Module 05 (CSR): Implement laaS solutions

1 Hr 15 Min Remaining

Instructions Resources Help  100%

Required Lab Setup

Sign in to the lab virtual machine

1. Hello Altaf Hussain, log on to [AZ-204T00A-SEA-DEV](https://labclient.labondemand.com/Instructions/2fc1bf76-643c-4fbb-b613-5d9c9edf6fbd?rc=10) click [Ctrl+Alt+Delete](https://labclient.labondemand.com/Instructions/2fc1bf76-643c-4fbb-b613-5d9c9edf6fbd?rc=10) to activate the **Ctrl + Alt + Delete** sequence and bring up the logon page.
2. Sign in as Admin with the password Pa55w.rd

Any links like the one above will send Ctrl+Alt+Delete to the selected machine. This can also be done the **Commands** menu (lightning bolt) in the upper-left hand corner of the screen.

The **Lightning Bolt** in the upper left hand of the screen can also be used to **Copy /Paste** strings and sentences from the Instructions **into the VM** rather than typing them out.

* + To accomplish this simply Highligh\*\*t the sentence in the instructions you wish to type text. **CTRL + C** (Copy)
  + Click the lightning bolt icon, in the Dropdown menu select **Type Text** > **Type Clipboard Text**
  + Click into the Type Text window, **CTRL + V** (Paste) to Paste the sentence into the window
  + Select OK

This Lab supports **Redirect Clipboard** functionality in addition to TypeText.

You can use **Redirect Clipboard** to quickly input code blocks and other strings from the Instructions and elsewhere into the virtual machine using standard copy and paste from your local machine's Clipboard directly into the VM. You may need to click **Allow** in your browser to allow access to your local Clipboard.

Note that due to an issue within **Azure Cloudshell**, use your mouse: **Right-click + Paste** instead of **CTRL+V** when using **Cloudshell** inside the VM.

Prepare cloud shell for later use

Mount storage in Azure

1. Sign in to the Azure Portal https://portal.azure.com using the below credentials:

|  |  |
| --- | --- |
| Username | LabUser-23789257@cloudslice.onmicrosoft.com |
| Password | B#1d!hA9Bg |

1. In the toolbar at the top of the Azure portal, select the **Cloud Shell** icon.
2. In the Welcome to Azure Cloud Shell dialog, select **Bash**.
3. On the you have no storage mounted screen select **Show advanced settings**.
4. In the advanced settings screen, fill in the following fields, then click **Create Storage**:

|  |  |
| --- | --- |
| Resource Group | Use existing (**ContainerCompute-lod23789257**) |
| Cloud Shell Region | **East US** |
| Storage account (Create new) | cloudshell23789257 |
| File share (create new) | shellstorage |

1. After the cloud shell initializes and puts you at a text prompt, exit the shell.

Any potential issue with the resolver upgrade in October 2020 with pip version 20.3 Can be fixed with the following.

* 1. After opening a cloud shell for the first time in the lab, ensure you are on BASH
* 2. Type az --version. It should show 2.26.0 still.
* 3. Type pip install azure-cli and wait while it installs
* 4. Close cloud shell and start a new one
* 5. Type az --version again and confirm you are now on 2.26.1.

At the end of this lab, you can skip the **Clean Up** exercise directing you to remove the resources from your Subscription or Resource Group(s). The clean up is handled automatically, after ending your lab.

Download the lab files

Download Files

This will copy the latest lab files from GitHub and configure the directories as needed for the remainder of the lab exercises. You will see a 'Success!' message above, once the process is complete.

Lab Credentials:

* Username: LabUser-23789257@cloudslice.onmicrosoft.com
* Password: B#1d!hA9Bg

Sites used:

* https://portal.azure.com/#home
* https://shell.azure.com

All the resource groups you require in this lab have been created for you as part of the lab setup. You can safely ignore any steps that ask you to create a resource group. Please use or select a pre-existing resource group that has a similar name to the one you are asked to create. If no similarly named resource group exists, use any existing resource group.

For access to Azure, use the following credentials:

* Username: LabUser-23789257@cloudslice.onmicrosoft.com
* Password: B#1d!hA9Bg

You can see your available Resource Groups by checking the Resources tab above.

**For Exercise 3, Task 3, Step 5** you will need to change the size to **2vCPU and 4GB Memory**.

Lab 05: Deploy compute workloads by using images and containers

Microsoft Azure user interface

Given the dynamic nature of Microsoft cloud tools, you might experience Azure UI changes that occur after this training content's development. As a result, the lab instructions and lab steps might not align correctly.

Microsoft updates this training course when the community alerts us to needed changes. However, cloud updates occur frequently, so you might encounter UI changes before this training content updates. **If this occurs, adapt to the changes, and then work through them in the labs as needed.**

Instructions

Before you start

Sign in to the lab environment

Sign in to your Windows 10 virtual machine (VM) by using the following credentials:

* Username: **Admin**
* Password: **Pa55w.rd**

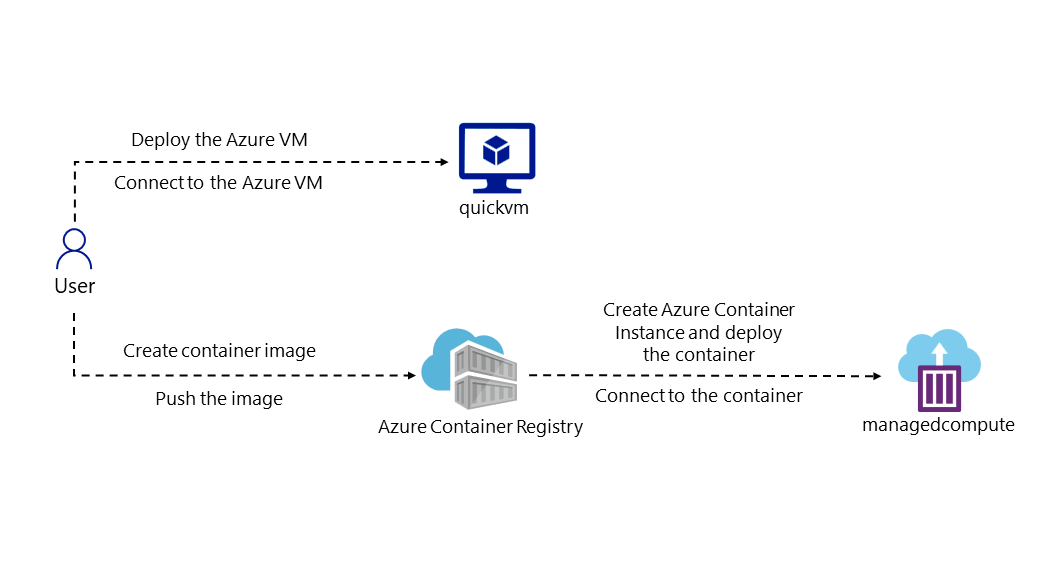
**Note**: Your instructor will provide instructions to connect to the virtual lab environment.

Review the installed applications

Find the taskbar on your Windows 10 desktop. The taskbar contains the icons for the applications that you'll use in this lab, including:

* Microsoft Edge
* File Explorer

Architecture diagram



Exercise 1: Create a VM by using the Azure Command-Line Interface (CLI)

Task 1: Open the Azure portal

1. On the taskbar, select the **Microsoft Edge** icon.
2. In the open browser window, navigate to the Azure portal https://portal.azure.com, and then sign in with the account you'll be using for this lab.

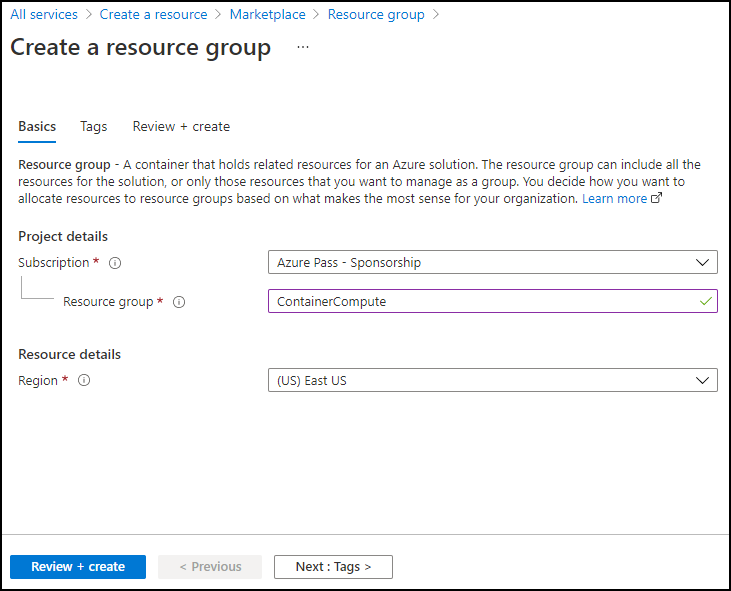
**Note**: If this is your first time signing in to the Azure portal, you'll be offered a tour of the portal. If you prefer to skip the tour, select **Get Started** to begin using the portal.

Task 2: Create a resource group

You can skip this task as the resource group has already been created for you.

1. On the Azure portal's navigation pane, use the **Search resources, services, and docs** text box to search for **Resource group**, and then in the list of results, select **Resource groups**.
2. On the **Resource groups** blade, select **Create**.
3. On the **Create a resource group** blade, on the **Basics** tab, perform the following actions, and then select **Review + create**:

| **Setting** | **Action** |
| --- | --- |
| **Subscription** drop-down list | Retain the default value. |
| **Resource group** text box | Enter **ContainerCompute-lod23789257** |
| **Region** drop-down list | Select **(US) East US** |

1. The following screenshot displays the configured settings on the **Create a resource group** blade.
2. 
3. On the **Review + create** tab, review the options that you selected during the previous steps.
4. Select **Create** to create the resource group by using your specified configuration.

**Note**: This has already been created in this **Cloud slice** lab. Please move on to the next task. All **Resource groups** have been pre created.

Task 3: Open Azure Cloud Shell

1. In the Azure portal, select the **Cloud Shell** icon Cloud Shell icon to open a new PowerShell session. If Cloud Shell defaults to a PowerShell session, select **PowerShell**, and then in the drop-down menu, select **Bash**.

**Note**: The **Cloud Shell** icon is represented by a greater than sign (>) and underscore character (\_).

**Note**: If this is the first time you're starting **Cloud Shell**, when prompted to select either **Bash** or **PowerShell**, select **Bash**. When you're presented with the **You have no storage mounted** message, select the subscription you're using in this lab, and then select **Create storage**.

1. At the **Cloud Shell** command prompt in the portal, run the following command to get the version of the Azure CLI tool:

az --version

Task 4: Use the Azure CLI commands

1. Run the following command to get a list of subgroups and commands at the root level of the CLI:

az --help

1. Run the following command to get a list of subgroups and commands for Azure Virtual Machines:

az vm --help

1. Run the following command to get a list of arguments and examples for the **Create Virtual Machine** command:

az vm create --help

1. Run the following command to create a new **virtual machine** with the following settings, be sure to record the password you are asked to create below you will need it later in the lab to access your virtual machine:
   * Resource group: **ContainerCompute-lod23789257**
   * Name: **quickvm**
   * Image: **Debian**
   * Admin-Username: **student**
   * Admin-Password: **<CreateYourPassword>**

Note: Replace **<CreateYourPassword>** in the command below with your own password.

az vm create --resource-group ContainerCompute-lod23789257 --name quickvm --image Debian --admin-username student --admin-password <CreateYourPassword>

**Note**: Wait for the VM to be created. After the process completes, the command will return a JavaScript Object Notation (JSON) file containing details about the machine.

1. Run the following command to get a more detailed JSON file that contains various metadata about the newly created VM:

az vm show --resource-group ContainerCompute-lod23789257 --name quickvm

1. Run the following command to list all the IP addresses associated with the VM:

az vm list-ip-addresses --resource-group ContainerCompute-lod23789257 --name quickvm

1. Run the following command to filter the output to only return the first IP address value:

az vm list-ip-addresses --resource-group ContainerCompute-lod23789257 --name quickvm --query '[].{ip:virtualMachine.network.publicIpAddresses[0].ipAddress}' --output tsv

1. Run the following command to store the results of the previous command in a new Bash shell variable named *ipAddress*:

ipAddress=$(az vm list-ip-addresses --resource-group ContainerCompute-lod23789257 --name quickvm --query '[].{ip:virtualMachine.network.publicIpAddresses[0].ipAddress}' --output tsv)

1. Run the following command to render the value of the Bash shell variable *ipAddress*:

echo $ipAddress

1. Run the following command to connect to the VM that you created previously in this lab, by using the Secure Shell (SSH) tool and the IP address stored in the Bash shell variable *ipAddress*:

ssh student@$ipAddress

1. The SSH tool informs you that the authenticity of the host can’t be established and then asks if you want to continue connecting. Enter **yes**, and then select Enter to continue connecting to the VM.
2. The SSH tool then asks for a password. Enter the password you created earlier, and then select Enter to authenticate with the VM.
3. After connecting to the VM by using SSH, run the following command to get metadata describing the Linux VM:

uname -a

1. Use the **exit** command to end your SSH session:

exit

1. Close the **Cloud Shell** pane in the portal.

Review

In this exercise, you used Cloud Shell to create a VM as part of an automated script.

Exercise 2: Create a Docker container image and deploy it to Azure Container Registry

Task 1: Open the Cloud Shell and editor

1. On the Azure portal's navigation pane, select the **Cloud Shell** icon to open a new shell instance.

**Note**: Wait for Cloud Shell to finish connecting to an instance before moving on with the lab.

1. At the **Cloud Shell** command prompt in the portal, run the following command to move from the root directory to the **~/clouddrive** directory:

cd ~/clouddrive

1. Run the following command to create a new directory named **ipcheck** in the **~/clouddrive** directory:

mkdir ipcheck

1. Run the following command to change the active directory from **~/clouddrive** to **~/clouddrive/ipcheck**:

cd ~/clouddrive/ipcheck

1. Run the following command to create a new .NET console application in the current directory:

dotnet new console --output . --name ipcheck

1. Run the following command to create a new file in the **~/clouddrive/ipcheck** directory named **Dockerfile**:

touch Dockerfile

1. Run the following command to open the embedded graphical editor in the context of the current directory:

code .

Task 2: Create and test a .NET application

1. In the graphical editor, on the **FILES** pane, select the **Program.cs** file to open it in the editor.
2. Delete the entire contents of the **Program.cs** file.
3. Copy and paste the following code into the **Program.cs** file:

csharp

public class Program

{

public static void Main(string[] args)

{

// Check if network is available

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

{

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

// Get host entry for current hostname

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

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

// Iterate over each IP address and render their values

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

{

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

}

}

else

{

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

}

}

}

1. Save the **Program.cs** file by using the menu in the graphical editor or the Ctrl+S keyboard shortcut. Don't close the graphical editor.
2. Back at the command prompt, run the following command to run the application:

dotnet run

1. Review the results of the run. At least one IP address should be listed for the Cloud Shell instance.
2. In the graphical editor, on the **FILES** pane of the editor, select the **Dockerfile** file to open it in the editor.
3. Copy and paste the following code into the **Dockerfile** file:
4. # Start using the .NET Core 3.1 SDK container image
5. FROM mcr.microsoft.com/dotnet/sdk:3.1-alpine AS build
6. # Change current working directory
7. WORKDIR /app
8. # Copy existing files from host machine
9. COPY . ./
10. # Publish application to the "out" folder
11. RUN dotnet publish --configuration Release --output out
12. # Start container by running application DLL

ENTRYPOINT ["dotnet", "out/ipcheck.dll"]

1. Save the **Dockerfile** file by using the menu in the graphical editor or by using the Ctrl+S keyboard shortcut.
2. Leave the Cloud Shell open for the next task.

Task 3: Create a Container Registry resource

1. At the **Cloud Shell** command prompt in the portal, run the following command to create a variable with a unique value for the Container Registry resource:

bash

registryName=conregistry$RANDOM

1. At the **Cloud Shell** command prompt in the portal, run the following command to verify the name created in the previous step is available:

bash

az acr check-name --name $registryName

If the results show the name is available, continue to the next step. If the name is not available then re-run the command in the previous step and verify availability again.

1. At the **Cloud Shell** command prompt in the portal, run the following command to create a Container Registry resource:

bash

az acr create --resource-group ContainerCompute-lod23789257 --name $registryName --sku Basic

**Note**: Wait for the creation task to complete before you continue with this lab.

Task 4: Store Container Registry metadata

1. At the **Cloud Shell** command prompt in the portal, run the following command to get a list of all container registries in your subscription:

az acr list

1. Run the following command, ensuring you see the name of your registry as output. If you see no output other than '[]', wait a minute and try running the command again.

az acr list --query "max\_by([], &creationDate).name" --output tsv

1. Run the following command:

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

1. Run the following command:

echo $acrName

Task 5: Deploy a Docker container image to Container Registry

1. Run the following command to change the active directory from **~/** to **~/clouddrive/ipcheck**:

cd ~/clouddrive/ipcheck

1. Run the following command to get the contents of the current directory:

dir

1. Run the following command to upload the source code to your container registry and build the container image as a Container Registry task:

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

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

1. Close the **Cloud Shell** pane in the portal.

Task 6: Validate your container image in Container Registry

1. On the Azure portal's **navigation** pane, select the **Resource groups** link.
2. From the **Resource groups** blade, select the **ContainerCompute-lod23789257** resource group that you created previously in this lab.
3. From the **ContainerCompute-lod23789257** blade, select the container registry that you created previously in this lab.
4. From the **Container Registry** blade, in the **Services** section, select the **Repositories** link.
5. In the **Repositories** section, select the **ipcheck** container image repository, and then select the **latest** tag.
6. Review the metadata for the version of your container image with the **latest** tag.

**Note**: You can also select the **Run ID** link to find metadata about the build task.

Review

In this exercise, you created a .NET console application to display a machine’s current IP address. You then added the **Dockerfile** file to the application so that it could be converted into a Docker container image. Finally, you deployed the container image to Container Registry.

Exercise 3: Deploy an Azure container instance

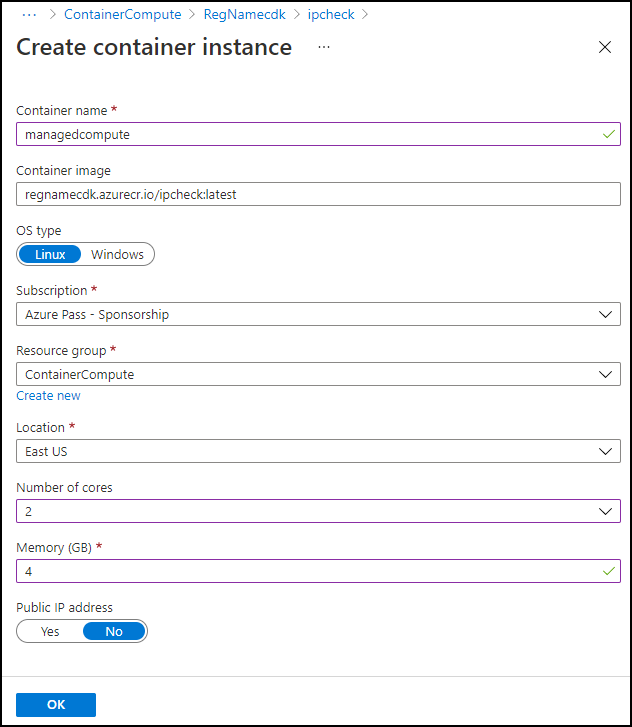
Task 1: Enable the admin user in Container Registry

1. On the Azure portal's **navigation** pane, select the **Resource groups** link.
2. On the **Resource groups** blade, select the **ContainerCompute-lod23789257** resource group that you created previously in this lab.
3. On the **ContainerCompute-lod23789257** blade, select the container registry that you created previously in this lab, and then select **Update**.
4. On the **Update container registry** blade, in the **Admin user** section, select **Enable**.
5. Select **Save**, and then close the **Update container registry** blade.

Task 2: Automatically deploy a container image to an Azure container instance

1. On the **Container Registry** blade, in the **Services** section, select the **Repositories** link.
2. In the **Repositories** section, select the **ipcheck** container image repository.
3. On the **Repository** blade, select the ellipsis menu associated with the **latest** tag entry, and then select **Run instance**.
4. On the **Create container instance** blade, perform the following actions, and then select **OK**:

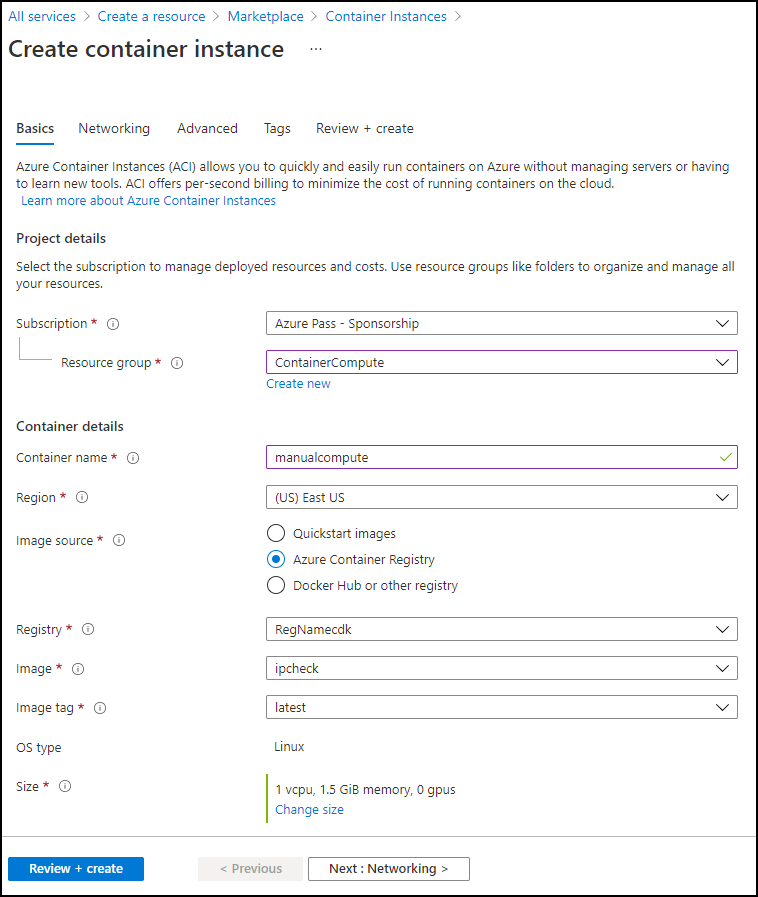
| **Setting** | **Action** |
| --- | --- |
| **Container name** text box | Enter **managedcompute** |
| **Container image** text box | Retain the default value. |
| **OS type** section | Select **Linux** |
| **Subscription** text box | Retain the default value. |
| **Resource group** drop-down list | Select **ContainerCompute-lod23789257** |
| **Location** drop-down list | Select **East US** |
| **Number of cores** drop-down list | Select **2** |
| **Memory (GB)** text box | Enter **4** |
| **Public IP address** section | Select **No** |

1. The following screenshot displays the configured settings on the **Create container instance** blade.
2. 
3. **Note**: Wait for the container instance to be created before you continue with this lab.

Task 3: Manually deploy a container image to Container Instances

1. On the Azure portal's **navigation** pane, select the **Create a resource** link.
2. On the **Create a resource** blade, in the **Search services and marketplace** text box, enter **container instances**, and then select Enter.
3. On the **Marketplace** search results blade, select the **Container Instances** result.
4. On the **Container Instances** blade, select **Create**.
5. On the **Create Container Instance** blade, on the **Basics** tab, perform the following actions, and then select **Review + create**:

| **Setting** | **Action** |
| --- | --- |
| **Subscription** drop-down list | Retain the default value |
| **Resource group** drop-down list | Select **ContainerCompute-lod23789257** |
| **Container name** text box | Enter **manualcompute** |
| **Region** drop-down list | Select **(US) East US** |
| **Image source** section | Select **Azure Container Registry** |
| **Registry** drop-down list | Select the **Azure Container Registry** resource that you created previously in this lab |
| **Image** drop-down list | Select **ipcheck** |
| **Image tag** drop-down list | Select **latest** |

1. The following screenshot displays the configured settings on the **Create container instance** blade, but does not show the needed size of **2vCPU and 4GB Memory**.
2. **For Exercise 3, Task 3, Step 5** you will need to change the size to **2vCPU and 4GB Memory**.
3. 
4. From the **Review + create** tab, review the selected options.
5. Select **Create** to create the container instance by using your specified configuration.

**Note**: Wait for the container instance to be created before you continue with this lab.

Task 4: Validate that the container instance ran successfully

1. On the Azure portal's navigation pane, select the **Resource groups** link.
2. On the **Resource groups** blade, select the **ContainerCompute-lod23789257** resource group that you created previously in this lab.
3. On the **ContainerCompute-lod23789257** blade, select the **manualcompute** container instance that you created previously in this lab.
4. On the **Container Instances** blade, in the **Settings** section, select the **Containers** link.
5. In the **Containers** section, review the list of **Events**.
6. Select the **Logs** tab, and then review the text logs from the container instance.

**Note**: You can also optionally find the **Events** and **Logs** from the **managedcompute** container instance.

**Note**: After the application finishes running, the container terminates because it has completed its work. For the manually created container instance, you indicated that a successful exit was acceptable, so the container ran once. The automatically created instance didn't offer this option, and it assumes the container should always be running, so you'll notice repeated restarts of the container.

Review

In this exercise, you used multiple methods to deploy a container image to an Azure container instance. By using the manual method, you were able to customize the deployment further and to run task-based applications as part of a container run.

Exercise 4: Clean up your subscription

You can skip the **Clean Up** exercise directing you to remove the resources from your Subscription or Resource Group(s). The clean up is handled automatically, after ending your lab.

Task 1: Open Azure Cloud Shell and list resource groups

1. In the Azure portal, select the **Cloud Shell** icon Cloud Shell icon to open a new Bash session. If Cloud Shell defaults to a PowerShell session, select **PowerShell**, and then in the drop-down menu, select **Bash**.

**Note**: If this is the first time you're starting **Cloud Shell**, when prompted to select either **Bash** or **PowerShell**, select **PowerShell**. When you're presented with the **You have no storage mounted** message, select the subscription you're using in this lab, and then select **Create storage**.

Task 2: Delete resource groups

1. On the **Cloud Shell** pane, run the following command to delete the **ContainerCompute-lod23789257** resource group:

az group delete --name ContainerCompute-lod23789257 --no-wait --yes

**Note**: The command executes asynchronously (as determined by the *--no-wait* parameter), so while you'll be able to run another Azure CLI command immediately afterwards within the same Bash session, it'll take a few minutes before the resource groups are actually removed.

1. Close the **Cloud Shell** pane in the portal.

Task 3: Close the active applications

* Close the currently running Microsoft Edge application.

Review

In this exercise, you cleaned up your subscription by removing the resource groups used in this lab.

Congratulations!

You have successfully completed this **Lab** press **End** to end your lab.

0% Tasks Complete

End

Live Chat