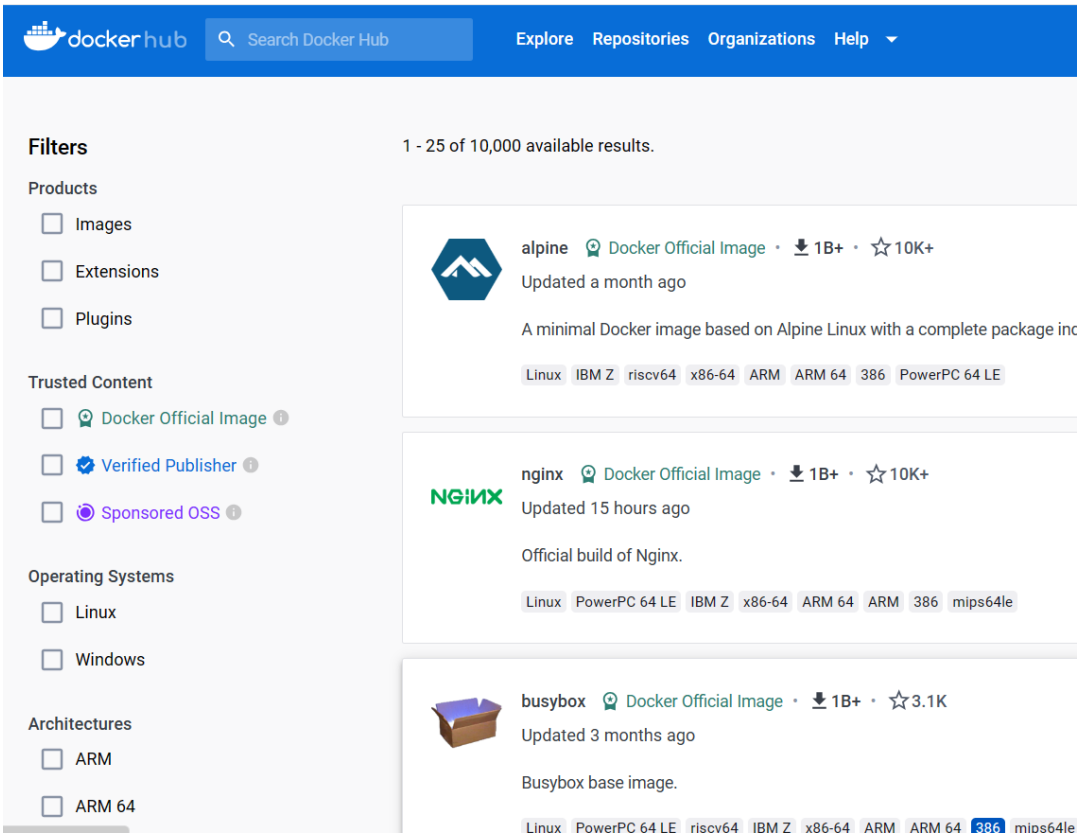
AKSHIT MITTAL
10/26/23

DOCKER – Dockerfile, Docker- compose

- ➔ Docker is the containerization tool is used to manage the containerized package of the application or the software.
- ➔ Images- The Read only files containing the set of rules to install all the packages of the applications, which when run, creates the container.
- ➔ Containers- The package of the software as a whole, including its dependencies, users information, IP addresses, Networks configurations and more, which are used to create and run the application on any infrastructure and environment successfully.
- ➔ Docker File – Image of the Docker is created via docker files. It contains the set of instructions to run the container.
- ➔ Docker Hub – A Public registry containing the set of all the images. [Explore Docker's Container Image Repository | Docker Hub]
- ➔ In some organizations, the docker is used, replacing the need of CM tool like ansible

→ Install Docker → Go to docker hub → pull the image you want to be present on your system



```
MINGW64:/c/Users/admin
hello-world   latest    9c7a54a9a43c   5 months ago   13.3kB

admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
3f4d90098f5b: Pulling fs layer
3f4d90098f5b: Verifying Checksum
3f4d90098f5b: Download complete
3f4d90098f5b: Pull complete
Digest: sha256:3fbc632167424a6d997e74f52b878d7cc478225cffac6bc977eedfe51c7f
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scou
```

```
admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
nginx         latest    593aee2afb64   15 hours ago   187MB
busybox       latest    a416a98b71e2   3 months ago   4.26MB
hello-world   latest    9c7a54a9a43c   5 months ago   13.3kB
```

You can add Images with giving tags, or they will be installed with latest version by default

```
admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker pull ubuntu:20.04
20.04: Pulling from library/ubuntu
96d54c3075c9: Pulling fs layer
96d54c3075c9: Verifying Checksum
96d54c3075c9: Download complete
96d54c3075c9: Pull complete
Digest: sha256:ed4a42283d9943135ed87d4ee34e542f7f5ad9ecf2f
Status: Downloaded newer image for ubuntu:20.04
docker.io/library/ubuntu:20.04
```

Removing the image from host repo (local machine)

```
admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
nginx         latest    593aee2afb64   16 hours ago   187MB
ubuntu        20.04     bf40b7bc7a11   3 weeks ago    72.8MB
busybox       latest    a416a98b71e2   3 months ago   4.26MB
hello-world   latest    9c7a54a9a43c   5 months ago   13.3kB

admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker rmi ubuntu
Error response from daemon: No such image: ubuntu:latest

admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker rmi ubuntu
Error response from daemon: No such image: ubuntu:latest

admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker rmi ubuntu:20.04
Untagged: ubuntu:20.04
Untagged: ubuntu@sha256:ed4a42283d9943135ed87d4ee34e542f7f5ad9ecf2f244870e23122f703f91c2
Deleted: sha256:bf40b7bc7a11b43785755d3c5f23dee03b08e988b327a2f10b22d01d5dc5259d
Deleted: sha256:6c3e7df31590f02f10cb71fc4eb27653e9b428df2e6e5421a455b062bd2e39f9

admin@DESKTOP-9UJRCUE MINGW64 ~
```

Please note: You can delete all the images available in your local machine by the command # docker system prune --all

→ Run the image to create the container

```
admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker run ubuntu
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
aece8493d397: Pulling fs layer
aece8493d397: Verifying Checksum
aece8493d397: Download complete
aece8493d397: Pull complete
Digest: sha256:2b7412e6465c3c7fc5bb21d3e6f1917c167358449fecac8176c6e496e5c1f
Status: Downloaded newer image for ubuntu:latest

admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
nginx         latest    593aee2afb64   16 hours ago   187MB
ubuntu        latest    e4c58958181a   2 weeks ago    77.8MB
busybox       latest    a416a98b71e2   3 months ago   4.26MB
hello-world   latest    9c7a54a9a43c   5 months ago   13.3kB
```

Running the container without installing image, automatically installs the image on the system before running.

→ Listing the created containers

```
admin@DESKTOP-9UJRCUE MINGW64 ~
$ docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED             STATUS
                    PORTS                   NAMES
02ffff9e4d1c   ubuntu        "/bin/bash"              2 minutes ago       Exited (0
) 2 minutes ago                             brave_lewin
3788ebd828e2   hello-world   "/hello"                 About an hour ago   Exited (0
) About an hour ago                         focused_bassi
ddd6264c98f3   nginx         "/docker-entrypoint.…"   5 hours ago         Up 5 hour
s                   0.0.0.0:80->80/tcp      elastic_shamir
```

→ docker run –it “img name” – Runs the image with attaching the user to the terminal of the container. Here –it means Interactive Terminal. Exit command will helps exiting the terminal

```
PS C:\Users\admin> docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
nginx         latest    593aee2afb64   16 hours ago   187MB
ubuntu        latest    e4c58958181a   2 weeks ago    77.8MB
busybox       latest    a416a98b71e2   3 months ago   4.26MB
hello-world   latest    9c7a54a9a43c   5 months ago   13.3kB
PS C:\Users\admin> docker run -it ubuntu
root@bae15a7c3361:/# exit
exit
```

→ You can assign the names to the containers. || To run the container without attaching user to its terminal is possible though detached –d mode, which helps container run in the background.

```
PS C:\Users\admin> docker run --name MyContainerNginx -d nginx
c114faede94cf7805b0bfd4e281151b4d5b99f7dc7d2c44340fecbdce052ac37
PS C:\Users\admin> docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS
NAMES
c114faede94c   nginx         "/docker-entrypoint.…"   9 seconds ago    Up 8 seconds
MyContainerNginx
ca3c25715325   nginx         "/docker-entrypoint.…"   14 minutes ago   Exited (0) 8
wizardly_lamarr
bae15a7c3361   ubuntu        "/bin/bash"              15 minutes ago   Exited (0) 14
brave_cannon
02ffff9e4d1c   ubuntu        "/bin/bash"              25 minutes ago   Exited (0) 25
brave_lewin
3788ebd828e2   hello-world   "/hello"                 2 hours ago      Exited (0) 2
focused_bassi
ddd6264c98f3   nginx         "/docker-entrypoint.…"   5 hours ago      Up 5 hours
elastic_shamir
PS C:\Users\admin>
```

→ inspect command gives all the information about the running container

```
elastic_shamir
PS C:\Users\admin> docker inspect MyContainerNginx
[
    {
        "Id": "c114faede94cf7805b0bfd4e281151b4d5b99f7dc7d2c44340fecbdc
        "Created": "2023-10-25T17:36:13.342509705Z",
        "Path": "/docker-entrypoint.sh",
        "Args": [
            "nginx",
            "-g",
            "daemon off;"
        ],
        "State": {
            "Status": "running",
            "Running": true,
            "Paused": false,
            "Restarting": false,
            "OOMKilled": false,
            "Dead": false,
            "Pid": 23514,
            "ExitCode": 0,
```

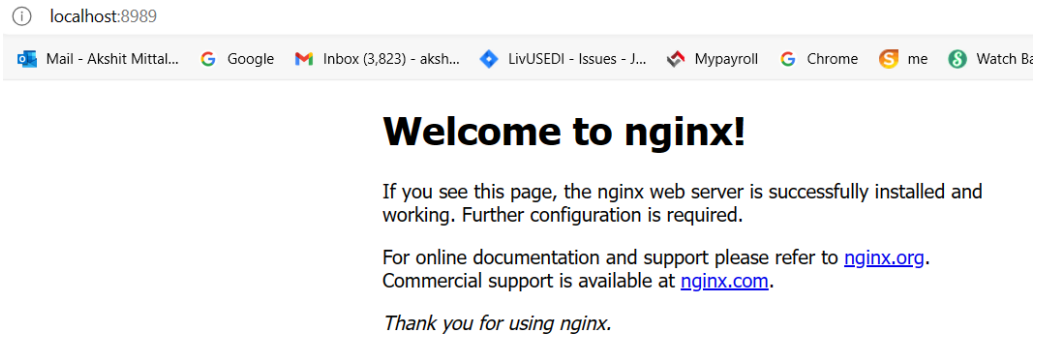
→ Port Mapping

This concepts allow user to access the container from the browser. This is only possible while creating the new container running stage and not on the already created and

running containers. Only available for web images or db images which can be access via web. We can manually map the port numbers for containers

```
PS C:\Users\admin> docker run -d --name webContainer -p 8989:80 nginx
020c553759b42fe8b38451371306295e4f1440acd6c9e68aa56d474e8c8d0c44
PS C:\Users\admin> docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS                      PORTS                   NAMES
020c553759b4   nginx         "/docker-entrypoint.…"   12 seconds ago   Up 11 seconds               0.0.0.0:8989->80/tcp    webContainer
c114faede94c   nginx         "/docker-entrypoint.…"   10 minutes ago   Up 10 minutes               80/tcp                  MyContainerNginx
ca3c25715325   nginx         "/docker-entrypoint.…"   25 minutes ago   Exited (0) 18 minutes ago                           wizardly_lamarr
bae15a7c3361   ubuntu        "/bin/bash"              25 minutes ago   Exited (0) 25 minutes ago                           brave_cannon
02ffff9e4d1c   ubuntu        "/bin/bash"              36 minutes ago   Exited (0) 36 minutes ago                           brave_lewin
3788ebd828e2   hello-world   "/hello"                 2 hours ago      Exited (0) 2 hours ago                              focused_bassi
ddd6264c98f3   nginx         "/docker-entrypoint.…"   5 hours ago      Up 5 hours                  0.0.0.0:80->80/tcp      elastic_shamir
PS C:\Users\admin> docker port webContainer
80/tcp -> 0.0.0.0:8989
PS C:\Users\admin> docker -d --name web2 -P httpd
```

We can see that port numbers has been mapped successfully and containers will be available on these port numbers.



→ We can let docker assign any random open port number to the container through –P

```
PS C:\Users\admin> docker run -d --name h1 -P nginx
f355a096210ae8ae353ef9dfc9c28b09ec8a02336e6e674121d26275e7ccb468
PS C:\Users\admin> docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS                      PORTS                   NAMES
f355a096210a   nginx         "/docker-entrypoint.…"   6 seconds ago    Up 5 seconds                0.0.0.0:32769->80/tcp   h1
c83386b2e588   httpd         "httpd-foreground"       5 minutes ago    Up 5 minutes                0.0.0.0:32768->80/tcp   web2
```

→ docker rm -f $(docker ps -aq) → This command deletes all the containers

```
PS C:\Users\admin> docker rm -f $(docker ps -aq)
f355a096210a
c83386b2e588
020c553759b4
c114faede94c
ca3c25715325
bae15a7c3361
02ffff9e4d1c
3788ebd828e2
ddd6264c98f3
PS C:\Users\admin> docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
PS C:\Users\admin>
```

→ Docker Volume: The file where data of containers directory can be preserved. Its like the backup folder of the container. If any file on the container terminal deletes, it will still be preserved In the volume folder

```
PS C:\Users\admin> docker volume create myvol
myvol
PS C:\Users\admin> docker volume inspect myvol
[
    {
        "CreatedAt": "2023-10-25T18:04:12Z",
        "Driver": "local",
        "Labels": null,
        "Mountpoint": "/var/lib/docker/volumes/myvol/_data",
        "Name": "myvol",
        "Options": null,
        "Scope": "local"
    }
]
PS C:\Users\admin>
```

→ You can only Map the volume to the container at runtime (during creation of the container)

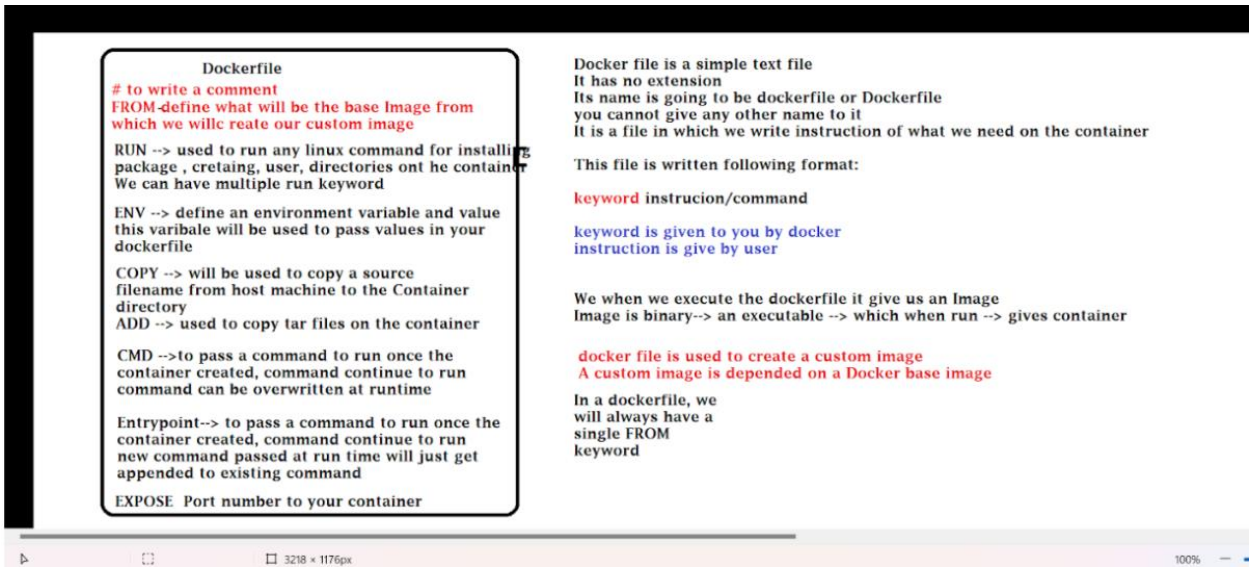
```
PS C:\Users\admin> docker run -it --name u1 -v myvol:/tmp ubuntu
root@4b08785fa21e:/# cd /tmp
root@4b08785fa21e:/tmp# touch file1 files2
root@4b08785fa21e:/tmp# cd /tmp
```

Now these 2 files will be visible in the volume folder. Any files added there will also be visible here.

TO come out of container terminal , without stopping the container– ctrl + pq
TO stop the running container and come out of the terminal - exit

Docker Files

➔ Create your own custom image



→ Make a directory MyDockerFile on which you will be writing your docker files. First,

we will create a html.txt file , which will be copied in the container

index - Notepad

File Edit Format View Help

```
<h1> file from docker </h1>
```

→ Then, we will create a docker file with name Dockerfile without any extension.

Dockerfile - Notepad

File Edit Format View Help

```
FROM ubuntu
#ubuntu will be our base image
ENV pkg1 nginx
RUN apt-get update
RUN apt-get install $pkg1 -y
COPY index.html /var/www/html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

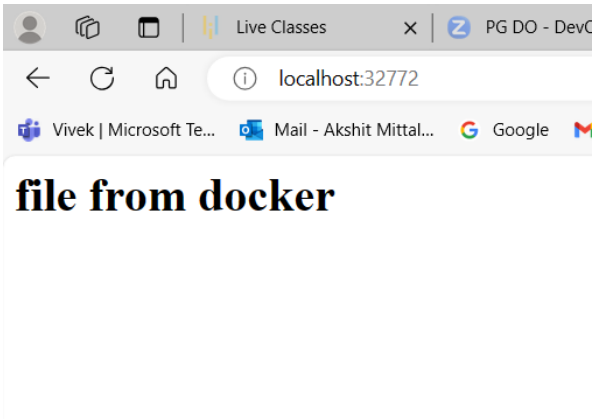
Docker file is always created based on some base image. Here the base image will be Ubuntu. We will define package nginx in Envelope section. CMD section defines the nginx to run in the foreground and generates the output log in the console window.

```
PS C:\Users\admin\MyDockerFile> PS C:\Users\admin\MyDockerFile> docker build -t mynginx .
>> [+] Building 0.1s (2/2) FINISHED                                  docker:default
>>  => [internal] load .dockerignore                                            0.0s
>>  => => transferring context: 2B                                              0.0s
>>  => [internal] load build definition from Dockerfile                         0.0s
>>  => => transferring dockerfile: 2B                                           0.0s
>> ERROR: failed to solve: failed to read dockerfile: open /var/lib/docker/tmp/buildkit-mount3596536272/Doc
>> PS C:\Users\admin\MyDockerFile> ls
>>
>>
>>     Directory: C:\Users\admin\MyDockerFile
>>
>>
>> Mode                 LastWriteTime         Length Name
>> ----                 -------------         ------ ----
>> -a----        10/26/2023  12:29 AM            187 Dockerfile.txt
>> -a----        10/26/2023  12:18 AM             29 index.html
>>
[+] Building 75.0s (6/8)                                            docker:default
 => [internal] load .dockerignore                                            0.0s
 => => transferring context: 2B                                              0.0s
 => [internal] load build definition from Dockerfile                         0.0s
 => => transferring dockerfile: 226B                                         0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest             0.0s
 => [internal] load build context                                            0.1s
 => => transferring context: 66B                                             0.0s
 => [1/4] FROM docker.io/library/ubuntu                                      0.0s
 => [2/4] RUN apt-get update                                                31.7s
 => [3/4] RUN apt-get install nginx -y                                      43.2s
 => => # Get:20 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libx11-
 => => # data all 2:1.7.5-1ubuntu0.3 [120 kB]
 => => # Get:21 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libx11-
 => => # 6 amd64 2:1.7.5-1ubuntu0.3 [667 kB]
 => => # Get:22 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-dejavu-co
 => => # re all 2.37-2build1 [1041 kB]
[+] Building 136.8s (9/9) FINISHED                                  docker:default
 => [internal] load .dockerignore                                            0.0s
 => => transferring context: 2B                                              0.0s
 => [internal] load build definition from Dockerfile                         0.0s
 => => transferring dockerfile: 226B                                         0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest             0.0s
 => [internal] load build context                                            0.1s
```

```
PS C:\Users\admin\MyDockerFile> notepad .\index.html
PS C:\Users\admin\MyDockerFile> docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
mynginx       latest    fa31ebd58663   16 minutes ago   179MB
nginx         latest    593aee2afb64   18 hours ago     187MB
httpd         latest    75a48b16cd56   6 days ago       168MB
ubuntu        latest    e4c58958181a   2 weeks ago      77.8MB
busybox       latest    a416a98b71e2   3 months ago     4.26MB
hello-world   latest    9c7a54a9a43c   5 months ago     13.3kB
PS C:\Users\admin\MyDockerFile>
```

Image has been successfully created

```
PS C:\Users\admin\MyDockerFile> docker run -d --name myweb -P mynginx
17c2697c39aa7ac0caa37b5138fe7e11e1225296753d6176cbcdf63f7d970b67
PS C:\Users\admin\MyDockerFile> docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                   NAMES
17c2697c39aa   mynginx   "nginx -g 'daemon of…"   6 seconds ago    Up 6 seconds    0.0.0.0:32772->80/tcp   myweb
```



File has been created successfully on the container port and container is also running successfully

➔ Push the image to public registry form local system.

```
PS C:\Users\admin\MyDockerFile> docker login
Authenticating with existing credentials...
Login Succeeded
PS C:\Users\admin\MyDockerFile> docker push mynginx
Using default tag: latest
The push refers to repository [docker.io/library/mynginx]
51dd07b4b233: Preparing
8d9290192b53: Preparing
58a78ea78e7c: Preparing
256d88da4185: Preparing
denied: requested access to the resource is denied
PS C:\Users\admin\MyDockerFile> docker tag mynginx akshitmittal20/mynginx
```

```
Windows PowerShell
PS C:\Users\admin\MyDockerFile> docker push akshitmittal20/mynginx
```

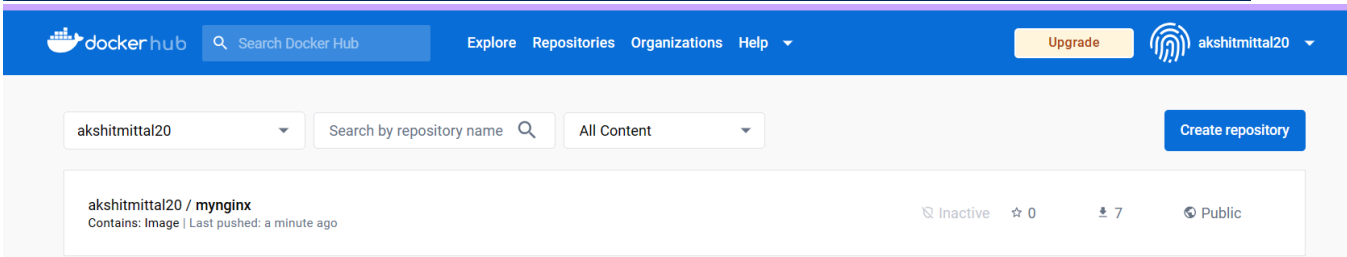


Image will be visible in your repository on docker hub.

Docker Compose Files:

→ If we want to create multiple containers from different images, we will use docker

compose file. It is written in YAML containing the information of containers we want.
→ install docker compose on your system
#curl -SL https://github.com/docker/compose/releases/download/v2.5.0/docker-compose-linux-x86_64 -o /usr/local/bin/docker-compose

#sudo chmod +x /usr/local/bin/docker-compose

#docker-compose –version

→ Create the directory for docker compose files where you will be writing all the docker compose files. Create the docker compose file as following, with name docker-compose

docker-compose - Notepad

File Edit Format View Help

```
version: '3'
services:
 comp-test:
   image: centos:7
   networks:
   - compnet
   command: sleep infinity
   depends_on:
   - comp-db
 comp-db:
   image: redis
   ports:
   - '6379:6379'
   networks:
   - compnet
networks:
 compnet:
```

Version tag with 3 value defines the version of the docker compose. Version 3 defines the compose files with networks, ports, and dependencies
Services defines the tags or data of all the services in the compose file
comp-test is the name of our first container
image: cent-os defines the operating system image, networks – compnet is the name of the private network via which our containers will be connected.
command: sleep infinity defines that containers should be running continuously

Depends on create the dependency on the container of db. If db container is not created, the test container will not start
In the next image, the port mapping is 6379, because the redis image is by default open at this port number

Now we will run this file, in the detached mode.
$ docker-compose up –d → This will run the compose file and create all the images and run the containers with networks mentioned

$ docker-compose ps – it will give information of all the containers status on compose file
$docker-compose exec comp-test /bin/bash → This command will let us to go to the console of the comp-test container, that is cent-os container. From here we will try to ping the db container via port number we mentioned.
To ping the other container within an container, we use the tool netcat (-nc) so we will try to install it prior.

```
 ⠿ Container composefies-comp-test-1  Started                                0.1s
PS C:\Users\admin\MyDockerFile\composefies> docker-compose up -d
[+] Building 0.0s (0/0)                                          docker:default
[+] Running 2/0
 ⠿ Container composefies-comp-db-1    Run...                                  0.0s
 ⠿ Container composefies-comp-test-1  R...                                    0.0s
PS C:\Users\admin\MyDockerFile\composefies> docker-compose ps
NAME                      IMAGE      COMMAND                                SERVICE     CREAT
composefies-comp-db-1     redis      "docker-entrypoint.sh redis-server"    comp-db     14 m
composefies-comp-test-1   centos:7   "sleep infinity"                       comp-test   14 m
PS C:\Users\admin\MyDockerFile\composefies> docker-compose exec compose-test /bin/ba
service "compose-test" is not running
PS C:\Users\admin\MyDockerFile\composefies> docker-compose exec comp-test /bin/bash
Loaded plugins: fastestmirror, ovl
```

```
exec comp-test /bin/bash [root@45e702126420 /]# yum install nc -y
```

```
Complete!
[root@45e702126420 /]# nc comp-db 6379
ping
+PONG
set name akshit
+OK
get name
$6
akshit
^C
[root@45e702126420 /]# ^C
```

The db container is successfully responding

→ Stop all the running container created with docker compose

```
exit
PS C:\Users\admin\MyDockerFile\composefies> docker-compose down
[+] Running 0/1
 - Container composefies-comp-test-1  Stopping                    6.0s
```