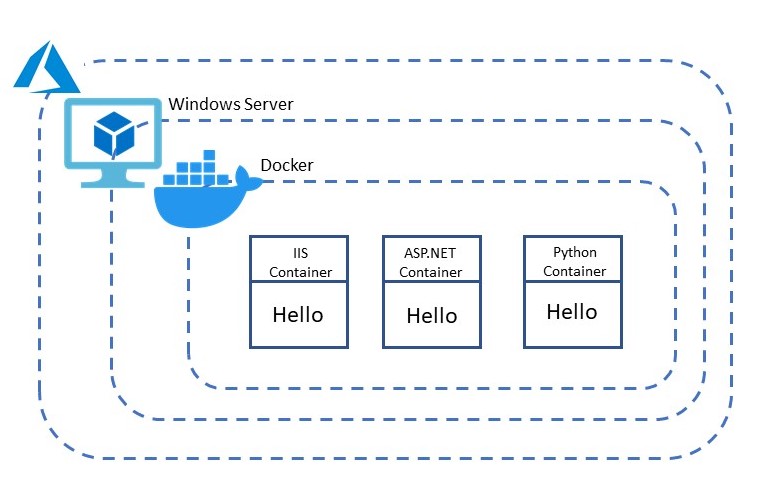
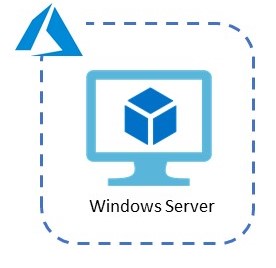
**Create & Deploy Docker Images**

Windows Server containers provide applications an isolated, portable and resource controlled operating environment. This isolation enables containerized applications to run without risk of dependencies and environmental configuration affecting the application. By sharing the same kernel and other key system components, containers exhibit rapid startup times and reduced resource overhead. Rapid startup helps in development and testing scenarios and continuous integration environments, while the reduced resource overhead makes them ideal for service-oriented architectures. The Windows Server container infrastructure allows for sharing, publishing and shipping of containers to anywhere the next wave of Windows Server is running. With this new technology millions of Windows developers familiar with technologies such as .NET, ASP.NET, PowerShell, and more will be able to leverage container technology.

Docker has done a fantastic job of building a vibrant open source ecosystem based on Linux container technologies, providing an easy user experience to manage the lifecycle of containers drawn from a huge collection of open and curated applications in Docker Hub. We will bring Windows Server containers to the Docker ecosystem to expand the reach of both developer communities.

**Normal Deployment vs Containerized Web Application**



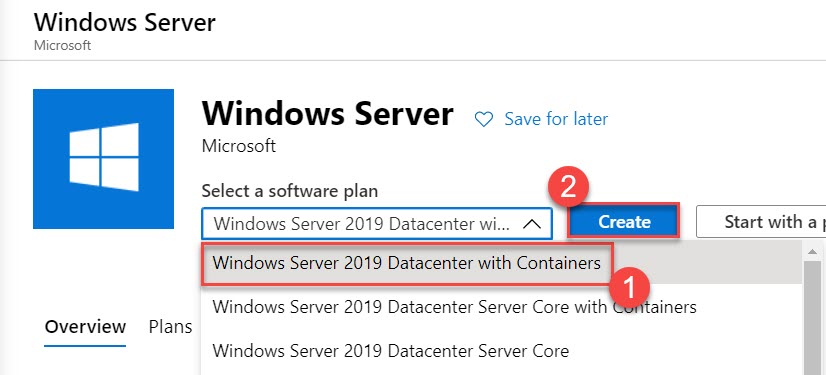
**Step 1:** Open **Microsoft Azure Portal**

[https://portal.azure.com](https://portal.azure.com/)

**Step 2:** Open below link in browser

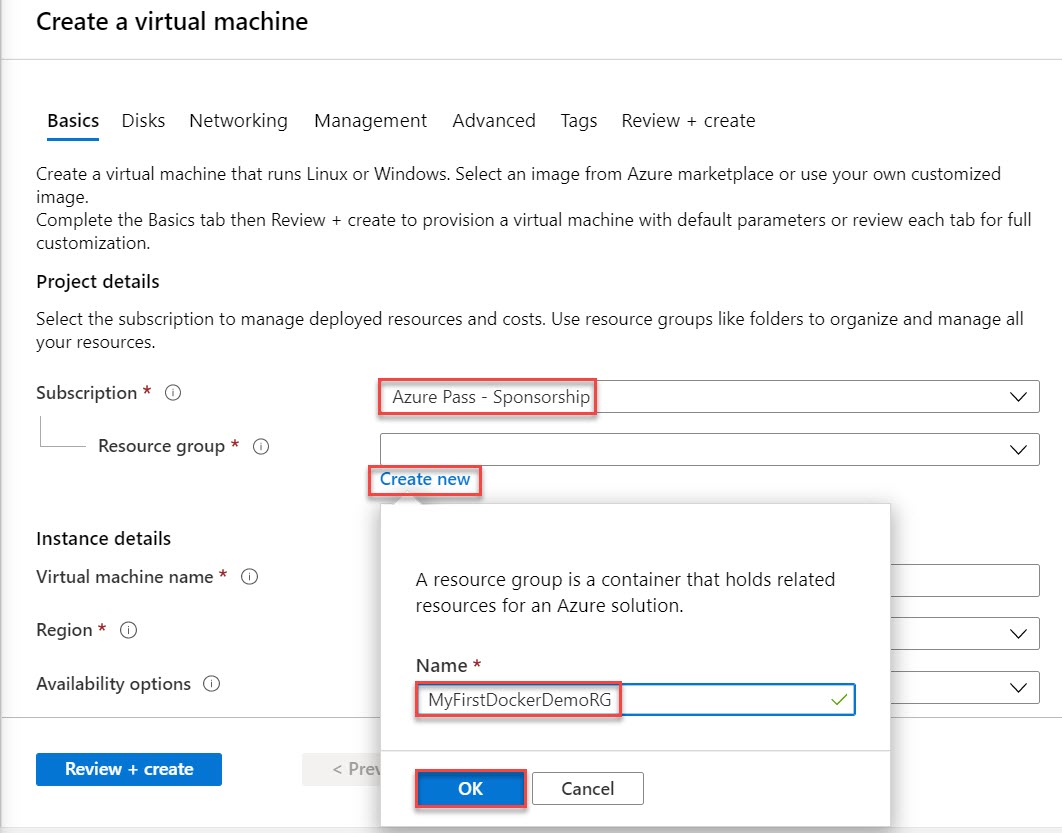
<https://portal.azure.com/#create/Microsoft.WindowsServer2019DatacenterwithContainers-ARM>

**Step 3:** Select **Windows Server 2019 Datacenter with Containers** and click on **Create** button



**Step 4:** Create New Virtual Machine.

Click on Create New Resource Group: **MyFirstDockerDemoRG**

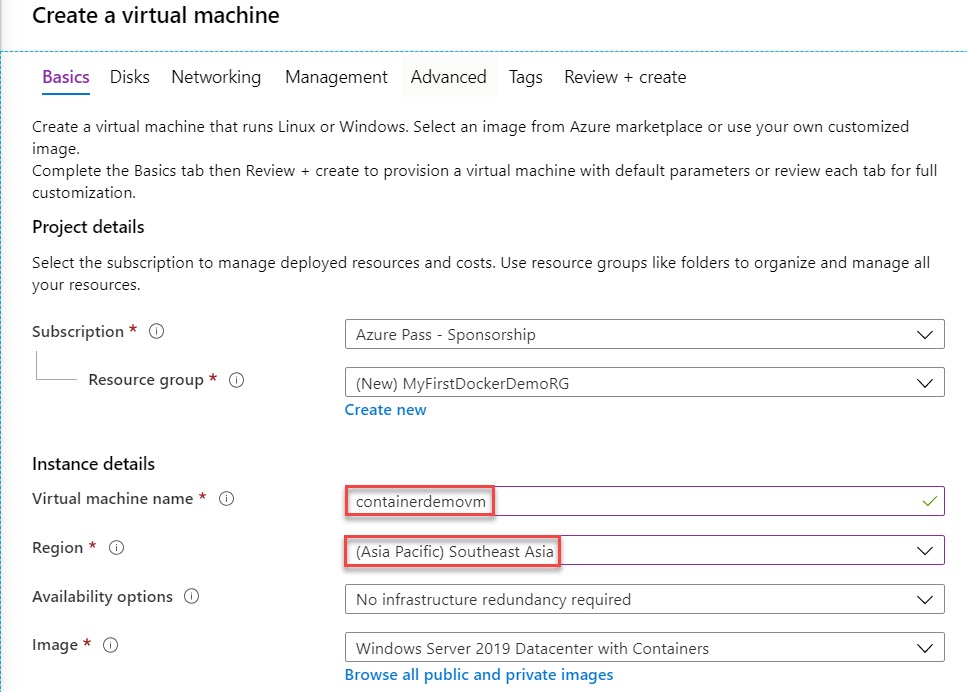


Provide Instance Details

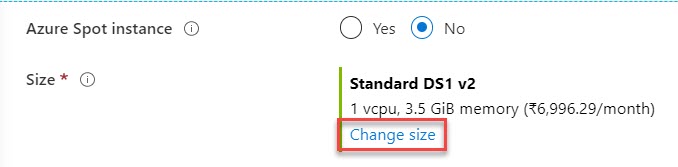
Virtual Machine Name: **containerdemovm**

Region: **Choose any nearest region** ex. **Southeast Asia**

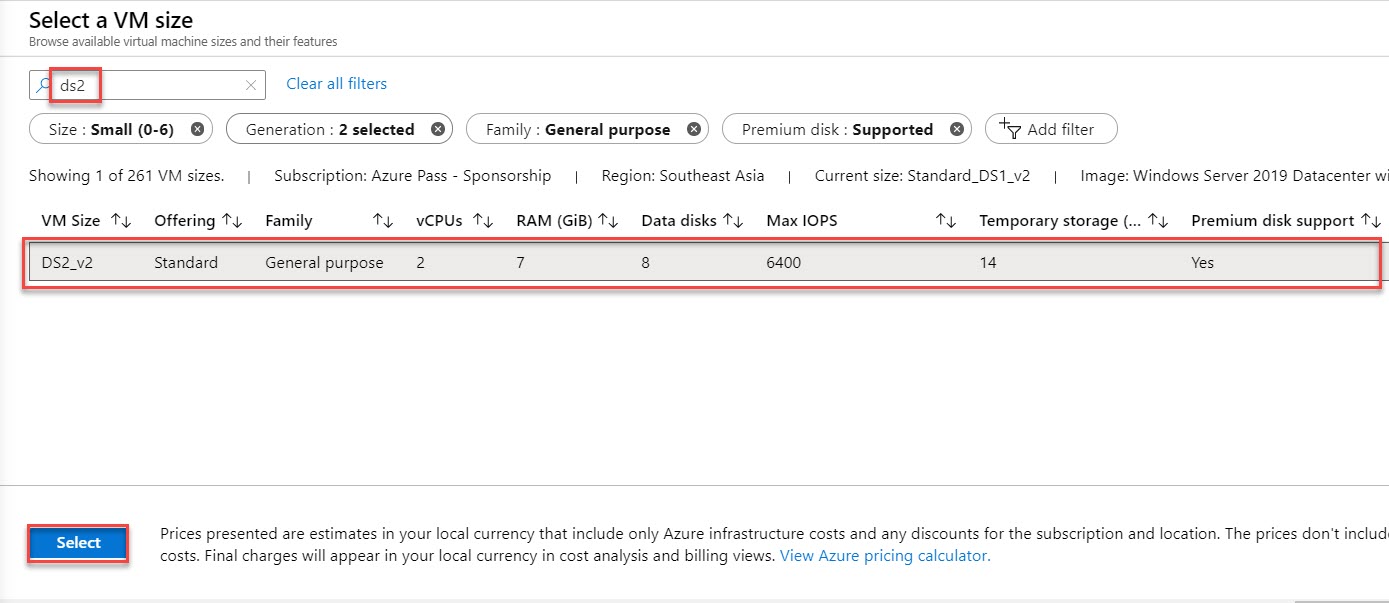
Image: **Windows Server 2019 Datacenter with Containers**



**Change Size** of Virtual Machine



Search for “**DS2**” and select **DS2\_v2**



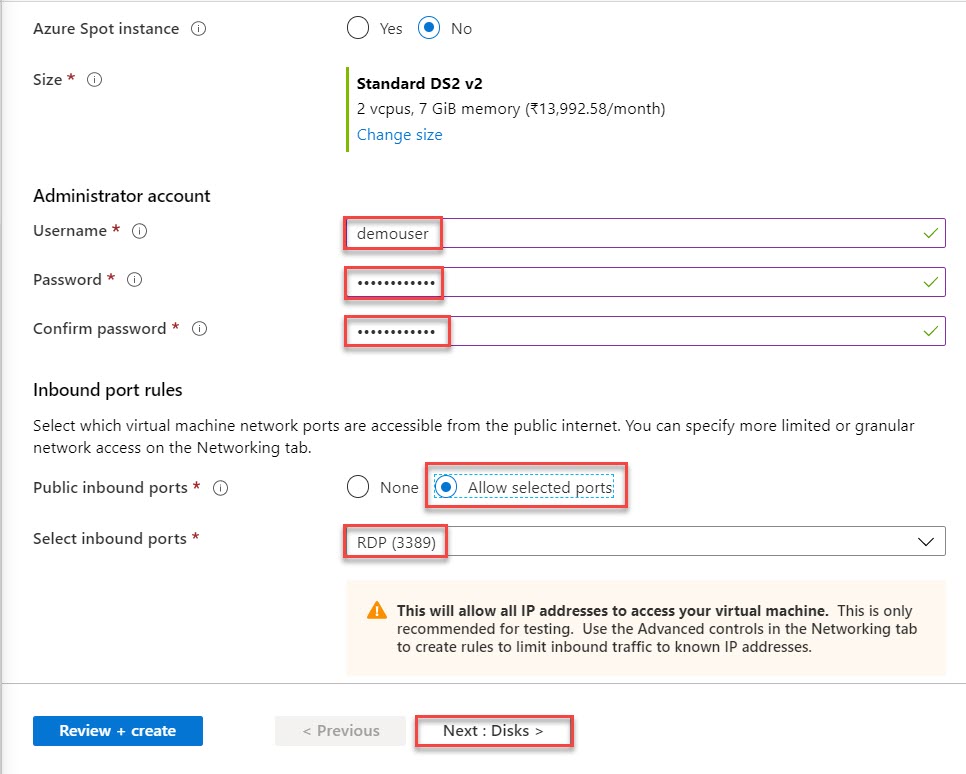
Administrator account

Username: **demouser**

Password: **demo@pass123**

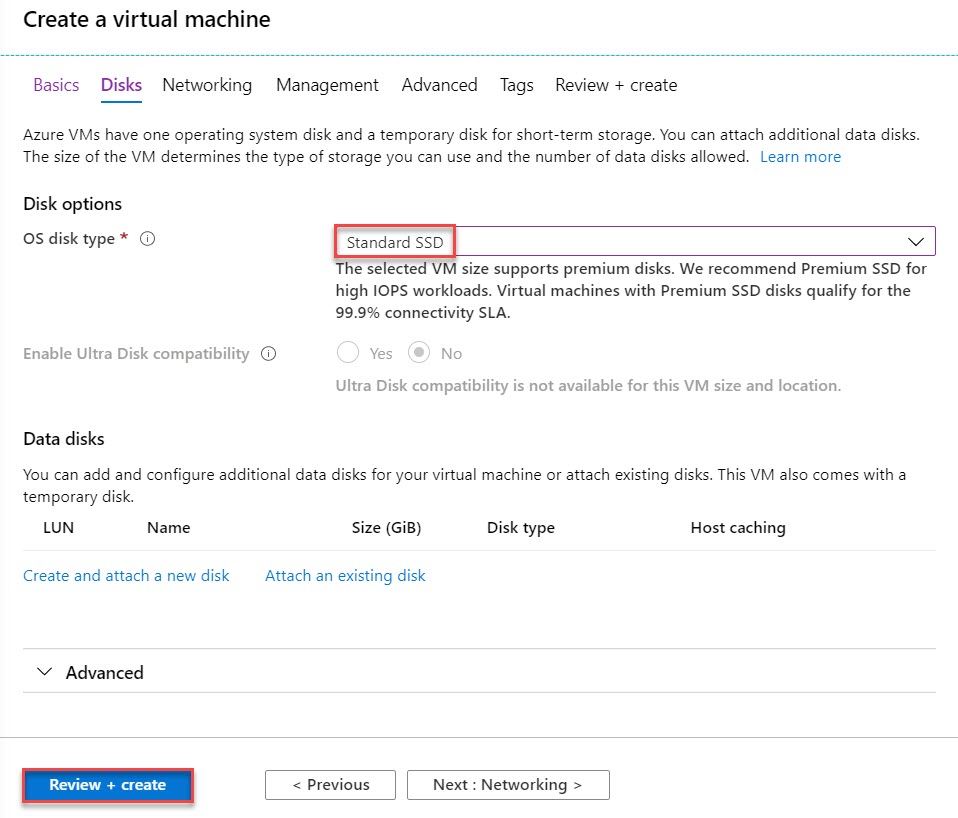
Public inbound ports: **Allow Select Ports** Ex. **RDP (3389)**

Click on **Next: Disks >**

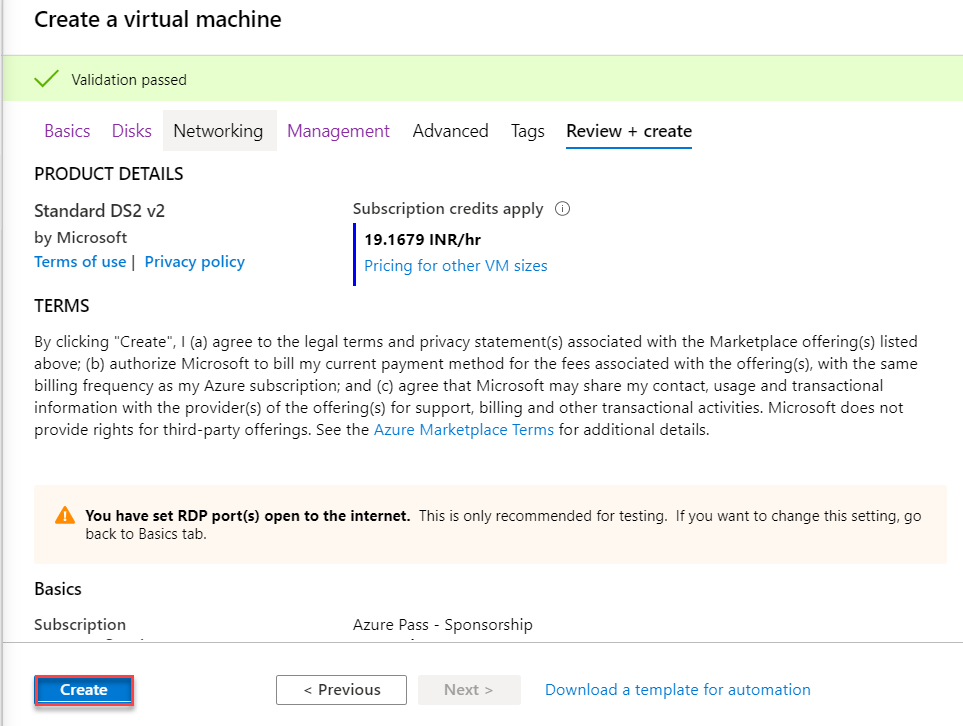


**Step 5:** Select Disk Type: **Standard SSD**

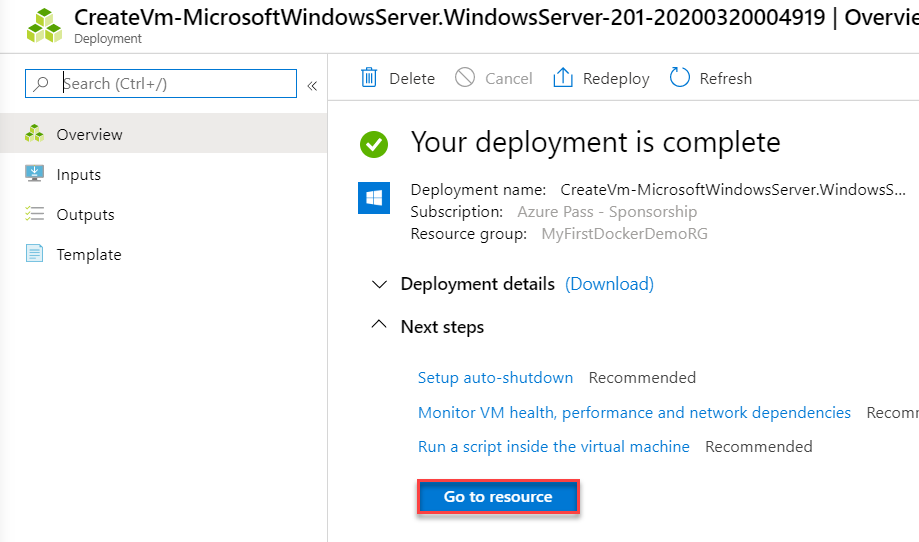
Click on **Next: Review + create >**



**Step 6:** Click on **Create** button.

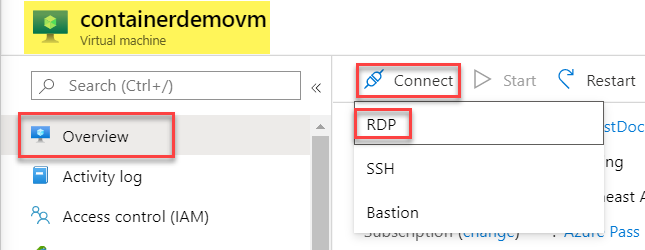


Virtual Machine Deployment will take couple of minutes. Click on **Go to resource** button

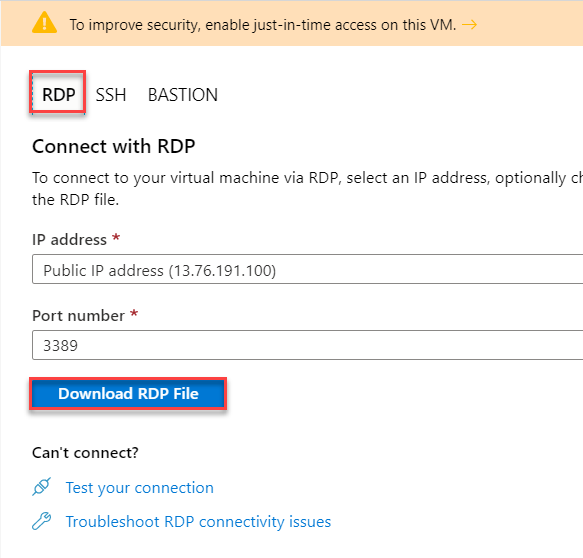


**Step 7:** Virtual Machine blade will open.

Click on **Connect** option and click on **RDP** option



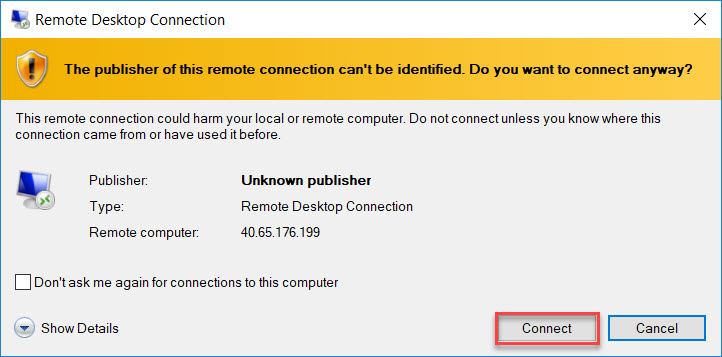
**Step 8:** Click on **Download RDP File** under RDP tab



It will download RDP file and double click on that.



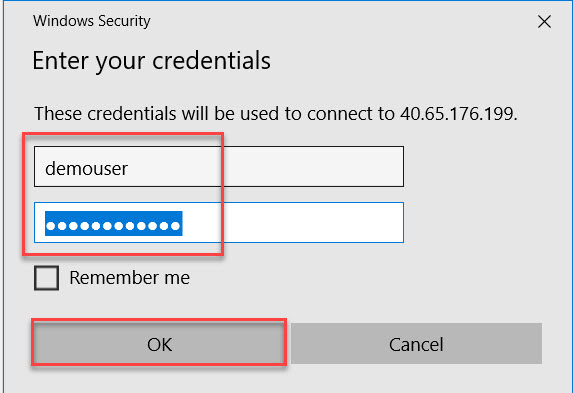
**Step 9:** Click on **Connect** button.



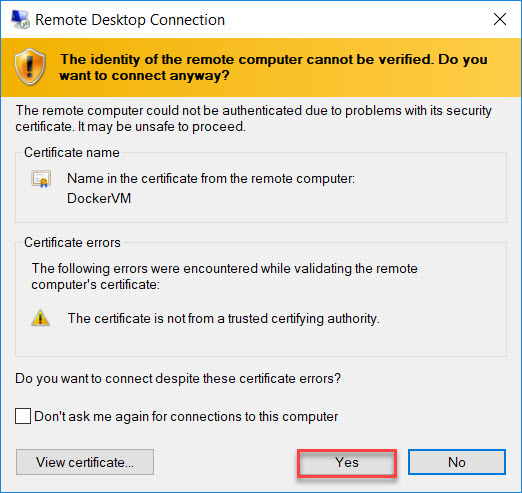
**Step 10:** Enter Virtual Machine Credentials:

Username: **demouser**

Password: **demo@pass123**

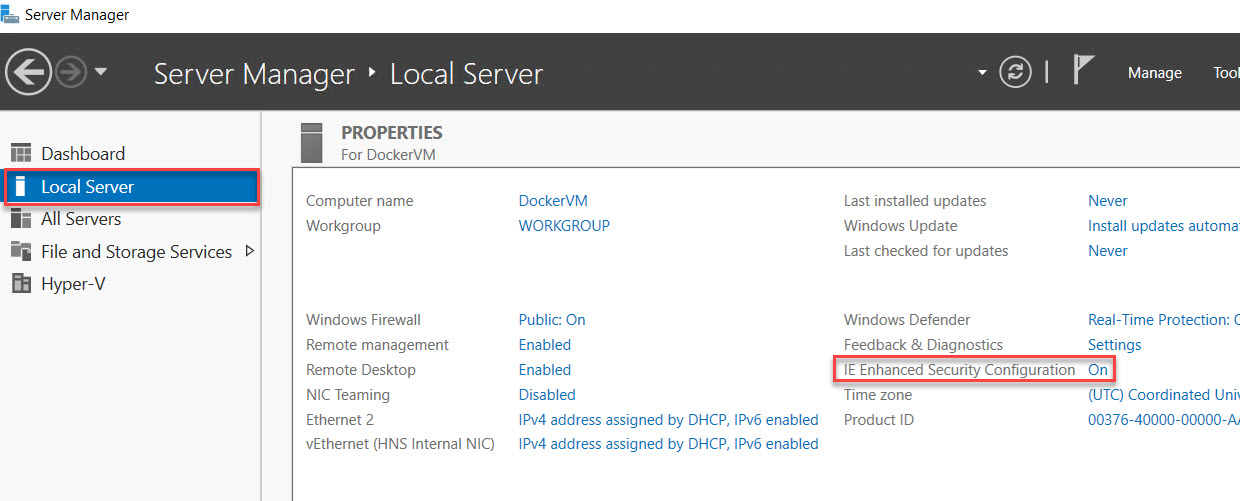


**Step 11:** Click on **Yes** button.

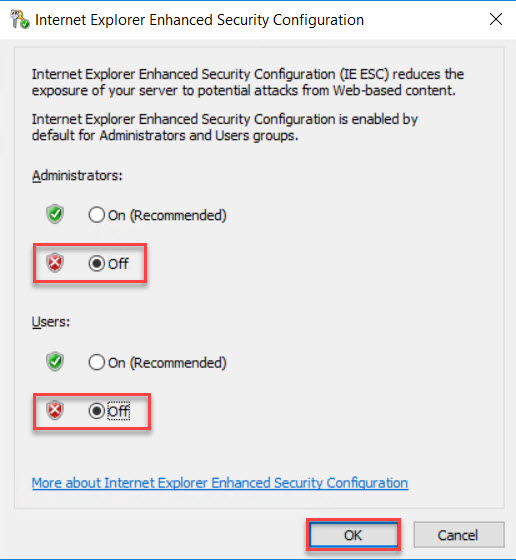


**Step 12:** Windows Server 2019 Datacenter will start.

Within couple of second **Server Manager** will open and Turn **off IE Enhanced Security Configuration**.



**Administrators: Off & Users: Off.**

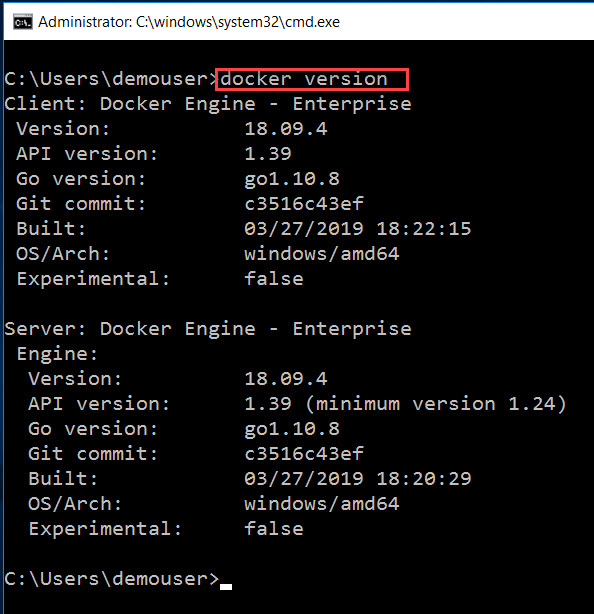


**Step 13:** Start **Command Prompt** and type below commands:

docker version

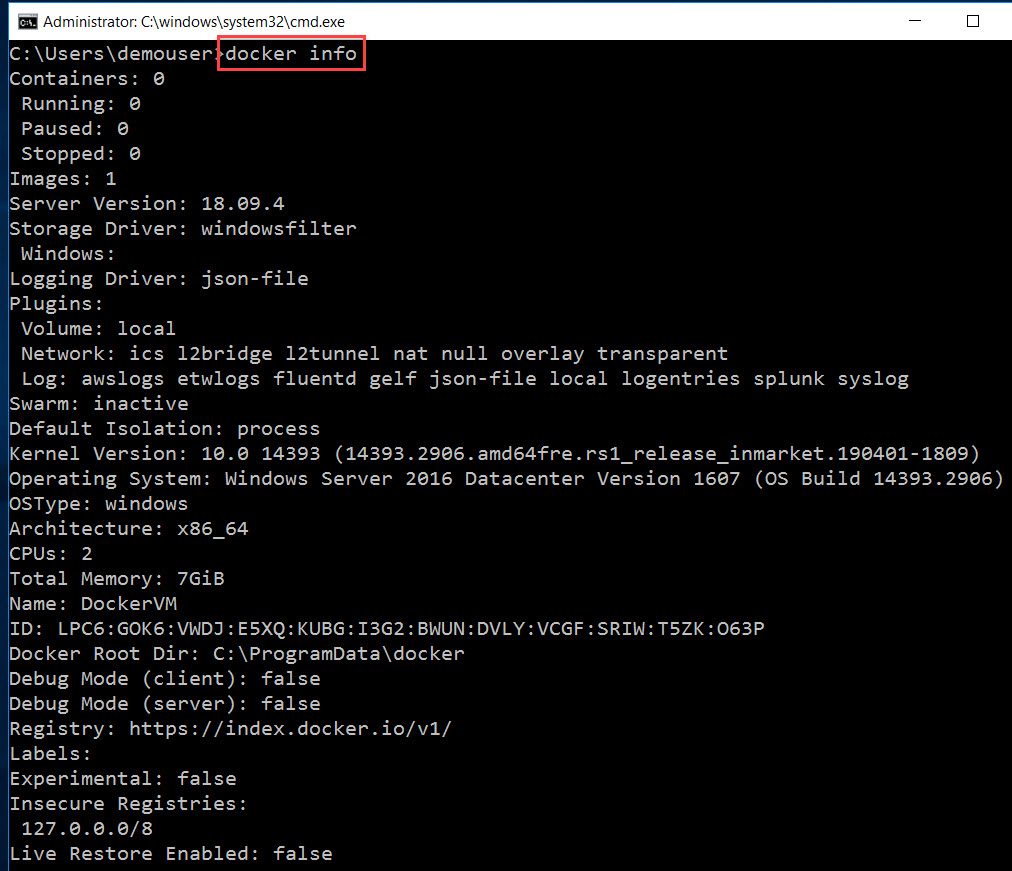
It will show Client and Server versions.

Here Client and Server installed on same machine. There any two-edition available: Community and Enterprise. Community edition is free of cost.



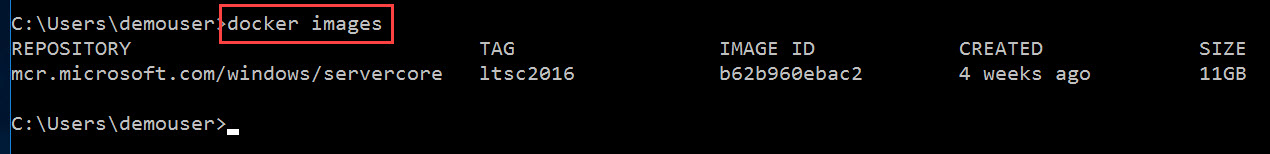
**Step 14:** To get system information run below command

docker info



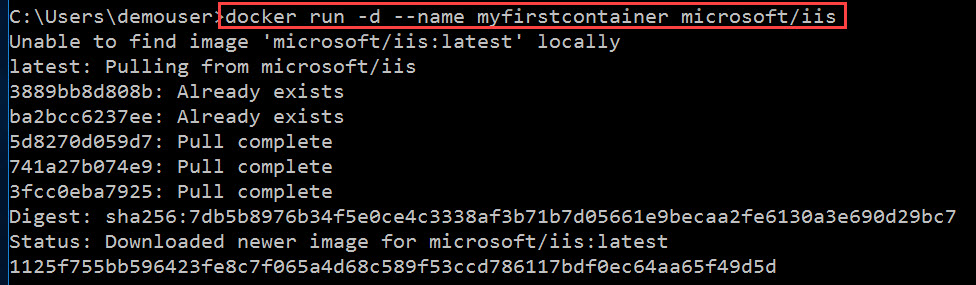
**Step 15:** Check available images on machine

Docker images



**Step 16:** Run Container

docker run -d --name myfirstcontainer microsoft/iis

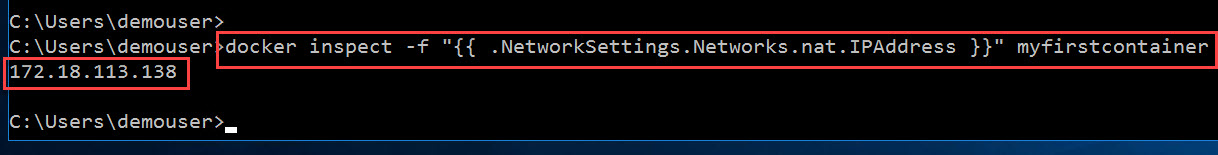


The output from Docker can be divided into several sections:

1. In the first line, Docker is telling you that it couldn't find the image you specified, **microsoft/iis**, on the Docker server. The *latest* portion after the colon (:) is a tag. The tag identifies which version of the image to use. By default, it looks for the *latest* version.
2. In the next line, it notifies you that it automatically pulled the image. You could manually perform that task using the command docker pull microsoft/iis. The *microsoft/iis* is the repository it's pulling from inside the Docker Hub registry. *microsoft*is the account name for Microsoft on Docker Hub. In general, images will come from repositories identified using the pattern *account/repository*. The account name for official Docker images is *library*.
3. The next six lines show the status of layers inside an image. The first two layers were already present and the next four had to be *pulled*/downloaded.
4. The next two lines confirm the image has been downloaded.
5. The last line is the ID of the container that started running. Usually you won't see the long form of the ID. You will usually only see the first 12 characters.

**Step 17:** To check IP Address of Container

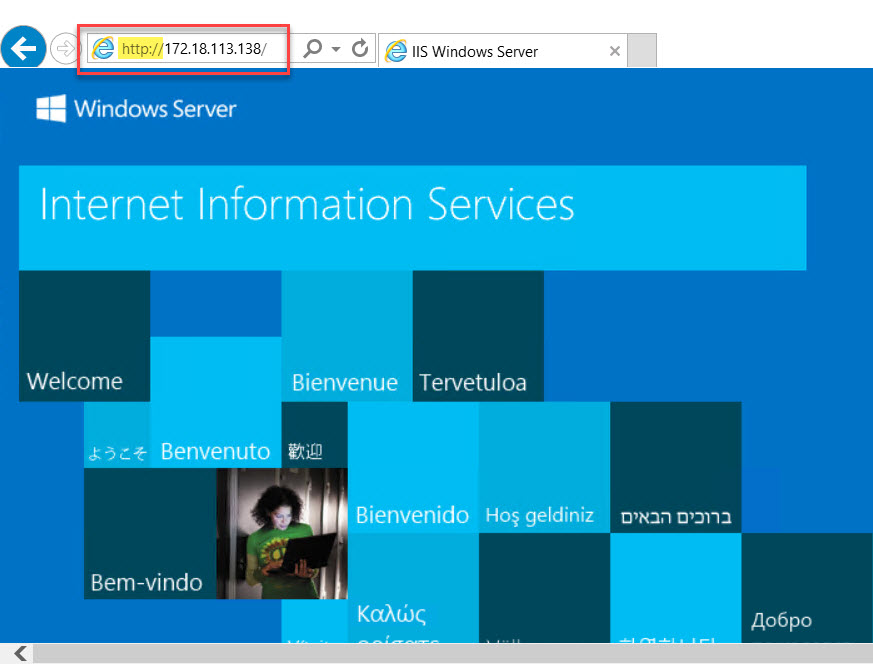
docker inspect -f "{{ .NetworkSettings.Networks.nat.IPAddress }}" myfirstcontainer



**Step 18:** Open Internet Explorer.

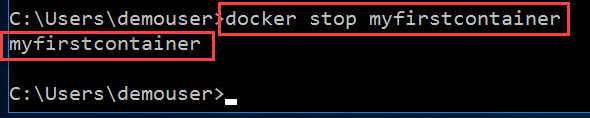
Type http://<IPADDRESS>

Ex. http://1XX.XX.XXX.XXX



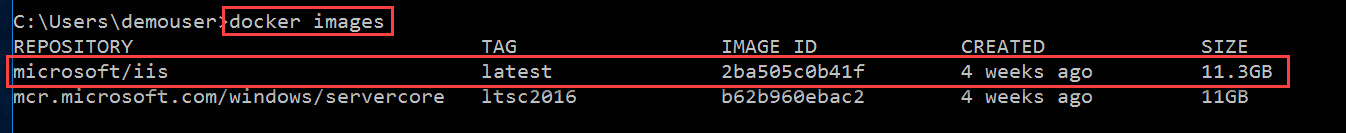
**Step 19:** Now stop this container (optional step)

docker stop myfirstcontainer



**Step 20:** Check all the images

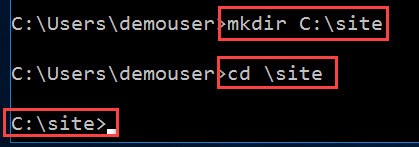
docker images



**Step 21:** Create one folder

mkdir c:\site

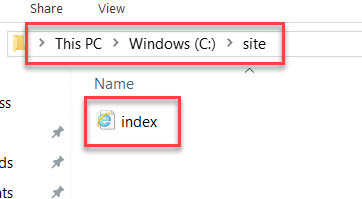
cd \site



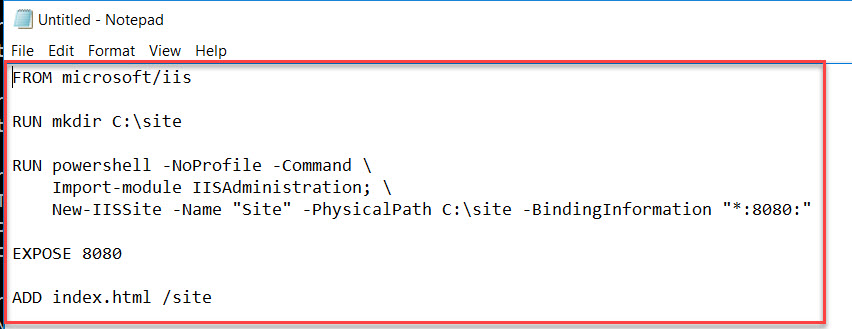
**Step 22:** Type below command to create web page

echo ^<!doctype html^>^<body^>^<h3^>My first web page on Docker^</h3^>^</body^>^</html^> > index.html

**Step 23:** navigate to C Drive and open site folder, one index.html page available.



**Step 24:** Now open notepad and add below code to docker file



FROM microsoft/iis

RUN mkdir C:\site

RUN powershell -NoProfile -Command \

Import-module IISAdministration; \

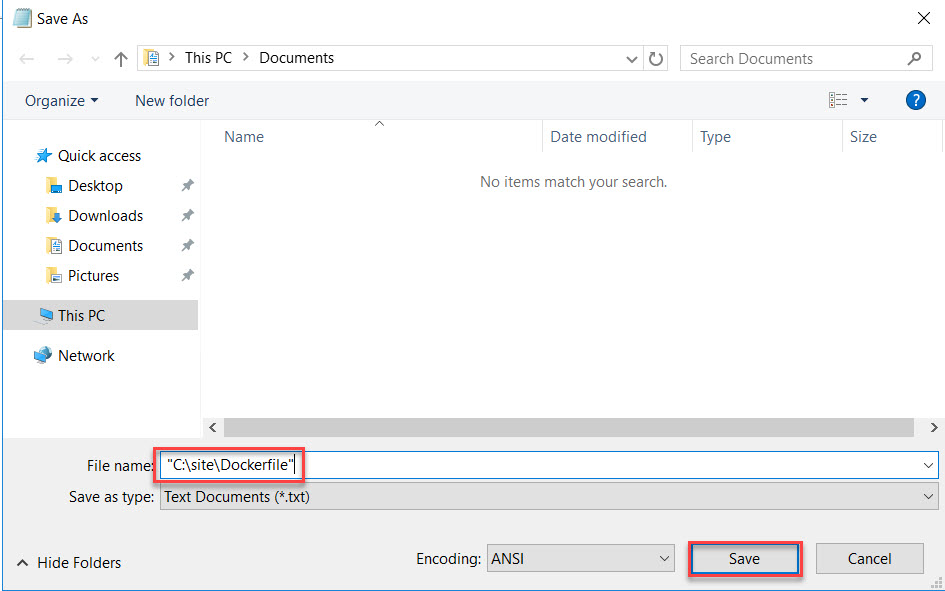
New-IISSite -Name "Site" -PhysicalPath C:\site -BindingInformation "\*:8080:"

EXPOSE 8080

ADD index.html /site

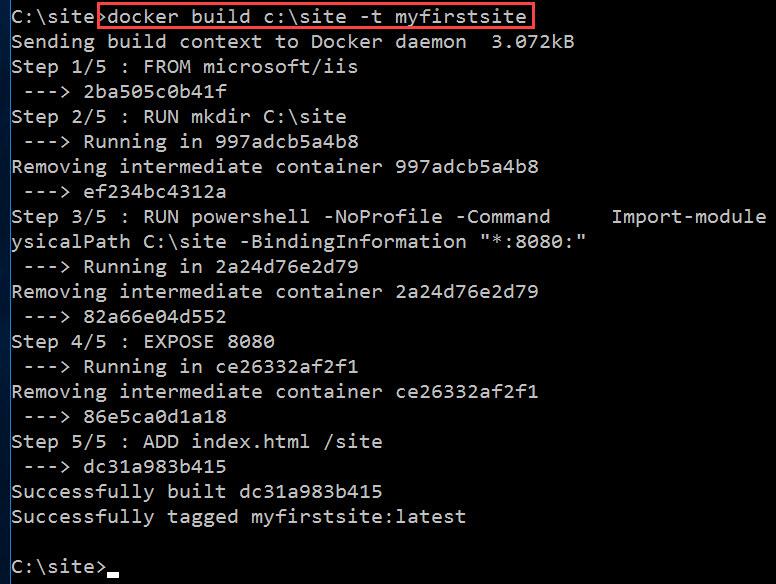
**Step 25:** Save the file and type this file name **"C:\site\Dockerfile"**

Note: include the quotation marks. Dockerfile name is also case sensitive.



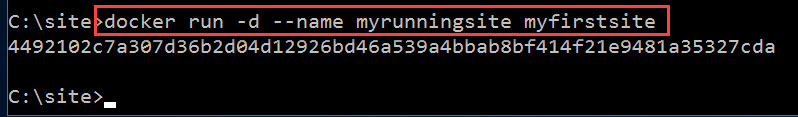
**Step 26:** Time to build an image to create container

docker build c:\site -t myfirstsite



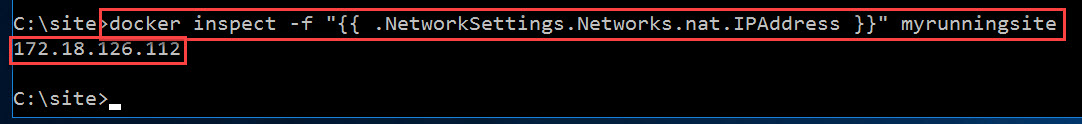
**Step 27:** Now run container using an image

docker run -d --name myrunningsite myfirstsite



**Step 28:** Get IP Address assigned to the container

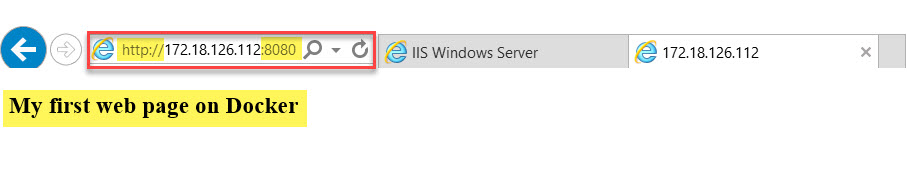
docker inspect -f "{{ .NetworkSettings.Networks.nat.IPAddress }}" myrunningsite



**Step 29:** Navigate to Internet Explorer and type.

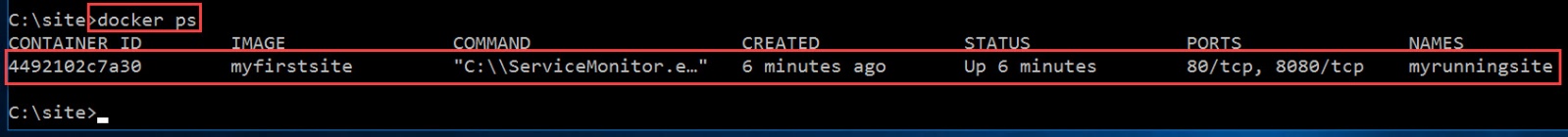
http://<IP-ADDRESS>:8080

Ex. http://1XX.XX.XX.XXX:8080



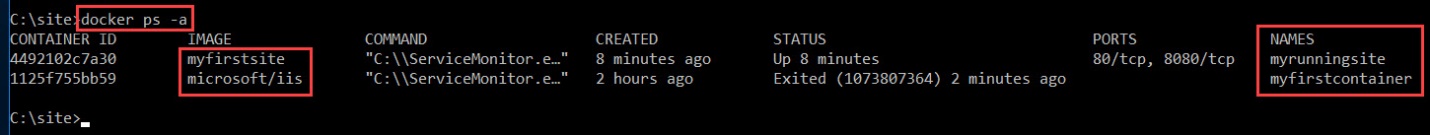
**Step 30:** To list running container

docker ps



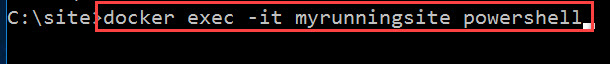
**Step 31:** To list running and stopped containers

docker ps -a

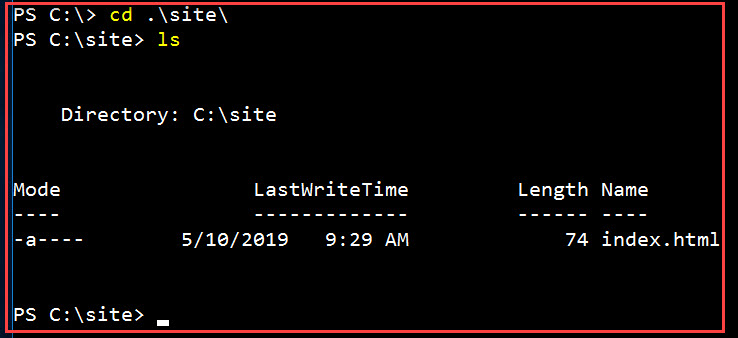


**Step 32:** Run other commands in a running container. Here PowerShell prompt will start

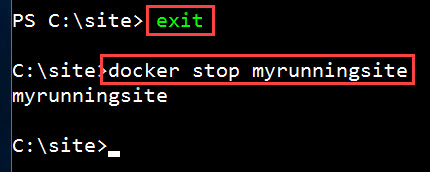
docker exec -it myrunningsite powershell



Type cd .\site\

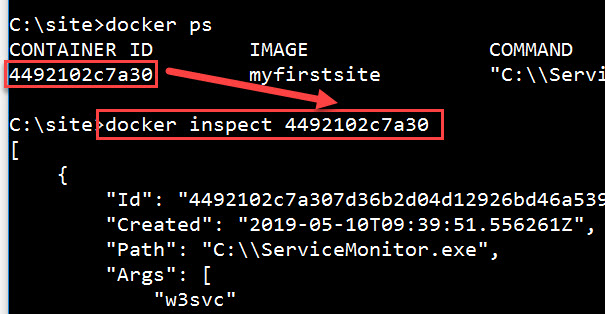


**Step 33:** type **Exit**



**Step 34:** Also, can check information from container id

docker inspect <container-id>

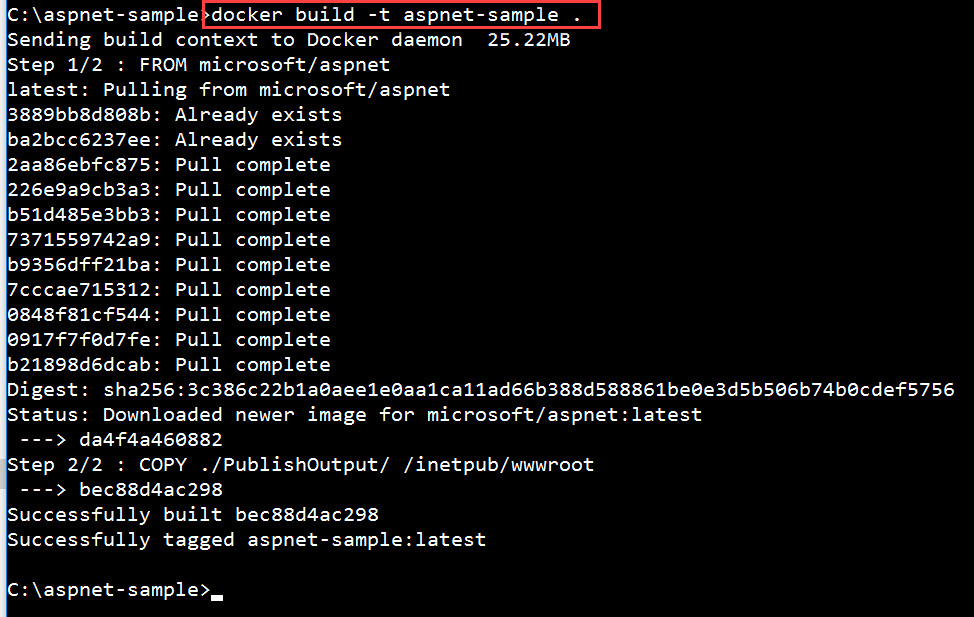


**Step 35:** Navigate to another folder (Copy from your system to Container Virtual Machine)

cd C:\aspnet-sample

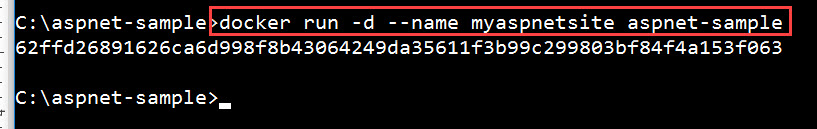
Build image

docker build -t aspnet-sample .



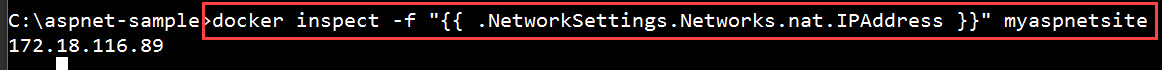
**Step 36:** run container

docker run -d –name myaspnetsite aspnet-sample



**Step 37:** Get IP Address

docker inspect -f "{{ .NetworkSettings.Networks.nat.IPAddress }}" myaspnetsite



**Step 38:** Open Browser and check the output

