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

SIT env

Application code

Dependencies: Tomcat server, Java 17, Angular 16, MySQL, OS

DEV env

Application code

Dependencies: Tomcat server, Java 17, Angular 16, MySQL, OS

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:

run

Windows VM container

build

Docker Image

Docker file

run

Mac VM container

run

push

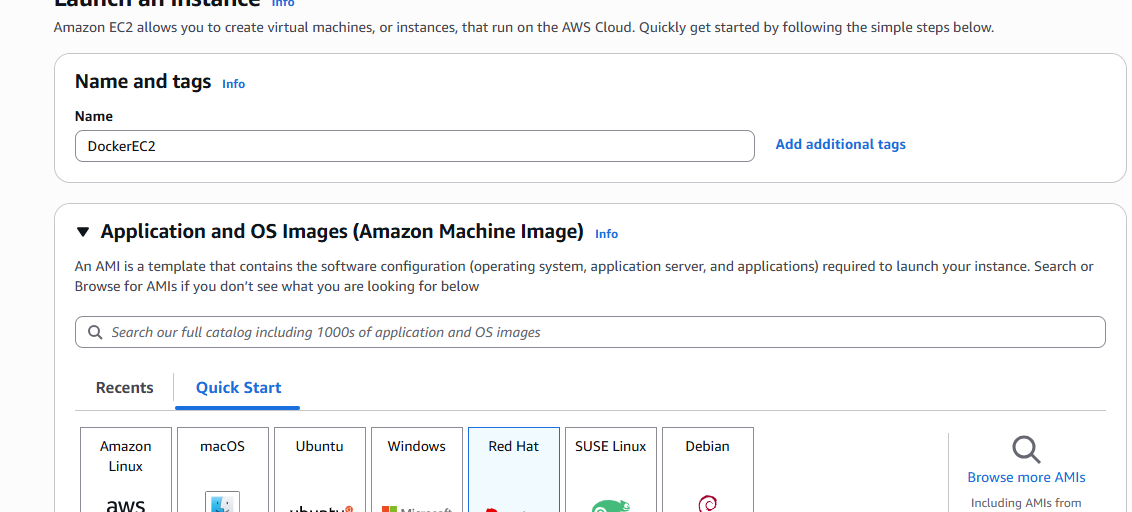
Docker Hub

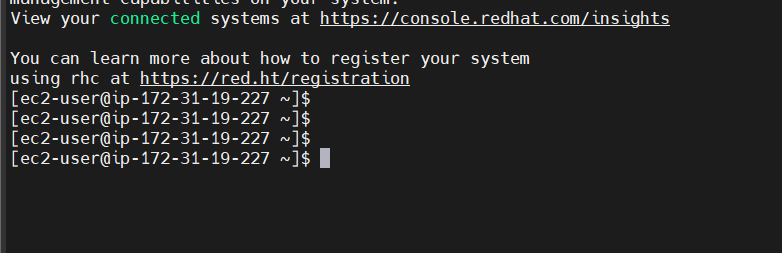
Linux VM container

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

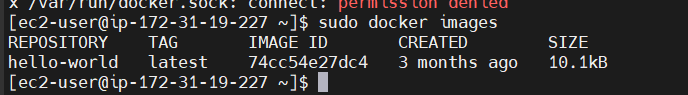
[ec2-user@ip-172-31-19-227 ~]$ sudo service docker start

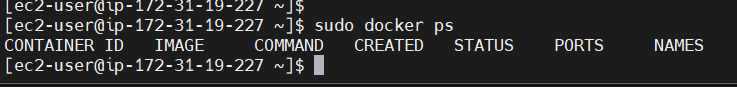
Redirecting to /bin/systemctl start docker.service

[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

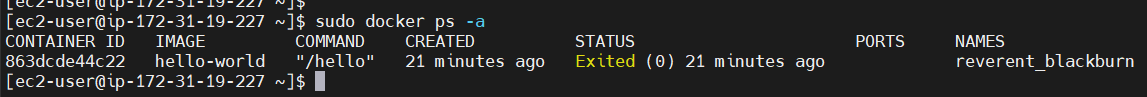
docker pull <image-id/image-name> --> To download Docker image from Docker hub

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



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> -->

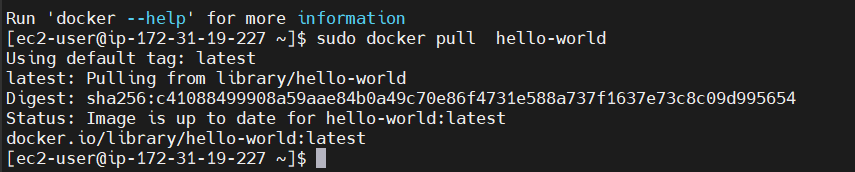
docker run -d <image-id/name> -->

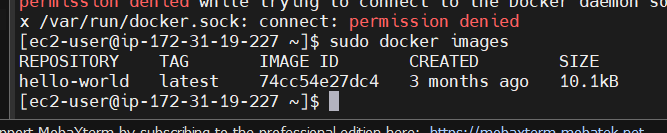
docker run -d -p <Linux\_port:Application\_port><image-id/name> --> 8484 is the port# on host machine and 8080 is the port# of application inside Guest VM (Docker Container)

-d --> represents detached mode

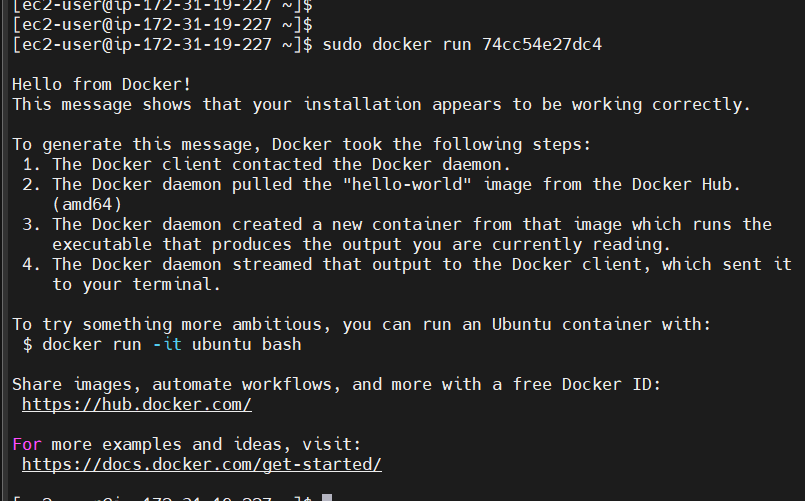
-p ---> represents port mapping

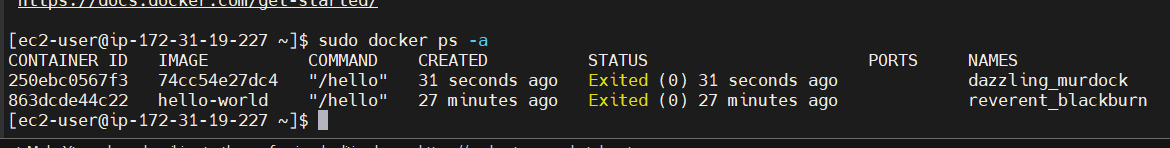
Two containers cannot be mapped to same port number in the host machine

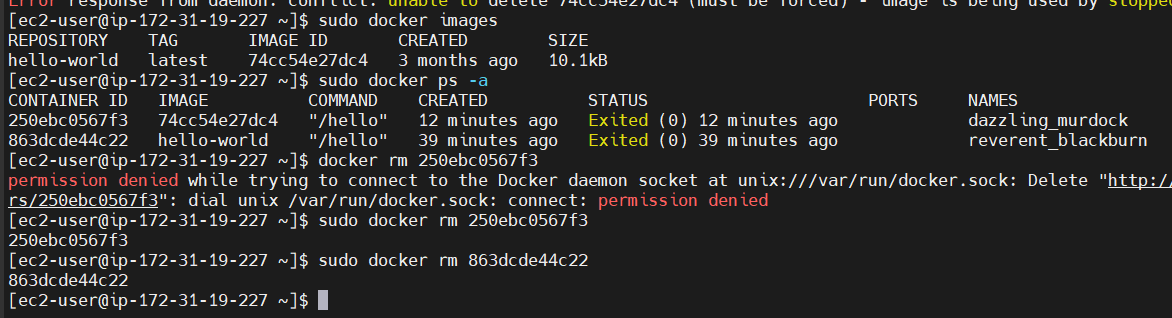




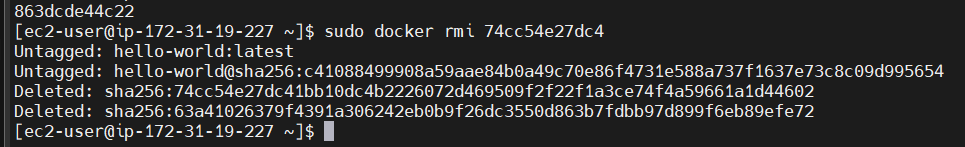
[ec2-user@ip-172-31-19-227 ~]$ sudo docker run 74cc54e27dc4

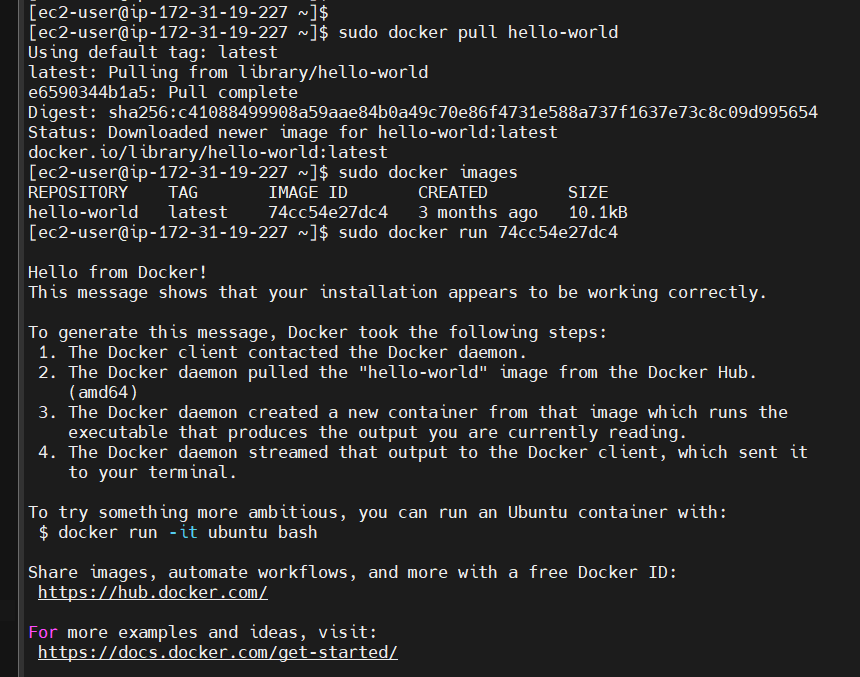






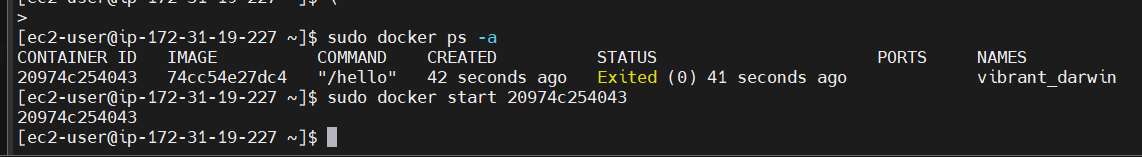
[ec2-user@ip-172-31-19-227 ~]$ sudo docker rmi 74cc54e27dc4



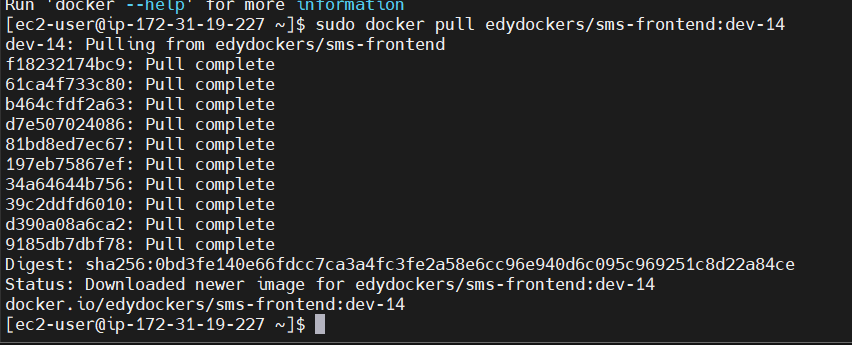


Now the container is in the stopped state

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



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



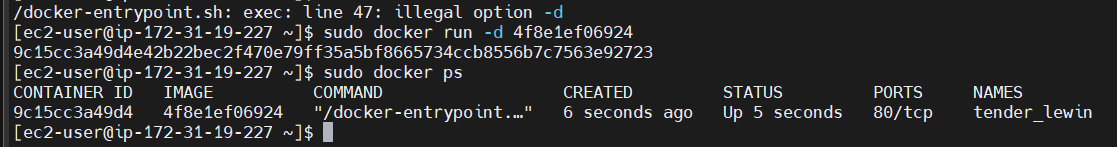
[ec2-user@ip-172-31-19-227 ~]$ sudo docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

edydockers/sms-frontend dev-14 4f8e1ef06924 27 hours ago 53.7MB

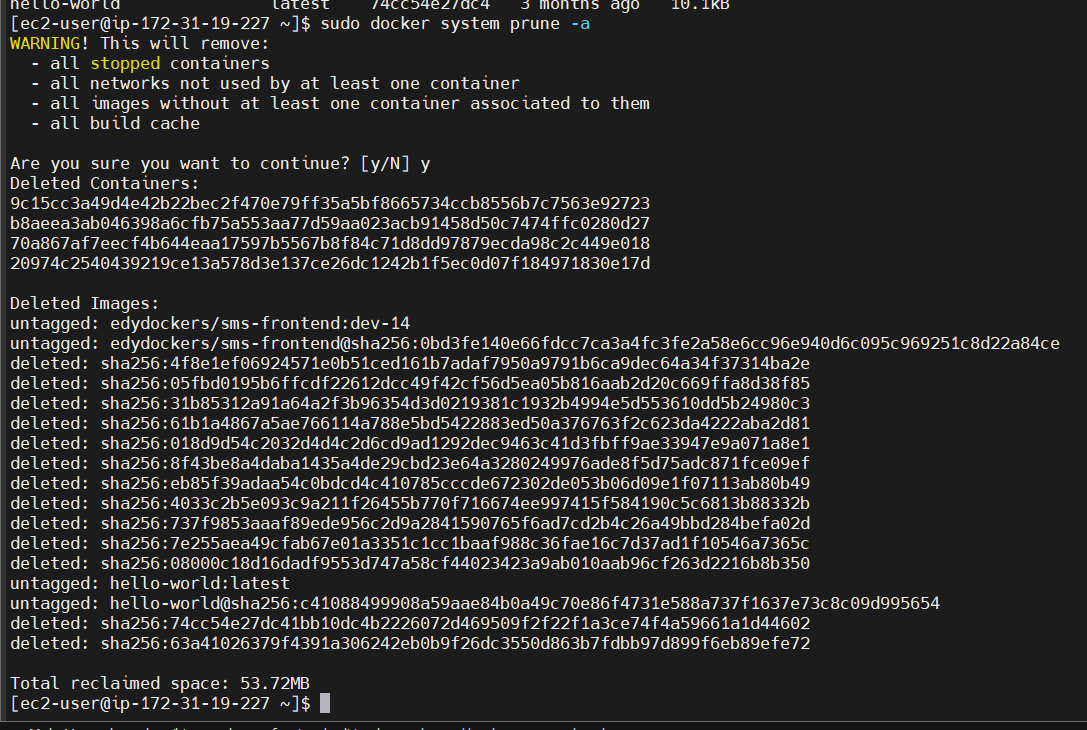
hello-world latest 74cc54e27dc4 3 months ago 10.1kB

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



Container is currently running

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



Port mapping:

Guest VM (Linux VM)

Docker Container

Docker Container

Docker Engine

-p 8584:8080 port mapping

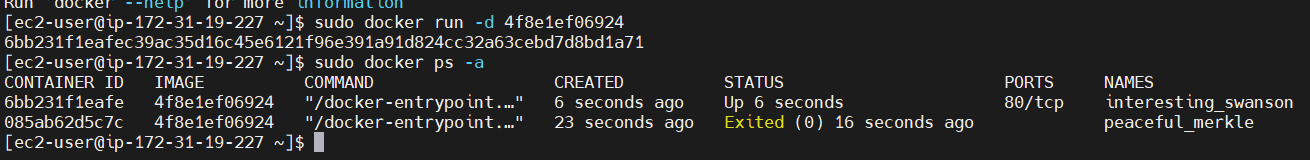
-p 9786:8080 port mapping

Linux OS

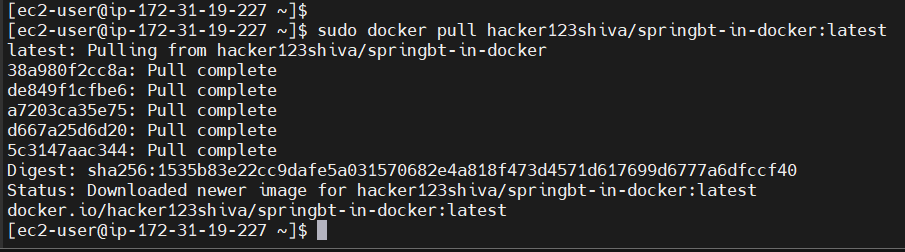
Public IP host machine

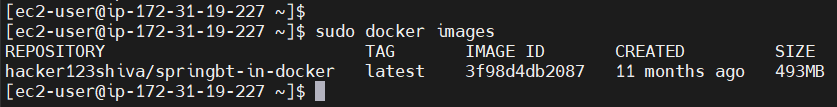
When I map Docker port mapping to Linux VM, I can access the application using public IP

-p 8485:9090



Public IP: 3.99.142.28

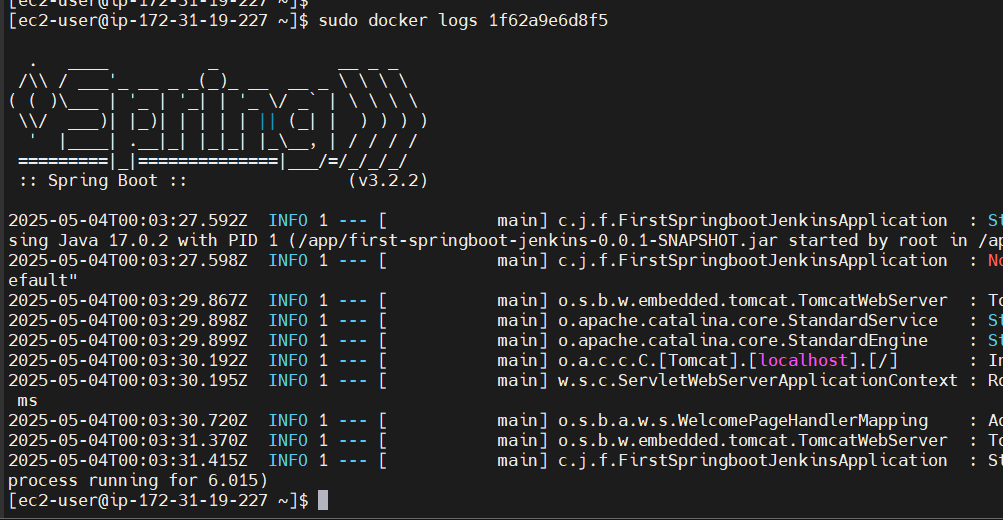




[ec2-user@ip-172-31-19-227 ~]$ sudo docker run -d 3f98d4db2087

1f62a9e6d8f5d6764eb231ff2d0409bd1ba4e3d2966f46622ee7c2d27fb273bc





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

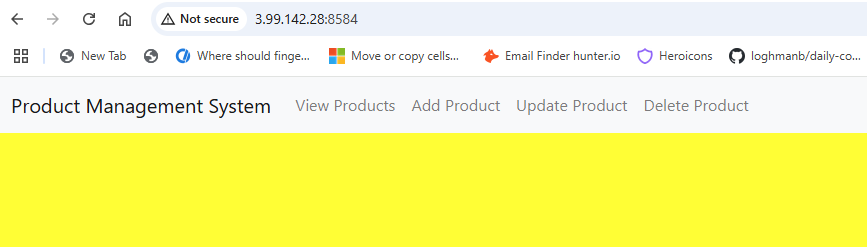
Where is this deployed application?

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