**Docker** (1st Nov 2019)

Types Container engines – Docker, RKT, LXD, LXC etc.

VM – Hardware Virtualization

Application Isolation - Containerization

In BIOS we need to enable virtualization

Add remove programs and features

Windows features on and off

Hyper-V is installed – Windows server / windows 10 pro

Container services

Docker Desktop must be installed.

Hyper-V Manager => MobyLinuxVM / DockerDesktopVM

Try Commands:

Docker version

Client and server versions

Docker images

**Images and Containers:**

Application – winword.exe

Process – when application goes to running state it is a process.

Images:

Containerized application is called image.

Image is just virtual collection.

Image is used to create instances of application.

The running instance is called container.

Every container have a unique name and id.

Every image also have unique name and id.

Image name must be in <imagename>:<tag> format

Tag is a version identifier or unique value that can be used to identify image.

Eg: eshop-web:1.0 , eshop-web:2.0 , eshop-web:3.0

If tag name is not supplied at the time of image creation, it will be assigned to ‘latest’ by default.

Eg: eshop-web:latest

An image contains a set of layers.

Every layer has unique id

**Image Repositories (registries):**

It is a storage place for code and data.

A Docker image repository is a place to store images.

Repositories can be:

Local

Store locally in Dev machine.

Local repo images name can be anything, without repo name.

Cloud

Publically accessible repo

Anyone can access it from here.

**Two types:**

**Public** – when uploading into a public repo it is accessible to all without any credentials.

Docker Hub is an example for public repo.

**Private** – secured repo

It needs credentials to push and pull images.

Azure container registry (ACR), Elastic container registry (ECR) and GCR

**To run the image file-**

>> docker run -p host-port:container-port --d --name <nameofcontainer> --rm <imagename>

--name = used to assign a name for the running container. If not use docker will assign a random name to the container.

-d = detached mode.

--rm = remove the container when stopped.

-p = map host machine port number to container application port.

To check Old images on PC:

>> Docker images

To check

To download image from docker

>> Docker pull <image name>

List all images located in local repo-

>> Docker images

List all running containers

>> Docker ps

To stop the running container

>> Docker stop <containerId/containerName>

To get list of running and stopped containers

>> Docker ps -a

Remove the stopped containers

>> Docker rm <containerId/name> <containerId/name> multiple Ids

Stop and remove running containers

>> Docker rm -f <containerId/Name>

**Building Images:**

A set of commands need to be executed

Commands can be written into a file called “Dockerfile”

The instruction to build an image may start with the following keywords:

FROM <baseimagename>

Specify the base image for our application.

Eg: FROM nginx:latest

LABEL <key>=<value>

Used to add meta information for image, such as author info, version, company etc.

Eg: LABEL author “Amol Patole”

Eg: LABEL version “1.2”

RUN <cmd> <args>

Execute a command while building the image.

Eg: RUN dotnet restore

Eg: RUN dotnet build

Eg: RUN dotnet publish

// single line RUN dotnet restore && dotnet build && dotnet publish

WORKDIR <dir\_path>

Set the current working directory inside the container.

Eg: WORKDIR /usr/app

Eg: WORKDIR C:\users\myapp

COPY <sourceFiles/pattern> <dest\_path>

COPY [-from <stage>] <sourceFiles/pattern> <dest\_path>

Copy files from local machine to docker image path.

Eg: COPY ./\*.\* ./

Eg: COPY ./\*.jpg ./

Eg: COPY ./\*.\* /usr/app

ENV <envVariableName> <value>

Used to set the environment variable value for the app.

EXPOSE <port-number>

Specify the port number to be opened.

CMD <command> <args>

Execute commands when the container is created.

When image is running.

ENTRYPOINT <cmd> <args>

Execute commands when the container is created.

When image is running.

It is used to specify the starting command of application.

**Building Docker images:**

**Create below Docker File**

FROM nginx:latest

LABEL author "amol patole"

LABEL company "Hexa"

LABEL version "1.0"

WORKDIR /urs/share/nginx/html/

COPY ./ ./

EXPOSE 80

Docker build --t <imagename:tag> <dockerfilepath>

Docker build --t mysampleapp:v1 .

**Run docker via port forwarding:**

Docker run --p 7010:80 --name “sampleapp” mysampleapp:v1

Start existing stopped container

Docker start <containerName>

Stop running container

Docker stop <containerName>

To remove Docker images

Docker rmi <imageid/name>

Containerizing .NET Core apps:

Download SDK and runtime images

Docker pull mcr.microsoft.com/dotnet/core/aspnet:2.2.7-bionic

Docker pull mcr.microsoft.com/dotnet/core/sdk:2.2.402-bionic

Multi Stage Building

Create docker file and .doclkerignore file by installing docker extension.

Hit F1

Search docker.add =>

Build image using => docker build –t <imagename> .

Run the image using => docker run –p hostport:containerport <imagename>:<tag?>