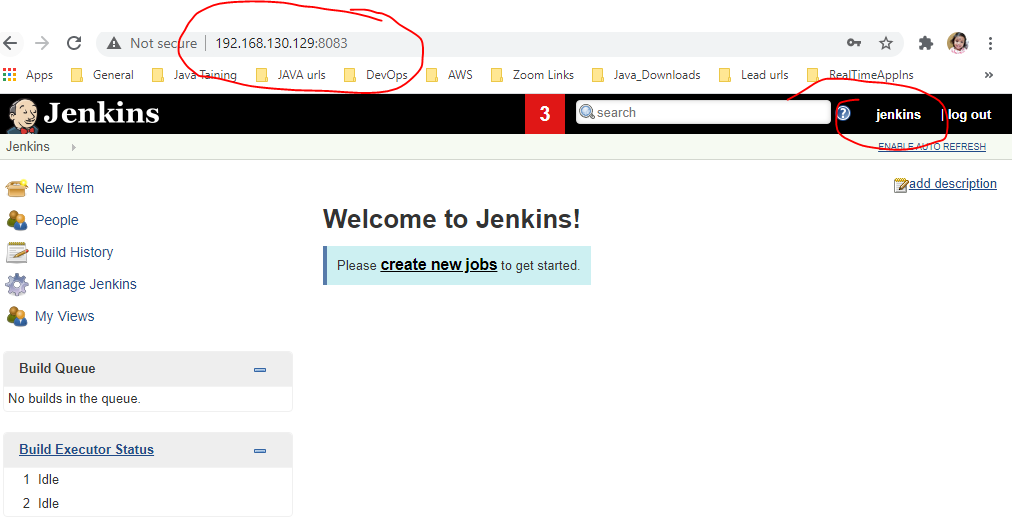




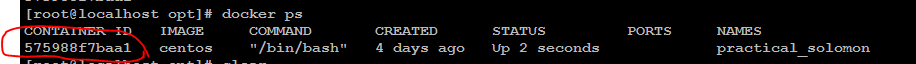


Refresh Jenkins url.

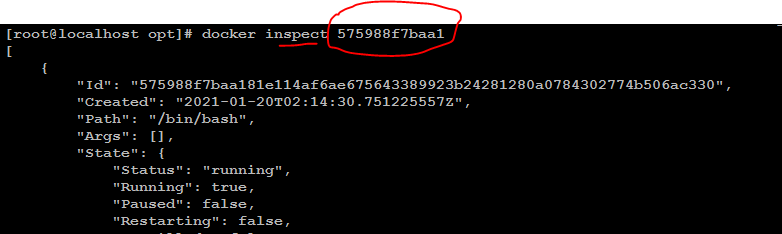




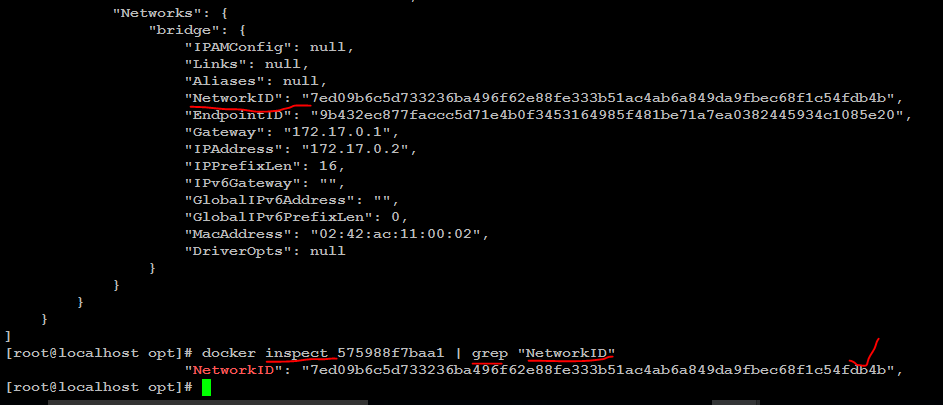
**How to get specific details from container using inspect cmnd:-**

****

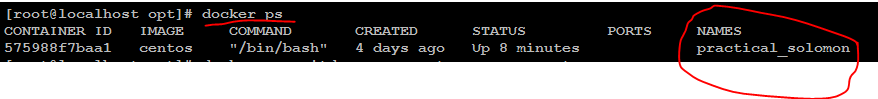
[root@localhost opt]# docker **inspect** **575988f7baa1**



[root@localhost opt]# docker **inspect** 575988f7baa1 | **grep** "NetworkID"

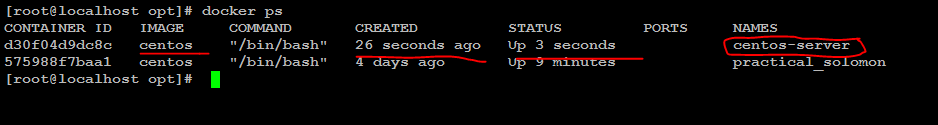


**Set name for running container services:-**

****

**[root@localhost opt]# docker run -itd --name centos-server centos (creating new img with service name)**

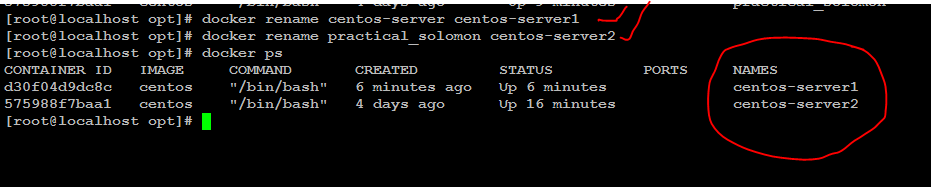
**(centos –img name, centos-server – new container service name)**

****

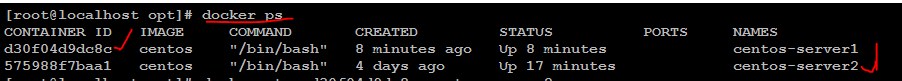
**How to change container service name for existing image:- changed 2 service names.**

**[root@localhost opt]# docker rename centos-server centos-server1**

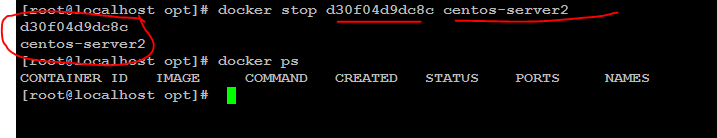
**[root@localhost opt]# docker rename practical\_solomon centos-server2**

****

**How to stop using cid and img\_id :-**

****

[root@localhost opt]# docker stop **d30f04d9dc8c** **centos-server2**



**How to change container hostname:-**

****

[root@localhost opt]# docker run -it --name **centos11** -h **lssofttech.com** **centos**

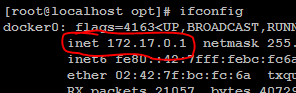
(centos11 – new service name, lssofttech.com – new host name, centos – img name )

****

****

****

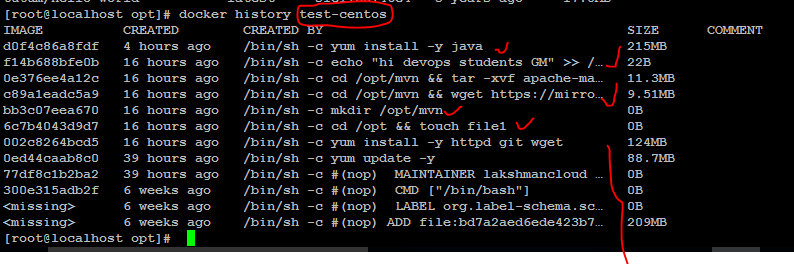
**Containers ip address based on this docker0 series .**

****

**How to check image history:-**

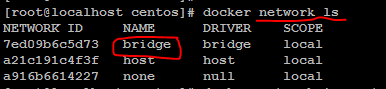
****

[root@localhost opt]# docker **history** **test-centos**

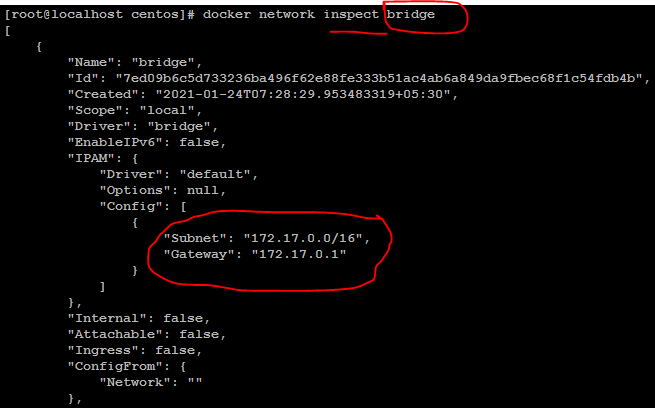


**Docker networking commands:-**

[root@localhost centos]# docker network ls



[root@localhost centos]# docker network **inspect** **bridge**

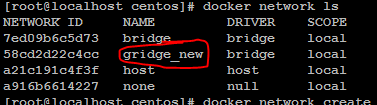


**How to create new network bridge gateway:-**

[root@localhost centos]# docker network create --**subnet** 10.1.0.0/25 --**gateway** 10.1.0.1 gridge\_new



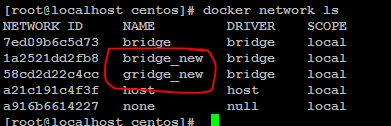
[root@localhost centos]# docker network ls



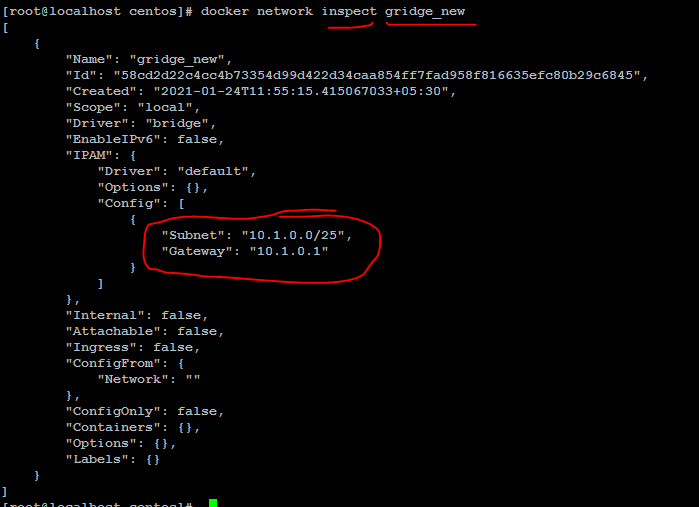
[root@localhost centos]# docker network create --subnet 10.1.1.1/24 --gateway 10.1.1.3 bridge\_new1



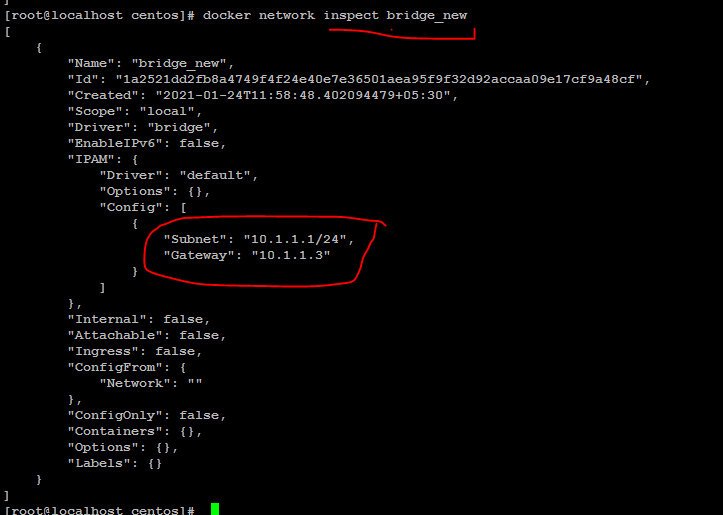
[root@localhost centos]# docker network ls



[root@localhost centos]# docker network inspect gridge\_new

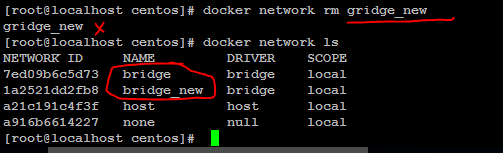


[root@localhost centos]# docker network inspect bridge\_new



**How to remove network : don’t remove default networks.**

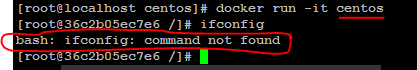
[root@localhost centos]# docker network rm **gridge\_new** (gridge\_new network is removed)



OR

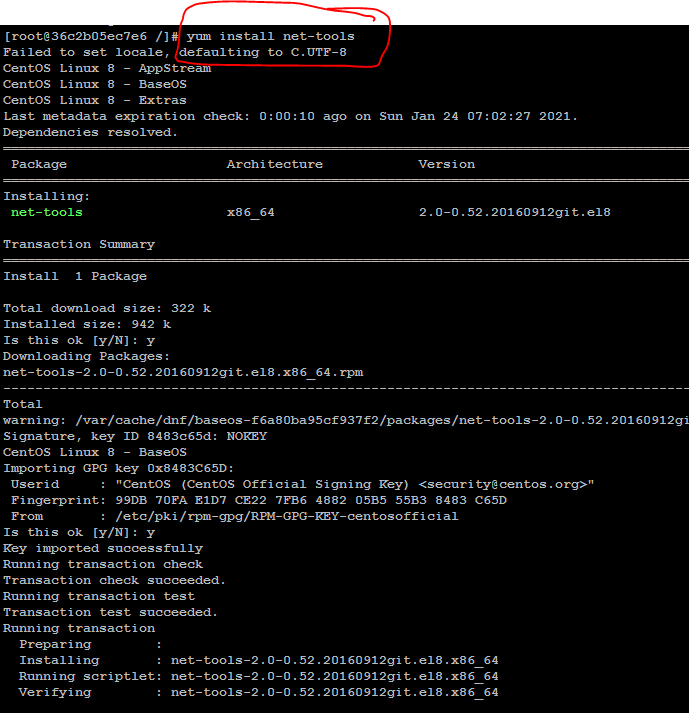
[root@localhost centos]# docker network create --subnet 10.1.0.0/16 --gateway 10.1.0.1 --ip-range=10.1.6.0/24 --driver=bridge --label=new-network bridge-2

[root@localhost centos]# docker run -it **centos**



**Install net tools.**

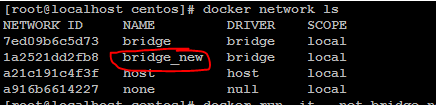
[root@36c2b05ec7e6 /]# yum install **net-tools**



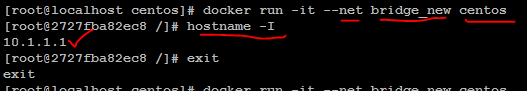
After install net tools you can get ifconfig details.

**How to assign bridge ip addrs to ur container:-**

[root@localhost centos]# docker network ls

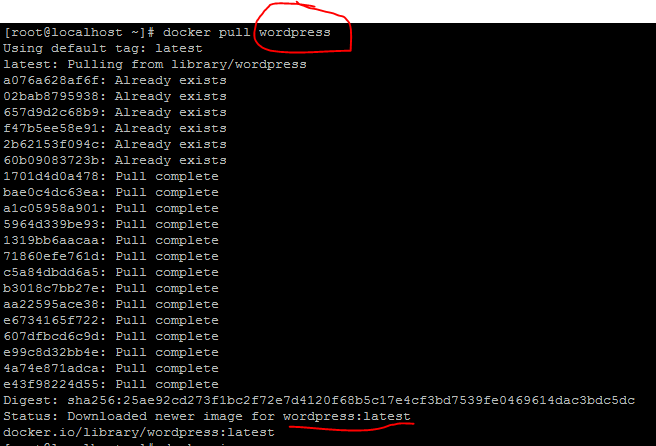


[root@localhost centos]# docker run -it --**net** **bridge\_new** **centos**



**Docker link:- wordpress+link+mysql**

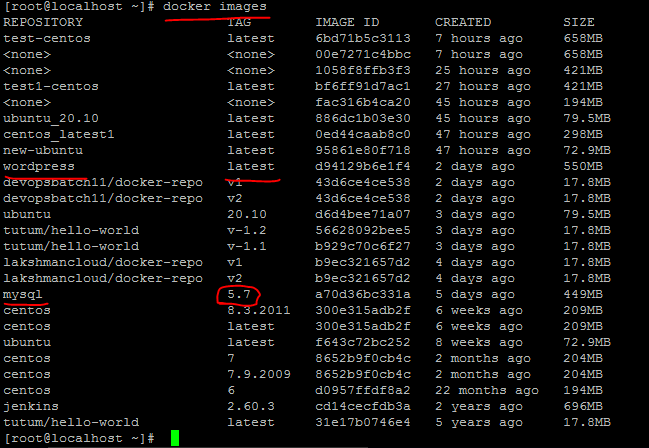
[root@localhost ~]# docker pull **wordpress**



[root@localhost ~]# docker pull **mysql:5.7**

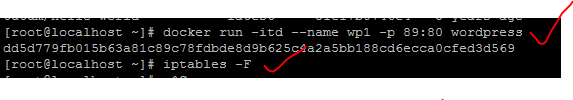


[root@localhost ~]# docker images



[root@localhost ~]# docker run -itd --name wp1 -p 89:80 wordpress

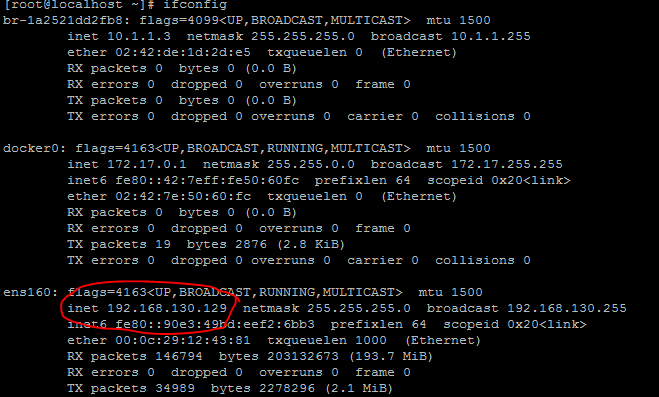
[root@localhost ~]# iptables –F



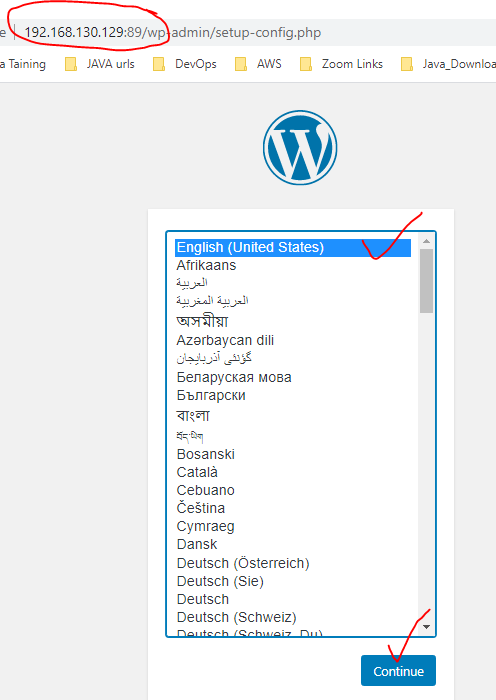
**[root@localhost ~]# netstat –ntlp -> to show open ports**

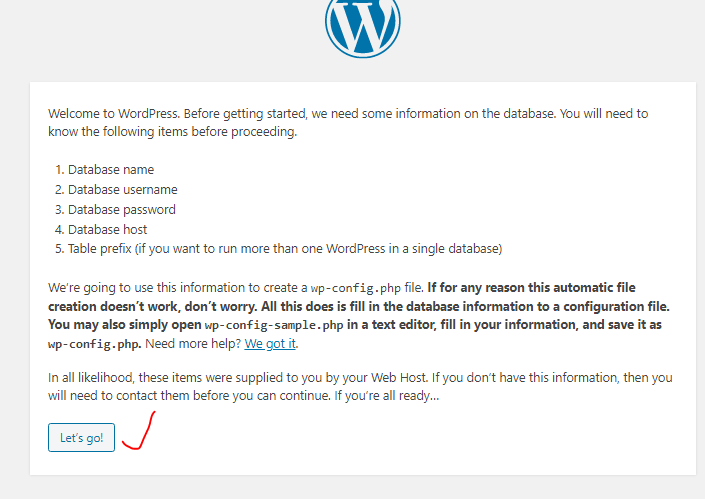
****

[root@localhost ~]# ifconfig



<http://192.168.130.129:89/>

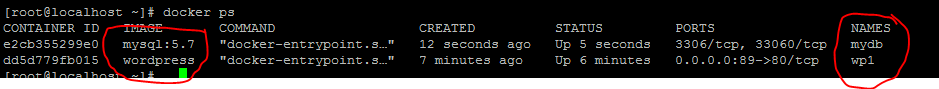




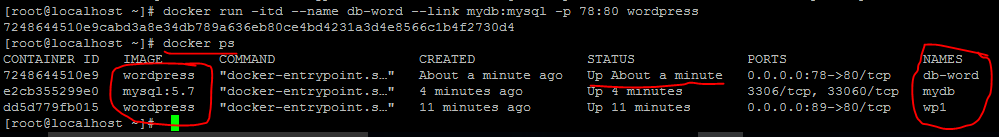
[root@localhost ~]# docker run -itd --name mydb -e MYSQL\_ROOT\_PASSWORD=123456 mysql:5.7



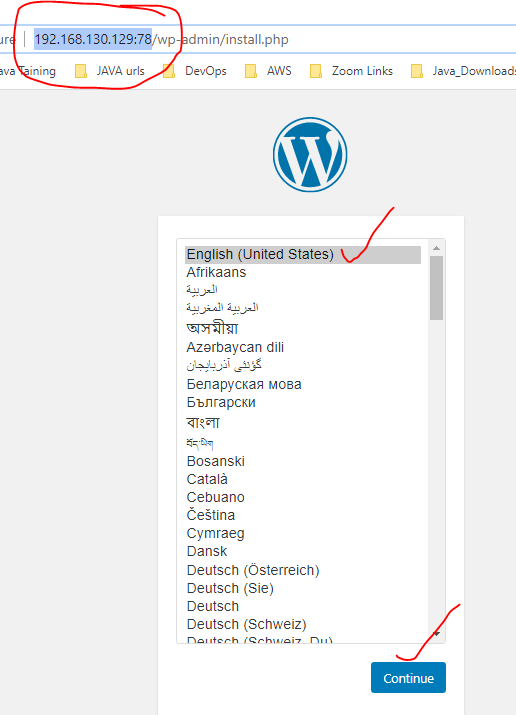
[root@localhost ~]# docker ps

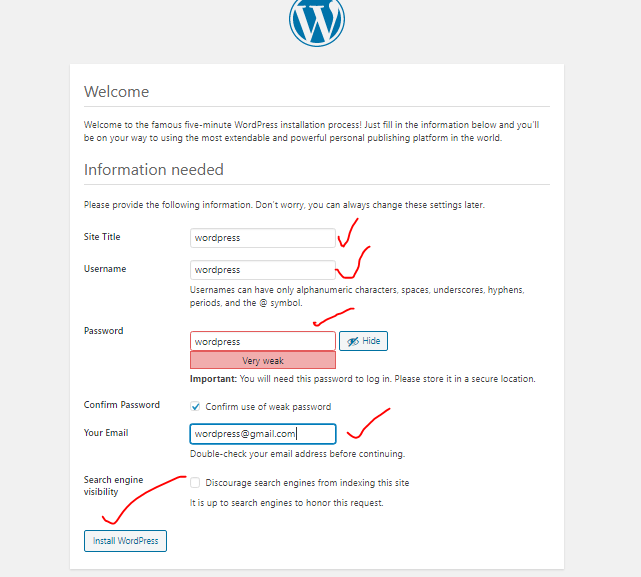


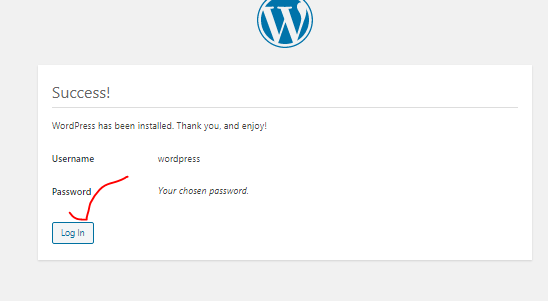
[root@localhost ~]# docker run -itd --name db-link --**link** mydb:mysql -p 78:80 wordpress

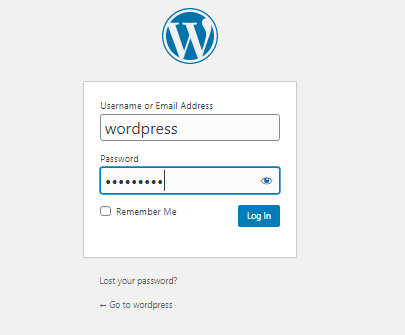


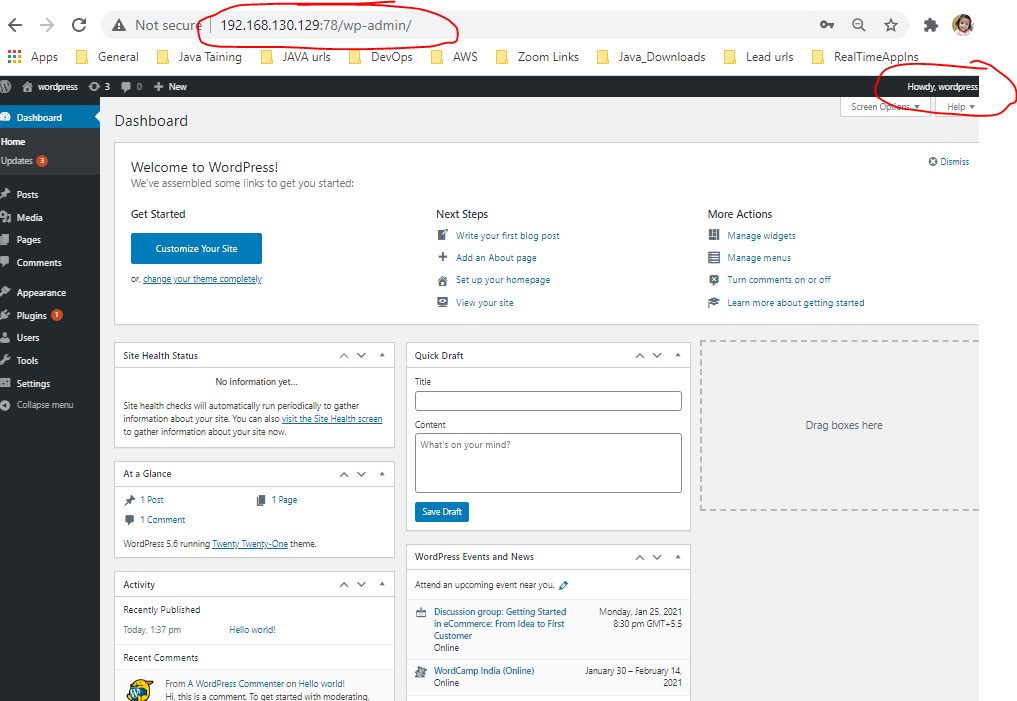
<http://192.168.130.129:78/>



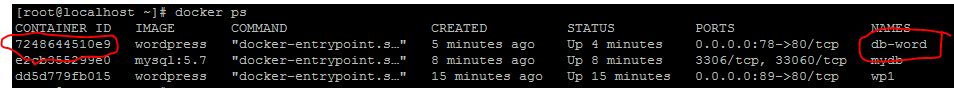






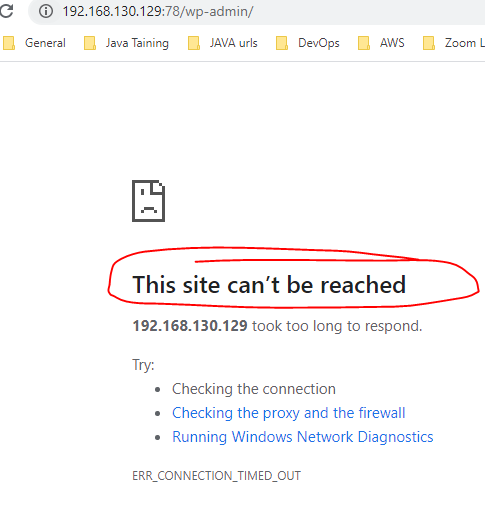


[root@localhost ~]# docker ps

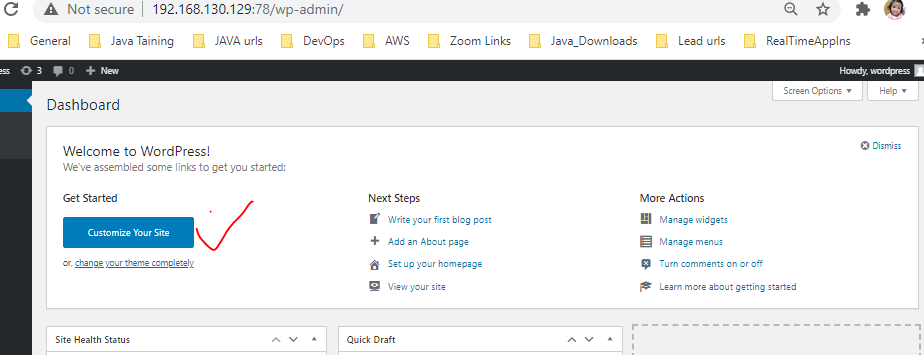


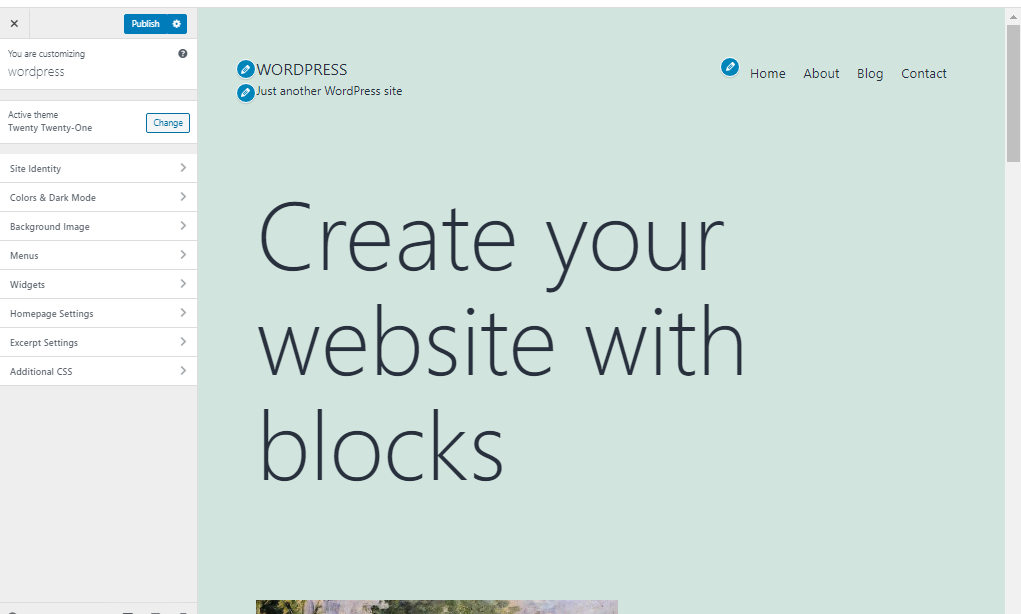
[root@localhost ~]# docker stop **7248644510e9**





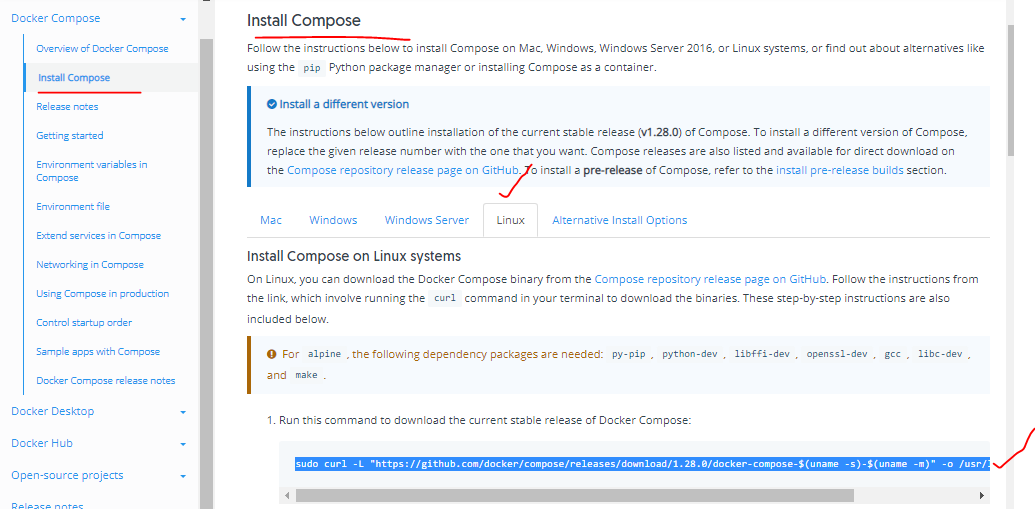


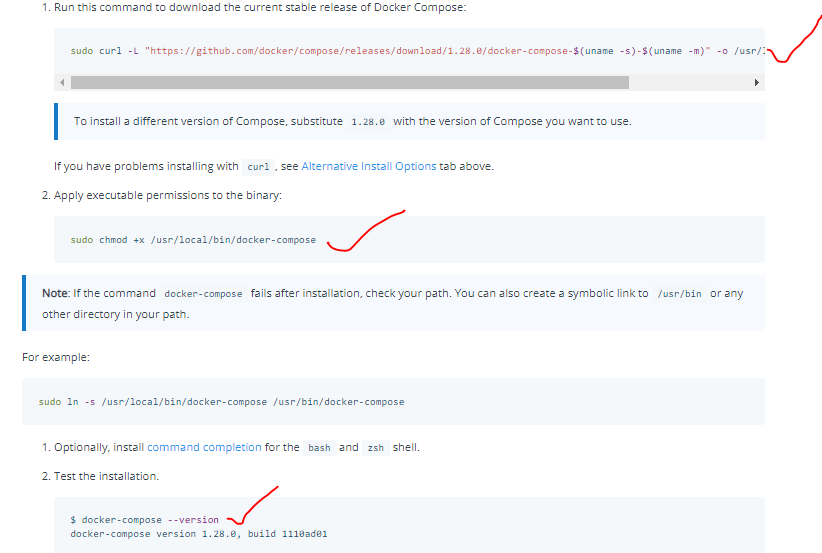




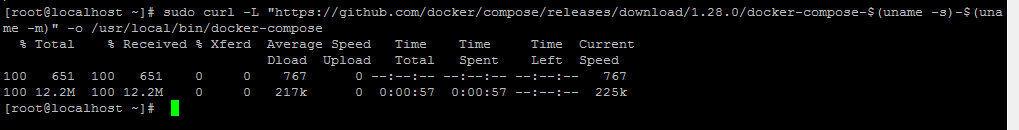
**Docker Compose :-**  compose file(.yaml or .yml) used for multiple containers up.

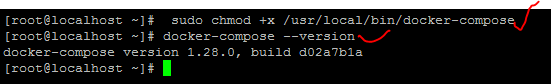
<https://docs.docker.com/compose/install/>



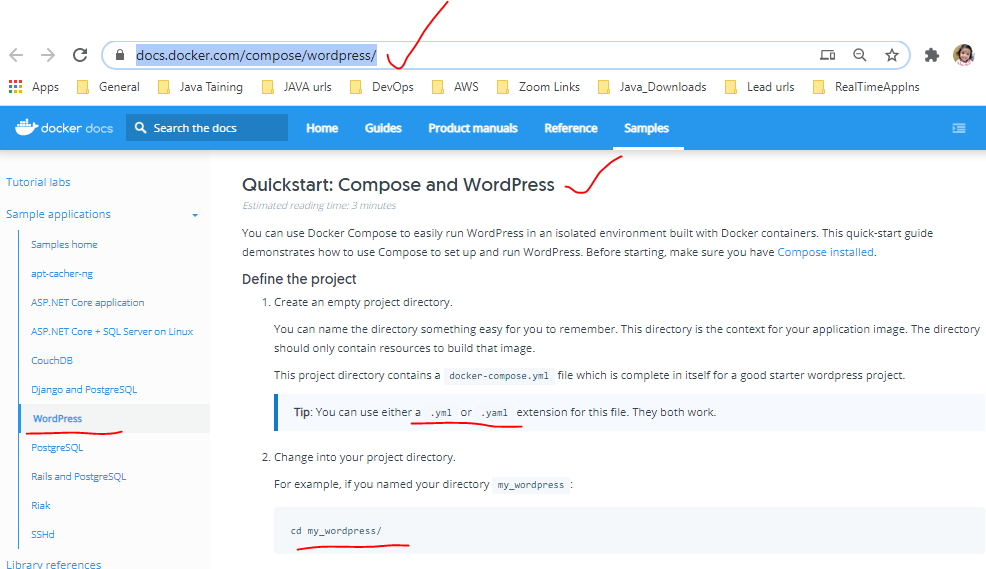


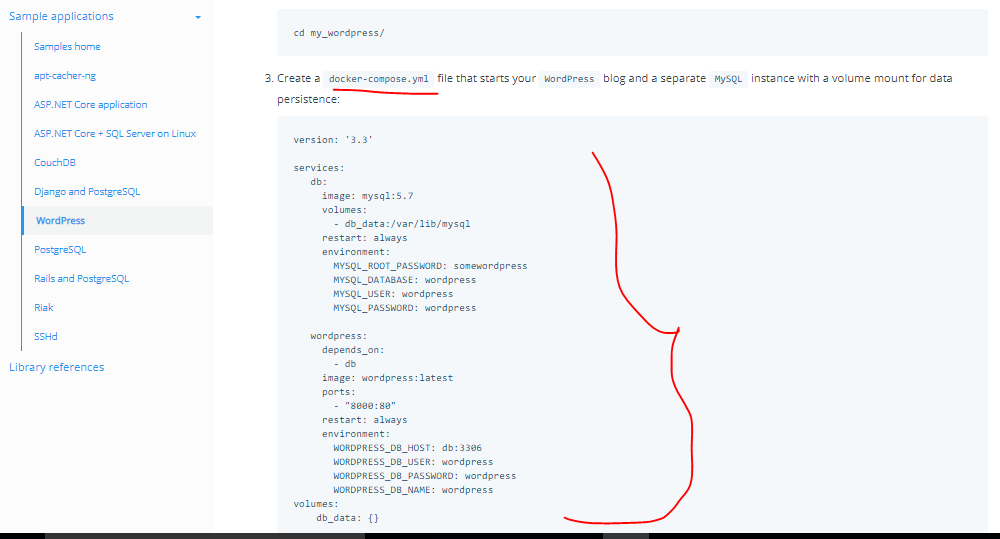
sudo curl -L "https://github.com/docker/compose/releases/download/1.28.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

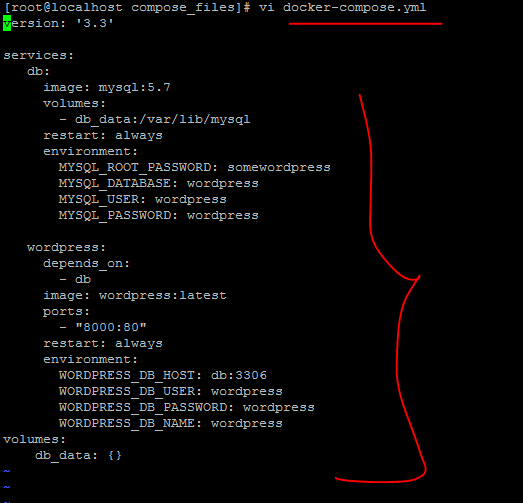
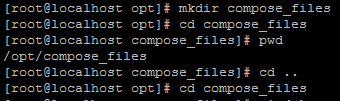




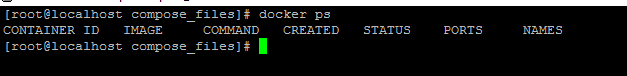
<https://docs.docker.com/compose/wordpress/>



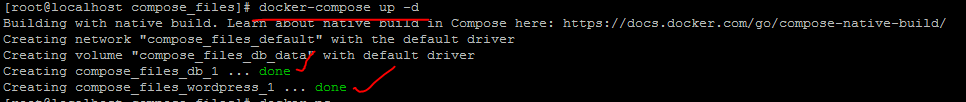




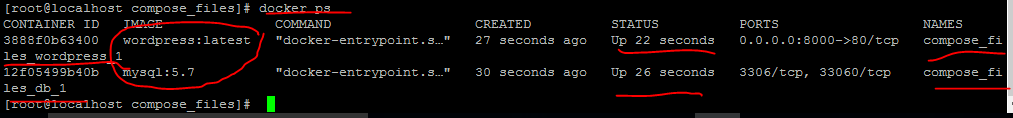
Not running any services



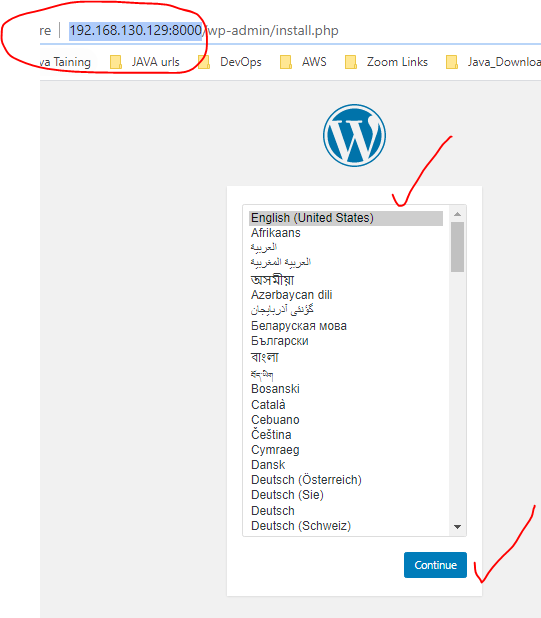
[root@localhost compose\_files]# dokcer-compose up –d

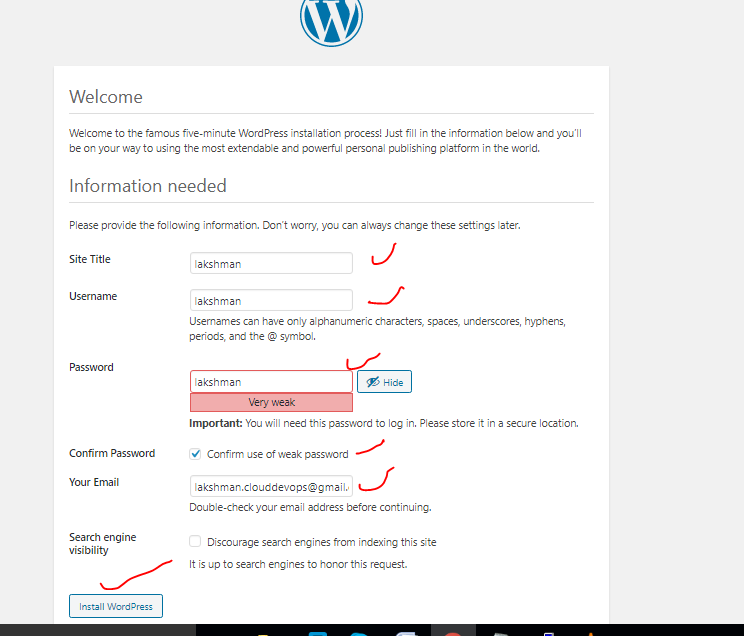


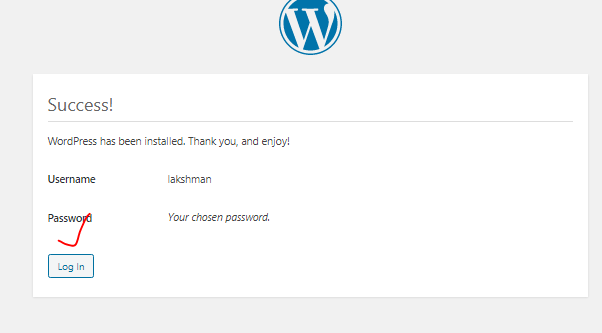
[root@localhost compose\_files]# docker ps

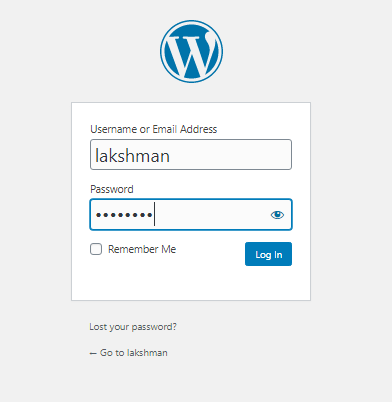


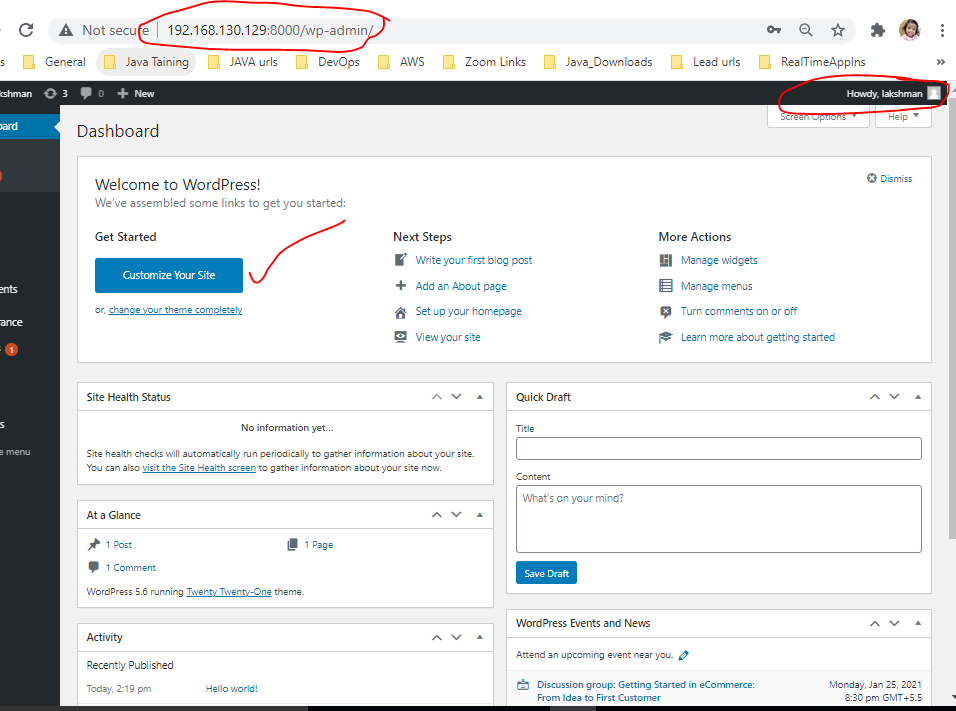
<http://192.168.130.129:8000/>

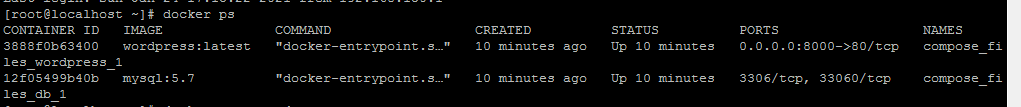




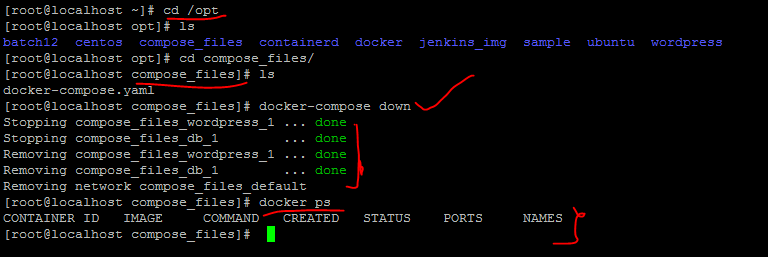








[root@localhost compose\_files]# docker-compose down



=======================================Docker Done=========================================