Docker :

Create a ubuntu machine

```
sudo su

apt update
apt install docker.io -y

service docker start

################


docker images   #to see the list of all the images locally

docker run -it --name c01 ubuntu /bin/bash   #create a ubuntu container with
container name c01   #-it interactive terminal (you will enter inside container)

    exit   # to stop the container and come out of it...if you want to come out of
the container without stopping press ctrl p and ctrl q

docker images  #you will see ubuntu image locally

docker ps -a   #to see the list of all the containers

docker exec -it c01 /bin/bash
 #to enter inside the container c01 ...if container is stopper start it first by
writing docker start c01



# to start a container
docker start c01

# to stop the container
docker stop c01

# to remove the container
docker rm c01
docker ps -a #you will see that container is removed
```

####

Lets assume a situation in which you are working in a project and you have created a container ..inside the container
there is a application which you have developer ...there is some files and softwares inside the container...
now your manager comes to you and says that bro please create a replica of the container ....now we can do it?

Ans: we will create a custom image of the container which will then be used to create the replica

```
docker run -it --name akshatcon ubuntu /bin/bash
   apt update
   apt install apache2 -y
   touch file1 file2 file3 file4
   exit
```

# now we will create a read only template(image) for akshatcon

```
docker commit akshatcon myimg
```

#with this command we will create a image with name myimg which would be read only template of akshatcon

```
docker images  (you will see myimg)
```

## now lets create the replica container

```
docker run -it --name mynewcon myimg /bin/bash

  ls
  which apache2
```

(you can confirm this mynewcon is replica of akshatcon)

#####

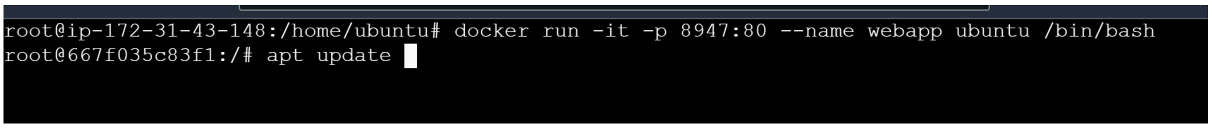
REMEMBER : CONTAINER DOES NOT HAVE ANY PUBLIC IP OF ITS OWN
IF we want to deploy any application which is accessible from the internet ...how we can deploy with the help of conainers?

# you need to do port expose only during creation of container...later on if you need it you need to replicate the container and do it .

In such case we will expose the port of the machine with the port of the container...
for example: if lets say we expose 8947(random number there are 65k ports) of the machine with 80 port of the container ...
in this case when user from internet opens publicip of the machine --> publicip:8947 the person will access the application running on port 80 of the container

Lets create a container with a machine (host) port exposed with port80 of the container

docker run -it -p 8947:80 --name webserver ubuntu /bin/bash



```
root@ip-172-31-43-148:/home/ubuntu# docker run -it -p 8947:80 --name webapp ubuntu /bin/bash
root@667f035c83f1:/# apt update
```

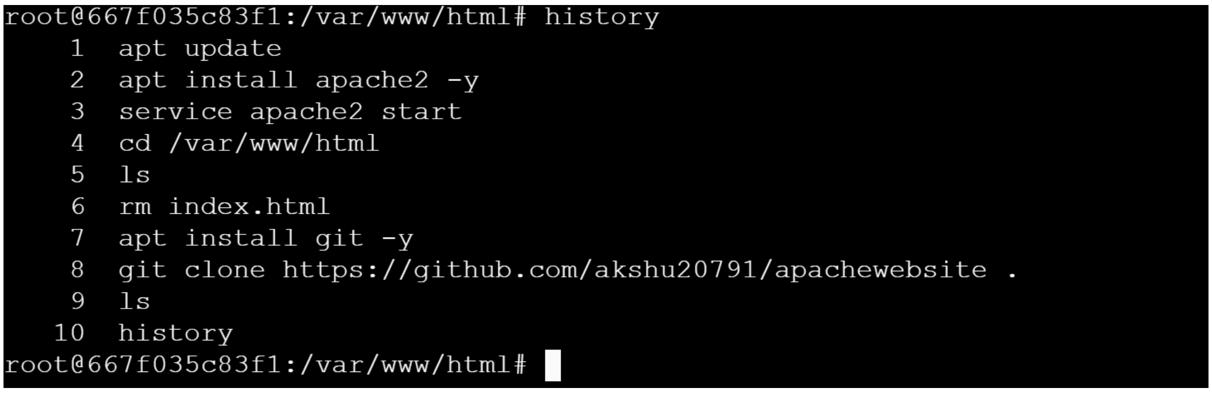
#in above we are exposing the port 8947 of the machine (ec2 /instance/virtual machine/host machine) with port 80 (http port)
of the container ...-p is the port ...

# you will enter inside the container

inside the container lets install apache

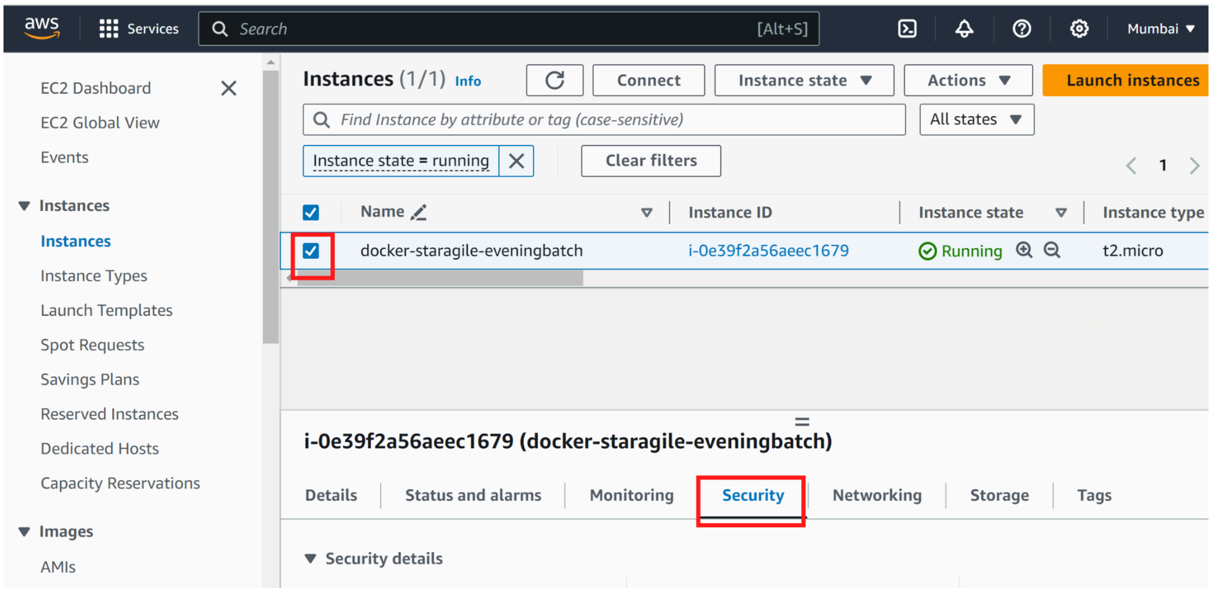
```
  apt update
  apt install apache2 -y  #this is the webserver . this is like tomcat , nginx …
  service apache2 start
  cd /var/www/html  #default location which is created by installing apache. if
we put files here it would be accessible on net
  rm index.html  # we need to delete the default homepage (apache
homepage) which comes up
  apt install git -y
  git clone https://github.com/akshu20791/apachewebsite .
```

(press ctrl p and ctrl q to come out of container without stopping the container)



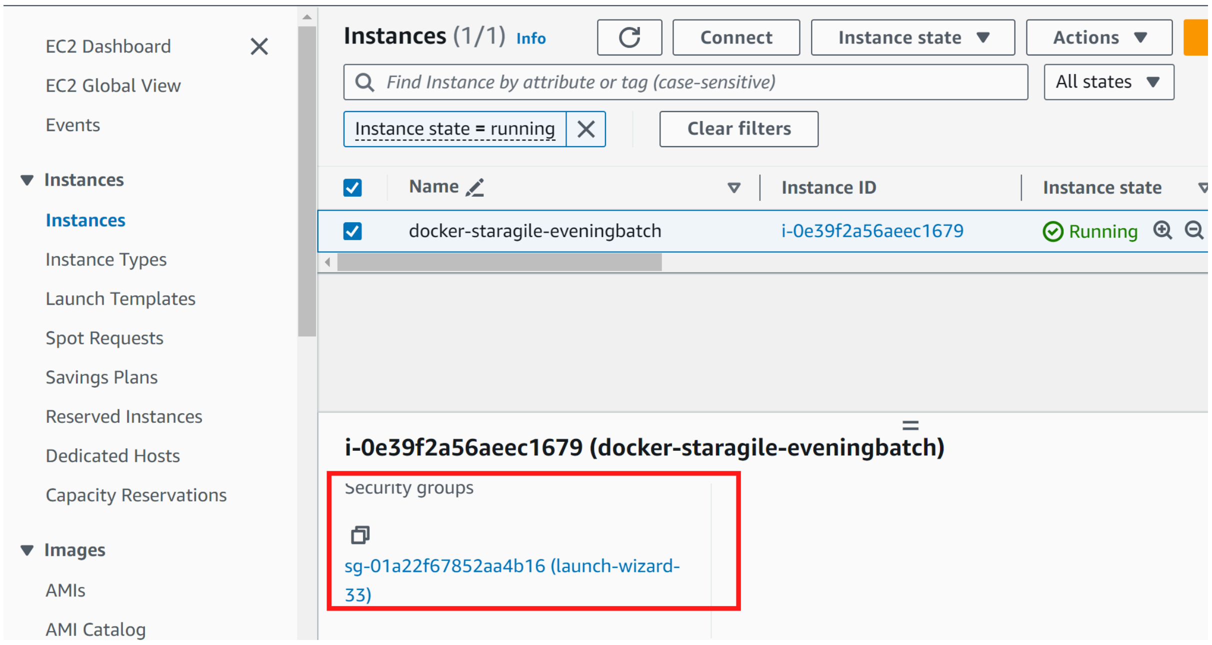
```
root@667f035c83f1:/var/www/html# history
    1  apt update
    2  apt install apache2 -y
    3  service apache2 start
    4  cd /var/www/html
    5  ls
    6  rm index.html
    7  apt install git -y
    8  git clone https://github.com/akshu20791/apachewebsite .
    9  ls
   10  history
root@667f035c83f1:/var/www/html# 
```

We now need to enable the firewall (inbound rule) on port 8947



Name	Instance ID	Instance state	Instance type
docker-staragile-eveningbatch	i-0e39f2a56aeec1679	Running	t2.micro

i-0e39f2a56aeec1679 (docker-staragile-eveningbatch)


EC2 Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog

Instances (1/1) Info

Connect | Instance state | Actions

Find Instance by attribute or tag (case-sensitive) | All states

Instance state = running | Clear filters

Name	Instance ID	Instance state
docker-staragile-eveningbatch	i-0e39f2a56aeec1679	Running

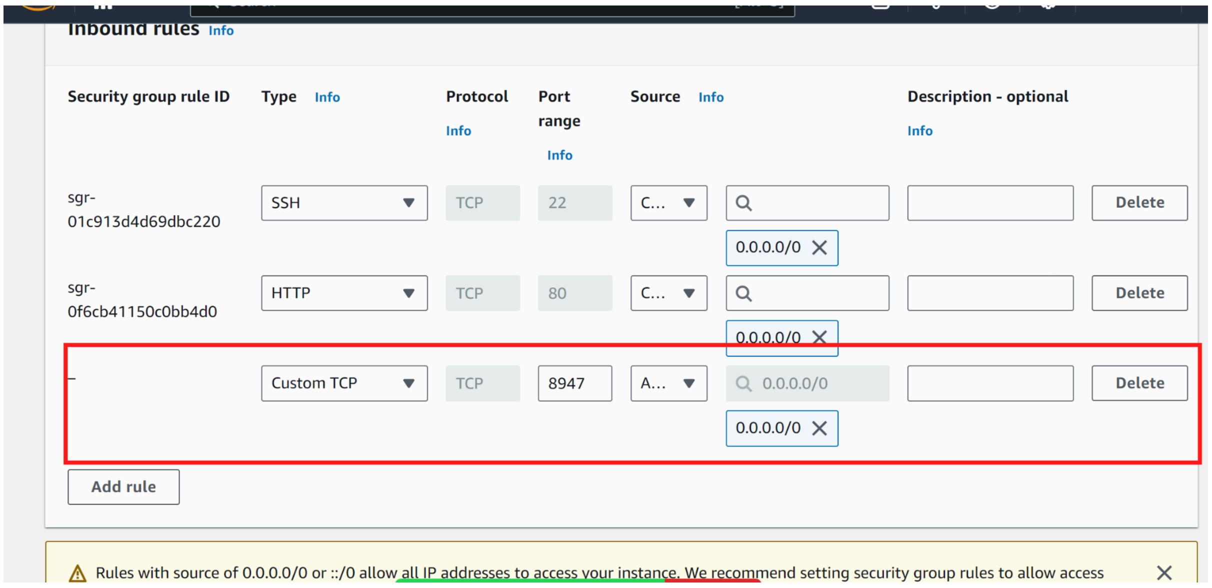
i-0e39f2a56aeec1679 (docker-staragile-eveningbatch)

Security groups

sg-01a22f67852aa4b16 (launch-wizard-33)

Click on edit inbound rule

Add rule


Inbound rules Info

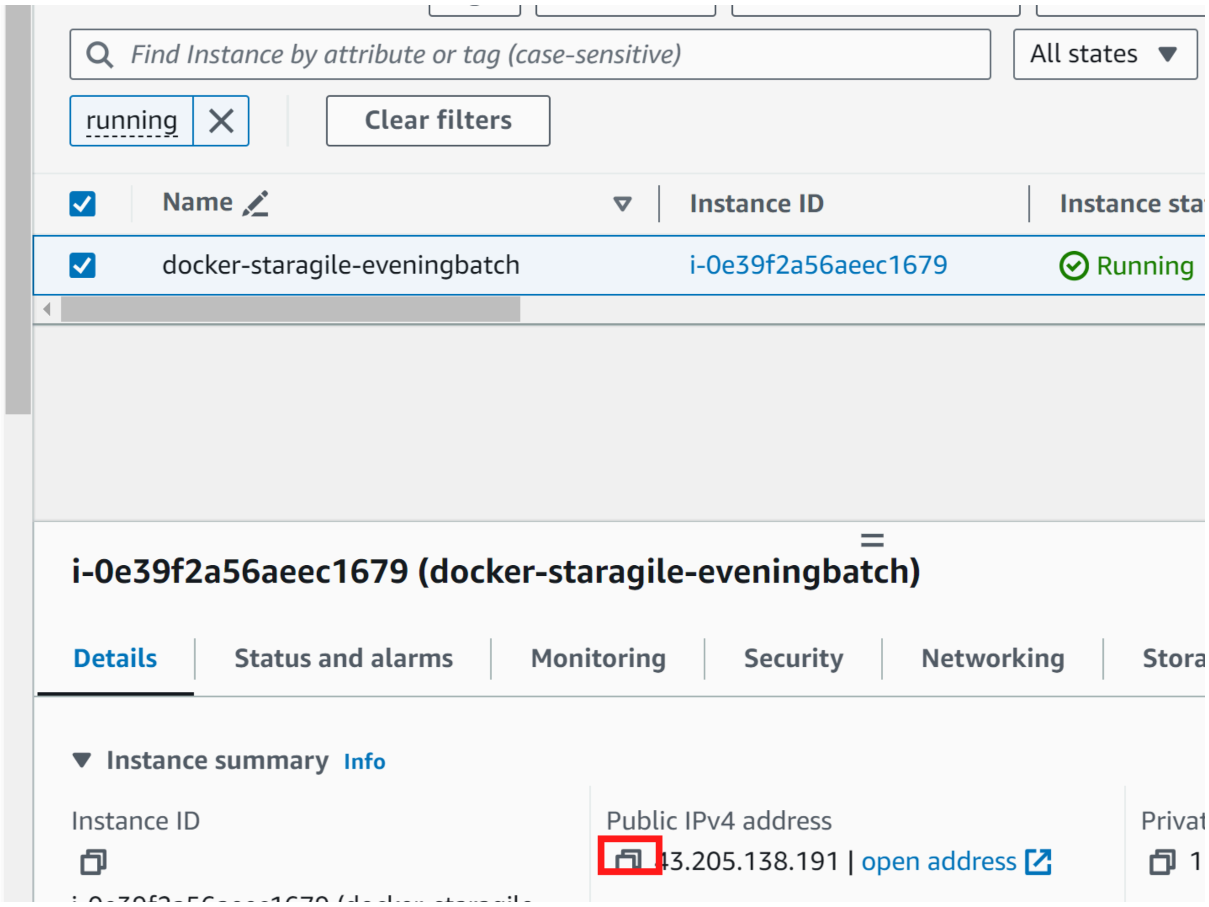
Security group rule ID	Type	Protocol	Port range	Source		Description - optional	
sgr-01c913d4d69dbc220	SSH	TCP	22	C...	0.0.0.0/0		Delete
sgr-0f6cb41150c0bb4d0	HTTP	TCP	80	C...	0.0.0.0/0		Delete
–	Custom TCP	TCP	8947	A...	0.0.0.0/0		Delete

Add rule

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access

Save

Now lets see if we can acces the app



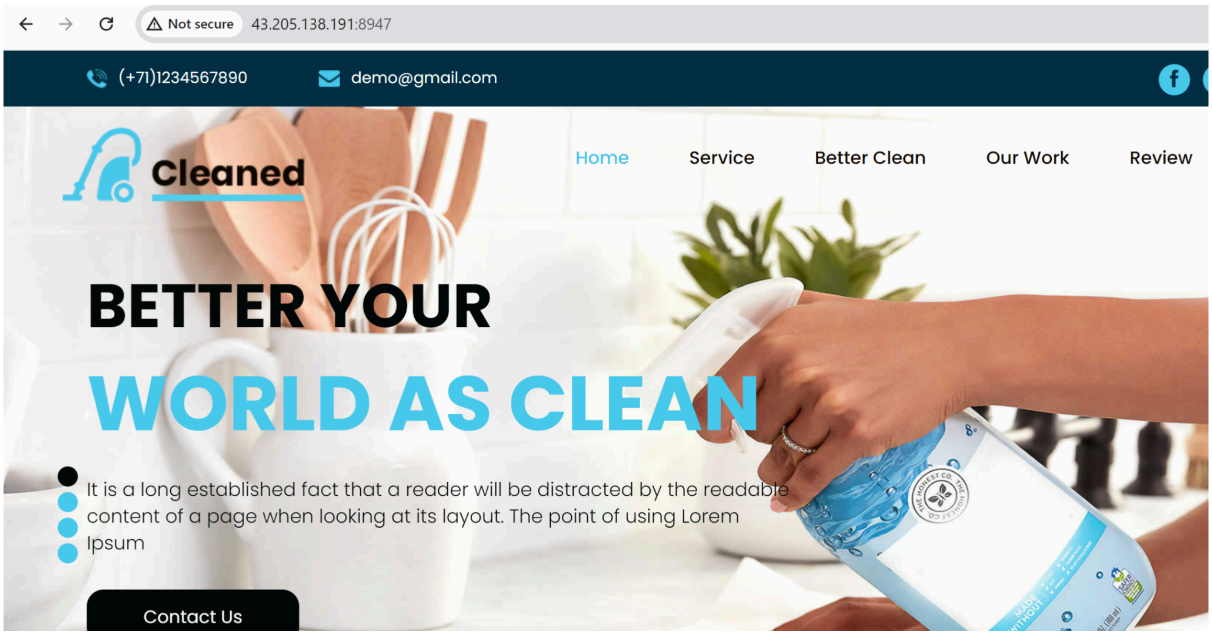
Name	Instance ID	Instance sta
docker-staragile-eveningbatch	i-0e39f2a56aeec1679	Running

i-0e39f2a56aeec1679 (docker-staragile-eveningbatch)

Instance ID

Public IPv4 address
43.205.138.191 | open address

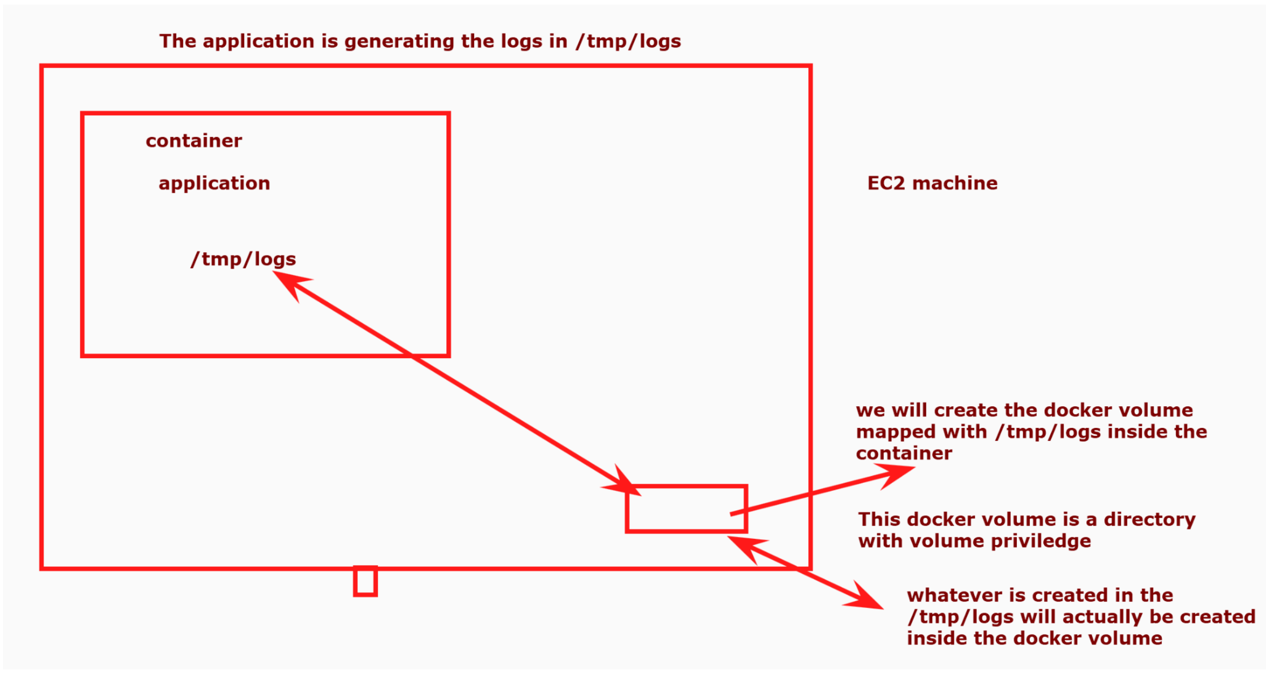
Privat
1


Not secure
43.205.138.191:8947
(+71)1234567890
demo@gmail.com
Cleaned
Home
Service
Better Clean
Our Work
Review
BETTER YOUR
WORLD AS CLEAN
It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum
Contact Us

################

DOCKER VOLUMES

Lets assume a situation ….lets say some logs are getting generated in your container..now when the container is deleted due to any reason those logs are also deleted….
And the problem is those logs have the reason why the container is deleted ..so in such cases We need to persist the logs ….which means that when the container is deleted the logs should be still be present ….in these cases we use concept of docker volume

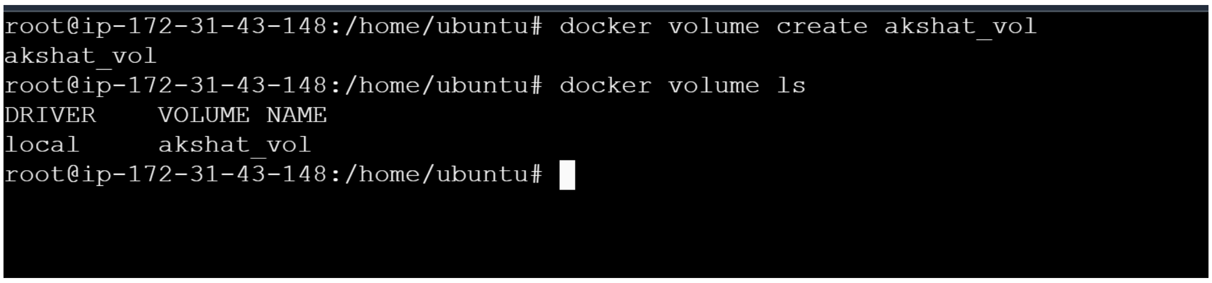

The application is generating the logs in /tmp/logs
container
application
/tmp/logs
EC2 machine
we will create the docker volume mapped with /tmp/logs inside the container
This docker volume is a directory with volume priviledge
whatever is created in the /tmp/logs will actually be created inside the docker volume

Remember that you cannot mount the docker volume with the existing container

# lets first create the docker volume

docker volume create akshat_vol
#akshat_vol is a directory created in ec2 machine with docker volume priviledge

docker volume ls
#to list all the volume …here the o/p would be akshat_vol

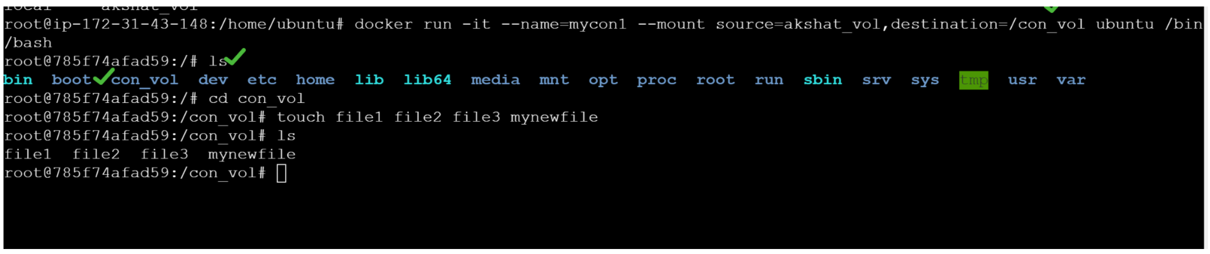

```
root@ip-172-31-43-148:/home/ubuntu# docker volume create akshat_vol
akshat_vol
root@ip-172-31-43-148:/home/ubuntu# docker volume ls
DRIVER    VOLUME NAME
local     akshat_vol
root@ip-172-31-43-148:/home/ubuntu#
```

# i want to create a container with the akshat_vol mapped with a directory inside the container

docker run -it --name=mycon1 --mount
source=akshat_vol,destination=/con_vol ubuntu /bin/bash

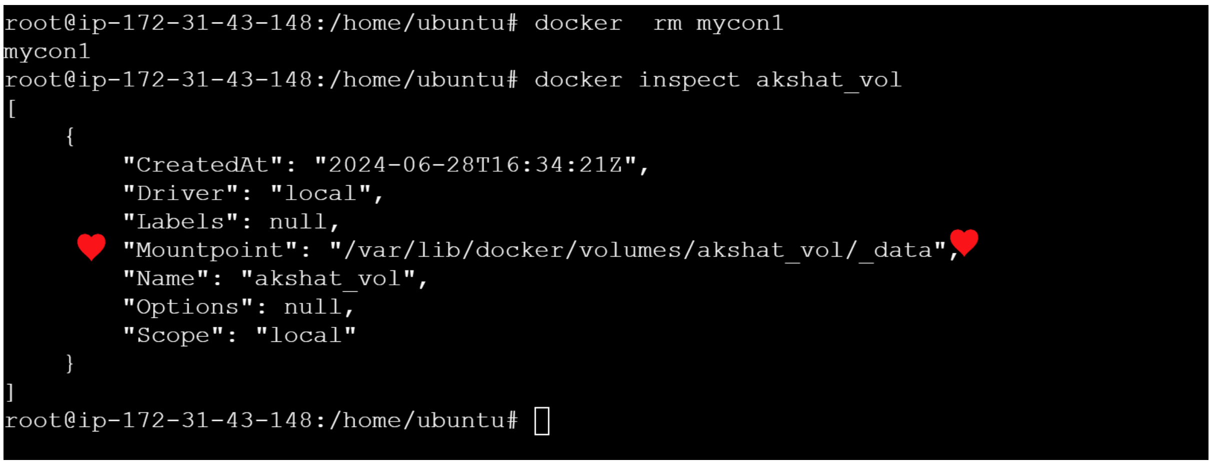
#in above command we will map the directory named con_vol of the container mycon1 with the docker volume created with the name akshat_vol
ls

(you will see con_vol)
cd con_vol
touch file1 file2 file4 file3
exit



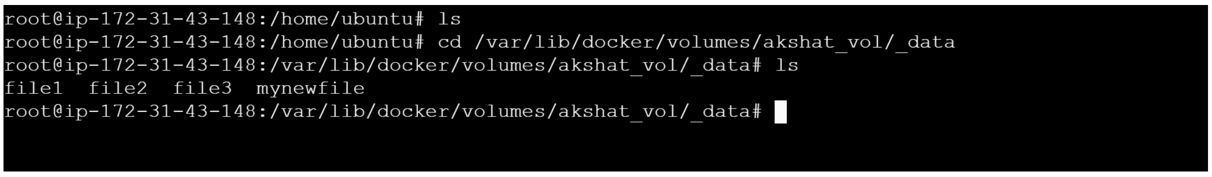
```
local     akshat_vol
root@ip-172-31-43-148:/home/ubuntu# docker run -it --name=mycon1 --mount source=akshat_vol,destination=/con_vol ubuntu /bin
/bash
root@785f74afad59:/# ls
bin  boot  con_vol  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
root@785f74afad59:/# cd con_vol
root@785f74afad59:/con_vol# touch file1 file2 file3 mynewfile
root@785f74afad59:/con_vol# ls
file1  file2  file3  mynewfile
root@785f74afad59:/con_vol# 
```

# just to ensure that if we delete the container still out data in the con_vol is persistent
docker rm mycon1
#now we want to go to location where akshat_vol is created
docker inspect akshat_vol



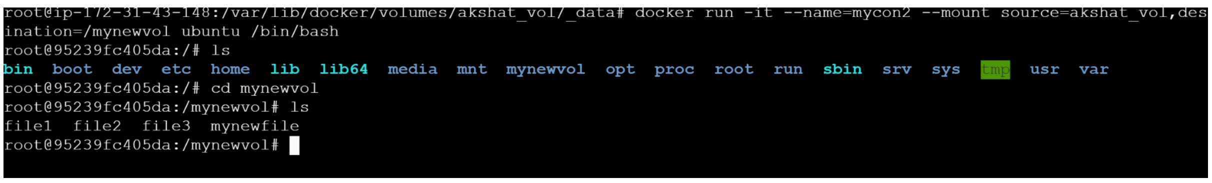
```
root@ip-172-31-43-148:/home/ubuntu# docker  rm mycon1
mycon1
root@ip-172-31-43-148:/home/ubuntu# docker inspect akshat_vol
[
    {
        "CreatedAt": "2024-06-28T16:34:21Z",
        "Driver": "local",
        "Labels": null,
        "Mountpoint": "/var/lib/docker/volumes/akshat_vol/_data",
        "Name": "akshat_vol",
        "Options": null,
        "Scope": "local"
    }
]
root@ip-172-31-43-148:/home/ubuntu# 
```

In the mountpoint we get the location where akshat_vol is actually present cd /var/lib/docker/volumes/akshat_vol/_data



```
root@ip-172-31-43-148:/home/ubuntu# ls
root@ip-172-31-43-148:/home/ubuntu# cd /var/lib/docker/volumes/akshat_vol/_data
root@ip-172-31-43-148:/var/lib/docker/volumes/akshat_vol/_data# ls
file1  file2  file3  mynewfile
root@ip-172-31-43-148:/var/lib/docker/volumes/akshat_vol/_data#
```

# we can map any number of container with the akshat_vol



```
root@ip-172-31-43-148:/var/lib/docker/volumes/akshat_vol/_data# docker run -it --name=mycon2 --mount source=akshat_vol,des
ination=/mynewvol ubuntu /bin/bash
root@95239fc405da:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  mynewvol  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
root@95239fc405da:/# cd mynewvol
root@95239fc405da:/mynewvol# ls
file1  file2  file3  mynewfile
root@95239fc405da:/mynewvol#
```

docker run -it --name=mycon2 --mount
source=akshat_vol,destination=/mynewvol ubuntu /bin/bash

cd mynewvol
ls
(you will see all the files present in akshat_vol)

######################

clic