AZ204 LABWORKS

Lab 05: Deploy compute workloads by using images and containers

Kaushik Ganguli

https://github.com/kaushikganguli

Table of Contents

[Architecture diagram 1](#_Toc92737584)

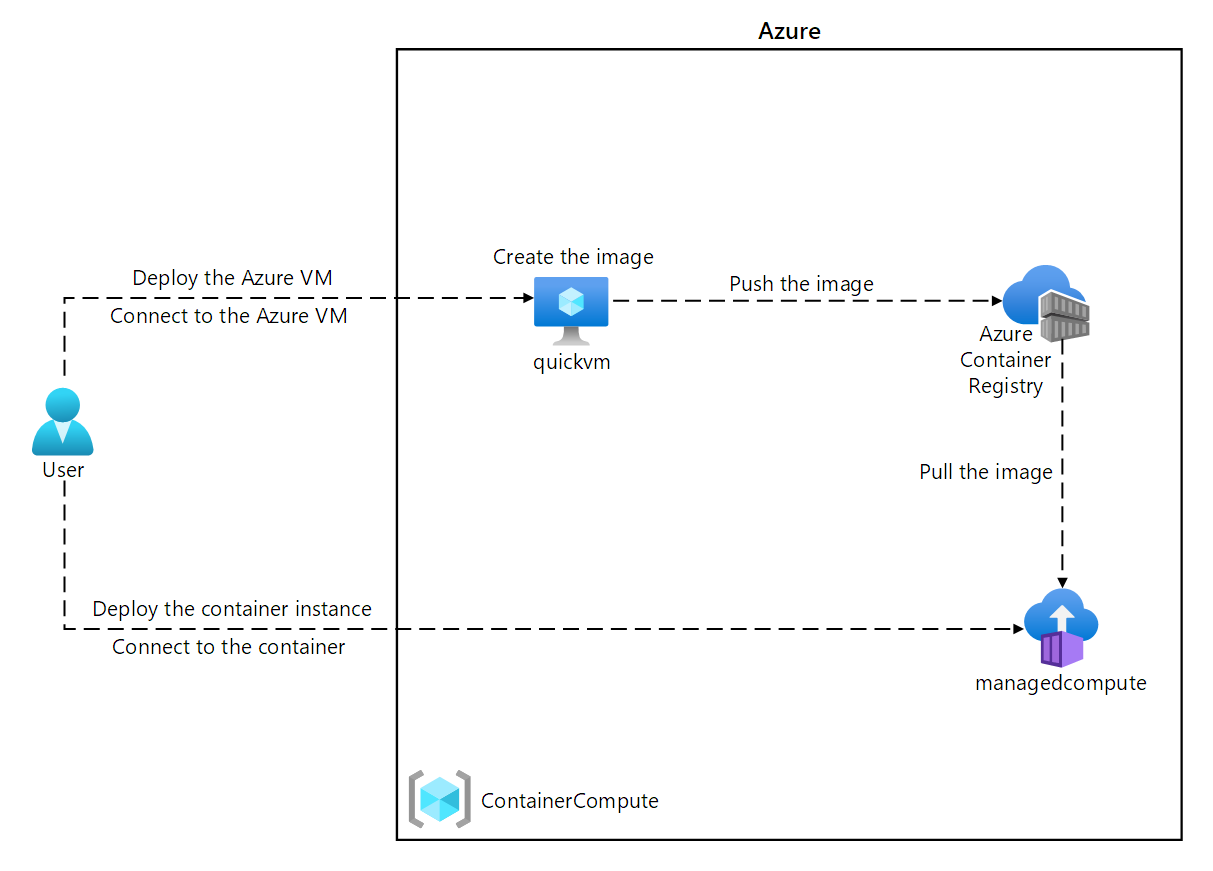
[Exercise 1: Create a VM by using the Azure Command-Line Interface (CLI) 1](#_Toc92737585)

[Exercise 2: Create a Docker container image and deploy it to Azure Container Registry 9](#_Toc92737586)

[Exercise 3: Deploy an Azure container instance 22](#_Toc92737587)

[Exercise 4: Clean up your subscription 29](#_Toc92737588)

Architecture diagram

[](https://microsoftlearning.github.io/AZ-204-DevelopingSolutionsforMicrosoftAzure/Instructions/Labs/media/Lab05-Diagram.png)

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](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

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**. |
| **Region** drop-down list | Select **(US) East US**. |

1. The following screenshot displays the configured settings on the **Create a resource group** blade.

Graphical user interface, text, application, email

Description automatically generated

1. On the **Review + create** tab, review the options that you selected during the previous steps.
2. Select **Create** to create the resource group by using your specified configuration.

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

Task 3: Open Azure Cloud Shell

1. In the Azure portal, select the **Cloud Shell** icon [Cloud Shell icon](https://microsoftlearning.github.io/AZ-204-DevelopingSolutionsforMicrosoftAzure/Instructions/Labs/media/az204_lab_CloudShell.png) 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:

Text

Description automatically generated

CodeCopy

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:

CodeCopy

az --help

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

CodeCopy

az vm --help

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

CodeCopy

az vm create --help

1. Run the following command to create a new **virtual machine** with the following settings:
   * Resource group: **ContainerCompute**
   * Name: **quickvm**
   * Image: **Debian**
   * Admin-Username: **student**
   * Admin-Password: **StudentPa55w.rd**

CodeCopy

az vm create --resource-group ContainerCompute --name quickvm --image Debian --admin-username student --admin-password StudentPa55w.rd

**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.

A picture containing graphical user interface

Description automatically generated

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

CodeCopy

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

Text

Description automatically generated

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

CodeCopy

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

Text

Description automatically generated

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

CodeCopy

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

Text

Description automatically generated

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

CodeCopy

ipAddress=$(az vm list-ip-addresses --resource-group ContainerCompute --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*:

CodeCopy

echo $ipAddress

Text

Description automatically generated

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*:

CodeCopy

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 **StudentPa55w.rd**, and then select Enter to authenticate with the VM.

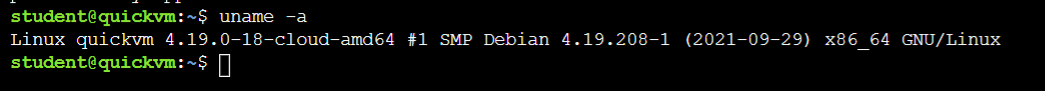
Graphical user interface, text

Description automatically generated

1. After connecting to the VM by using SSH, run the following command to get metadata describing the Linux VM:

CodeCopy

uname -a



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

CodeCopy

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:

CodeCopy

cd ~/clouddrive

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

CodeCopy

mkdir ipcheck

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

CodeCopy

cd ~/clouddrive/ipcheck

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

CodeCopy

dotnet new console --output . --name ipcheck

Text

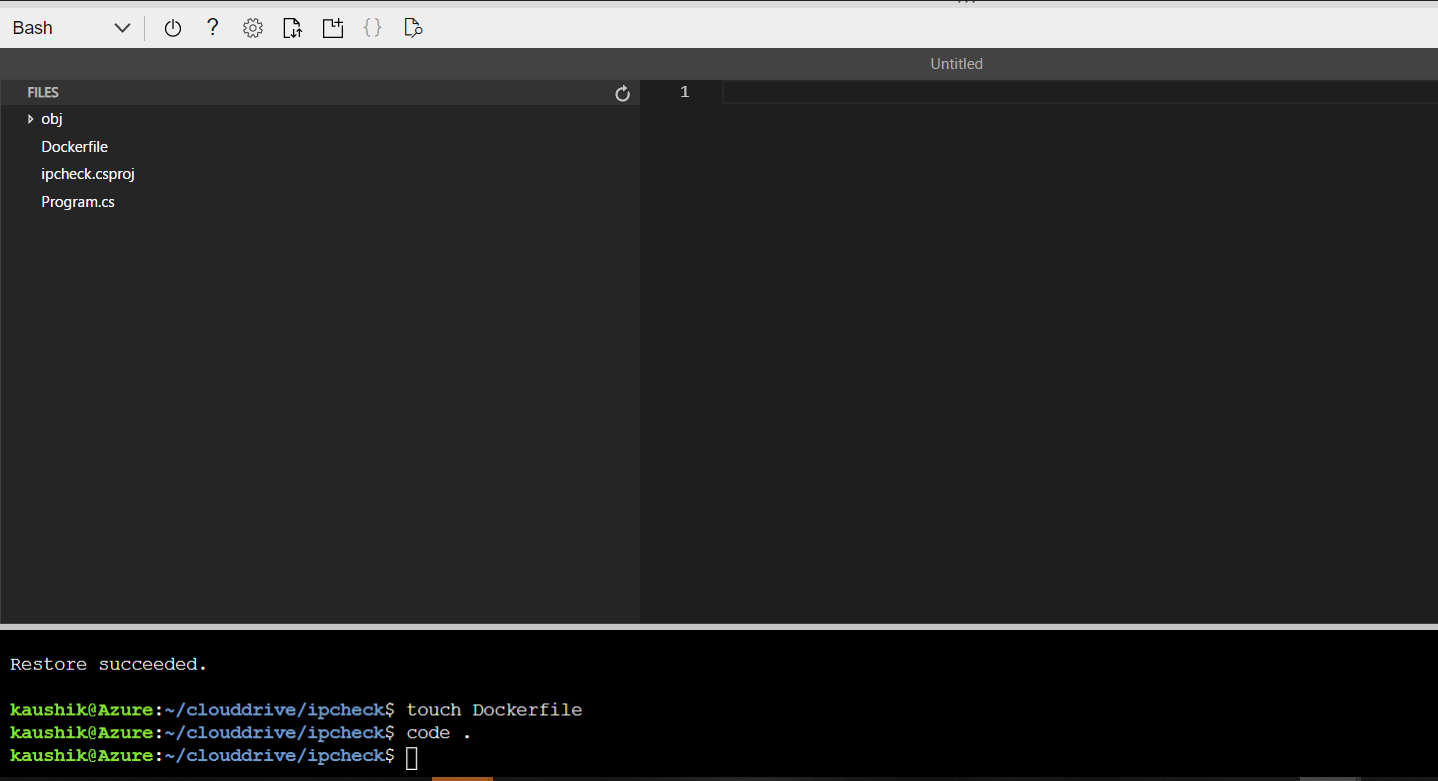
Description automatically generated

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

CodeCopy

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:

C#Copy

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");

}

}

}

Graphical user interface, text

Description automatically generated

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:

CodeCopy

dotnet run

Text

Description automatically generated

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.

Text

Description automatically generated

1. Copy and paste the following code into the **Dockerfile** file:

CodeCopy

# Start using the .NET Core 3.1 SDK container image

FROM mcr.microsoft.com/dotnet/sdk:3.1-alpine AS build

# Change current working directory

WORKDIR /app

# Copy existing files from host machine

COPY . ./

# Publish application to the "out" folder

RUN dotnet publish --configuration Release --output out

# Start container by running application DLL

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

Text

Description automatically generated

1. Save the **Dockerfile** file by using the menu in the graphical editor or by using the Ctrl+S keyboard shortcut.
2. Close the **Cloud Shell** pane in the portal.

Task 3: Create a Container Registry resource

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 Registry**, and then select Enter.
3. On the **Marketplace** search results blade, select the **Container Registry** result.

Graphical user interface, text, application, chat or text message

Description automatically generated

1. On the **Container Registry** blade, select **Create**.
2. On the **Create container registry** 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 the existing **ContainerCompute** option. |
| **Registry name** text box | Enter a globally unique name. Azure will automatically check the name for uniqueness and inform you if you’re required to choose a different name. |
| **Location** drop-down list | Select **East US**. |
| **Availability zones** check box | Not selected. |
| **SKU** drop-down list | Select **Basic**. |

1. The following screenshot displays the configured settings on the **Create container registry** blade.

Graphical user interface, application

Description automatically generated

1. On the **Review + Create** tab, review the options that you selected in the previous steps.
2. Select **Create** to create the container registry by using your specified configuration.

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

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

1. In the Azure portal, select the **Cloud Shell** icon to open a new shell instance.

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

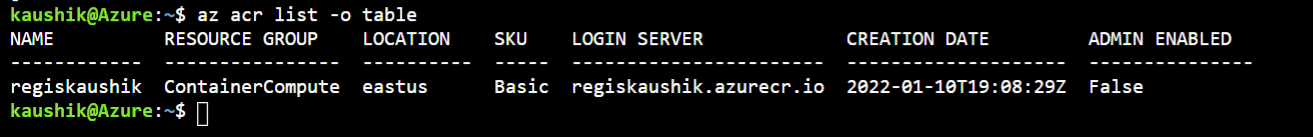
1. At the **Cloud Shell** command prompt in the portal, run the following command to get a list of all container registries in your subscription:

CodeCopy

az acr list

Text

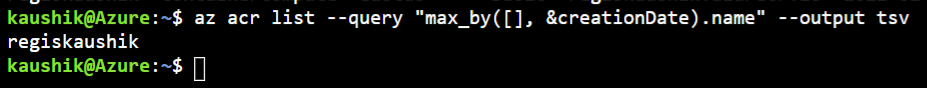
Description automatically generated



1. Run the following command:

CodeCopy

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



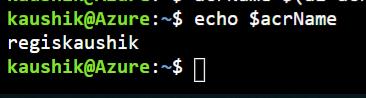
1. Run the following command:

CodeCopy

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

1. Run the following command:

CodeCopy



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**:

CodeCopy

cd ~/clouddrive/ipcheck

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

CodeCopy

dir

Text

Description automatically generated

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

CodeCopy

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

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

Text

Description automatically generated

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** resource group that you created previously in this lab.
3. From the **ContainerCompute** blade, select the container registry that you created previously in this lab.

Graphical user interface, text, application, email

Description automatically generated

1. From the **Container Registry** blade, in the **Services** section, select the **Repositories** link.

Graphical user interface, text, application, chat or text message

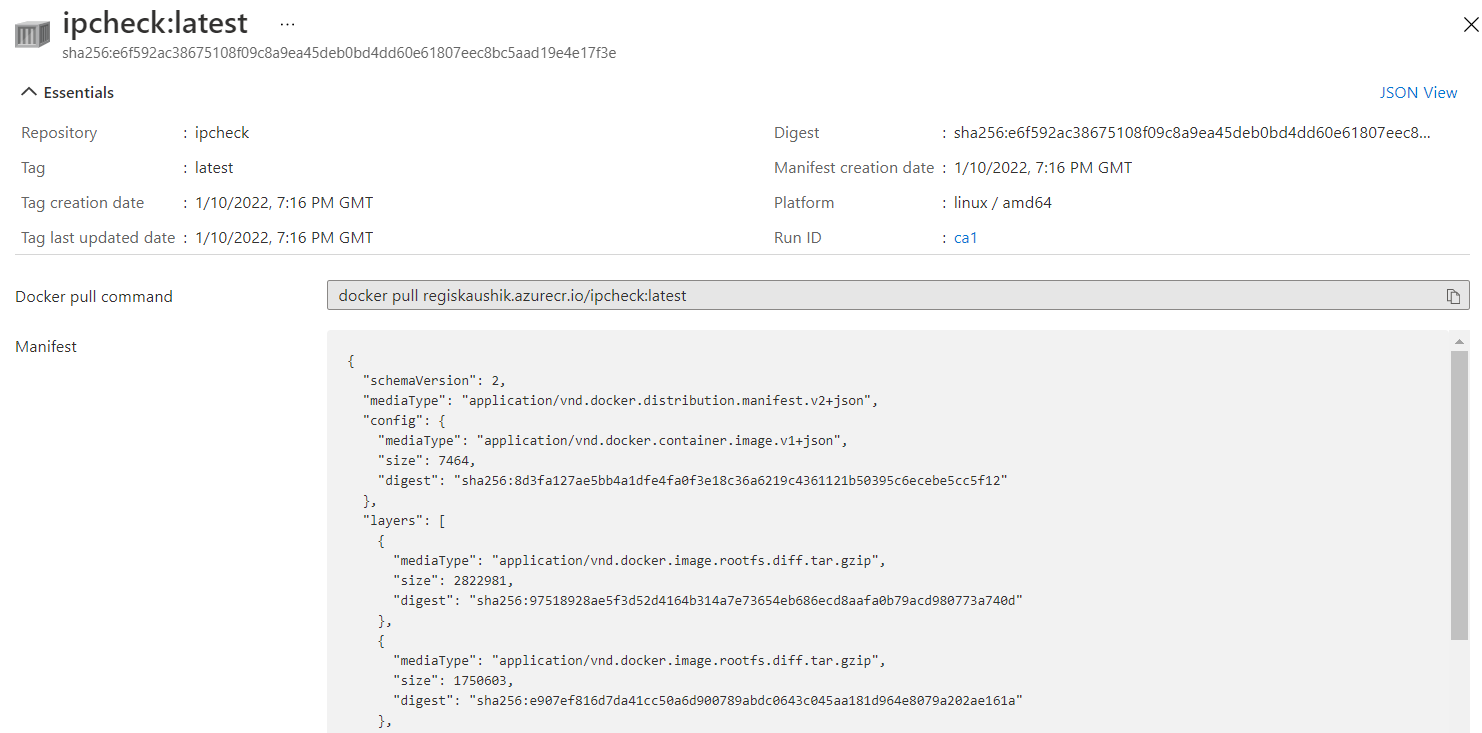
Description automatically generated

1. In the **Repositories** section, select the **ipcheck** container image repository, and then select the **latest** tag.

Graphical user interface, application

Description automatically generated

1. Review the metadata for the version of your container image with the **latest** tag.



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** resource group that you created previously in this lab.
3. On the **ContainerCompute** blade, select the container registry that you created previously in this lab, and then select **Update**.

A screenshot of a computer

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. On the **Update container registry** blade, in the **Admin user** section, select **Enable**.

Graphical user interface, text, application, email

Description automatically generated

1. 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.

Graphical user interface, text, application, chat or text message

Description automatically generated

1. In the **Repositories** section, select the **ipcheck** container image repository.
2. On the **Repository** blade, select the ellipsis menu associated with the **latest** tag entry, and then select **Run instance**.

Graphical user interface, application, email

Description automatically generated

1. 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**. |
| **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.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

1. **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.

Graphical user interface, text, application, chat or text message

Description automatically generated

1. On the **Create a resource** blade, in the **Search services and marketplace** text box, enter **container instances**, and then select Enter.
2. On the **Marketplace** search results blade, select the **Container Instances** result.
3. On the **Container Instances** blade, select **Create**.
4. 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**. |
| **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.

Graphical user interface, text, application, email

Description automatically generated

1. From the **Review + create** tab, review the selected options.
2. 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** resource group that you created previously in this lab.
3. On the **ContainerCompute** blade, select the **manualcompute** container instance that you created previously in this lab.

Graphical user interface, text, application, email

Description automatically generated

1. On the **Container Instances** blade, in the **Settings** section, select the **Containers** link.

Graphical user interface, text, application

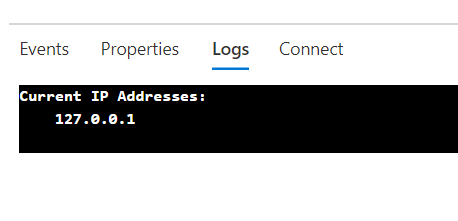
Description automatically generated

1. In the **Containers** section, review the list of **Events**.

Graphical user interface, application

Description automatically generated

1. 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

Task 1: Open Azure Cloud Shell and list resource groups

1. In the Azure portal, select the **Cloud Shell** icon [Cloud Shell icon](https://microsoftlearning.github.io/AZ-204-DevelopingSolutionsforMicrosoftAzure/Instructions/Labs/media/az204_lab_CloudShell.png) 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** resource group:

CodeCopy

az group delete --name ContainerCompute --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.