Exercise: Create a virtual machine using the Azure portal

Task 1: Open the Azure portal

Task 2: Create a resource group

Create a new resource group with the following details:

Name: ContainerCompute

Location: (US) East US

Wait for the creation task to complete before moving forward with this lab.

Task 3: Create a Linux virtual machine resource

Create a new Virtual Machine with the following details:

Operating system: Ubuntu Server 18.04 LTS

Name: simplevm

Disk type: SSD

Username: <user name>

Password: <password>

Resource group: ContainerCompute

Location: (US) East US

Size: Standard B1s

Public inbound ports: SSH (22)

Wait for the creation task to complete before moving on with this lab.

Task 4: Validate the virtual machine

Access the simplevm VM that you created earlier in this lab.

Exercise: Create a Docker container image and deploy it to Azure Container Registry

Task 1: Open the Cloud Shell and editor

Open a new Cloud Shell instance in the Azure Portal.

Within the Cloud Shell command prompt, change the active directory to ~/clouddrive.

In the Cloud Shell command prompt, create a new directory ipcheck within the ~/clouddrive directory.

Use mkdir <directory name>.

Change the active directory to ~/clouddrive/ipcheck.

Use the dotnet new console --output . --name ipcheck command to create a new .NET Core console application in the current directory.

Create a new file in the ~/clouddrive/ipcheck directory named Dockerfile.

Use touch <file name>. The filename Dockerfile (case sensitive)

Open the embedded graphical editor in the context of the current directory.

Task 2: Create and test a .NET Core application

In the graphical editor, open the Program.cs file and replace its contents with the following code, and then save the file:

```
public class Program

{

   public static void Main(string[] args)

   {

      if (System.Net.NetworkInformation.NetworkInterface.GetIsNetworkAvailable())

      {

         System.Console.WriteLine("Current IP Addresses:");

         string hostname = System.Net.Dns.GetHostName();

         System.Net.IPHostEntry host = System.Net.Dns.GetHostEntry(hostname);

         foreach(System.Net.IPAddress address in host.AddressList)

         {

            System.Console.WriteLine($"\t{address}");

         }

      }

      else

      {

         System.Console.WriteLine("No Network Connection");

      }

   }

}
```

Use the dotnet run command in the command prompt to execute the application and validate that it finds one or more IP addresses.

Open the Dockerfile file in the graphical editor, replace its contents with the following code, and then Save the file (use Ctrl + S)

```
FROM mcr.microsoft.com/dotnet/core/sdk:2.2-alpine AS build

WORKDIR /app

COPY *.csproj ./
```

```
RUN dotnet restore

COPY . ./

RUN dotnet publish --configuration Release --output out

FROM mcr.microsoft.com/dotnet/core/runtime:2.2-alpine

WORKDIR /app

COPY --from=build /app/out .

ENTRYPOINT ["dotnet", "ipcheck.dll"]
```

Close the Cloud Shell pane.

Task 3: Create an Azure Container Registry resource

Create a new container registry with the following details:

Name: < name>

Resource group: ContainerCompute

Location: East US

Admin user: Disable

SKU: Basic

Wait for the creation task to complete before moving on with this lab.

Task 4: Open Cloud Shell and store Azure Container Registry metadata

Open a new Cloud Shell instance.

Within the Cloud Shell command prompt, use the az acr list command to view a list of all container registries in your subscription.

Use the following command to output the name of the most recently created container registry:

```
az acr list --query "max_by([], &creationDate).name" --output tsv
```

Use the following command to save the name of the most recently created container registry in a Bash shell variable acrName:

```
$acrName=$(az acr list --query "max_by([], &creationDate).name" --output tsv)

echo $acrName
```

Task 5: Deploy a Docker container image to Azure Container Registry

Change the active directory to ~/clouddrive/ipcheck.

Use the dir command to view the contents of the current directory.

Use the following command to upload the source code to your container registry and build the container image as an Azure Container Registry Task:

```
az acr build --registry $acrName --image ipcheck:latest .
```

Wait for the build task to complete before moving forward with this lab.

Close the Cloud Shell pane.

Task 6: Validate your container image in Azure Container Registry

Access the container registry that you created earlier in this lab.

Select the Repositories link to view your images stored in the registry.

Proceed through the Images and Tags blades to view the metadata associated with the ipcheck image with the latest tag.

Exercise: Deploy an Azure container instance

Task 1: Enable Admin User in Azure Container Registry

Access the container registry that you created earlier in this lab.

Select the Update button to view the settings for the Container Registry.

Enable the Admin User.

Task 2: Deploy a container image automatically to Azure container instance

Select the Repositories link to view your images stored in the registry.

Select the ipcheck image and view the latest tag for that image.

Select/right-click the latest tag for the ipcheck container image to deploy a new Azure container instance with the following settings:

Container name: managedcompute

OS type: Linux

Resource group: ContainerCompute

Location: East US

Number of cores: 2

Memory (GB): 4

Public IP address: No

Wait for the creation task to complete before moving on with this lab.

Task 3: Deploy a container image manually to an Azure container instance

Access the container registry that you created earlier in this lab.

Select the Access keys link to view the credentials necessary to access your container registry from another service. Record the following values from this section to use later

Login server

Username

Password

Create a new container instance with the following details, using the Access keys credentials you recorded earlier in this lab:

Container name: manualcompute

Image type: Private

Image name: <Login server recorded earlier in the lab >/ipcheck:latest

Image registry login server: <Login server recorded earlier in the lab>

Image registry user name: <Username recorded earlier in the lab >

Image registry password: <Password recorded earlier in the lab >

Resource group: ContainerCompute

Region: East US

OS type: Linux

Number of cores: 1

Memory (GB): 1.5

Public IP address: Yes

Port: 80

Open additional ports: No

Port protocol: TCP

Restart policy: On failure

Wait for the creation task to complete before moving forward with this lab.

Task 4: Validate that the container instance ran successfully

Access the manualcompute container instance that you created earlier in this lab.

Select the Containers link to view a list of the current containers running.

View the contents of the Events list for the container instance that ran your ipcheck application.

Add another instance with FQDN with following command

```
az container create --resource-group computecontainer --name mycontainer --image mcr.microsoft.com/azuredocs/aci-helloworld --dns-name-label <unique name> --ports 80
```

find the FQDN by command

```
az container show --resource-group containercompute--name mycontainer --query "{FQDN:ipAddress.fqdn,ProvisioningState:provisioningState}" --out table
```

Use the FQDN to view the image

Exercise: Clean up subscription

Task 1: Open Cloud Shell and list resource groups

At the top of the Azure portal, select the Cloud Shell icon to open a new shell instance.

In the Cloud Shell command prompt at the bottom of the portal, type the following command and press Enter to list all resource groups in the subscription:

```
az group list
```

Type the following command and press Enter to view a list of possible commands to delete a resource group:

```
az group delete --help
```

Task 2: Delete resource groups

Type the following command and press Enter to delete the ContainerCompute resource group:

```
az group delete --name ContainerCompute --no-wait --yes
```