

7. Deploy a web application using docker

- Name: Om Deore
- PRN: 22110318
- Div: TY(A)
- RollNo: 321022
- github: [All Assignments](#)

Theory

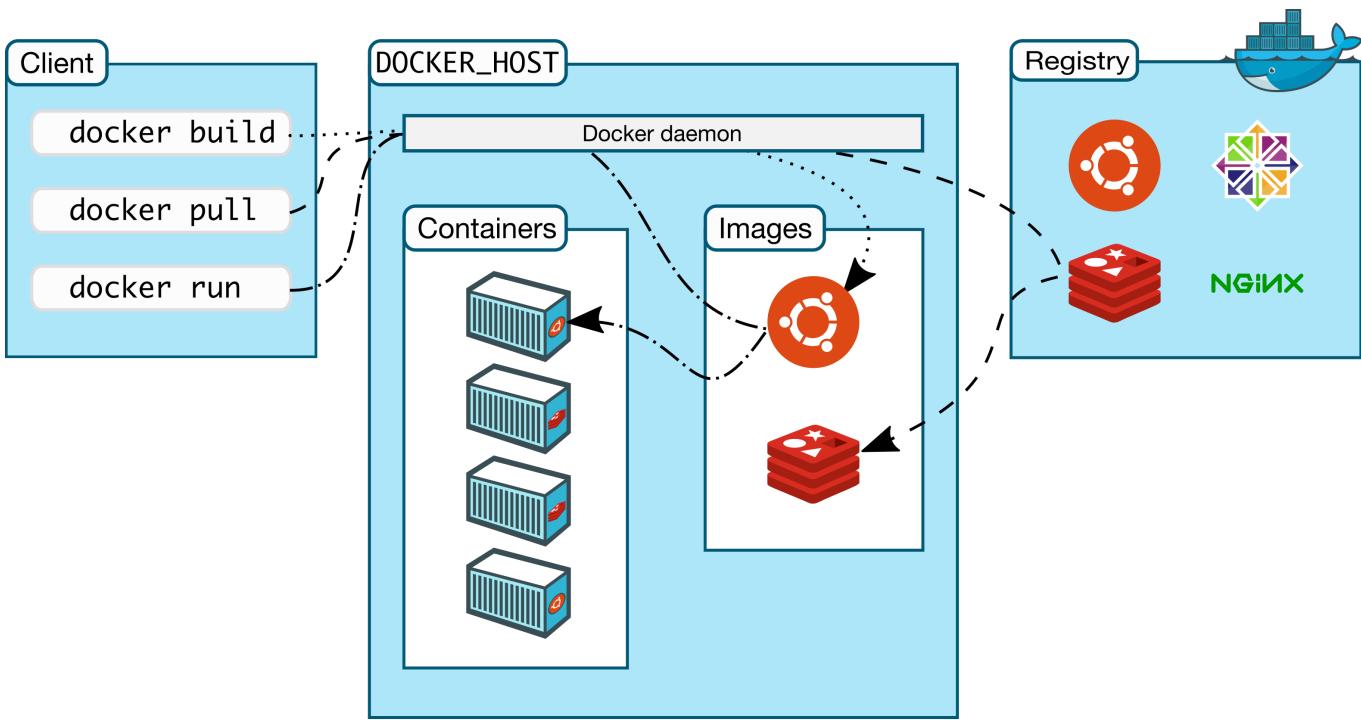
1) What is Docker?

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code, you can significantly reduce the delay between writing code and running it in production.

2) Docker Architecture

Docker architecture consists of three main components:

- Docker Engine : The runtime that runs and manages containers on a host system.
- Docker Images : Lightweight, standalone, executable packages that contain everything needed to run a piece of software, including the code, runtime, libraries, and dependencies.
- Docker Registries : Repositories for Docker images, allowing users to store and share container images.



3) Difference between Docker and VM

1. Virtualisation : From our understanding thus far, both virtual machines and Docker containers provide isolated environments to run applications. The key difference between the two is in *how* they facilitate this isolation. Recall that a VM boots up its own guest OS. Therefore, it virtualises both the operating system kernel and the application layer. A Docker container virtualises *only* the application layer, and runs on top of the host operating system
2. Compatibility : A virtual machine uses its own operating system and is *independent* of the host operating system that it's running on. Therefore, a VM is compatible with all operating systems. A Docker container, on the other hand, is compatible with *any* Linux distribution. You may run into some problems running Docker on a Windows machine or an older Mac.
3. Size : A Docker image is lightweight and is typically in the order of kilobytes. A Docker image denotes the artifact containing the application, its associated dependencies, and configuration. A running instance of the Docker image is called a container. A VM instance can be as large as a few gigabytes or even terabytes
4. Performance : In terms of performance, Docker containers provide near-native performance. Because they are lightweight, you can start them in a few milliseconds. Starting a VM is equivalent to setting up a standalone machine inside your computer. It can take as long as a few minutes to start a VM instance.
5. Security : Docker containers run on top of the host operating system. Therefore, if the host OS is susceptible to security vulnerabilities, so are the Docker containers. Virtual machines, on the other hand, boot up their own operating system, and are more secure.

Recall: each virtual machine is a fully blown machine running inside another. If you have stringent security constraints to be met for sensitive applications, you should consider using a virtual machine instead.

4) Docker Commands

- Build an Image from a Dockerfile

```
docker build -t <image_name>
```

- List local images

```
docker images
```

- Create and run a container from an image, with a custom name:

```
docker run --name <container_name> <image_name>
```

- Run a container with and publish a container's port(s) to the host.

```
docker run -p <host_port>:<container_port> <image_name>
```

- Run a container in the background

```
docker run -d <image_name>
```

- Start or stop an existing container:

```
docker start|stop <container_name> (or <container-id>)
```

- Login into Docker

```
docker login -u <username>
```

- Publish an image to Docker Hub

```
docker push <username>/<image_name>
```

- Search Hub for an image

```
docker search <image_name>
```

- Pull an image from a Docker Hub

```
docker pull <image_name>
```

5) What is Dockerfile

A Dockerfile is a text document that contains instructions for building a Docker image. It specifies the base image to use, any additional dependencies to install, and commands to run when the container starts. Docker can build images automatically by reading the instructions from a Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. This page describes the commands you can use in a Dockerfile.

6) Docker-compose and Docker-swarm

1. Docker-compose - A tool for defining and running multi- container Docker applications. It uses a YAML file to configure the application's services, networks, and volumes, making it easy to manage complex applications.
2. Docker-swarm - A native clustering and orchestration tool for Docker. It allows users to create and manage a cluster of Docker nodes, providing features like service discovery, load balancing, and rolling updates for containerized applications.

Implementation:

Step 1) Install Docker

Run following script in bash

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
"deb [arch=$(dpkg --print-architecture) signed-
```

```
by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \
$(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-
plugin docker-compose-plugin
```

After running this script verify your docker install and run demo docker image 'hello-world'

The screenshot shows a terminal window titled 'bash - Konsole' with a dark blue background. At the top, there's a status bar with icons for signal strength, battery, and date/time (23 Apr 2024 20:20). The terminal output is as follows:

```
ubuntu@ip-172-31-14-190:~$ docker --version
Docker version 26.1.0, build 9714adc
ubuntu@ip-172-31-14-190:~$ docker run hello-world
docker: permission denied while trying to connect to the Docker daemon socket at unix
:///var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping": dial unix /v
ar/run/docker.sock: connect: permission denied.
See 'docker run --help'.
ubuntu@ip-172-31-14-190:~$ sudo docker run hello-world

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/

ubuntu@ip-172-31-14-190:~$
```

In the bottom right corner of the terminal window, there is a timestamp: "deblinu" 20:20 23-Apr-24.

Step 2) Getting files into current folder

scp project folder into our server

The screenshot shows a terminal window with two panes. The left pane shows the user transferring files from their local machine to a remote Ubuntu host via SCP:

```
om_d@deblinu:~/College/CC/Assignments/07_assignment :: (main)
> scp -r -i ~/cc_aws_key.pem portfolio ubuntu@13.50.251.111
cp: -r not specified; omitting directory 'portfolio'

om_d@deblinu:~/College/CC/Assignments/07_assignment :: (main)
> scp -r -i ~/cc_aws_key.pem portfolio ubuntu@13.50.251.111

om_d@deblinu:~/College/CC/Assignments/07_assignment :: (main)
> ls
01_docker_installed.png portfolio README.md
om_d@deblinu:~/College/CC/Assignments/07_assignment :: (main)
> rm -r ubuntu@13.50.251.111

om_d@deblinu:~/College/CC/Assignments/07_assignment :: (main)
> scp -i ~/cc_aws_key.pem portfolio ubuntu@13.50.251.111:/home/ubuntu
The authenticity of host '13.50.251.111 (13.50.251.111)' can't be established.
ED25519 key fingerprint is SHA256:bV+ZIicityXDzG7atyMIdFQUEg2xaHa2gnJl0NR0fro.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:9: [hashed name]
  ~/.ssh/known_hosts:12: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '13.50.251.111' (ED25519) to the list of known hosts.
scp: local "portfolio" is not a regular file
scp: failed to upload file portfolio to /home/ubuntu

om_d@deblinu:~/College/CC/Assignments/07_assignment :: (main)
> scp -r -i ~/cc_aws_key.pem portfolio ubuntu@13.50.251.111:/home/ubuntu
style.css                      100%   616    2.4KB/s  00:00
java-seeklogo.svg               100%   17KB   37.2KB/s  00:00
linux-tux-seeklogo.svg          100%   681KB  276.3KB/s  00:02
phong-opengl.png                100% 1069KB  563.4KB/s  00:01
unity-seeklogo.svg              100%   5192   26.3KB/s  00:00
opengl-seeklogo.eps             100%   135KB  742.9KB/s  00:00
opengl-seeklogo.svg              100%   86KB   467.4KB/s  00:00
opengl-seeklogo_1.svg            100%   72KB   177.8KB/s  00:00
Unity-New-2021.svg              100% 2466    15.4KB/s  00:00
c-seeklogo.svg                  100% 2739    16.2KB/s  00:00
index.html                       100% 4024   20.5KB/s  00:00

om_d@deblinu:~/College/CC/Assignments/07_assignment :: (main)
>
[CC] 0:ssh*
```

The right pane shows the user running Docker commands on the remote host:

```
ubuntu@ip-172-31-14-190:~$ docker --version
Docker version 26.1.0, build 9714adc
ubuntu@ip-172-31-14-190:~$ docker run hello-world
docker: permission denied while trying to connect to the Docker daemon socket at unix
:///var/run/docker.sock: Head "http:///%2Var%2Frun%2Fdocker.sock/_ping": dial unix /v
ar/run/docker.sock: connect: permission denied.
See 'docker run --help'.
ubuntu@ip-172-31-14-190:~$ sudo docker run hello-world
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/
```

Additional Docker commands shown in the right pane include creating a directory, moving files, listing contents, and changing directories.

Step 3) Dockerfile

Create a docker file for this application



23 Apr 2024 21:10

~ : bash – Konsole

ubuntu@ip-172-31-14-190:~/docker-demo\$ batcat Dockerfile

	File: Dockerfile
1	FROM ubuntu:latest
2	WORKDIR /var/www/html
3	COPY portfolio/ /var/www/html
4	
5	ENV DEBIAN_FRONTEND=noninteractive
6	
7	EXPOSE 80
8	
9	RUN chown -R www-data:www-data /var/www/html
10	
11	CMD service apache2 start

ubuntu@ip-172-31-14-190:~/docker-demo\$ █

"deblinu" 21:10 23-Apr-24

Step 4) Build the docker image

```
●○○○
~ :bash – Konsole
ubuntu@ip-172-31-14-190:~/docker-demo$ nvim Dockerfile
ubuntu@ip-172-31-14-190:~/docker-demo$ sudo docker build -t portfolio .
[+] Building 18.6s (10/10) FINISHED                                            docker:default
=> [internal] load build definition from Dockerfile                         0.0s
=> => transferring dockerfile: 291B                                         0.0s
=> [internal] load metadata for docker.io/library/ubuntu:latest             0.9s
=> [internal] load .dockerignore                                           0.0s
=> => transferring context: 2B                                             0.0s
=> [1/5] FROM docker.io/library/ubuntu:latest@sha256:1b8d8ff4777f36f19bfe73ee4df61e 0.0s
=> [internal] load build context                                           0.0s
=> => transferring context: 686B                                         0.0s
=> CACHED [2/5] WORKDIR /var/www/html                                      0.0s
=> CACHED [3/5] COPY portfolio/ /var/www/html                            0.0s
=> [4/5] RUN apt-get update && apt-get install -y apache2                15.9s
=> [5/5] RUN chown -R www-data:www-data /var/www/html                      0.6s
=> exporting to image                                                 1.1s
=> => exporting layers                                              1.1s
=> => writing image sha256:f642a016dbf169727955e7648b11ffe2a77e30eebeb37808a27e32d1 0.0s
=> => naming to docker.io/library/portfolio                                0.0s
ubuntu@ip-172-31-14-190:~/docker-demo$ ls -a
. .. Dockerfile portfolio
ubuntu@ip-172-31-14-190:~/docker-demo$ █
```

"deblinu" 21:16 23-Apr-24

Step 5) run Docker image

A screenshot of a Linux terminal window titled "Konsole". The terminal shows the following command and its execution:

```
ubuntu@ip-172-31-14-190:~/docker-demo$ sudo docker run -p 8000:80 portfolio
 * Starting Apache httpd web server apache2
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.4. Set the 'ServerName' directive globally to suppress this message
 *
```

The terminal prompt is "ubuntu@ip-172-31-14-190:~/docker-demo\$". At the bottom right of the terminal window, there is a timestamp: "deblinu" 21:17 23-Apr-24.

Step 6) Login to docker hub and push image

```
●○○○
~ : bash – Konsole
ubuntu@ip-172-31-14-190:~/docker-demo$ sudo docker login -u om101deore
Password:
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
ubuntu@ip-172-31-14-190:~/docker-demo$ sudo docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
portfolio       latest   f642a016dbf1  20 minutes ago  241MB
<none>          <none>  17fa2f084a77  25 minutes ago  82.1MB
nginx           latest   2ac752d7aeb1  6 days ago    188MB
hello-world     latest   d2c94e258dcb  11 months ago  13.3kB
ubuntu@ip-172-31-14-190:~/docker-demo$ sudo docker tag portfolio:latest om101deore/portfolio:latest
ubuntu@ip-172-31-14-190:~/docker-demo$ sudo docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
portfolio       latest   f642a016dbf1  20 minutes ago  241MB
om101deore/portfolio  latest   f642a016dbf1  20 minutes ago  241MB
<none>          <none>  17fa2f084a77  25 minutes ago  82.1MB
nginx           latest   2ac752d7aeb1  6 days ago    188MB
hello-world     latest   d2c94e258dcb  11 months ago  13.3kB
ubuntu@ip-172-31-14-190:~/docker-demo$ sudo docker push om101deore/portfolio:latest
The push refers to repository [docker.io/om101deore/portfolio]
3d793682cdf0: Pushed
16ce2b35c3e9: Pushed
957891dcbb8d: Pushed
7af4fc59b326: Pushed
e0a9f5911802: Mounted from library/ubuntu
latest: digest: sha256:63b505e0d96ae3336bed5f4fde7c78db020c18ea82349a88756202ece4468bce size: 1370
ubuntu@ip-172-31-14-190:~/docker-demo$ █
"deblinu" 21:39 23-Apr-24
```

Step 7) Image on Docker hub

This way we have created our docker image and hosted it on docker hub. Now anyone can use

this image

The screenshot shows a browser window with the URL https://hub.docker.com/u/om101deore. The Docker Hub header includes the logo, navigation links for Explore, Repositories, and Organizations, a search bar with placeholder 'Search Docker Hub' and a 'ctrl+K' keyboard shortcut, and a notifications icon showing 0 notifications. The user profile for 'Om Deore' is displayed, featuring a blue fingerprint icon, the name 'Om Deore' with an 'Edit profile' link, and status 'Community User' joined on April 23, 2024. Below the profile, there are tabs for 'Repositories', 'Starred', and 'Contributed', with 'Repositories' being the active tab. A message indicates 'Displaying 1 to 1 repositories'. A single repository card for 'om101deore/portfolio' is shown, created by 'om101deore' and updated 'a few seconds ago'. It includes a small image thumbnail, the repository name, the creator's name, the update timestamp, and two small circular icons for '0' stars and '0' forks.