Course- BTech/BCA/B.Sc: (B.Tech)
Course Code- ECSE460L

Year- III (VI Sem)
Date- 24-01-2022

Type- Core/Elective (Elective)
Course Name: DevOps Engineering Practices
Semester- Even/Odd (Even)
Batch- B1-B14

A- Type- Lab Assignment (Week 3, Lab 3)

Objective

1) Docker Overview
2) Installation of Docker on Windows/Mac
3) Installation of Virtual Linux platform (if working on windows)
4) Basic Docker Commands
5) Assignment

Docker Overview: **(15)**

Docker, is a DevOps tool that use the containers to design, develop and deploy any application faster as compared to traditional methods.

Requirements for Docker Windows 10: **(5)**

1) Download Docker for desktop from:
   https://docs.docker.com/docker-for-windows/install/
2) System Requirement:
   a) Windows 10 64-bit: Home or Pro 2004 (build 19041) or higher, or Enterprise or Education 1909 (build 18363) or higher.
   b) Enable the WSL 2 feature on Windows
   c) Hardware Prerequisites: 64-bit processor, Atleast 4 GB RAM. virtualization support must be enabled in the BIOS setting.

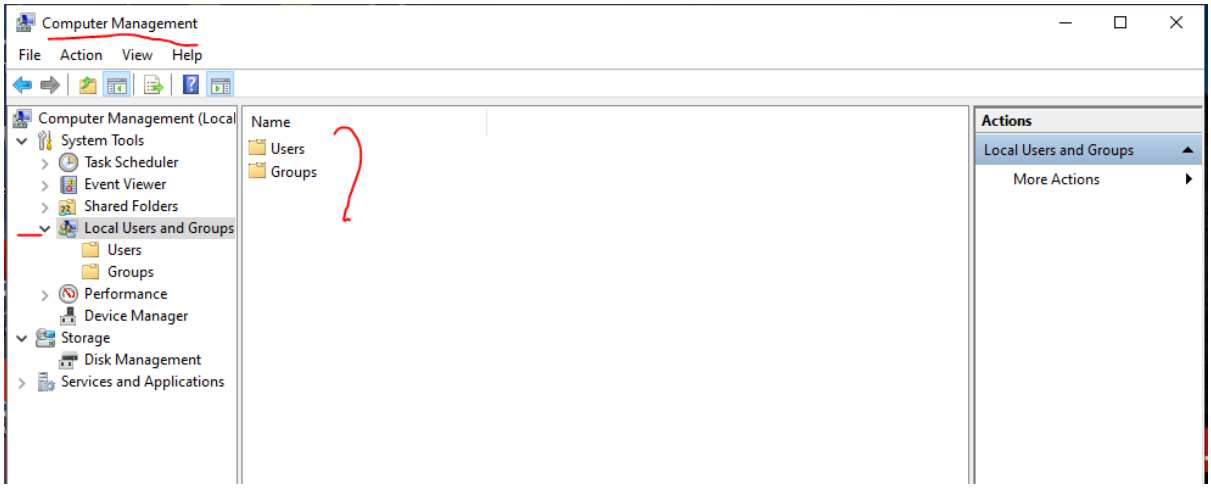
3) Download and install the Linux kernel update package.

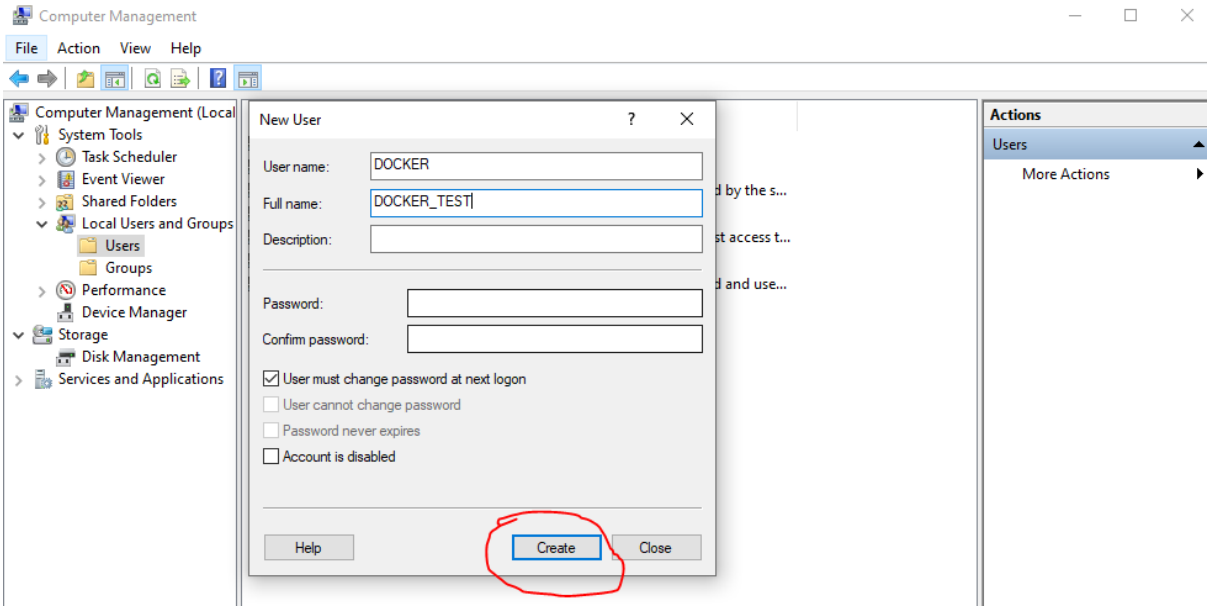
Docker Installation: **(15)**

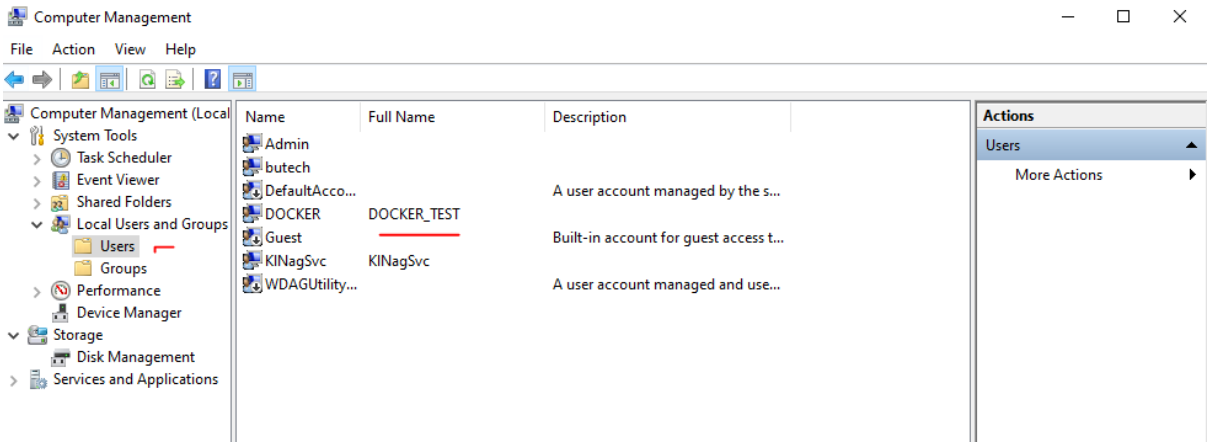
1) Run **Docker Desktop Installer.exe**
2) Ensure that **Enable Hyper-V Windows Features** or the **Install required Windows components for WSL 2** option is selected on the Configuration page.
3) Follow the instruction and keep on proceeding by clicking next

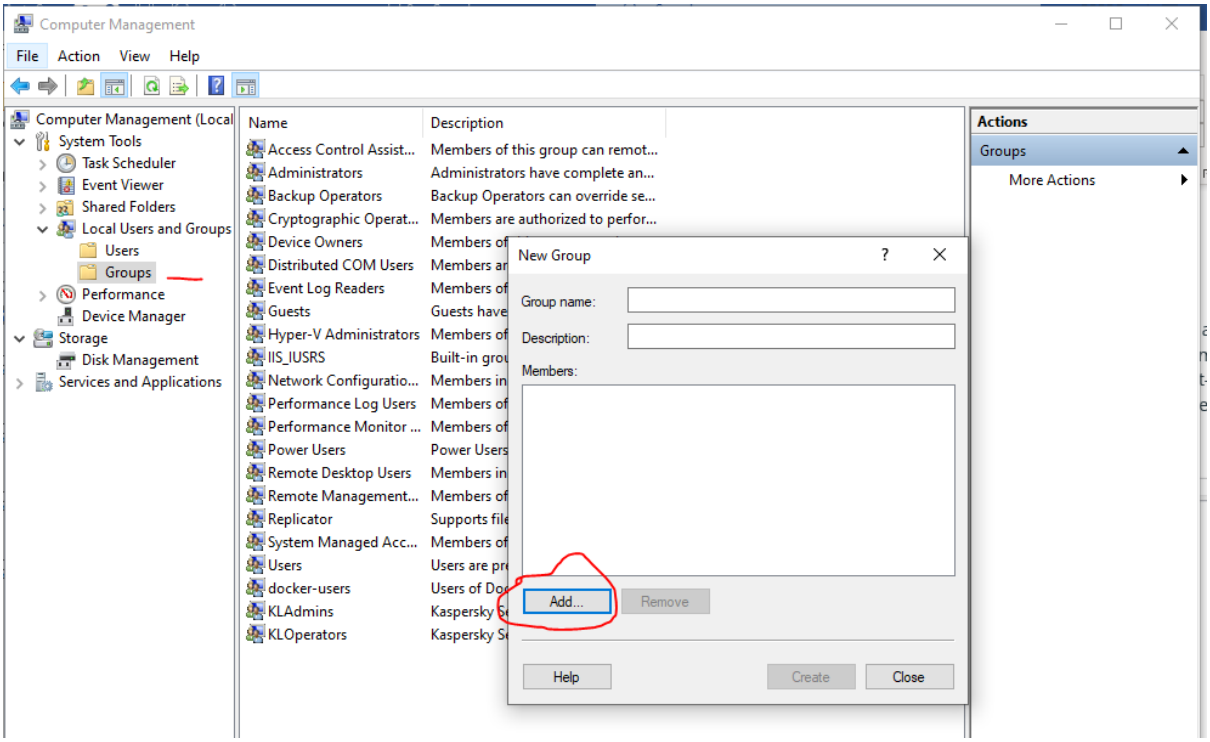
NOTE:

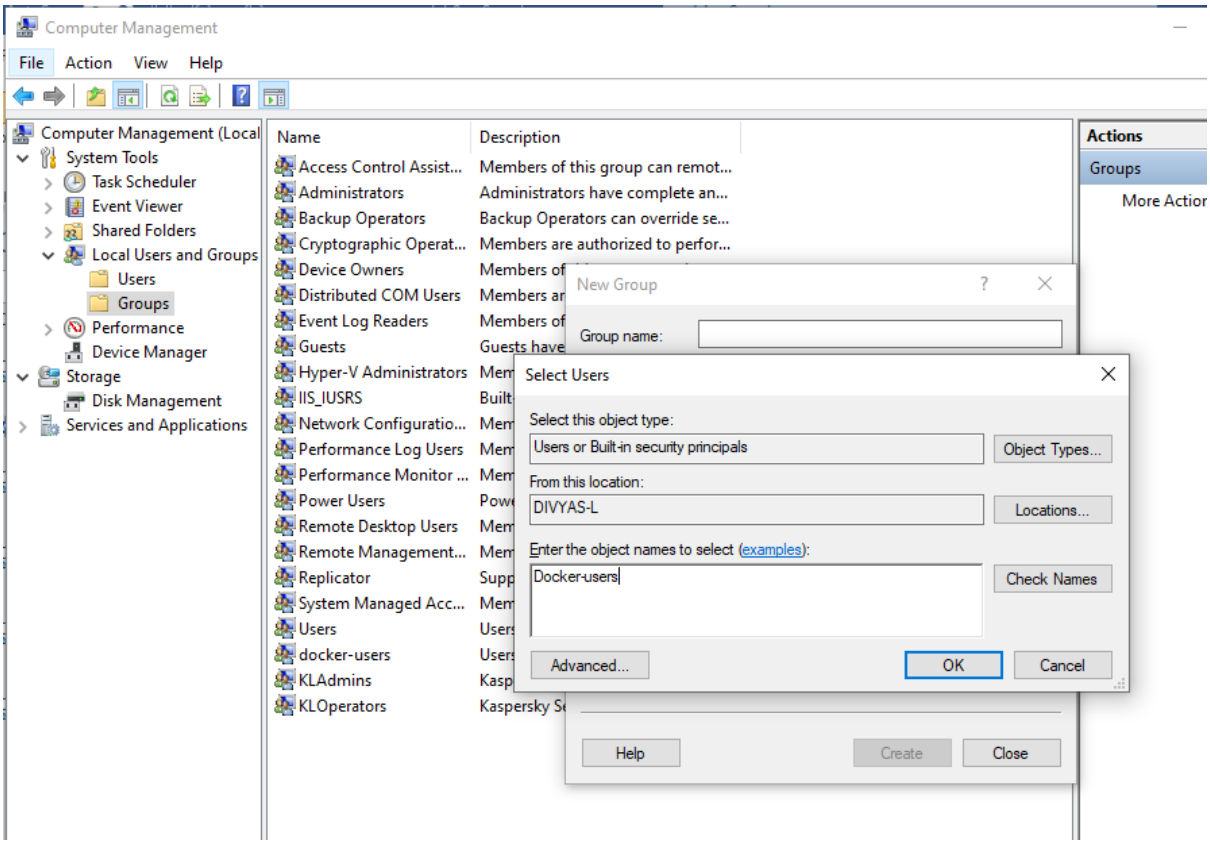
if your admin account is different to your user account, you must add the user to the **docker-users** group. Run **Computer Management** as an administrator and navigate to **Local Users and Groups** > **Groups** > **docker-users**. Right-click to add the user to the group. Log out and log back in for the changes to take effect.

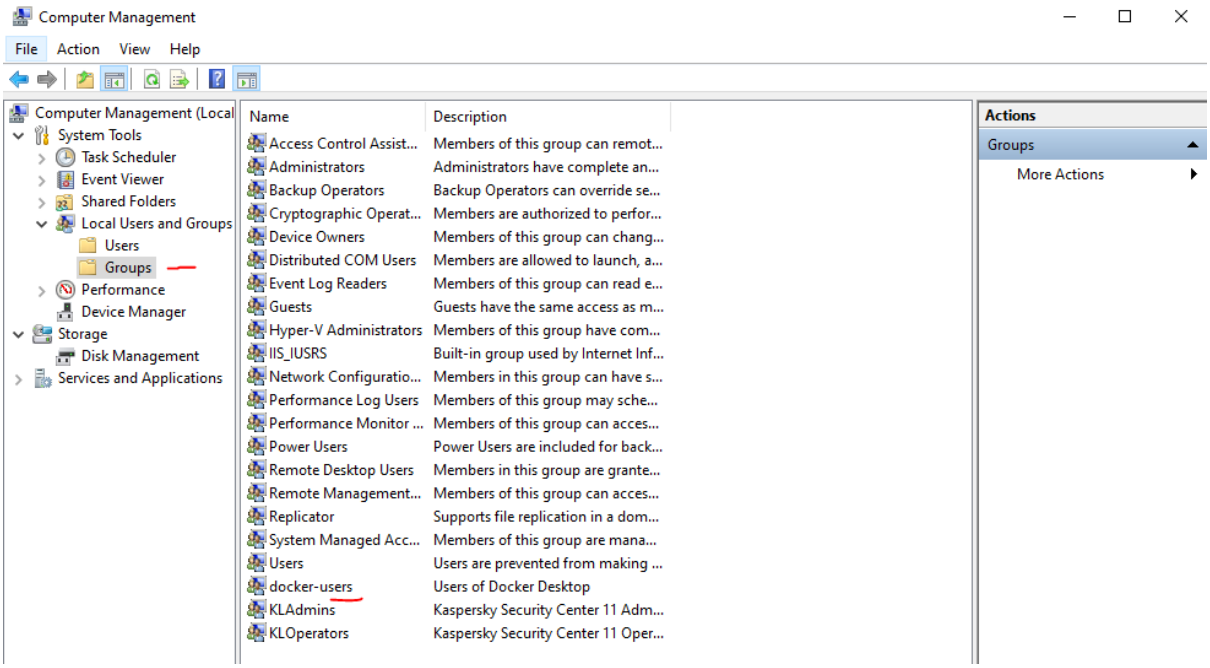



Computer Management
File Action View Help
Computer Management (Local
System Tools
Task Scheduler
Event Viewer
Shared Folders
Local Users and Groups
Users
Groups
Performance
Device Manager
Storage
Disk Management
Services and Applications
New User
User name: DOCKER
Full name: DOCKER_TEST
Description:
Password:
Confirm password:
User must change password at next logon
User cannot change password
Password never expires
Account is disabled
Help
Create
Close
Actions
Users
More Actions


Computer Management
File Action View Help
Computer Management (Local
System Tools
Task Scheduler
Event Viewer
Shared Folders
Local Users and Groups
Users
Groups
Performance
Device Manager
Storage
Disk Management
Services and Applications
Name
Full Name
Description
Admin
butech
DefaultAcco...
A user account managed by the s...
DOCKER
DOCKER_TEST
Guest
Built-in account for guest access t...
KINagSvc
KINagSvc
WDAGUtility...
A user account managed and use...
Actions
Users
More Actions

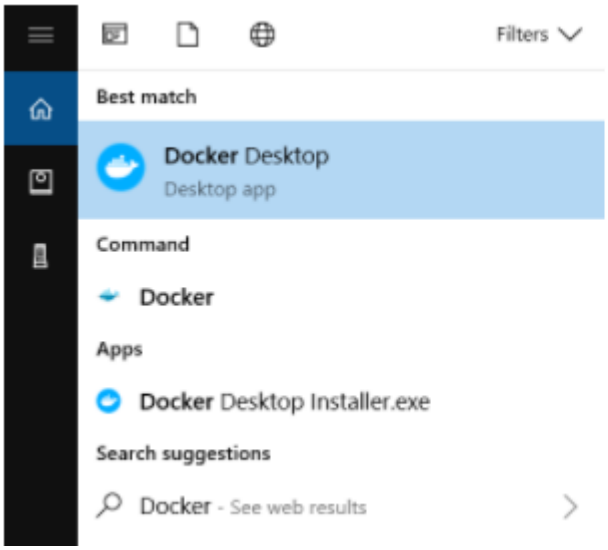

Computer Management
File Action View Help
Computer Management (Local
System Tools
Task Scheduler
Event Viewer
Shared Folders
Local Users and Groups
Users
Groups
Performance
Device Manager
Storage
Disk Management
Services and Applications
Name
Description
Access Control Assist... Members of this group can remot...
Administrators Administrators have complete an...
Backup Operators Backup Operators can override se...
Cryptographic Operat... Members are authorized to perfor...
Device Owners
Distributed COM Users
Event Log Readers
Guests
Hyper-V Administrators
IIS_IUSRS
Network Configuratio...
Performance Log Users
Performance Monitor ...
Power Users
Remote Desktop Users
Remote Management...
Replicator
System Managed Acc...
Users
docker-users
KLAdmins
KLOperators
Actions
Groups
More Actions
New Group
Group name:
Description:
Members:
Add...
Remove
Help
Create
Close


Computer Management
File Action View Help
Computer Management (Local
System Tools
Task Scheduler
Event Viewer
Shared Folders
Local Users and Groups
Users
Groups
Performance
Device Manager
Storage
Disk Management
Services and Applications
Name
Description
Access Control Assist... Members of this group can remot...
Administrators Administrators have complete an...
Backup Operators Backup Operators can override se...
Cryptographic Operat... Members are authorized to perfor...
Actions
Groups
More Actions
New Group
Group name:
Select Users
Select this object type:
Users or Built-in security principals
Object Types...
From this location:
DIVYAS-L
Locations...
Enter the object names to select (examples):
Docker-users
Check Names
Advanced...
OK
Cancel
Help
Create
Close

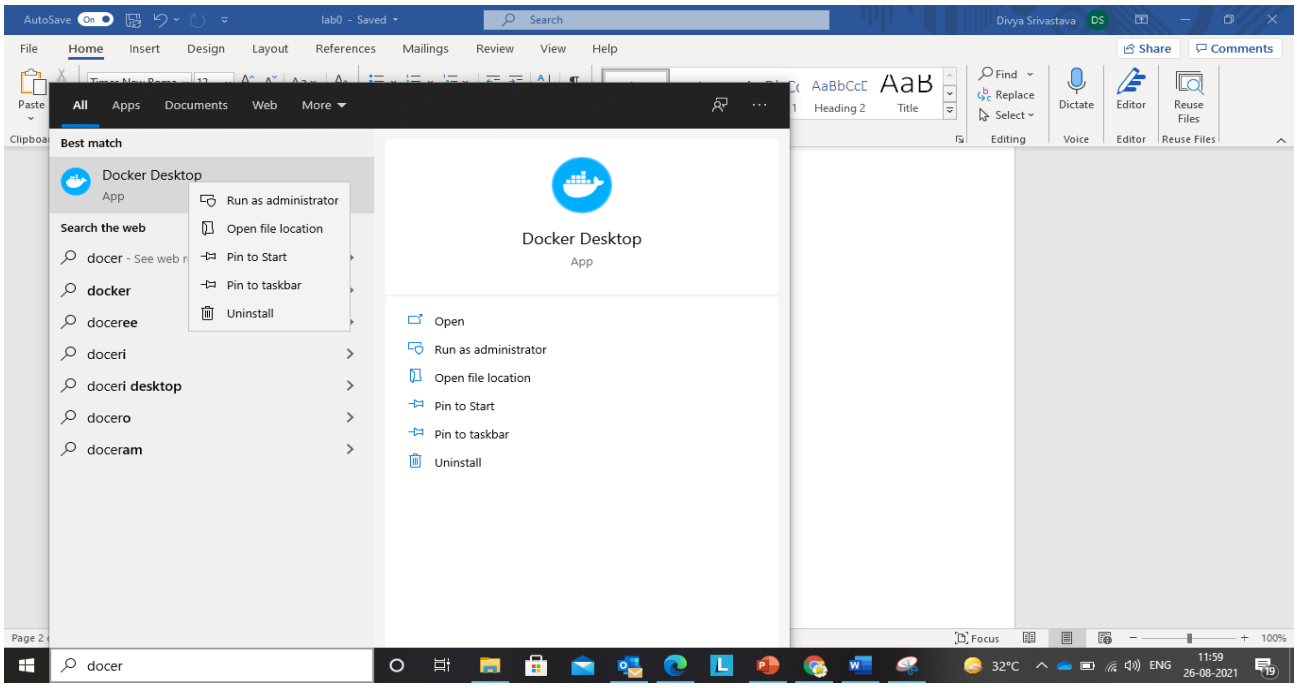


To start working on Docker, you should restart your system, so the changes are reflected.

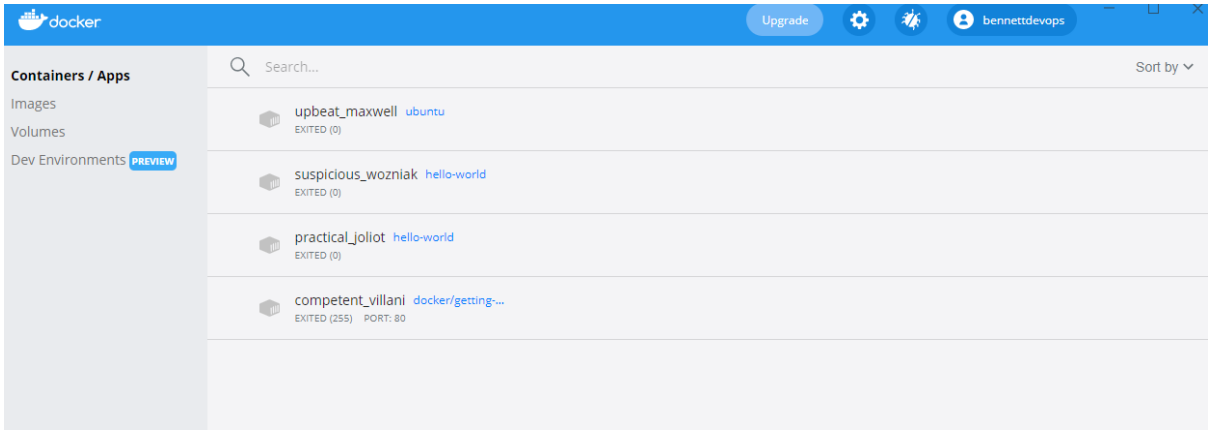
Once Docker is installed, you can see it in your start menu as: (20)



Open Docker as: Run as administrator:



Once the Docker Engine is started the following screen will appear:



Demonstration of some Basic Docker Commands: **(15)**

1) Open command prompt (cmd)
2) Type docker
3) Various docker commands and their usage will be listed down.

```
C:\Users\divya.srivastava>docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:
      --config string      Location of client config files (default
                           "C:\\Users\\divya.srivastava\\.docker")
  -c, --context string     Name of the context to use to connect to the
                           daemon (overrides DOCKER_HOST env var and
                           default context set with "docker context use")
  -D, --debug              Enable debug mode
  -H, --host list          Daemon socket(s) to connect to
  -l, --log-level string   Set the logging level
                           ("debug"|"info"|"warn"|"error"|"fatal")
                           (default "info")
      --tls                Use TLS; implied by --tlsverify
      --tlscacert string   Trust certs signed only by this CA (default
                           "C:\\Users\\divya.srivastava\\.docker\\ca.pem")
      --tlscert string     Path to TLS certificate file (default
                           "C:\\Users\\divya.srivastava\\.docker\\cert.pem")
      --tlskey string      Path to TLS key file (default
                           "C:\\Users\\divya.srivastava\\.docker\\key.pem")
      --tlsverify          Use TLS and verify the remote
  -v, --version            Print version information and quit
```

1) docker --version

```
C:\Users\divya.srivastava>docker --version
Docker version 20.10.7, build f0df350

C:\Users\divya.srivastava>
```

2) Docker run hello-world

```
C:\WINDOWS\system32>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/
```

3) docker image

```
C:\WINDOWS\system32>docker images
REPOSITORY              TAG       IMAGE ID       CREATED        SIZE
docker/getting-started  latest    083d7564d904   2 months ago   28MB
hello-world             latest    d1165f221234   5 months ago   13.3kB
```

4) docker pull ubuntu

```
C:\WINDOWS\system32>docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
16ec32c2132b: Pull complete
Digest: sha256:82becede498899ec668628e7cb0ad87b6e1c371cb8a1e597d83a47fac21d6af3
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

5) docker run -it -d ubuntu

```
C:\WINDOWS\system32>docker run -it -d ubuntu
47f79e0ceaed3e2cdc601d7c725b8b8079a57fae0099fd2075d4d92ce936cf3e

C:\WINDOWS\system32>docker ps -a
CONTAINER ID   IMAGE                  COMMAND                  CREATED         STATUS                     PORTS                                NAMES
47f79e0ceaed   ubuntu                 "bash"                   32 seconds ago  Up 31 seconds                                           upbeat_maxwell
41c1a7efe409   hello-world            "/hello"                 5 minutes ago   Exited (0) 5 minutes ago                                suspicious_wozniak
234325d85374   hello-world            "/hello"                 18 minutes ago  Exited (0) 18 minutes ago                               practical_joliot
f55695774ddb   docker/getting-started "/docker-entrypoint.…"   22 minutes ago  Exited (255) 10 minutes ago  0.0.0.0:80->80/tcp, :::80->80/tcp  competent_villani
```

6) docker ps -a

```
C:\WINDOWS\system32>docker ps -a
CONTAINER ID   IMAGE                  COMMAND                  CREATED         STATUS                     PORTS                                NAMES
47f79e0ceaed   ubuntu                 "bash"                   32 seconds ago  Up 31 seconds                                           upbeat_maxwell
41c1a7efe409   hello-world            "/hello"                 5 minutes ago   Exited (0) 5 minutes ago                                suspicious_wozniak
234325d85374   hello-world            "/hello"                 18 minutes ago  Exited (0) 18 minutes ago                               practical_joliot
f55695774ddb   docker/getting-started "/docker-entrypoint.…"   22 minutes ago  Exited (255) 10 minutes ago  0.0.0.0:80->80/tcp, :::80->80/tcp  competent_villani

C:\WINDOWS\system32>docker exec -it 47f79e0ceaed bash
root@47f79e0ceaed:/#
```

7) Execution of inbuilt container, exit command, stop running any particular container, creation of new container:

   a) docker exec -it <container name> bash
   b) exit
   c) docker stop <container name>
   d) docker commit <container name> <new image name>

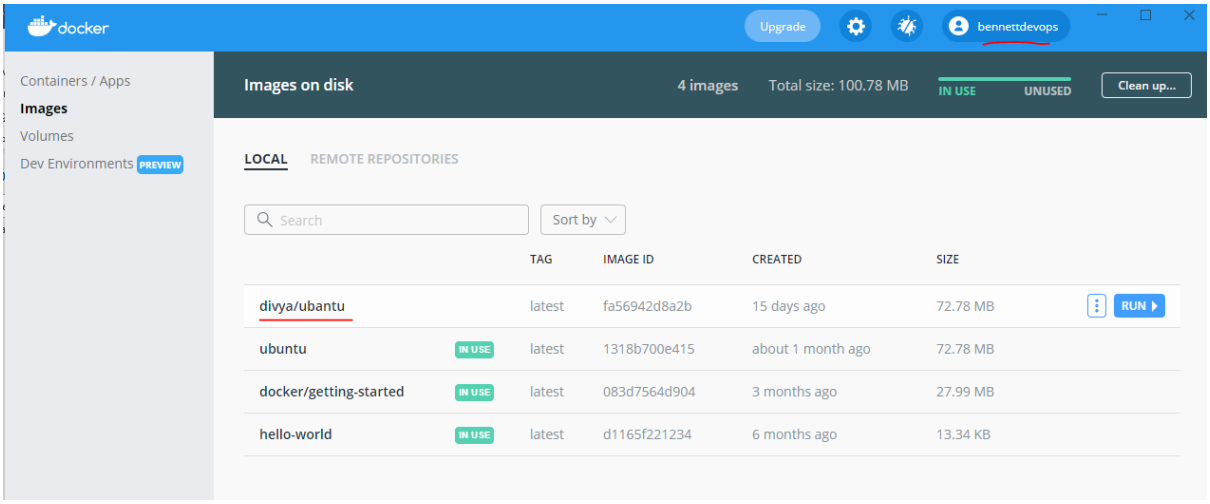
```
C:\WINDOWS\system32>docker exec -it 47f79e0ceaed bash
root@47f79e0ceaed:/# echo hello
hello
root@47f79e0ceaed:/# exit
exit

C:\WINDOWS\system32>docker stop 47f79e0ceaed
47f79e0ceaed

C:\WINDOWS\system32>docker commit 47f79e0ceaed divya/ubantu
sha256:fa56942d8a2bc082abcc6820669485c9cda5a1bfda8f0341490ebf0b76881f34

C:\WINDOWS\system32>docker images
REPOSITORY                TAG       IMAGE ID       CREATED          SIZE
divya/ubantu              latest    fa56942d8a2b   12 seconds ago   72.8MB
ubuntu                    latest    1318b700e415   2 weeks ago      72.8MB
docker/getting-started    latest    083d7564d904   2 months ago     28MB
hello-world               latest    d1165f221234   5 months ago     13.3kB
```

8) New image is created and is reflected in the Docker engine



9) Go to hub.docker.com and create your account over there. Click on repository section and create a public repository. Attach the account with your Github profile as well.

Docker Fie creation and assignment **(20)**

10) Create a Dockerfile and a python file with a simple print command.

11) Write the following commands in your dockerfile

```
# set base image (host OS)
FROM python:3.8

# make directory to hold scripts
RUN mkdir -p /src

# set the working directory in the container
WORKDIR /src

# copy the content of the local src directory to the working directory
ADD ./ /src

# copy the dependencies file to the working directory
COPY ./requirements.txt /src

# install dependencies
RUN pip3 install -r requirements.txt
```

```
# command to run on container start
CMD [ "python3", "labs.py" ]
```

12) Type this in your terminal - docker build -t <username>/<repo name>:<tag> and wait for your image to build

13) Push your docker image to docker hub by -

docker push <username>/<repo name>:<tag>

and check your docker hub to see if it was pushed

14) To be done by yourself – open your ec2 instance and run your dockerfile there by calling your image from your docker hub

docker run <username>/<repo name>:<tag>