AZ-204

# Developing solutions for Microsoft Azure

## Lab 05

Deploying compute workloads by using images and containers

## Table of Contents

# 1 Pre-requisites

## 1.1 Sign in to the lab virtual machine

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

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

**Note**: Instructions to connect to the virtual lab environment will be provided by your instructor.

## 1.2 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:

- Microsoft Edge
- File Explorer
- Azure CLI
- Windows PowerShell

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

## 3.1 Task 1: Open the Azure portal

1. On the taskbar, select the **Microsoft Edge** icon.
2. In the open browser window, go to the Azure portal (https://portal.azure.com).
3. Enter the email address for your Microsoft account, and then select **Next**.
4. Enter the password for your Microsoft account, and then select **Sign in**.

   **Note**: If this is your first time signing in to the Azure portal, you will be offered a tour of the portal. Select **Get Started** to begin using the portal.

## 3.2 Task 2: Create a resource group

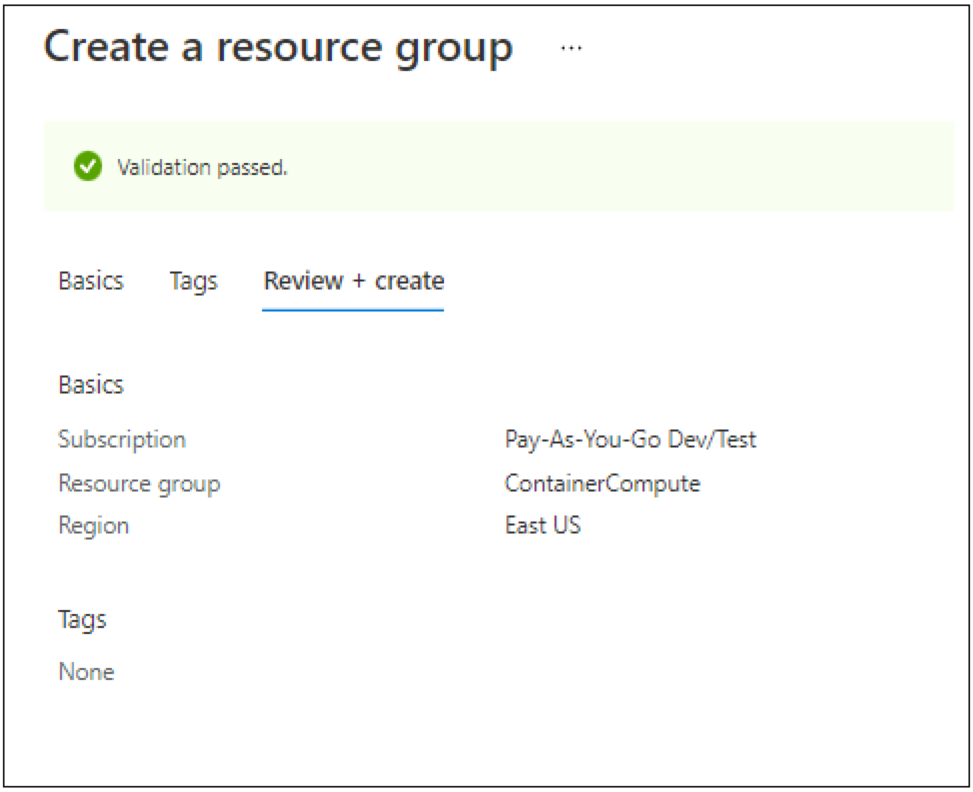
1. In the Azure portal's navigation pane, select the **Create a resource** link.

   **Note**: If you can't find the **Create a resource** link, the **Create a resource** icon is a plus sign (+) character from the portal.

2. From the **Create a resource** blade, find the **Search services and marketplace** text box.
3. In the search box, enter the text **Resource Group**, and then select Enter.
4. From the **Marketplace** search results blade, select the **Resource group** result.
5. From the **Resource group** blade, select **Create**.
6. From the additional **Resource group** blade, find the tabs from the blade, such as **Basics**.

   **Note**: Each tab represents a step in the workflow to create a new resource group. You can select **Review + Create** at any time to skip the remaining tabs.

7. From the **Basics** tab, perform the following actions:
   1. Leave the **Subscription** text box set to its default value.
   2. In the **Resource group** text box, enter the value **ContainerCompute**.
   3. In the **Region** drop-down list, select the **(US) East US** location.
   4. Select **Review + Create**.
8. From the **Review + Create** tab, review the options that you selected during the previous steps.



Create a resource group

✓ Validation passed.

Basics

| | |
|---|---|
| Subscription | Pay-As-You-Go Dev/Test |
| Resource group | ContainerCompute |
| Region | East US |

Tags

None

9. 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.

## 3.3 Task 3: Open Azure Cloud Shell

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

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

2. If this is your first time opening Cloud Shell using your subscription, you can use the **Welcome to Azure Cloud Shell Wizard** to configure Cloud Shell for first-time usage. Perform the following actions in the wizard:
   1. A dialog box prompts you to configure the shell. Select **Bash**, review the selected subscription, and then select **Create storage**.

   **Note**: Wait for Cloud Shell to finish its initial setup procedures before moving forward with the lab. If you don’t notice the **Cloud Shell** configuration options, this is most likely because you’re using an existing subscription with this course’s labs. The labs are written with the presumption that you’re using a new subscription.

3. At the **Cloud Shell** command prompt in the portal, enter the following command, and then select Enter to get the version of the Azure CLI tool:

```
az --version
```

## 3.4 Task 4: Use the Azure CLI commands

1. Enter the following command, and then select Enter to get a list of subgroups and commands at the root level of the CLI:

```
az --help
```

2. Enter the following command, and then select Enter to get a list of subgroups and commands for Azure Virtual Machines:

```
az vm --help
```

3. Enter the following command, and then select Enter to get a list of arguments and examples for the **Create Virtual Machine** command:

```
az vm create --help
```

4. Enter the following command, and then select Enter to create a new **virtual machine** with the following settings:

    - Resource group: **ContainerCompute**
    - Name: **quickvm**
    - Image: **Debian**
    - Username: **Student**
    - Password: **StudentPa55w.rd**

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

**Note**: Wait for the VM creation process to complete. After the process completes, the command will return a JSON file containing details about the machine.

5. Enter the following command, and then select Enter to get a more detailed JavaScript Object Notation (JSON) file that contains various metadata about the newly created VM:

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

6. Enter the following command, and then select Enter to list all the IP addresses associated with the VM:

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

7. Enter the following command, and then select Enter to filter the output to only return the first IP address value:

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

Enter the following command, and then select Enter 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 --name quickvm --query
'[].{ip:virtualMachine.network.publicIpAddresses[0].ipAd
dress}' --output tsv)
```

8. Enter the following command, and then select Enter to render the value of the Bash shell variable *ipAddress*:

```
echo $ipAddress
```

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

```
ssh student@$ipAddress
```

The SSH tool will first inform you that the authenticity of the host can't be established and then ask if you want to continue connecting. Enter **yes**, and then select Enter to continue connecting to the VM.

The SSH tool will then ask you for a password. Enter **StudentPa55w.rd**, and then select Enter to authenticate with the VM.

10. After connecting to the VM by using SSH, enter the following command, and then select Enter to get metadata describing the Linux VM:

```
uname -a
```

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

```
exit
```

14. Close the Cloud Shell pane in the portal.

## 3.5 Review

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

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

## 4.1 Task 1: Open the Cloud Shell and editor

1. In 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.

2. At the **Cloud Shell** command prompt in the portal, enter the following command, and then select Enter to move from the root directory to the **~/clouddrive** directory:

   ```
   cd ~/clouddrive
   ```

3. Enter the following command, and then select Enter to create a new directory named **ipcheck** in the ~/**clouddrive** directory:

   ```
   mkdir ipcheck
   ```

4. Enter the following command, and then select Enter to change the active directory from ~/**clouddrive** to ~/**clouddrive**/**ipcheck**:

   ```
   cd ~/clouddrive/ipcheck
   ```

Enter the following command, and then select Enter to create a new .NET console application in the current directory:

```
dotnet new console --output . --name ipcheck
```

Enter the following command, and then select Enter to create a new file in the **~/clouddrive/ipcheck** directory named **Dockerfile**:

```
touch Dockerfile
```

Enter the following command, and then select Enter to open the embedded graphical editor in the context of the current directory:

```
code .
```

## 4.2 Task 2: Create and test a .NET application

1. In the graphical editor, find the FILES pane, and then open 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:

```
using System;

namespace ipcheck
{
    class Program
    {
        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");
        }

        }
    }
}
```

4.
5. 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.
6. Back at the command prompt, enter the following command, and then select Enter to run the application:

```
dotnet run
```

Find the results of the run. At least one IP address should be listed for the Cloud Shell instance.

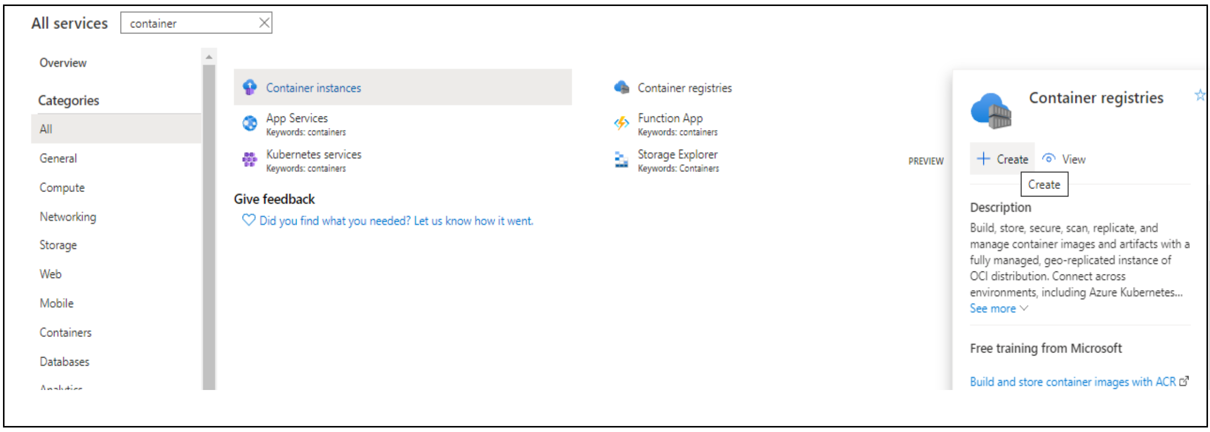
7. In the graphical editor, find the FILES pane of the editor, and then open the **Dockerfile** file to open it in the editor.
8. Copy and paste the following code into the **Dockerfile** file:

```
# 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"]
8.
```

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

## 4.3 Task 3: Create a Container Registry resource

1. In the Azure portal's navigation pane, select the **Create a resource** link.
2. From the **Create a resource** blade, find the **Search services and marketplace** text box.
3. In the search box, enter **Container Registry**, and then select Enter.
4. From the **Marketplace** search results blade, select the **Container Registry** result.
5. From the **Container Registry** blade, select **Create**.

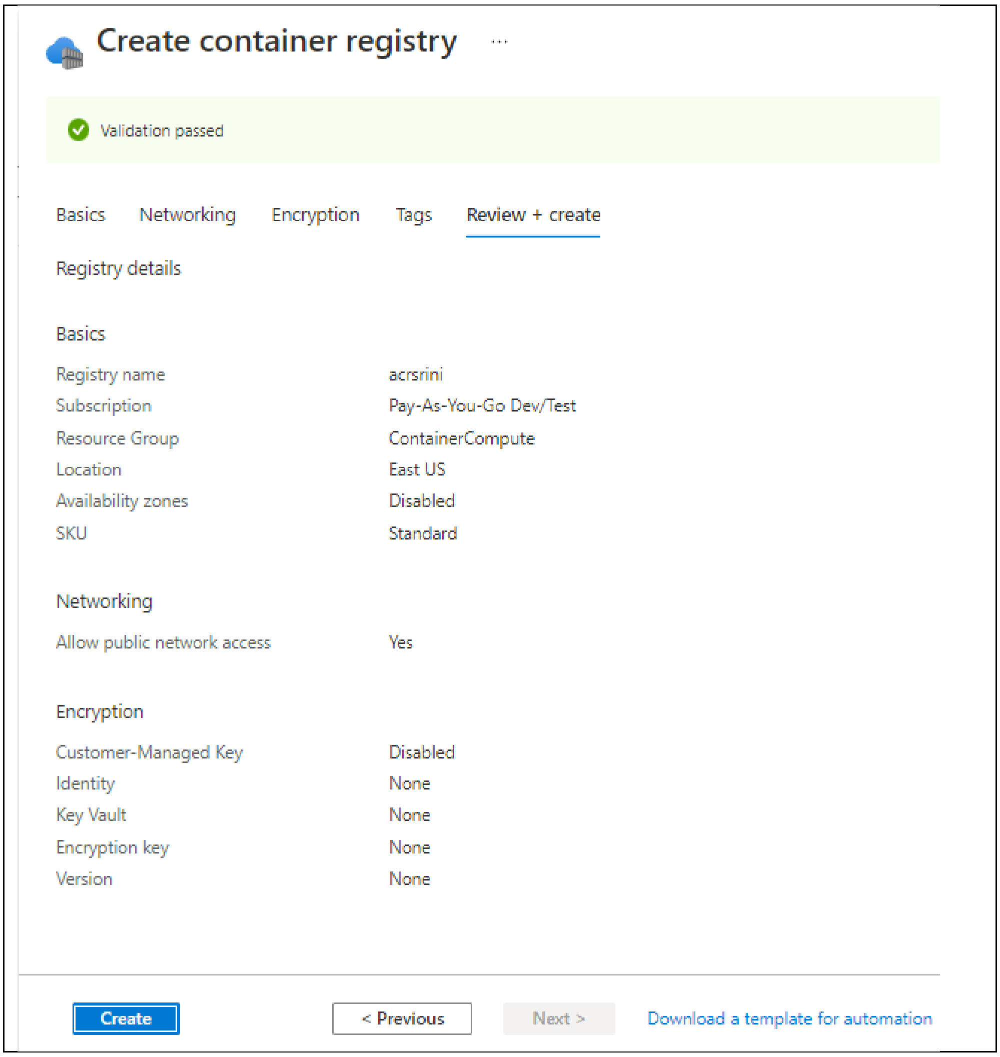


Build, store, secure, scan, replicate, and manage container images and artifacts with a fully managed, geo-replicated instance of OCI distribution. Connect across environments, including Azure Kubernetes...

6. From the **Create container registry** blade, perform the following actions:
    1. In the **Registry name** text box, give your registry a globally unique name.

       **Note**: The blade will automatically check the name for uniqueness and inform you if you're required to choose a different name.

    2. Leave the **Subscription** text box set to its default value.
    3. In the **Resource group** drop-down list, select the existing **ContainerCompute** option.
    4. In the **Location** text box, select **East US**.
    5. In the **SKU** drop-down list, select **Basic**.
    6. Select **Review + Create**.
7. From the **Review + Create** tab, review the options that you selected in the previous steps.



Create container registry

✓ Validation passed

Registry details

Basics

| | |
|---|---|
| Registry name | acrsrini |
| Subscription | Pay-As-You-Go Dev/Test |
| Resource Group | ContainerCompute |
| Location | East US |
| Availability zones | Disabled |
| SKU | Standard |

Networking

| | |
|---|---|
| Allow public network access | Yes |

Encryption

| | |
|---|---|
| Customer-Managed Key | Disabled |
| Identity | None |
| Key Vault | None |
| Encryption key | None |
| Version | None |

8. Select **Create** to create the container registry by using your specified configuration.

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

## 4.4 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.

2. At the **Cloud Shell** command prompt in the portal, enter the following command, and then select Enter to get a list of all container registries in your subscription:

```
az acr list
```

3. Enter the following command, and then select Enter:

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

4. Enter the following command, and then select Enter:

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

5. Enter the following command, and then select Enter:

```
echo $acrName
```

## 4.5 Task 5: Deploy a Docker container image to Container Registry

1. Enter the following command, and then select Enter to change the active directory from ~/ to **~/clouddrive/ipcheck**:

```
cd ~/clouddrive/ipcheck
```

2. Enter the following command, and then select Enter to get the contents of the current directory:

```
dir
```

3. Enter the following command, and then select Enter to upload the source code to your container registry and build the container image as a Container Registry task:

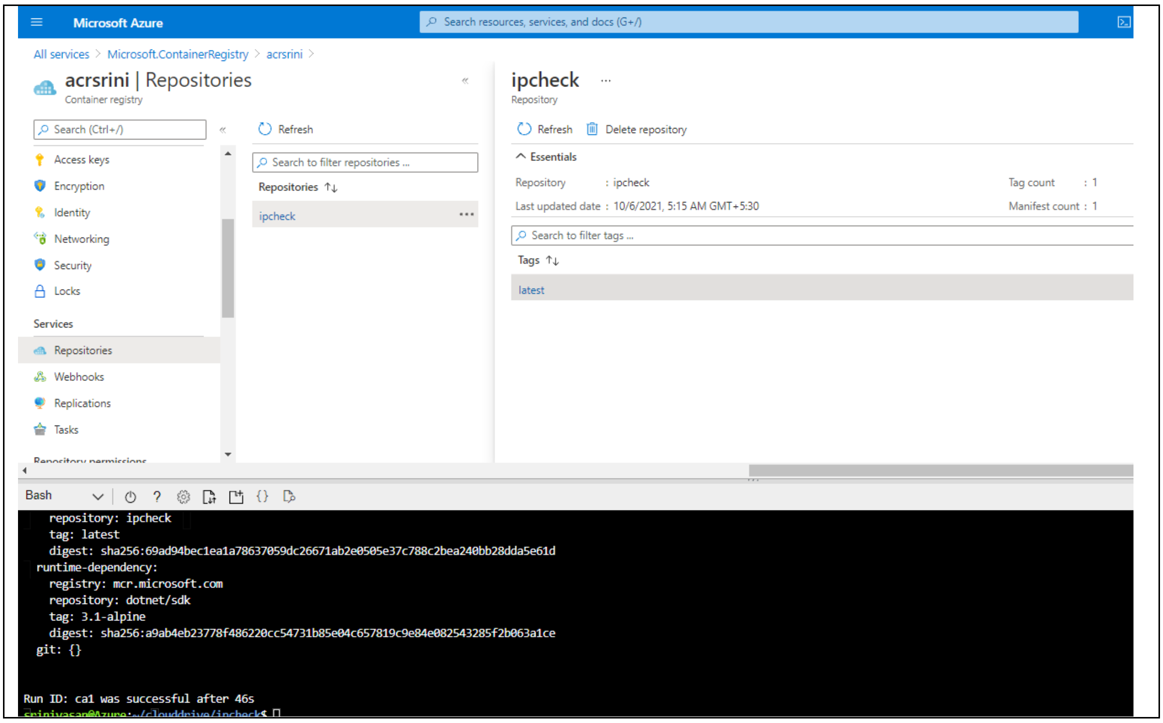
```
az acr build --registry $acrName --image ipcheck:latest .
```

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

3. Close the Cloud Shell pane in the portal.

## 4.6 Task 6: Validate your container image in Container Registry

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



Repository : ipcheck
Last updated date : 10/6/2021, 5:15 AM GMT+5:30
Tag count : 1
Manifest count : 1

```
    repository: ipcheck
    tag: latest
    digest: sha256:69ad94bec1ea1a78637059dc26671ab2e0505e37c788c2bea240bb28dda5e61d
  runtime-dependency:
    registry: mcr.microsoft.com
    repository: dotnet/sdk
    tag: 3.1-alpine
    digest: sha256:a9ab4eb23778f486220cc54731b85e04c657819c9e84e082543285f2b063a1ce
  git: {}


Run ID: ca1 was successful after 46s
srinivasan@Azure:~/clouddrive/ipcheck$
```

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

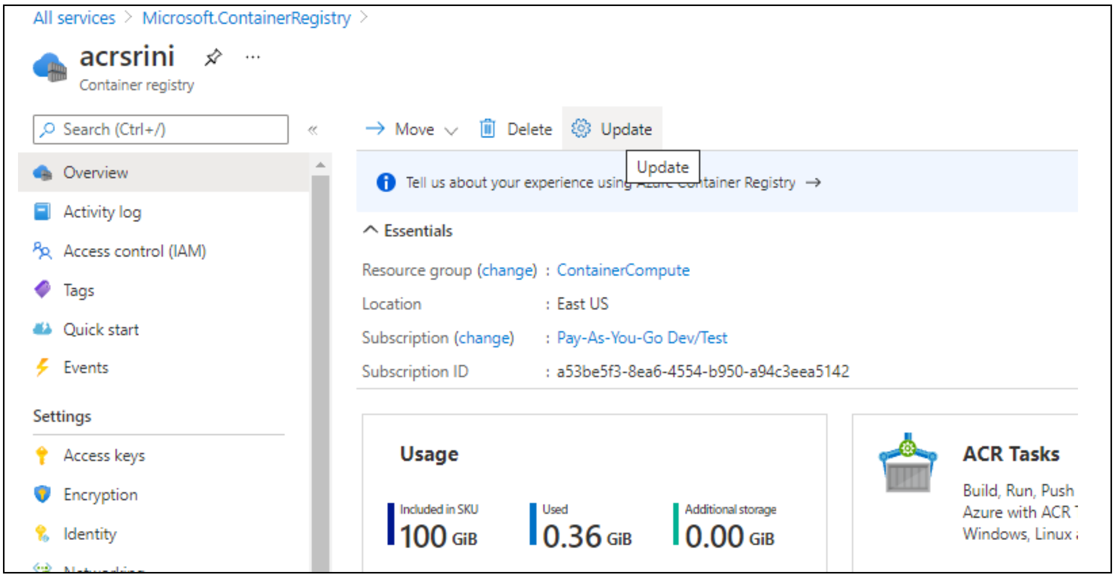
## 4.7 Review

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.

# 5 Exercise 3: Deploy an Azure container instance

## 5.1 Task 1: Enable the admin user in Container Registry

1. In the Azure portal's navigation pane, select the **Resource groups** link.
2. From the **Resource groups** blade, find and then select the **ContainerCompute** resource group that you created earlier in this lab.
3. From the **ContainerCompute** blade, select the container registry that you created earlier in this lab.
4. From the **Container Registry** blade, select **Update**.



All services > Microsoft.ContainerRegistry >

acrsrini

Container registry

Search (Ctrl+/)

→ Move ⌄ Delete Update

Update

Tell us about your experience using Azure Container Registry →

Overview

Activity log

Access control (IAM)

Tags

Quick start

Events

Settings

Access keys

Encryption

Identity

^ Essentials

Resource group (change) : ContainerCompute

Location : East US

Subscription (change) : Pay-As-You-Go Dev/Test

Subscription ID : a53be5f3-8ea6-4554-b950-a94c3eea5142

Usage

Included in SKU 100 GiB

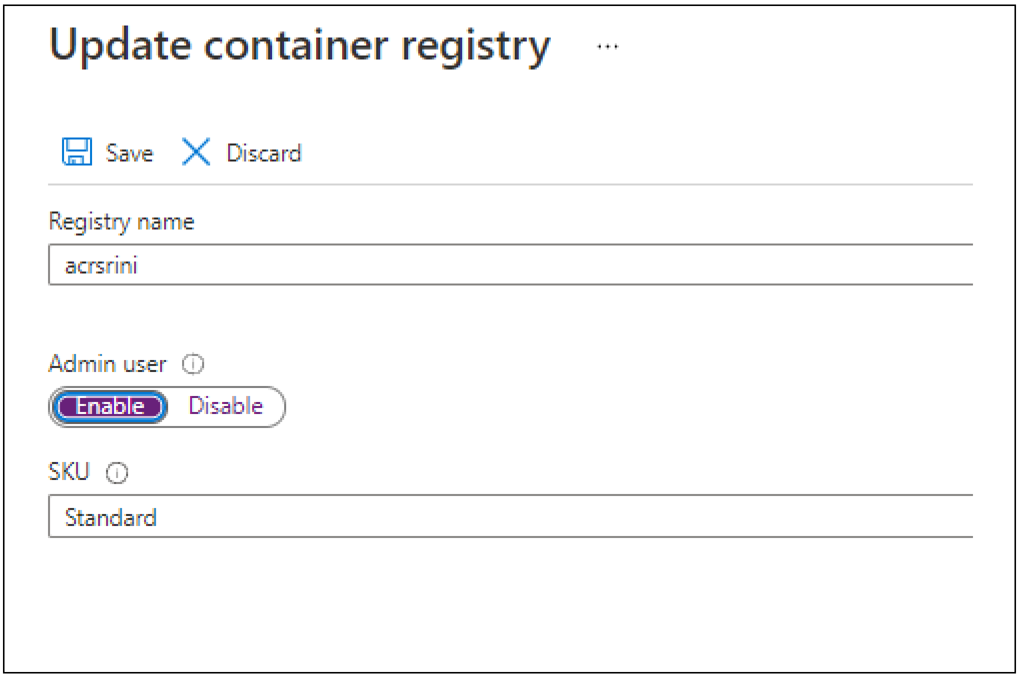
Used 0.36 GiB

Additional storage 0.00 GiB

ACR Tasks

Build, Run, Push Azure with ACR Windows, Linux

5. From the **Update container registry** blade, perform the following actions:
    1. In the **Admin user** section, select **Enable**.
    2. Select **Save**.



Update container registry ···

Save Discard

Registry name

acrsrini

Admin user ⓘ

[✓ Enable] Disable

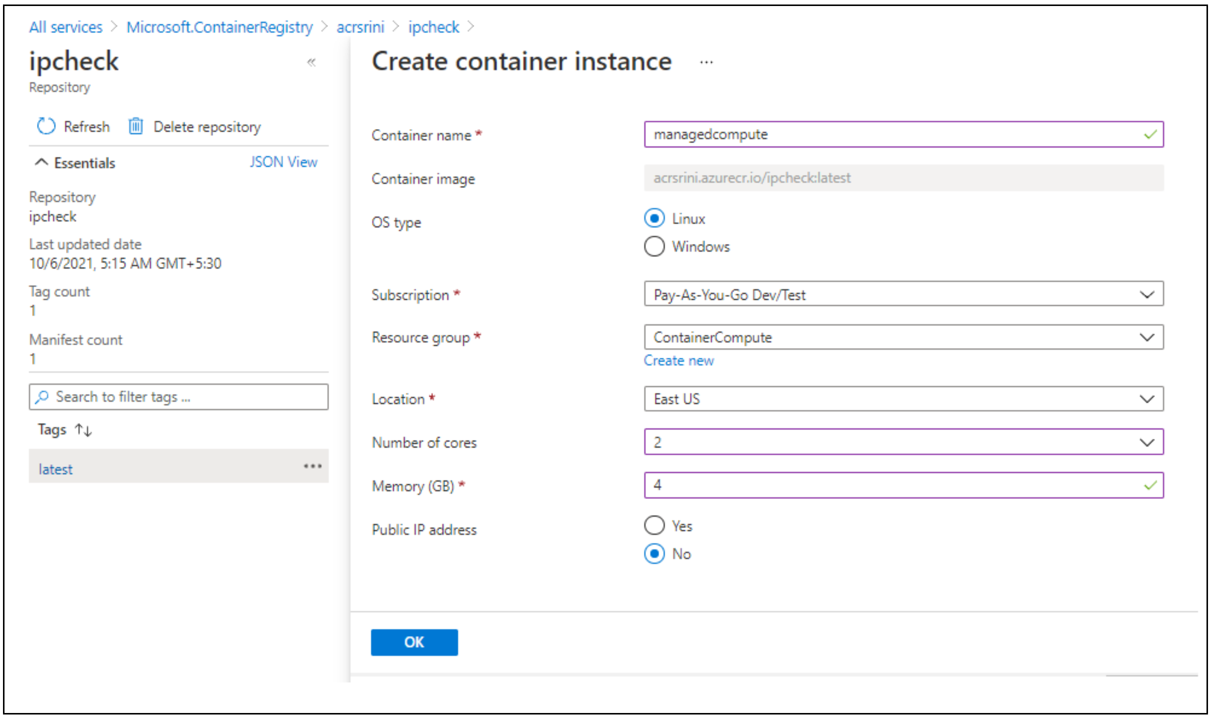
SKU ⓘ

Standard

    3. Close the blade.
6. Close the **Update container registry** blade.

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

1. From the **Container Registry** blade, find the **Services** section, and then select the **Repositories** link.
2. In the **Repositories** section, select the **ipcheck** container image repository.
3. From the **Repository** blade, select the ellipsis menu associated with the **latest** tag entry.
4. In the pop-up menu, select the **Run instance** link.
5. From the **Create container instance** blade, perform the following actions:
    1. In the **Container name** text box, enter **managedcompute**.
    2. Leave the **Container image** text box set to its default value.
    3. In the **OS type** section, select **Linux**.
    4. Leave the **Subscription** text box set to its default value.
    5. In the **Resource group** drop-down list, select **ContainerCompute**.
    6. In the **Location** drop-down list, select **East US**.
    7. In the **Number of cores** drop-down list, select **2**.
    8. In the **Memory (GB)** text box, enter **4**.
    9. In the **Public IP address** section, select **No**.


All services > Microsoft.ContainerRegistry > acrsrini > ipcheck >
ipcheck
Repository
Refresh
Delete repository
Essentials
JSON View
Search to filter tags ...
Tags
latest
...
OK

Repository
ipcheck

Last updated date
10/6/2021, 5:15 AM GMT+5:30

Tag count
1

Manifest count
1

Create container instance

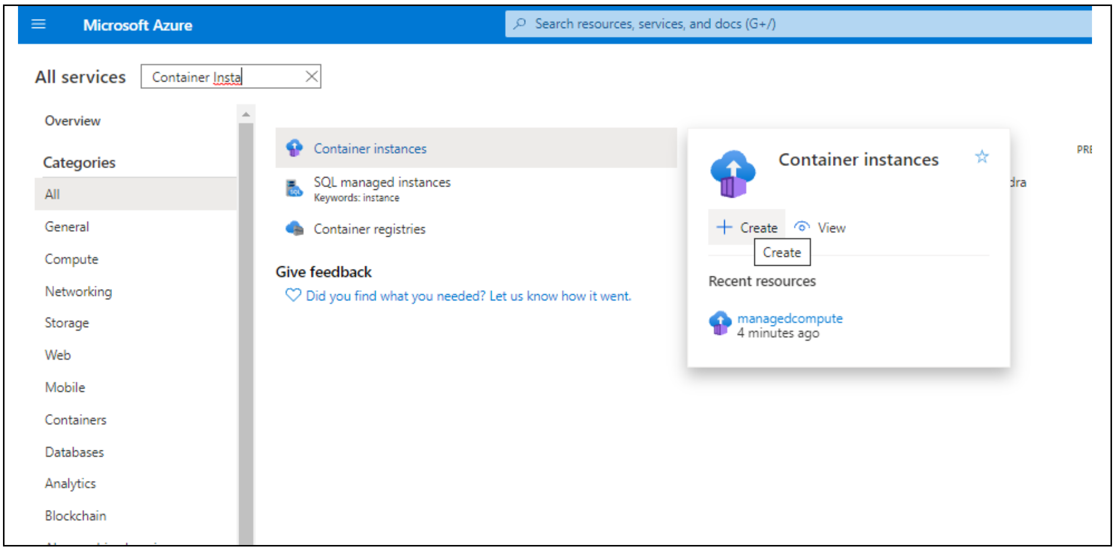
| | |
|---|---|
| Container name * | managedcompute ✓ |
| Container image | acrsrini.azurecr.io/ipcheck:latest |
| OS type | ● Linux<br>○ Windows |
| Subscription * | Pay-As-You-Go Dev/Test |
| Resource group * | ContainerCompute<br>Create new |
| Location * | East US |
| Number of cores | 2 |
| Memory (GB) * | 4 ✓ |
| Public IP address | ○ Yes<br>● No |

    10. Select **OK**.

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

## 5.3 Task 3: Manually deploy a container image to Container Instances

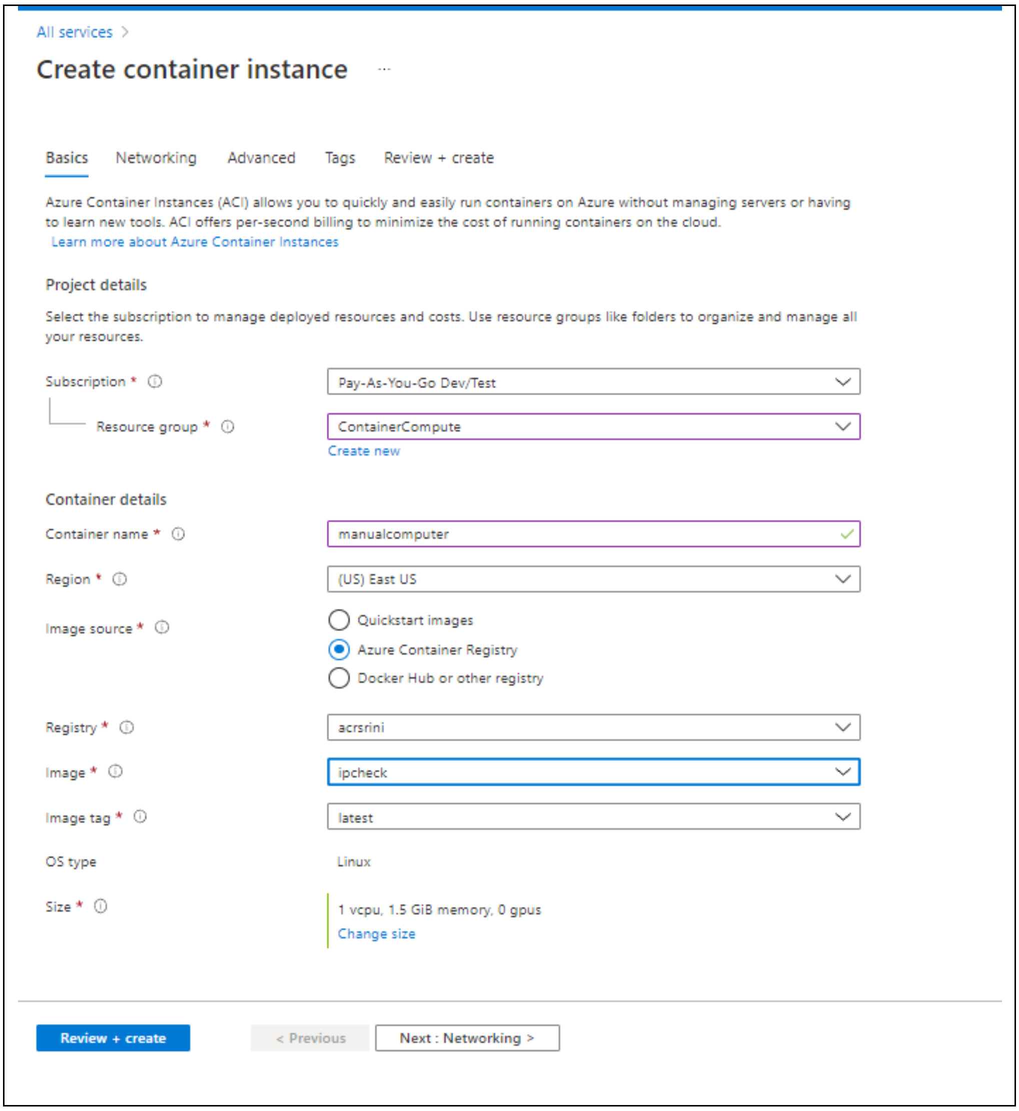
1. In the Azure portal’s navigation pane, select the **Create a resource** link.
2. From the **Create a resource** blade, find the **Search services and marketplace** text box.
3. In the search box, enter **container instances**, and then select Enter.
4. From the **Marketplace** search results blade, select the **Container Instances** result.
5. From the **Container Instances** blade, select **Create**.



6. Find the tabs from the **Create Container Instances** blade, such as **Basics**, **Networking**, and **Advanced**.

   **Note**: Each tab represents a step in the workflow to create a new container instance.

7. From the **Basics** tab, perform the following actions:
   1. Leave the **Subscription** text box set to its default value.
   2. In the **Resource group** drop-down list, select **ContainerCompute**.
   3. In the **Container name** text box, enter **manualcompute**.
   4. In the **Region** drop-down list, select **(US) East US**.
   5. In the **Image source** section, select **Azure Container Registry**.
   6. In the **Registry** drop-down list, select the **Azure Container Registry** resource that you created earlier in this lab.
   7. In the **Image** drop-down list, select **ipcheck**.
   8. In the **Image tag** drop-down list, select **latest**.
   9. Select **Review + Create**.



Azure Container Instances (ACI) allows you to quickly and easily run containers on Azure without managing servers or having to learn new tools. ACI offers per-second billing to minimize the cost of running containers on the cloud.
Learn more about Azure Container Instances

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *: Pay-As-You-Go Dev/Test

Resource group *: ContainerCompute

Create new

Container details

Container name *: manualcomputer ✓

Region *: (US) East US

Image source *:
○ Quickstart images
● Azure Container Registry
○ Docker Hub or other registry

Registry *: acrsrini

Image *: ipcheck

Image tag *: latest

OS type: Linux

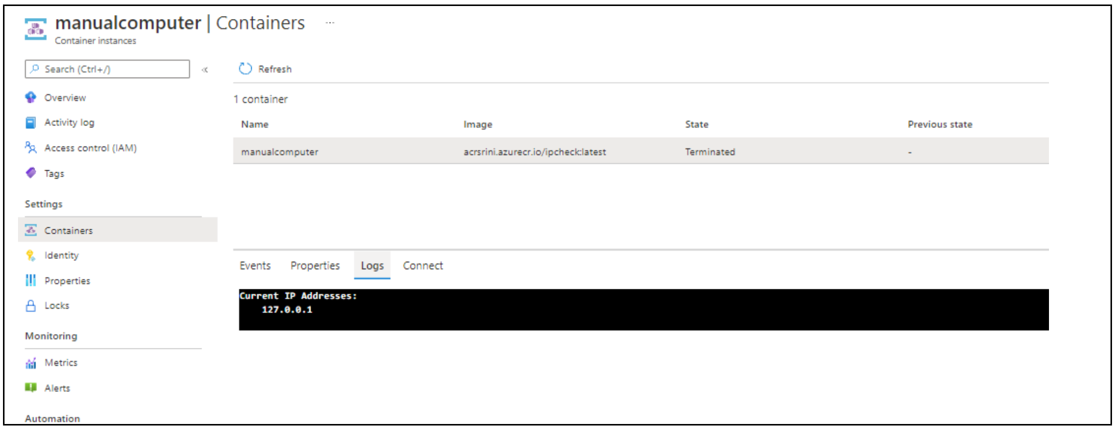
Size *: 1 vcpu, 1.5 GiB memory, 0 gpus
Change size

8. From the **Review + Create** tab, review the selected options.
9. Select **Create** to create the container instance by using your specified configuration.

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

## 5.4 Task 4: Validate that the container instance ran successfully

1. In the Azure portal's navigation pane, select the **Resource groups** link.
2. From the **Resource groups** blade, find and then select the **ContainerCompute** resource group that you created earlier in this lab.
3. From the **ContainerCompute** blade, select the **manualcompute** container instance that you created earlier in this lab.
4. From the **Container Instance** blade, find the **Settings** section, and then select the **Containers** link.
5. In the **Containers** section, find the list of **Events**.



1 container

| Name | Image | State | Previous state |
|---|---|---|---|
| manualcomputer | acrsrini.azurecr.io/ipcheck:latest | Terminated | - |

```
Current IP Addresses:
    127.0.0.1
```

6. Select the **Logs** tab, and then find 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.

## 5.5 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 also able to customize the deployment further and to run task-based applications as part of a container run.

# 6 Exercise 4: Clean up your subscription

## 6.1 Task 1: Open Azure Cloud Shell and list resource groups

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

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

2. If this is your first time opening Cloud Shell using your subscription, you can use the **Welcome to Azure Cloud Shell Wizard** to configure Cloud Shell for first-time usage. Perform the following actions in the wizard:
   1. A dialog box prompts you to configure the shell. Select **Bash**, review the selected subscription, and then select **Create storage**.

   **Note**: Wait for Cloud Shell to finish its initial setup procedures before moving forward with the lab. If you don’t notice Cloud Shell configuration options, this is most likely because you are using an existing subscription with this course’s labs. The labs are written with the presumption that you are using a new subscription.

## 6.2 Task 2: Delete resource groups

1. Enter the following command, and then select Enter to delete the **ContainerCompute** resource group:

   code

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

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

## 6.3 Task 3: Close the active applications

- Close the currently running Microsoft Edge application.

## 6.4 Review

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