

TASK-01 DOCKER

Date:26/4/24

Q.1 Write a note on difference between Virtualization vs Containerization vs Bare Metal

Virtualization:

Virtualization involves creating virtual instances of hardware platforms, operating systems, storage devices, or network resources. It enables running multiple operating systems and applications on a single physical machine. Each virtual instance, known as a virtual machine (VM), operates independently, with its own allocated resources such as CPU, memory, storage, and networking. Hypervisors manage the virtualization process, allowing multiple VMs to run concurrently on the same hardware. Virtualization offers benefits like hardware resource optimization, isolation between VMs, and the ability to run legacy applications.

Containerization:

Containerization is a lightweight form of virtualization that encapsulates an application and its dependencies into a single package called a container. Containers share the host operating system's kernel and resources, making them more lightweight and efficient compared to VMs. They provide consistency across different environments, enabling applications to run reliably from development to production. Containerization platforms like Docker and Kubernetes have gained popularity due to their ability to streamline the development, deployment, and scaling of applications. Containers offer benefits such as portability, scalability, and faster deployment times compared to traditional virtualization.

Bare Metal:

Bare metal refers to running applications directly on the physical hardware without any intervening layer of abstraction like virtualization or containerization. In a bare metal environment, the operating system interacts directly with the underlying hardware, maximizing performance and resource

utilization. Bare metal setups are often used in high-performance computing (HPC), real-time applications, and scenarios where minimal overhead and maximum control are required. While bare metal environments offer the highest level of performance and control, they may require more management effort compared to virtualized or containerized environments.

Parameters	Virtualization	Containerization	Bare Metal
Resource Allocation	Virtualization divides physical resources into virtual ones, allowing multiple isolated instances to share the same hardware.	Containerization shares the host operating system's resources among containers, providing lightweight isolation.	Bare metal environments utilize the entire physical hardware for each application.
Performance Overhead	Virtualization introduces some overhead due to the hypervisor layer	Containerization has less overhead since containers share the host OS kernel.	Bare metal environments offer the best performance with minimal overhead .
Isolation	Virtualization provides strong isolation between VMs since each VM operates independently.	Containerization offers lighter isolation , sharing the host OS kernel but providing separate user spaces.	Bare metal environments offer no isolation between applications running directly on the hardware
Portability	It is portable but require additional management overhead.	Containers are highly portable, allowing applications to run consistently	Bare metal setups are less portable since applications are tightly coupled with the

		across different environments.	underlying hardware.
Management Complexity	It require less manual management compared to Bare metal or containerized envirinments.	It require medium manual management compared to virtulized envirinments.	Bare metal environments require more manual management compared to virtualized or containerized environments.

Q.2 Create an httpd container, inside that container create 2 more html pages (eg. Home.html and about.html) which will be accessible from browser.

To Create an httpd container use the following command:

- `docker run -d --name<Containername> -p80:80 <imagename>`
- eg. `docker run -d --name httpdcontainer -p80:80 httpd`

To see container :

- `docker ps`

```
root@ip-172-31-26-137: /home/ubuntu
httpd latest 356125da0595 6 weeks ago 147MB
root@ip-172-31-26-137:/home/ubuntu# docker run -d --name httpdcontainer -p80:80 httpd
5c8a3ab3aa1e8f98ff41ddd07b61b9f3d9598277c1b7b45ab5fdc148cd9cb04c
root@ip-172-31-26-137:/home/ubuntu# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS
5c8a3ab3aa1e   httpd     "httpd-foreground"      3 seconds ago Up 2 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp
httpdcontainer
root@ip-172-31-26-137:/home/ubuntu# docker exec -it httpdcontainer /bin/bash
root@5c8a3ab3aa1e:/usr/local/apache2# ls
bin build cgi-bin conf error htdocs icons include logs modules
root@5c8a3ab3aa1e:/usr/local/apache2# cd htdocs/
root@5c8a3ab3aa1e:/usr/local/apache2/htdocs# ls
index.html
root@5c8a3ab3aa1e:/usr/local/apache2/htdocs# nano index.html
bash: nano: command not found
root@5c8a3ab3aa1e:/usr/local/apache2/htdocs# apt update
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8786 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [13.8 kB]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [156 kB]
Fetched 9210 kB in 2s (6058 kB/s)
Reading package lists... Done
```

To go inside the container use following command:

- `docker exec -it <container name> /bin/bash`
- eg. `docker exec -it httpdcontainer /bin/bash`

To create html pages:

- Go to , `cd /usr/local/apache2/htdocs`
- `apt update`
- `apt install nano -y`
- create page , `nano home.html` (see below screenshot)
- create page , `nano about.html` (see below screenshot)

To see page on browser:

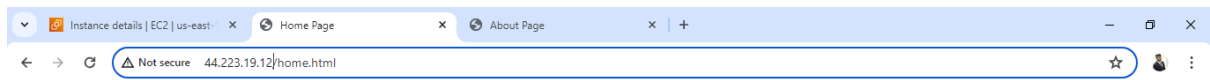
- <http://PUBLIC IP of server/page name>
- <http://44.223.19.12/home.html>
- <http://44.223.19.12/about.html>

```
root@ip-172-31-26-137: /home/ubuntu
GNU nano 7.2 home.html
<!-- home.html -->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Home Page</title>
</head>
<body>
  <h1>Welcome to the Home Page!</h1>
  <p>This is the home page content.</p>
</body>
</html>

[ Read 14 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/_ Go To Line  M-E Redo
```

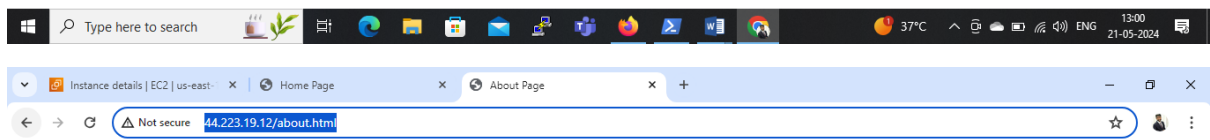
```
root@ip-172-31-26-137: /home/ubuntu
GNU nano 7.2 about.html
<!-- about.html -->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>About Page</title>
</head>
<body>
  <h1>Welcome to About Page</h1>
  <h2>Swapnil Sagar , 7709628645 </h2>
</body>
</html>

[ Read 13 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/_ Go To Line  M-E Redo
```



Welcome to the Home Page!

This is the home page content.



Welcome to About Page

Swapnil Sagar , 7709628645

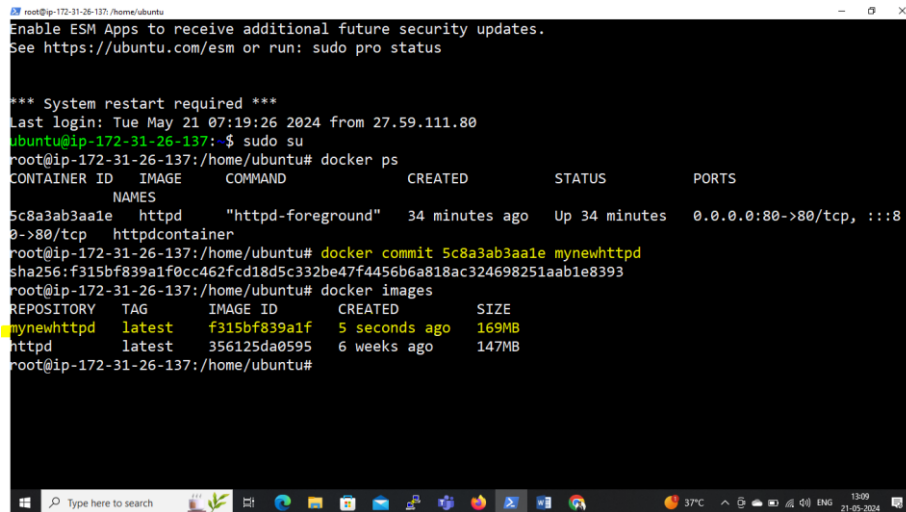


Q.3 Create image from httpd container and push that image to

- a. Docker hub
- b. ECR (AWS students)

To Create image from container use the command:

- docker commit <containerid/name> image name
- Eg. docker commit 5c8a3ab3aa1e mynewhttpd



```
root@ip-172-31-26-137: /home/ubuntu
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

*** System restart required ***
Last login: Tue May 21 07:19:26 2024 from 27.59.111.80
ubuntu@ip-172-31-26-137:~$ sudo su
root@ip-172-31-26-137:/home/ubuntu# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS
5c8a3ab3aa1e   httpd     "httpd-foreground"      34 minutes ago Up 34 minutes 0.0.0.0:80->80/tcp, :::80->80/tcp
httpdcontainer
root@ip-172-31-26-137:/home/ubuntu# docker commit 5c8a3ab3aa1e mynewhttpd
sha256:f315bf839a1f0cc462fcd18d5c332be47f4456b6a818ac324698251aab1e8393
root@ip-172-31-26-137:/home/ubuntu# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
mynewhttpd    latest    f315bf839a1f   5 seconds ago  169MB
httpd         latest    356125da0595   6 weeks ago   147MB
root@ip-172-31-26-137:/home/ubuntu#
```

A. To push this image to Docker Hub:

Create a repository on docker-hub:

- Eg. image-repo_httpd

To give a tag to the image who want to push:

- docker tag <imagename>
<dockerhubusername/repositoryname>:<tagname>
- Eg. docker tag mynewhttpd thenameis/image-repo_httpd:V1.0

```
root@ip-172-31-26-137: /home/ubuntu
Expanded Security Maintenance for Applications is not enabled.

4 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

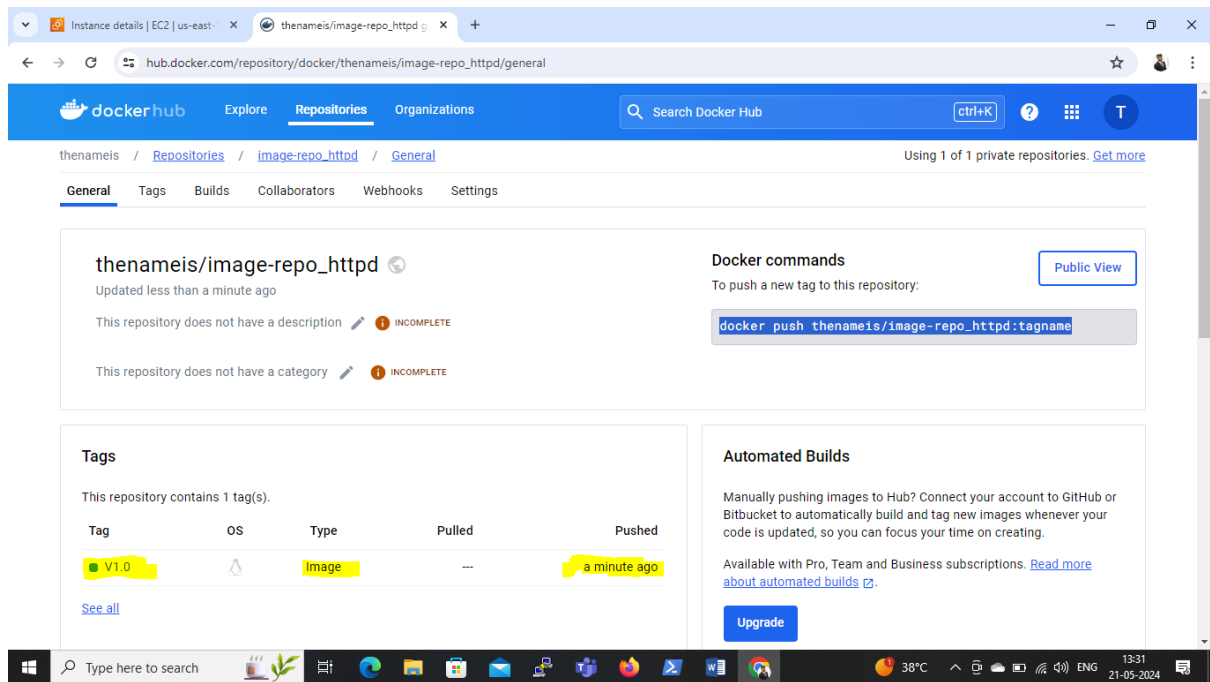
*** System restart required ***
Last login: Tue May 21 07:46:45 2024 from 27.59.111.80
ubuntu@ip-172-31-26-137:~$ sudo su
root@ip-172-31-26-137:/home/ubuntu# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
mynewhttpd    latest   f315bf839a1f   15 minutes ago 169MB
httpd         latest   356125da0595   6 weeks ago    147MB
root@ip-172-31-26-137:/home/ubuntu# docker tag mynewhttpd thenameis/image-repo_httpd:V1.0
root@ip-172-31-26-137:/home/ubuntu# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
mynewhttpd    latest   f315bf839a1f   16 minutes ago 169MB
thenameis/image-repo_httpd  V1.0     f315bf839a1f   16 minutes ago 169MB
httpd         latest   356125da0595   6 weeks ago    147MB
root@ip-172-31-26-137:/home/ubuntu# docker push thenameis/image-repo_httpd:V1.0
The push refers to repository [docker.io/thenameis/image-repo_httpd]
```

To Push the image to docker hub use the following command:

```
docker push thenameis/image-repo_httpd:tagname
```

➤ docker push thenameis/image-repo_httpd:V1.0

```
root@ip-172-31-26-137: /home/ubuntu
*** System restart required ***
Last login: Tue May 21 07:46:45 2024 from 27.59.111.80
ubuntu@ip-172-31-26-137:~$ sudo su
root@ip-172-31-26-137:/home/ubuntu# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
mynewhttpd    latest   f315bf839a1f   15 minutes ago 169MB
httpd         latest   356125da0595   6 weeks ago    147MB
root@ip-172-31-26-137:/home/ubuntu# docker tag mynewhttpd thenameis/image-repo_httpd:V1.0
root@ip-172-31-26-137:/home/ubuntu# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
mynewhttpd    latest   f315bf839a1f   16 minutes ago 169MB
thenameis/image-repo_httpd  V1.0     f315bf839a1f   16 minutes ago 169MB
httpd         latest   356125da0595   6 weeks ago    147MB
root@ip-172-31-26-137:/home/ubuntu# docker push thenameis/image-repo_httpd:V1.0
The push refers to repository [docker.io/thenameis/image-repo_httpd]
588a6279c0ba: Pushed
3f5306ccc4fdb: Mounted from library/httpd
2e035843b69b: Mounted from library/httpd
d138aa37a32d: Mounted from library/httpd
5f70bf18a086: Mounted from library/httpd
4cc26374e331: Mounted from library/httpd
5d4427064ecc: Mounted from library/httpd
V1.0: digest: sha256:2edee8d2a8951e9271df7c997e4f06c4923fcc27a3076154d2abead28881485d8 size: 1784
root@ip-172-31-26-137:/home/ubuntu#
```

B. To push this image to ECR:

Create a Public Repository in AWS

- Repository Name: httpd-image-repo

Install AWS CLI with using following command:

- snap install aws-cli --classic

```
root@ip-172-31-26-137:/home/ubuntu
Fetched 163 kB in 1s (265 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
4 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-26-137:/home/ubuntu# apt install aws cli
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package aws
E: Unable to locate package cli
root@ip-172-31-26-137:/home/ubuntu# aws configure
Command 'aws' not found, but can be installed with:
snap install aws-cli # version 1.15.58, or
apt install awscli # version 2.14.6-1
See 'snap info aws-cli' for additional versions.
root@ip-172-31-26-137:/home/ubuntu# snap install aws-cli
error: This revision of snap "aws-cli" was published using classic confinement and thus may perform
arbitrary system changes outside of the security sandbox that snaps are usually confined to,
which may put your system at risk.

If you understand and want to proceed repeat the command including --classic.
root@ip-172-31-26-137:/home/ubuntu# snap install aws-cli --classic
aws-cli (v2/stable) 2.15.54 from Amazon Web Services (aws) installed
root@ip-172-31-26-137:/home/ubuntu#
```

To configure AWS: aws configure

- Access key ID:
- Secret access key:
- Default region Name: us-east-1
- Default output format: json

From view push commands on repository:

- `aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws/s1v6e7h4`

```
root@ip-172-31-26-137: /home/ubuntu
Command 'aws' not found, but can be installed with:
snap install aws-cli # version 1.15.58, or
apt install awscli # version 2.14.6-1
See 'snap info aws-cli' for additional versions.
root@ip-172-31-26-137:/home/ubuntu# snap install aws-cli
error: This revision of snap "aws-cli" was published using classic confinement and thus may perform
arbitrary system changes outside of the security sandbox that snaps are usually confined to,
which may put your system at risk.

If you understand and want to proceed repeat the command including --classic.
root@ip-172-31-26-137:/home/ubuntu# snap install aws-cli --classic
aws-cli (v2/stable) 2.15.54 from Amazon Web Services (aws) installed
root@ip-172-31-26-137:/home/ubuntu# aws configure
AWS Access Key ID [None]: AKIAU6GDU6KSZQOKRB5J
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
root@ip-172-31-26-137:/home/ubuntu# aws ecr-public get-login-password --region us-east-1 | docker login
--username AWS --password-stdin public.ecr.aws/s1v6e7h4
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
root@ip-172-31-26-137:/home/ubuntu#
```

To give a tag to the image who want to push:

➤ `docker tag mynewhttpd public.ecr.aws/s1v6e7h4/httpd-image-repo:V1.1`

```
root@ip-172-31-26-137: /home/ubuntu
root@ip-172-31-26-137:/home/ubuntu# aws configure
AWS Access Key ID [None]: AKIAU6GDU6KSZQOKRB5J
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
root@ip-172-31-26-137:/home/ubuntu# aws ecr-public get-login-password --region us-east-1 | docker login
--username AWS --password-stdin public.ecr.aws/s1v6e7h4
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
root@ip-172-31-26-137:/home/ubuntu# docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
mynewhttpd          latest       f315bf839a1f      8 hours ago     169MB
thenameis/image-repo_httpd  V1.0        f315bf839a1f      8 hours ago     169MB
httpd                latest       356125da0595      6 weeks ago     147MB
root@ip-172-31-26-137:/home/ubuntu# docker tag mynewhttpd public.ecr.aws/s1v6e7h4/httpd-image-repo:V1.1
root@ip-172-31-26-137:/home/ubuntu# docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
mynewhttpd          latest       f315bf839a1f      8 hours ago     169MB
thenameis/image-repo_httpd  V1.0        f315bf839a1f      8 hours ago     169MB
public.ecr.aws/s1v6e7h4/httpd-image-repo  V1.1        f315bf839a1f      8 hours ago     169MB
httpd                latest       356125da0595      6 weeks ago     147MB
root@ip-172-31-26-137:/home/ubuntu#
```

To Push the image to ECR use the following command:

➤ `docker push public.ecr.aws/s1v6e7h4/httpd-image-repo:V1.1`

```
root@ip-172-31-26-137:/home/ubuntu# docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
mynewhttpd          latest      f315bf839a1f  8 hours ago   169MB
thenameis/image-repo_httpd  V1.0       f315bf839a1f  8 hours ago   169MB
httpd                latest      356125da0595  6 weeks ago   147MB
root@ip-172-31-26-137:/home/ubuntu# docker tag mynewhttpd public.ecr.aws/s1v6e7h4/httpd-image-repo:V1.1
root@ip-172-31-26-137:/home/ubuntu# docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
mynewhttpd          latest      f315bf839a1f  8 hours ago   169MB
thenameis/image-repo_httpd  V1.0       f315bf839a1f  8 hours ago   169MB
public.ecr.aws/s1v6e7h4/httpd-image-repo  V1.1       f315bf839a1f  8 hours ago   169MB
httpd                latest      356125da0595  6 weeks ago   147MB
root@ip-172-31-26-137:/home/ubuntu# docker push public.ecr.aws/s1v6e7h4/httpd-image-repo:V1.1
The push refers to repository [public.ecr.aws/s1v6e7h4/httpd-image-repo]
588a6279c0ba: Pushed
3f5306cc4fdb: Pushed
2e035843b69b: Pushed
d138aa37a32d: Pushed
5f70bf18a086: Pushed
4cc26374e331: Pushed
5d4427064ecc: Pushed
V1.1: digest: sha256:2edee8d2a8951e9271df7c997e4f06c4923fcc27a3076154d2abea28881485d8 size: 1784
root@ip-172-31-26-137:/home/ubuntu#
```

