Docker notes:

Docker is used for containerization.

Kubernetes is used for Orchestration

Jenkins is used for CI/CD

Application Architecture:

1. User Interface ---> Frontend of the application ---> End user will use application through UI (HTML, CSS, Angular, React)

2. Backend ---> Business logic (Java, Python, .NET)

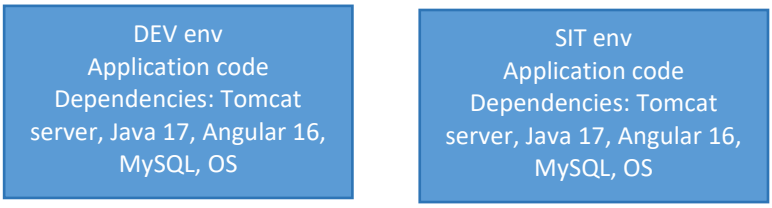
3. Database ---> It Is to store the information or data (MySQL, SQL server, Oracle, MongoDB)

Application code ---> developed application in Java, MySQL, Angular --> need server software to deploy application say Tomcat ---> only if all of them are there, we can make our application work ---> if we want to run our application code, we must setup all required dependencies (Required softwares to run our application) in the machine

Example: Java + MySQL + Angular + Tomcat server

Application environments:

It will go through Dev environment then System in Test (SIT), User Acceptance Testing (UAT), Pilot, Production environment where the application will be deployed



Dev Env is used by Developers for code integration and testing

SIT Env is used by Testers for system integration testing

UAT Env is used by client team for acceptance testing (Go or NoGo) Go ahead for production

Pilot Env is used for pre-production testing

Prod Env is used for the live deployment where end users can access our application

As a DevOps engineer, we are responsible to setup infrastructure to run our application --> basically we need to setup dependencies in all the environments (Dev, SIT, UAT, Pilot, Prod), everywhere we need to do it over and over again to run our application on all the environment. To setup infrastructure everytime over and over again, then there is a possibility of making mistakes. There is a high chance of making mistakes in dependency installation process (Version compatibility issues may occur).

To simplify application execution in any machine we can go with Docker

Docker:

Docker is a free and open-source software/tool.

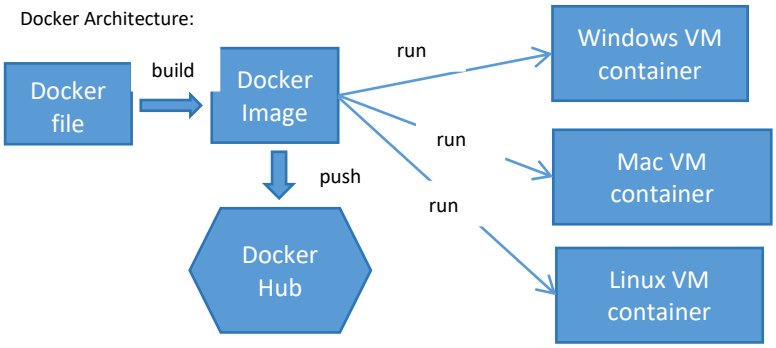
Used for containerization because of which it makes sure that we can run our application in any machine.

Container --> Application code + Application dependencies together

(Docker will take care of dependencies installation required for our application)

It will make our application portable using Docker

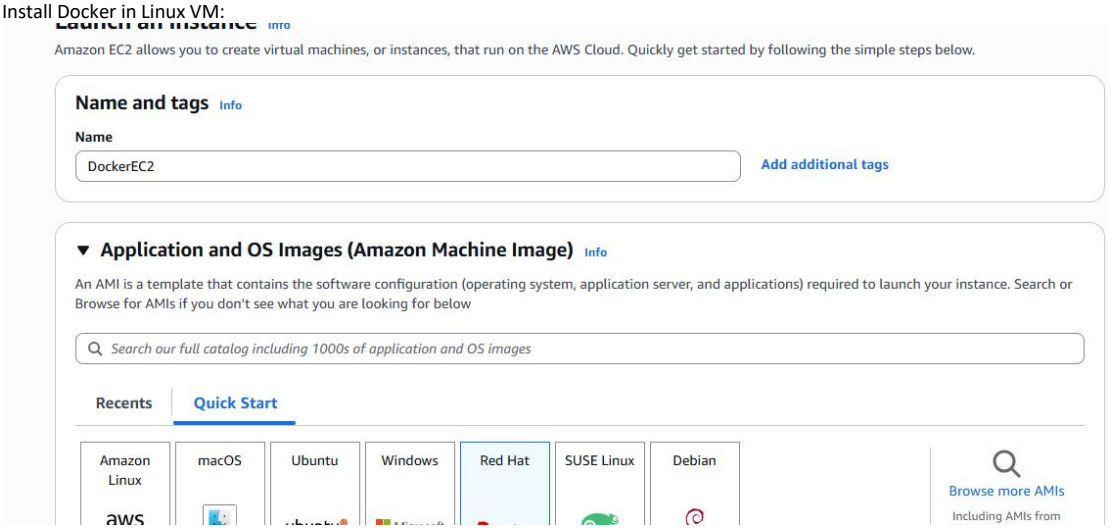
Docker Architecture:

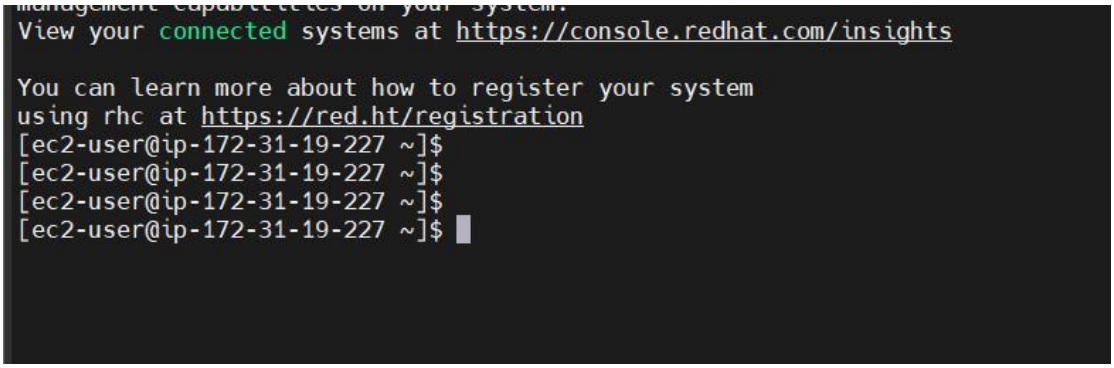


Dockerfile, Docker Image, Docker Registry, Docker container

Dockerfile is used to specify where our application code and to run the application what dependencies are required for our application execution ---> Docker file is required to build Docker Image ---> Docker image is a package, which contains code + dependencies (Everything that's required to run our application) ---> Docker Registry is used to store Docker Images ---> When we run Docker image then Docker container will be created (Docker container is a Linux VM where your entire application is running including code + dependencies) ---> Docker container is used to run our application

Install Docker in Linux VM:





Step 1: Create EC2 Linux VM and connect with VM using MobaXterm

Step 2: Execute commands sudo yum install docker

Step 3: Add EC2 user into docker group: Sudo usermod -aG docker ec2-user

Step 4: Verify Docker installation: sudo docker -v

To update yum package manager -y for auto-approve

[ec2-user@ip-172-31-19-227 ~]$ sudo yum update -y

To install Docker

[ec2-user@ip-172-31-19-227 ~]$ sudo yum install docker

sudo dnf update -y

sudo dnf install docker -y

sudo systemctl enable docker

sudo systemctl start docker

[ec2-user@ip-172-31-19-227 ~]$ sudo service docker start

Do this in case of error

sudo dnf remove docker docker-engine docker.io containerd runc -y

sudo dnf install docker -y

sudo systemctl enable --now docker

sudo dnf remove podman -y

sudo dnf install docker -y

sudo systemctl enable --now docker

1. Remove any old versions of Docker/Podman

sudo dnf remove podman docker docker-client docker-client-latest docker-common docker-latest docker-latest-logrotate docker-logrotate docker-engine containerd runc -y

2. Add Docker's official repo

sudo dnf config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo

3. Install Docker from Docker's official repo

sudo dnf install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y

4. Start and enable Docker

sudo systemctl enable --now docker

sudo docker run hello-world

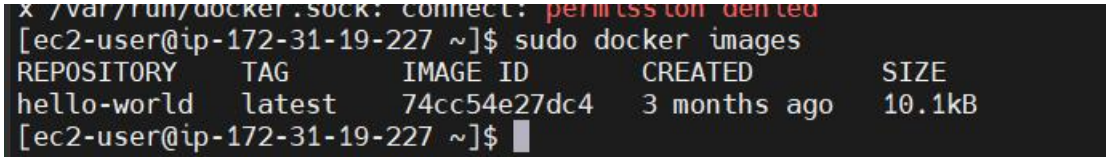
[ec2-user@ip-172-31-19-227 ~]$ sudo service docker start

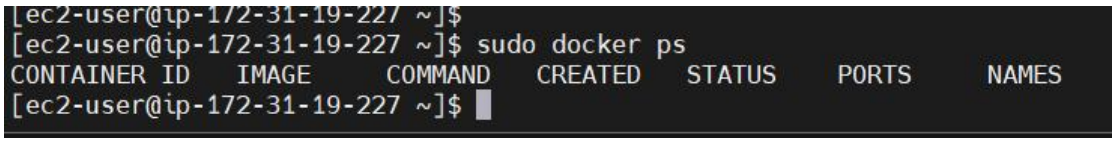
Redirecting to /bin/systemctl start docker.service

[ec2-user@ip-172-31-19-227 ~]$ sudo usermod -aG docker ec2-user

[ec2-user@ip-172-31-19-227 ~]$ docker -v

Docker version 28.1.1, build 4eba377





Docker Commands:
docker images --> It will display all the docker images available in our machine
docker ps --> To display running docker containers
docker ps -a --> To display the running + stopped containers
docker pull <image-id/image-name> --> To download Docker image from Docker hub
docker pull hello-world
docker images
docker run <image-id/name> ---> To create/run docker container

[ec2-user@ip-172-31-19-227 ~]$ sudo docker run 74cc54e27dc4

```
[ec2-user@ip-172-31-19-227 ~]$ sudo docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED          STATUS                      PORTS     NAMES
863dcde44c22   hello-world   "/hello"   21 minutes ago   Exited (0) 21 minutes ago             reverent_blackburn
[ec2-user@ip-172-31-19-227 ~]$
```

It's a Docker public image
[ec2-user@ip-172-31-19-227 ~]$ sudo docker pull hello-world
docker ps -a --> Display all running + stopped containers
docker rmi <image-id> --> To delete Docker image
[ec2-user@ip-172-31-19-227 ~]$ sudo docker rmi 74cc54e27dc4
Error response from daemon: conflict: unable to delete 74cc54e27dc4 (must be forced) - image is being used by stopped container 250ebc0567f3
docker rm <container-id> ---> To delete docker container
docker stop <container-id> --> To stop the docker container
docker start <container-id> --> To start the docker container
docker system prune -a --> System clean up everything is deleted, stopped containers, unused images
docker logs <container-id> -->
docker run -d <image-id/name> -->
docker run -d -p <Linux_port:Application_port><image-id/name> --> 8484 is the port# on host machine and 8080 is the port# of application inside Guest VM (Docker Container)
-d --> represents detached mode
-p ---> represents port mapping
Two containers cannot be mapped to same port number in the host machine

```
Run 'docker --help' for more information
[ec2-user@ip-172-31-19-227 ~]$ sudo docker pull  hello-world
Using default tag: latest
latest: Pulling from library/hello-world
Digest: sha256:c41088499908a59aae84b0a49c70e86f4731e588a737f1637e73c8c09d995654
Status: Image is up to date for hello-world:latest
docker.io/library/hello-world:latest
[ec2-user@ip-172-31-19-227 ~]$
```

```
permission denied while trying to connect to the Docker daemon so
x /var/run/docker.sock: connect: permission denied
[ec2-user@ip-172-31-19-227 ~]$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    74cc54e27dc4   3 months ago   10.1kB
[ec2-user@ip-172-31-19-227 ~]$
```


[ec2-user@ip-172-31-19-227 ~]$ sudo docker run 74cc54e27dc4

```
[ec2-user@ip-172-31-19-227 ~]$
[ec2-user@ip-172-31-19-227 ~]$ sudo docker run 74cc54e27dc4

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

```
[ec2-user@ip-172-31-19-227 ~]$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND    CREATED          STATUS                      PORTS     NAMES
250ebc0567f3   74cc54e27dc4   "/hello"   31 seconds ago   Exited (0) 31 seconds ago             dazzling_murdock
863dcde44c22   hello-world    "/hello"   27 minutes ago   Exited (0) 27 minutes ago             reverent_blackburn
[ec2-user@ip-172-31-19-227 ~]$
```

```
[ec2-user@ip-172-31-19-227 ~]$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    74cc54e27dc4   3 months ago   10.1kB
[ec2-user@ip-172-31-19-227 ~]$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND    CREATED          STATUS                      PORTS     NAMES
250ebc0567f3   74cc54e27dc4   "/hello"   12 minutes ago   Exited (0) 12 minutes ago             dazzling_murdock
863dcde44c22   hello-world    "/hello"   39 minutes ago   Exited (0) 39 minutes ago             reverent_blackburn
[ec2-user@ip-172-31-19-227 ~]$ docker rm 250ebc0567f3
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Delete "http://
rs/250ebc0567f3": dial unix /var/run/docker.sock: connect: permission denied
[ec2-user@ip-172-31-19-227 ~]$ sudo docker rm 250ebc0567f3
250ebc0567f3
[ec2-user@ip-172-31-19-227 ~]$ sudo docker rm 863dcde44c22
863dcde44c22
[ec2-user@ip-172-31-19-227 ~]$
```

[ec2-user@ip-172-31-19-227 ~]$ sudo docker rmi 74cc54e27dc4

```
863dcde44c22
[ec2-user@ip-172-31-19-227 ~]$ sudo docker rmi 74cc54e27dc4
Untagged: hello-world:latest
Untagged: hello-world@sha256:c41088499908a59aae84b0a49c70e86f4731e588a737f1637e73c8c09d995654
Deleted: sha256:74cc54e27dc41bb10dc4b2226072d469509f2f22f1a3ce74f4a59661a1d44602
Deleted: sha256:63a41026379f4391a306242eb0b9f26dc3550d863b7fdbb97d899f6eb89efe72
[ec2-user@ip-172-31-19-227 ~]$
```

```
[ec2-user@ip-172-31-19-227 ~]$
[ec2-user@ip-172-31-19-227 ~]$ sudo docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
e6590344b1a5: Pull complete
Digest: sha256:c41088499908a59aae84b0a49c70e86f4731e588a737f1637e73c8c09d995654
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
[ec2-user@ip-172-31-19-227 ~]$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    74cc54e27dc4   3 months ago   10.1kB
[ec2-user@ip-172-31-19-227 ~]$ sudo docker run 74cc54e27dc4

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Now the container is in the stopped state

[ec2-user@ip-172-31-19-227 ~]$ sudo docker start 20974c254043

```
>
[ec2-user@ip-172-31-19-227 ~]$ sudo docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED          STATUS                      PORTS     NAMES
20974c254043   74cc54e27dc4   "/hello"   42 seconds ago   Exited (0) 41 seconds ago             vibrant_darwin
[ec2-user@ip-172-31-19-227 ~]$ sudo docker start 20974c254043
20974c254043
[ec2-user@ip-172-31-19-227 ~]$
```

[ec2-user@ip-172-31-19-227 ~]$ sudo docker pull edydockers/sms-frontend:dev-14

```
Run 'docker --help' for more information
[ec2-user@ip-172-31-19-227 ~]$ sudo docker pull edydockers/sms-frontend:dev-14
dev-14: Pulling from edydockers/sms-frontend
f18232174bc9: Pull complete
61ca4f733c80: Pull complete
b464cfdf2a63: Pull complete
d7e507024086: Pull complete
81bd8ed7ec67: Pull complete
197eb75867ef: Pull complete
34a64644b756: Pull complete
39c2ddfd6010: Pull complete
d390a08a6ca2: Pull complete
9185db7dbf78: Pull complete
Digest: sha256:0bd3fe140e66fdcc7ca3a4fc3fe2a58e6cc96e940d6c095c969251c8d22a84ce
Status: Downloaded newer image for edydockers/sms-frontend:dev-14
docker.io/edydockers/sms-frontend:dev-14
[ec2-user@ip-172-31-19-227 ~]$
```

[ec2-user@ip-172-31-19-227 ~]$ sudo docker images
REPOSITORY TAG IMAGE ID CREATED SIZE

edydockers/sms-frontend dev-14 4f8e1ef06924 27 hours ago 53.7MB
hello-world latest 74cc54e27dc4 3 months ago 10.1kB

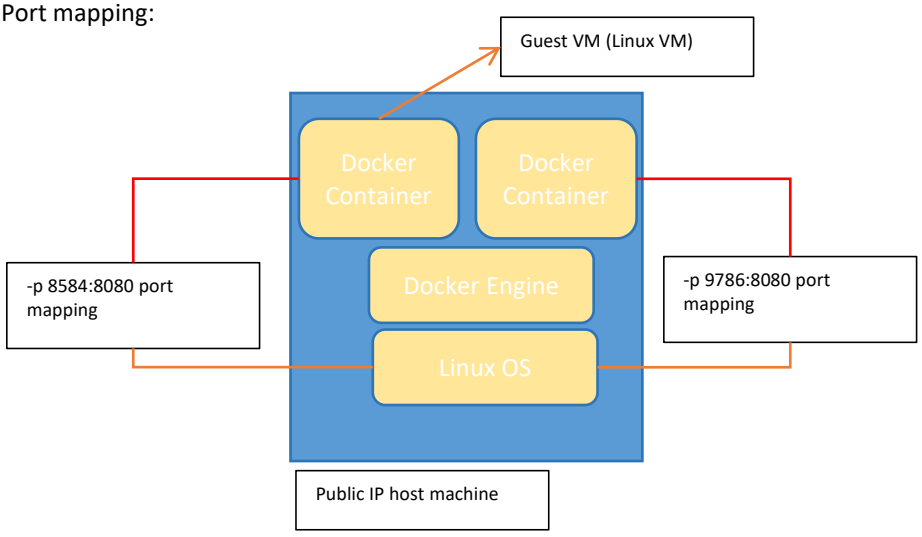
[ec2-user@ip-172-31-19-227 ~]$ sudo docker run 4f8e1ef06924

```
/docker-entrypoint.sh: exec: line 47: illegal option -d
[ec2-user@ip-172-31-19-227 ~]$ sudo docker run -d 4f8e1ef06924
9c15cc3a49d4e42b22bec2f470e79ff35a5bf8665734ccb8556b7c7563e92723
[ec2-user@ip-172-31-19-227 ~]$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS     NAMES
9c15cc3a49d4   4f8e1ef06924   "/docker-entrypoint.…"   6 seconds ago   Up 5 seconds   80/tcp    tender_lewin
[ec2-user@ip-172-31-19-227 ~]$
```

Container is currently running
[ec2-user@ip-172-31-19-227 ~]$ sudo docker system prune -a

```
hello-world               latest    74cc54e27dc4   3 months ago   10.1kB
[ec2-user@ip-172-31-19-227 ~]$ sudo docker system prune -a
WARNING! This will remove:
  - all stopped containers
  - all networks not used by at least one container
  - all images without at least one container associated to them
  - all build cache

Are you sure you want to continue? [y/N] y
Deleted Containers:
9c15cc3a49d4e42b22bec2f470e79ff35a5bf8665734ccb8556b7c7563e92723
b8aeea3ab046398a6cfb75a553aa77d59aa023acb91458d50c7474ffc0280d27
70a867af7eecf4b644eaa17597b5567b8f84c71d8dd97879ecda98c2c449e018
20974c2540439219ce13a578d3e137ce26dc1242b1f5ec0d07f184971830e17d

Deleted Images:
untagged: edydockers/sms-frontend:dev-14
untagged: edydockers/sms-frontend@sha256:0bd3fe140e66fdcc7ca3a4fc3fe2a58e6cc96e940d6c095c969251c8d22a84ce
deleted: sha256:4f8e1ef06924571e0b51ced161b7adaf7950a9791b6ca9dec64a34f37314ba2e
deleted: sha256:05fbd0195b6ffcdf22612dcc49f42cf56d5ea05b816aab2d20c669ffa8d38f85
deleted: sha256:31b85312a91a64a2f3b96354d3d0219381c1932b4994e5d553610dd5b24980c3
deleted: sha256:61b1a4867a5ae766114a788e5bd5422883ed50a376763f2c623da4222aba2d81
deleted: sha256:018d9d54c2032d4d4c2d6cd9ad1292dec9463c41d3fbff9ae33947e9a071a8e1
deleted: sha256:8f43be8a4daba1435a4de29cbd23e64a3280249976ade8f5d75adc871fce09ef
deleted: sha256:eb85f39adaa54c0bdcd4c410785cccde672302de053b06d09e1f07113ab80b49
deleted: sha256:4033c2b5e093c9a211f26455b770f716674ee997415f584190c5c6813b88332b
deleted: sha256:737f9853aaaf89ede956c2d9a2841590765f6ad7cd2b4c26a49bbd284befa02d
deleted: sha256:7e255aea49cfab67e01a3351c1cc1baaf988c36fae16c7d37ad1f10546a7365c
deleted: sha256:08000c18d16dadf9553d747a58cf44023423a9ab010aab96cf263d2216b8b350
untagged: hello-world:latest
untagged: hello-world@sha256:c41088499908a59aae84b0a49c70e86f4731e588a737f1637e73c8c09d995654
deleted: sha256:74cc54e27dc41bb10dc4b2226072d469509f2f22f1a3ce74f4a59661a1d44602
deleted: sha256:63a41026379f4391a306242eb0b9f26dc3550d863b7fdbb97d899f6eb89efe72

Total reclaimed space: 53.72MB
[ec2-user@ip-172-31-19-227 ~]$
```

Port mapping:



When I map Docker port mapping to Linux VM, I can access the application using public IP
-p 8485:9090

```
Run 'docker --help' for more information
[ec2-user@ip-172-31-19-227 ~]$ sudo docker run -d 4f8e1ef06924
6bb231f1eafec39ac35d16c45e6121f96e391a91d824cc32a63cebd7d8bd1a71
[ec2-user@ip-172-31-19-227 ~]$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS                      PORTS     NAMES
6bb231f1eafe   4f8e1ef06924   "/docker-entrypoint.…"   6 seconds ago    Up 6 seconds                80/tcp    interesting_swanson
085ab62d5c7c   4f8e1ef06924   "/docker-entrypoint.…"   23 seconds ago   Exited (0) 16 seconds ago             peaceful_merkle
[ec2-user@ip-172-31-19-227 ~]$
```

Public IP: 3.99.142.28

```
[ec2-user@ip-172-31-19-227 ~]$
[ec2-user@ip-172-31-19-227 ~]$ sudo docker pull hacker123shiva/springbt-in-docker:latest
latest: Pulling from hacker123shiva/springbt-in-docker
38a980f2cc8a: Pull complete
de849f1cfbe6: Pull complete
a7203ca35e75: Pull complete
d667a25d6d20: Pull complete
5c3147aac344: Pull complete
Digest: sha256:1535b83e22cc9dafe5a031570682e4a818f473d4571d617699d6777a6dfccf40
Status: Downloaded newer image for hacker123shiva/springbt-in-docker:latest
docker.io/hacker123shiva/springbt-in-docker:latest
[ec2-user@ip-172-31-19-227 ~]$
```

```
[ec2-user@ip-172-31-19-227 ~]$
[ec2-user@ip-172-31-19-227 ~]$ sudo docker images
REPOSITORY                          TAG       IMAGE ID       CREATED         SIZE
hacker123shiva/springbt-in-docker   latest    3f98d4db2087   11 months ago   493MB
[ec2-user@ip-172-31-19-227 ~]$
```

[ec2-user@ip-172-31-19-227 ~]$ sudo docker run -d 3f98d4db2087
1f62a9e6d8f5d6764eb231ff2d0409bd1ba4e3d2966f46622ee7c2d27fb273bc

```
[ec2-user@ip-172-31-19-227 ~]$
[ec2-user@ip-172-31-19-227 ~]$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS      NAMES
1f62a9e6d8f5   3f98d4db2087   "java -jar first-spr…"   27 seconds ago   Up 27 seconds   8080/tcp   adoring_johnson
[ec2-user@ip-172-31-19-227 ~]$
```

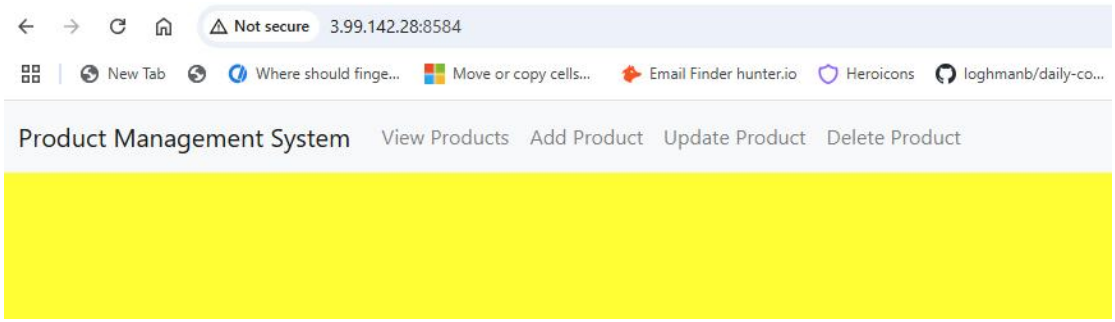
```
[ec2-user@ip-172-31-19-227 ~]$ sudo docker logs 1f62a9e6d8f5

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v3.2.2)

2025-05-04T00:03:27.592Z  INFO 1 --- [           main] c.j.f.FirstSpringbootJenkinsApplication  : St
sing Java 17.0.2 with PID 1 (/app/first-springboot-jenkins-0.0.1-SNAPSHOT.jar started by root in /ap
2025-05-04T00:03:27.598Z  INFO 1 --- [           main] c.j.f.FirstSpringbootJenkinsApplication  : No
efault"
2025-05-04T00:03:29.867Z  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : To
2025-05-04T00:03:29.898Z  INFO 1 --- [           main] o.apache.catalina.core.StandardService   : St
2025-05-04T00:03:29.899Z  INFO 1 --- [           main] o.apache.catalina.core.StandardEngine    : St
2025-05-04T00:03:30.192Z  INFO 1 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : In
2025-05-04T00:03:30.195Z  INFO 1 --- [           main] w.s.c.ServletWebServerApplicationContext : Ro
 ms
2025-05-04T00:03:30.720Z  INFO 1 --- [           main] o.s.b.a.w.s.WelcomePageHandlerMapping    : Ad
2025-05-04T00:03:31.370Z  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : To
2025-05-04T00:03:31.415Z  INFO 1 --- [           main] c.j.f.FirstSpringbootJenkinsApplication  : St
process running for 6.015)
[ec2-user@ip-172-31-19-227 ~]$
```

Application is running on port: 8080
2025-05-04T00:03:29.867Z INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer :
Tomcat initialized with port 8080 (http)

Still if I run : 3.99.142.28:8080 I wont see anything
Where is this deployed application?

It is in the container.
[ec2-user@ip-172-31-19-227 ~]$ sudo docker run -d -p 8584:8080 hacker123shiva/springbt-in-docker:latest

I want to map port 8584 in the host machine to 8080 in container
3.99.142.28:8584



What's a Dockerfile?
It is a blueprint for the Docker image
Dockerfile contains instructions to build docker image
(Used to specify where the application code is and what dependencies are required for our application execution) --> Blueprint of our Docker image
==> To write dockerfile we use the following keywords
FROM (To specify base image in the application)
MAINTAINER (To specify who's maintaining this dockerfile, who has made changes)
RUN (Specify instructions at the time of docker image creation)
CMD (What to execute at the time of Docker container creation)
COPY (You want to copy something from Host machine into Container machine)
WORKDIR
EXPOSE
ENTRYPOINT (To specify something has to be created at the time of container creation)
USER
RUN is say you want to execute at the time of Image creation, while CMD is you want to execute at the time of container creation