

Domain 5: Deploy and manage Azure compute resources

Introduction to Azure virtual machines

1. Introduction

<https://learn.microsoft.com/en-us/training/modules/intro-to-azure-virtual-machines/1-introduction>

Introduction

Completed

Suppose you work for a company doing medical research and you're responsible for managing the on-premises servers. The servers you administer run all the company infrastructure, from web servers to databases. However, the hardware is aging and starting to struggle to keep up with some of the new data analysis applications being deployed to it.

You could upgrade all the hardware, but that's not appealing for several reasons:

- The servers are physically scattered all around the world with minimal staff in each location. We'd like to centralize the upgrade to our home office.
- The company runs custom data analysis software on several versions and flavors of Windows and Linux, sometimes set up with odd configurations that aren't entirely understood. We need a way to test our deployments completely and try different configurations to make sure everything is working before we transition the work.
- Business is booming, and the company is growing fast. It's likely that the load on the internal servers, particularly the databases, will continue to grow. This growth requires us to either buy for the future or come up with a scaling plan to handle the growth.

For these reasons, you decide that it's time to explore the cloud to see if it can help solve the load and scale problem. Since you have a bunch of mixed servers and custom software, it makes sense to look at trying to move servers one at a time into Azure using Azure Virtual Machines (VMs).

Azure VMs are one of several types of on-demand, scalable computing resources that Azure offers. With VMs, you have total control over the configuration and can install anything you need to perform the work. You don't need to purchase physical hardware when you need to scale or extend your datacenter. Finally, Azure provides other services to monitor, secure, and manage updates and patches to the OS.

We're going to look at the decisions made before creating a VM, the options to create and manage the VM, and the extensions and services you use to manage your VM.

Learning objectives

In this module, you learn how to:

- Compile a checklist for creating a virtual machine
- Describe the options to create and manage virtual machines
- Describe the other services available to administer virtual machines

2. Compile a checklist for creating an Azure Virtual Machine

<https://learn.microsoft.com/en-us/training/modules/intro-to-azure-virtual-machines/2-compile-a-checklist-for-creating-a-vm>

Compile a checklist for creating an Azure Virtual Machine

Completed

Performing a migration of on-premises servers to Azure requires planning and care. You can move them all at once, or more likely, in small batches or even individually. Before you create a single VM, you should sit down and sketch out your current infrastructure model and see how it might map to the cloud.

What is an Azure resource?

An **Azure resource** is a manageable item in Azure. Just like a physical computer in your datacenter, VMs have several elements that are needed to do their job:

- The VM itself
- Disks for storage
- Virtual network
- Network interface to communicate on the network
- Network Security Group (NSG) to secure the network traffic
- An IP address (public, private, or both)

Azure creates all of these resources if necessary, or you can supply existing ones as part of the deployment process. Each resource needs a name that's used to identify it. If Azure creates the resource, it uses the VM name to generate a resource name - another reason to be consistent with your VM names!

Required resources for IaaS Virtual Machines

Let's walk through a checklist of things to think about.

- The network
- VM name
- Location
- The VM size
- Disks
- Operating system

The network

The first thing you should think about isn't the virtual machine at all - it's the network. Take a look at one of your on-premises servers:

- What does the server communicate with?
- Which ports are open?

Virtual networks (VNETs) are used in Azure to provide private connectivity between Azure Virtual Machines and other Azure services. VMs and services that are part of the same virtual network can access one another. By default, services outside the virtual network can't connect to services within the virtual network. You can, however, configure the network to allow access to the external service, including your on-premises servers.

This latter point is why you should spend some time thinking about your network configuration. Network addresses and subnets aren't trivial to change once you have them set up. If you plan to connect your private company network to the Azure services, you want to make sure you consider the topology before putting any VMs into place.

When you set up a virtual network, you specify the available address spaces, subnets, and security. If the VNet is connected to other VNETs, you must select address ranges that aren't overlapping. This is the range of private addresses that the VMs and services in your network can use. You can use unroutable IP addresses such as 10.0.0.0/8, 172.16.0.0/12, or 192.168.0.0/16, or define your own range. Azure treats any address range as part of the private VNet IP address space if it's only reachable within the VNet, within interconnected VNETs, and from your on-premises location. If someone else is responsible for the internal networks, you should work with that person

before selecting your address space to make sure there's no overlap. Let them know what space you want to use, so they don't try to use the same range of IP addresses.

Segregate your network

After deciding the virtual network address space(s), you can create one or more subnets for your virtual network. You create these subnets to break up your network into more manageable sections. For example, you might assign 10.1.0.0 to VMs, 10.2.0.0 to back-end services, and 10.3.0.0 to SQL Server VMs.

Note

Azure reserves the first four addresses and the last address in each subnet for its use.

Secure the network

By default, there's no security boundary between subnets, so services in each of these subnets can talk to one another. However, you can set up Network Security Groups (NSGs), which allow you to control the traffic flow to and from subnets and to and from VMs. NSGs act as software firewalls, applying custom rules to each inbound or outbound request at the network interface and subnet level. This way, you can fully control every network request coming in or out of the VM.

Plan each VM deployment

Once you have mapped out your communication and network requirements, you can start thinking about the VMs you want to create. A good plan is to select a server and take an inventory:

- Which OS is used?
- How much disk space is in use?
- What kind of data does this use? Are there restrictions (legal or otherwise) around how it's stored or where it's physically located?
- What sort of CPU, memory, and disk I/O load does the server have? Is there burst traffic to account for?

We can then start to answer some of the questions Azure has for a new virtual machine.

VM name

The VM name is used as the computer name, which is configured as part of the operating system. You can specify a name of up to 64 characters on a Linux VM and 15 characters on a Windows VM.

This name also defines a manageable **Azure resource**, and it's not trivial to change later. That means you should choose names that are meaningful and consistent, so you can easily identify what the VM does. A good convention is to include the following information in the name:

Element	Example	Notes
Environment	dev, prod, QA	Identifies the environment for the resource
Location	<code>eus</code> for East US, <code>jw</code> for Japan West	Identifies the region into which the resource is deployed
Instance	01, 02	For resources that have more than one named instance (web servers, etc.)
Product or Service	service	Identifies the product, application, or service that the resource supports
Role	sql, web, messaging	Identifies the role of the associated resource

For example, `deveus-webvm01` might represent the first development web server hosted in the East US location.

Decide the location for the VM

Azure has datacenters all over the world filled with servers and disks. These datacenters are grouped into geographic *regions* ('West US', 'North Europe', 'Southeast Asia', etc.) to provide redundancy and availability.

When you create and deploy a virtual machine, you must select a region where you want to allocate the resources. You can place your VMs as close as possible to your users to improve performance and to meet any legal, compliance, or tax requirements.

Two other things to think about regarding the location choice. First, the location can limit your available options. Each region has different hardware available and some configurations aren't available in all regions. Second, there are price differences between

locations. If your workload isn't bound to a specific location, it can be very cost effective to check your required configuration in multiple regions to find the lowest price.

Determine the size of the VM

Once you have the name and location set, you need to decide on the [size of your VM](#). Rather than specify processing power, memory, and storage capacity independently, Azure provides different *VM sizes* that offer variations of these elements in different sizes. Azure provides a wide range of VM size options allowing you to select the appropriate mix of compute, memory, and storage for what you want to do.

The best way to determine the appropriate VM size is to consider the type of workload your VM needs to run. Based on the workload, you're able to choose from a subset of available VM sizes. Workload options are classified as follows on Azure:

Option	Description
General purpose	General-purpose VMs are designed to have a balanced CPU-to-memory ratio. Ideal for testing and development, small to medium databases, and low to medium traffic web servers.
Compute optimized	Compute optimized VMs are designed to have a high CPU-to-memory ratio. Suitable for medium traffic web servers, network appliances, batch processes, and application servers.
Memory optimized	Memory optimized VMs are designed to have a high memory-to-CPU ratio. Great for relational database servers, medium to large caches, and in-memory analytics.
Storage optimized	Storage optimized VMs are designed to have high disk throughput and IO. Ideal for VMs running databases.
GPU	GPU VMs are specialized virtual machines targeted for heavy graphics rendering and video editing. These VMs are ideal options for model training and inferencing with deep learning.
High performance compute	High performance compute is the fastest and most powerful CPU virtual machines with optional high-throughput network interfaces.

You're able to filter on the workload type when you configure the VM size in the Azure. The size you choose directly affects the cost of your service. The more CPU, memory, and GPU you need, the higher the price point.

What if my size needs change?

Azure allows you to change the VM size when the existing size no longer meets your needs. You can upgrade or downgrade the VM, as long as your current hardware configuration is allowed in the new size. The ability to change VM size provides a fully agile and scalable approach to VM management.

The VM size can be changed while the VM is running, as long as the new size is available in the current hardware cluster the VM is running on. The Azure portal makes the size options obvious by only showing you available size choices. The command line tools report an error if you attempt to resize a VM to an unavailable size. Changing a running VM size automatically reboots the machine to complete the request.

If you stop and deallocate the VM, you can then select any size available in your region since deallocation removes your VM from the cluster it was running on.

Warning

Be careful about resizing production VMs - they will be rebooted automatically which can cause a temporary outage and change some configuration settings such as the IP address.

Parts of a VM and how they're billed

When you create a virtual machine, you're also creating resources that support the virtual machine. These resources come with their own costs that should be considered.

The default resources supporting a virtual machine and how they're billed are detailed in the following table:

Resource	Description	Cost
Virtual network	For giving your virtual machine the ability to communicate with other resources	Virtual Network pricing

Resource	Description	Cost
A virtual Network Interface Card (NIC)	For connecting to the virtual network	There is no separate cost for NICs. However, there is a limit to how many NICs you can use based on your VM's size . Size your VM accordingly and reference Virtual Machine pricing .
A private IP address and sometimes a public IP address.	For communication and data exchange on your network and with external networks	IP Addresses pricing
Network security group (NSG)	For managing the network traffic too and from your VM. For example, you might need to open port 22 for SSH access, but you might want to block traffic to port 80. Blocking and allowing port access is done through the NSG.	There are no additional charges for network security groups in Azure.
OS Disk and possibly separate disks for data.	It's a best practice to keep your data on a separate disk from your operating system, in case you ever have a VM fail, you can simply detach the data disk, and attach it to a new VM.	All new virtual machines have an operating system disk and a local disk. Azure doesn't charge for local disk storage. The operating system disk, which is usually 127GiB but is smaller for some images, is charged at the regular rate for disks . You can see the cost for attach Premium (SSD based) and Standard (HDD) based disks to your virtual machines on the Managed Disks pricing page .
In some cases, a license for the OS	For providing your virtual machine runs to run the OS	The cost varies based on the number of cores on your VM, so size your VM accordingly . The cost can be reduced through the Azure Hybrid Benefit .

Understanding the pricing model

There are two separate costs the subscription is charged for every VM: compute and storage. By separating these costs, you scale them independently and only pay for what you need.

Compute costs - Compute expenses are priced on a per-hour basis but billed on a per-minute basis. For example, you're only charged for 55 minutes of usage if the VM is deployed for 55 minutes. You're not charged for compute capacity if you stop and deallocate the VM since deallocation releases the hardware. The hourly price varies based on the VM size and OS you select. Linux-based instances are cheaper because there's no operating system license charge. For Windows, the cost for a VM includes the charge for the operating system.

Tip

You might be able to save money by reusing existing licenses with the **Azure Hybrid benefit** for [Linux](#) or [Windows](#).

You're able to choose from two payment options for compute costs.

Option	Description
Pay as you go	With the pay-as-you-go option, you pay for compute capacity by the second, with no long-term commitment or upfront payments. You're able to increase or decrease compute capacity on demand and start or stop at any time. Select this option if you run applications with short-term or unpredictable workloads that can't be interrupted. For example, if you're doing a quick test, or developing an app in a VM, pay-as-you-go is the appropriate option.
Reserved Virtual Machine Instances	The Reserved Virtual Machine Instances (RI) option is an advance purchase of a virtual machine for one or three years in a specified region. The commitment is made up front, and in return, you get up to 72% price savings compared to pay-as-you-go pricing. RI s are flexible and can easily be exchanged or returned for an early termination fee. Select this option if the VM has to run continuously, or you need budget predictability, and you can commit to using the VM for at least a year.

Storage costs - You're charged separately for the storage the VM uses. The status of the VM has no relation to the storage charges that are incurred. If the VM is stopped/deallocated and you aren't billed for the running VM, you're still charged for the storage used by the disks.

Storage for the VM

All Azure virtual machines have at least two virtual hard disks (VHDs). The first disk stores the operating system, and the second is used as temporary storage. You should add more data disks to store application data. Separating out the data to different disks allows you to

manage the disks independently. The VM size determines the maximum number of data disks you can attach to your VM, typically two per vCPU.

There are five disk types, each intended to address a specific customer scenario:

- [Ultra disks](#)
- [Premium SSD v2 \(preview\)](#)
- [Premium SSDs \(solid-state drives\)](#)
- [Standard SSDs](#)
- [Standard HDDs \(hard disk drives\)](#)

The following table provides a comparison of the five disk types to help you decide which to use.

	Ultra disk	Premium SSD v2	Premium SSD	Standard SSD	Standard HDD
Disk type	SSD	SSD	SSD	SSD	HDD
Scenario	IO-intensive workloads such as SAP HANA , top tier databases (for example, SQL, Oracle), and other transaction-heavy workloads.	Production and performance-sensitive workloads that consistently require low latency and high IOPS and throughput	Production and performance sensitive workloads	Web servers, lightly used enterprise applications and dev/test	Backup, noncritical, infrequent access
Max disk size	65,536 gibibytes (GiB)	65,536 GiB	32,767 GiB	32,767 GiB	32,767 GiB
Max throughput	4,000 MB/s	1,200 MB/s	900 MB/s	750 MB/s	500 MB/s
Max IOPS	160,000	80,000	20,000	6,000	2,000
Usable as OS Disk?	No	No	Yes	Yes	Yes

Select an operating system

Azure provides various OS images that you can install into the VM, including many Linux distributions. The choice of OS might influence your hourly compute pricing as Azure bundles the cost of the OS license into the price.

If you're looking for more than just base OS images, you can search the Azure Marketplace for more sophisticated images that include the OS and popular software tools for specific scenarios. For example, if you needed a new WordPress site, the standard technology stack would consist of a Linux server, Apache web server, a MySQL database, and PHP. Instead of setting up and configuring each component, you can use a Marketplace image and install the entire stack all at once.

Finally, if you can't find a suitable OS image, you can create your own image with what you need, and use them to create VMs. You can create individual images for use in development and test. Or, you can create an [Azure Compute Gallery](#) to manage multiple images and replicate them to the regions where they're needed.

3. Exercise - Create a VM using the Azure portal

<https://learn.microsoft.com/en-us/training/modules/intro-to-azure-virtual-machines/3-create-a-vm>

Exercise - Create a VM using the Azure portal

Completed

You've planned out the network infrastructure and identified a few VMs to migrate to the cloud. You have several choices for creating your VMs. The choice you make depends on the environment you're comfortable with. Azure supports a web-based portal for creating and administering resources. You can also choose to use command-line tools that run on Linux, macOS, and Windows.

Note

This exercise is optional. If you want to complete this exercise, you'll need to create an Azure subscription before you begin. If you don't have an Azure account or you don't want to create one at this time, you can read through the instructions so you understand the information that's being presented.

Note

You need to use a resource group to complete the steps in this exercise. You can use a resource group that you already created, or you can create a new resource group specifically for this exercise. If you choose to create a new resource group, that will make it easier to clean up any resources that you create as you complete the exercise. If you don't have an existing resource group or you want to create a new one specifically for this exercise, you can follow the steps in [Use the Azure portal and Azure Resource Manager to manage resource groups](#) to create a resource group by using the Azure portal, or you can follow the steps in [Manage Azure resource groups by using Azure CLI](#) to create a resource group by using the the Azure CLI.

Note

Throughout this exercise, replace **myResourceGroupName** in the examples with the name of an existing resource group, or the name of the resource group that you created for this exercise.

Options to create and manage VMs

Let's explore the Azure portal first - it's the easiest way to start with Azure.

Azure portal

The **Azure portal** provides an easy-to-use browser-based user interface that enables you to create and manage all your Azure resources. For example, you can set up a new database, increase the compute power of your virtual machines, and monitor your monthly costs. It's also a great learning tool, because you can survey all available resources and use guided wizards to create the ones you need.

Create an Azure VM with the Azure portal

Let's assume you want to create a VM running a web server on Ubuntu. Setting up a site isn't difficult, but there are a couple of things to keep in mind. You need to install and configure an operating system, configure a website, install a database, and worry about things like firewalls. We're going to cover creating VMs in the next few modules, but let's create one here to see how easy it is. We don't go through all the options - check out one of the **Create a VM** modules to get complete details on each option.

1. Sign in to the [Azure portal](#).
2. On the Azure home page, under **Azure services**, select **Create a resource**. The **Create a resource** pane appears, displaying popular products for Azure services.

[Home](#) >

Create a resource ...

Get Started

Recently created

Categories

AI + Machine Learning

Analytics

Blockchain

Compute

Containers

Databases

Developer Tools

DevOps

Identity

Integration

Search services and marketplace

Getting Started? [Try our Quickstart center](#)

Popular Azure services [See more in All services](#)

 **Virtual machine**
[Create](#) | [Learn more](#)

 **Kubernetes Service**
[Create](#) | [Docs](#) | [MS Learn](#)

 **Azure Cosmos DB**
[Create](#) | [Docs](#) | [MS Learn](#)

 **Function App**
[Create](#) | [Docs](#)

 **SQL Database**
[Create](#) | [Docs](#) | [MS Learn](#)

Popular Marketplace products [See more in Marketplace](#)

 **Windows Server 2019 Datacenter**
[Create](#) | [Learn more](#)

 **Ubuntu Server 20.04 LTS**
[Create](#) | [Learn more](#)

 **Windows 10 Pro, version 20H2**
[Create](#) | [Learn more](#)

 **Ubuntu Server 18.04 LTS**
[Create](#) | [Learn more](#)

 **Free 100**
[Set up + subscribe](#) | [Learn more](#)

3. We want to create a VM, so select **Virtual machine**.

4. The **Create virtual machine** pane appears.

Configure the VM

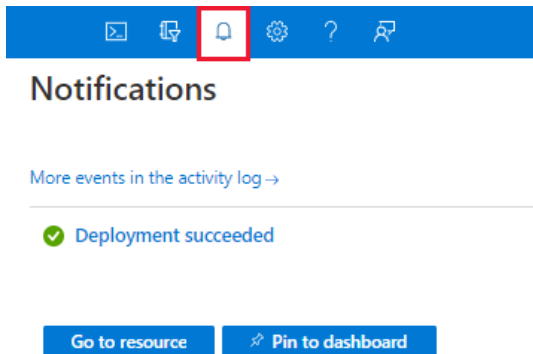
You need to configure the basic parameters of your virtual machine. If some of the options at this point are unfamiliar to you, that's OK. We're going to describe all of these options in a future module. You're welcome to copy the values used here.

1. On the **Basics** tab, enter the following values for each setting.

Setting	Value
Project details	
Subscription	Select your subscription
Resource group	Select myResourceGroupName from the drop-down
Instance details	
Virtual machine name	Enter <i>test-ubuntu-cus-vm</i>
Region	From the dropdown list, select a geographical location close to you.
Availability options	No infrastructure redundancy required
Security type	Standard
Image	Ubuntu Server 24.04 LTS - Gen2
VM architecture	x64
Run with Azure Spot discount	Unchecked
Size	Standard D2s V3
Administrator account	
Authentication type	SSH public key
Username	Enter a username
SSH public key source	Generate a new key pair
Key pair name	test-ubuntu-cus-vm_key
Inbound port rules	
Public inbound ports	Allow selected ports
Select inbound ports	SSH (22)

2. There are several other tabs you can explore to see the settings you can influence during the VM creation. After you're finished exploring, select **Review + create** to review and validate the settings.

- Azure validates your configuration settings for a resource before it creates it. You might need to supply some additional information based on the requirements of the image creator built into Azure. It's simple; just open the tab that has an error. Verify all the settings are set the way you want, and then select **Create** to deploy and create the VM.
- The **Generate new key pair** window opens. Select **Download private key and create resource**.
- You can monitor the deployment in the **Deployment details** on the **Overview** pane or through the **Notifications** pane. Select the notifications icon in the top right toolbar to show or hide the Notifications pane.



The VM deployment process takes a few minutes to complete. You receive a notification informing you that the deployment succeeded.

- Select **Go to resource**. The **Overview** page of your VM appears.

Here, you can see all the information and configuration options for your newly created Ubuntu VM. One of the pieces of information is the **Public IP address**.

When you enabled SSH public key authentication in an earlier step, the user interface also gave an option to enable SSH. SSH allows you to connect to your VM via the public IP using any SSH client.

Congratulations! With a few steps, you deployed a VM that runs Linux. Let's explore some other ways we could have created a VM.

4. Describe the options available to create and manage an Azure Virtual Machine

Describe the options available to create and manage an Azure Virtual Machine

Completed

The Azure portal is the easiest way to create resources such as VMs when you're getting started. However, it's not necessarily the most efficient or quickest way to work with Azure, particularly if you need to create several resources together. In our case, we eventually create dozens of VMs to handle different tasks. Creating them manually in the Azure portal wouldn't be a fun task!

Let's look at some other ways to create and administer resources in Azure:

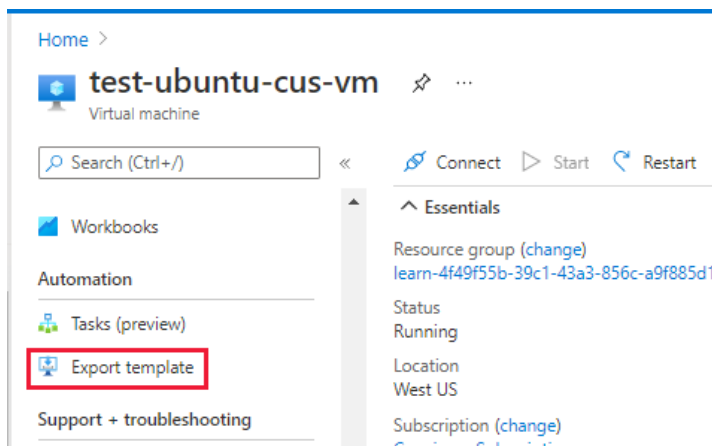
- Azure Resource Manager templates
- Azure PowerShell
- Azure CLI
- Azure REST API
- Azure Client SDK
- Azure VM Extensions
- Azure Automation Services

Resource Manager templates

Let's assume you want to create a copy of a VM with the same settings. You could create a VM image, upload it to Azure, and reference it as the basis for your new VM. This process is inefficient and time-consuming. Azure provides you with the option to create a template from which to create an exact copy of a VM.

Resource Manager templates are JSON files that define the resources you need to deploy for your solution.

You can create a resource template for your VM. From the VM menu, under **Automation** select **Export template**.



Note

The policies for the resources included in the sandbox for this Learn module prevent you from being able to export the VM you just created; that said, an exported template is an easy-to-edit JSON file. You have the option to download or save a template for later use, or immediately deploy a new VM based on the template. For example, you might create a VM from a template in a test environment, and find it doesn't quite work to replace your on-premises machine. You can delete the resource group, which deletes all of the resources, tweak the template, and try again. If you only want to make changes to the existing deployed resources, you can change the template used to create it, and redeploy it. Resource Manager will change the resources to match the new template.

After you have it working the way you want it, you can use that template to easily replicate multiple versions of your infrastructure, such as staging and production. You can parameterize fields such as the VM name, network name, storage account name, and so on, and load the template repeatedly, using different parameters to customize each environment.

For more information about using templates, see [Quickstart: Create an Ubuntu Linux virtual machine using an ARM template](#).

Azure CLI

An option for scripting and command-line Azure interaction is the **Azure CLI**.

The Azure CLI is Microsoft's cross-platform command-line tool for managing Azure resources such as virtual machines and disks from the command line. It's available for Linux, macOS, Windows, or in a browser using the Cloud Shell.

For example, from the CLI, you can create an Azure VM with the `az vm create` command.

```
az vm create \
  --resource-group TestResourceGroup \
  --name test-wp1-eus-vm \
  --image Ubuntu2204 \
  --admin-username azureuser \
  --generate-ssh-keys
```

The Azure CLI can be used with other scripting languages, like Ruby and Python.

Learn more about creating and managing VMs in the **Manage virtual machines with the Azure CLI tool** module.

For more information about using the Azure CLI to create VMs, see [Quickstart: Create a Linux virtual machine using the CLI](#).

Azure PowerShell

Azure PowerShell is ideal for one-off interactive tasks and/or the automation of repeated tasks.

Note

PowerShell is a cross-platform shell that provides services like the shell window and command parsing. Azure PowerShell is an optional add-on package that adds the Azure-specific commands (referred to as **cmdlets**). You can learn more about installing and using Azure PowerShell in a separate training module.

For example, you can use the `New-AzVM` cmdlet to create a new Debian-based Azure virtual machine.

```
New-AzVm `
  -ResourceGroupName "TestResourceGroup" `
  -Name "test-wp1-eus-vm" `
  -Location "East US" `
  -Image Debian11 `
  -VirtualNetworkName "test-wp1-eus-network" `
  -SubnetName "default" `
  -SecurityGroupName "test-wp1-eus-nsg" `
  -PublicIpAddressName "test-wp1-eus-pubip" `
  -GenerateSshKey `
  -SshKeyName myPSKey
  -OpenPorts 22
```

As shown here, you supply various parameters to handle the large number of VM configuration settings available. Most of the parameters have reasonable values; you only need to specify the required parameters. Learn more about creating and managing VMs with Azure PowerShell in the **Automate Azure tasks using scripts with PowerShell** module.

For more information about using PowerShell to create VMs, see [Quickstart: Create a Linux virtual machine using PowerShell](#).

Terraform

Azure also has a Terraform provider, so you can easily use Terraform to create and manage your VMs. Terraform enables the definition, preview, and deployment of cloud infrastructure. Using Terraform, you create configuration files using HCL syntax. The HCL syntax allows you to specify the cloud provider - such as Azure - and the elements that make up your cloud infrastructure. After you create your configuration files, you create an execution plan that allows you to preview your infrastructure changes before they're deployed. Once you verify the changes, you apply the execution plan to deploy the infrastructure.

For more information, see the [Azure Terraform Provider](#) and [Quickstart: Use Terraform to create a VM](#).

Programmatic (APIs)

Generally speaking, both Azure PowerShell and Azure CLI are good options if you have simple scripts to run and want to stick to command-line tools. When it comes to more complex scenarios, where the creation and management of VMs form part of a larger application with complex logic, another approach is needed.

You can interact with every type of resource in Azure programmatically.

Azure REST API

The Azure REST API provides developers with operations categorized by resource and the ability to create and manage VMs. Operations are exposed as URLs with corresponding HTTP methods (GET , PUT , POST , DELETE , and PATCH) and a corresponding response.

The Azure Compute APIs give you programmatic access to virtual machines and their supporting resources.

For more information, see the [Virtual Machines REST API reference](#).

Azure Client SDK

Even though the REST API is platform and language agnostic, most often developers look toward a higher level of abstraction. The Azure Client SDK encapsulates the Azure REST API, making it much easier for developers to interact with Azure.

The Azure Client SDKs are available for various languages and frameworks, including .NET-based languages such as C#, Java, Node.js, PHP, Python, Ruby, and Go.

Here's an example snippet of C# code to create an Azure VM using the `Microsoft.Azure.Management.Fluent` NuGet package.

```
var azure = Azure
    .Configure()
    .WithLogLevel(HttpLoggingDelegatingHandler.Level.Basic)
    .Authenticate(credentials)
    .WithDefaultSubscription();
// ...
var vmName = "test-wp1-eus-vm";

azure.VirtualMachines.Define(vmName)
    .WithRegion(Region.USEast)
    .WithExistingResourceGroup("TestResourceGroup")
    .WithExistingPrimaryNetworkInterface(networkInterface)
    .WithLatestWindowsImage("MicrosoftWindowsServer", "WindowsServer", "2012-R2-Datacenter")
    .WithAdminUsername("jenc")
    .WithAdminPassword("aReallyGoodPasswordHere")
    .WithComputerName(vmName)
    .WithSize(VirtualMachineSizeTypes.StandardDS1)
    .Create();
```

Here's the same snippet in Java using the **Azure Java SDK**.

```
String vmName = "test-wp1-eus-vm";
// ...
VirtualMachine virtualMachine = azure.virtualMachines()
    .define(vmName)
    .withRegion(Region.US_EAST)
    .withExistingResourceGroup("TestResourceGroup")
    .withExistingPrimaryNetworkInterface(networkInterface)
    .withLatestWindowsImage("MicrosoftWindowsServer", "WindowsServer", "2012-R2-Datacenter")
    .withAdminUsername("jenc")
    .withAdminPassword("aReallyGoodPasswordHere")
    .withComputerName(vmName)
    .withSize("Standard_DS1")
    .create();
```

Azure VM extensions

Let's assume you want to configure and install more software on your virtual machine after the initial deployment. You want this task to use a specific configuration, monitored and executed automatically.

Azure VM extensions are small applications that enable you to configure and automate tasks on Azure VMs after initial deployment.

For more information, see [Azure virtual machine extensions and features](#).

Azure Automation services

Saving time, reducing errors, and increasing efficiency are some of the most significant operational management challenges faced when managing remote infrastructure. If you have numerous infrastructure services, you might want to consider using higher-level services in Azure to help you operate from a higher level.

Azure Automation enables you to integrate services that allow you to automate frequent, time-consuming, and error-prone management tasks with ease. These services include **process automation**, **configuration management**, and **update management**.

- **Process Automation.** Let's assume you have a VM that is monitored for a specific error event. You want to take action, and fix the problem as soon as it's reported. Process automation enables you to set up watcher tasks that can respond to events that may occur in your datacenter.
- **Configuration Management.** Perhaps you want to track software updates that become available for the operating system that runs on your VM. There are specific updates you may want to include or exclude. Configuration management enables you to track these updates, and take action as required. You use **Microsoft Endpoint Configuration Manager** to manage your company's PC, servers, and mobile devices. You can extend this support to your Azure VMs with Configuration Manager.
- **Update Management.** Use this service to manage updates and patches for your VMs. With this service, you're able to assess the status of available updates, schedule installation, and review deployment results to verify updates applied successfully. Update management incorporates services that provide process and configuration management. You enable update management for a VM directly from your **Azure Automation** account. You can also enable update management for a single virtual machine from the virtual machine pane in the portal.

Auto-shutdown

Auto-shutdown is a feature in Azure that allows you to automatically shut down your VMs on a schedule. Use Auto-shutdown to save costs by ensuring that your VMs are not running when they aren't needed. You can set the schedule for auto-shutdown to occur daily or weekly, and you can also specify the time zone for the schedule.

To navigate to the Auto-shutdown feature in a VM in the Azure portal, go to the VM's blade in the portal, click on "Auto-shutdown" under the "Operations" section, and then configure the auto-shutdown settings according to your preferences.

The screenshot shows the Azure portal interface for a virtual machine named 'myVM'. The breadcrumb navigation at the top reads 'Home > myVM'. The main header area displays 'myVM | Auto-shutdown' with a clock icon and a star icon. Below the header is a search bar and a row of action buttons: '<<', 'Save', 'Discard', and 'Feedback'. On the left side, under the 'Operations' section, there is a list of options: 'Bastion', 'Auto-shutdown' (which is highlighted with a red border), and 'Backup'. The main content area on the right shows the configuration for 'Auto-shutdown':

- 'Enabled' is set to 'On' with a selected radio button.
- 'Scheduled shutdown' is set to '7:00:00 PM' in a text input field.
- 'Time zone' is set to '(UTC) Coordinated Universal Time' in a text input field.
- 'Send notification before auto-shutdown?' is set to 'No' with a selected radio button.

For more information, see [Auto-shutdown](#).

As you can see, Azure provides various tools to create and administer resources so that you can integrate management operations into a process *that works for you*. Let's examine some of the other Azure services to make sure your infrastructure resources are running smoothly.

5. Manage the availability of your Azure VMs

<https://learn.microsoft.com/en-us/training/modules/intro-to-azure-virtual-machines/5-high-availability>

Manage the availability of your Azure VMs

Completed

Frequently, the success of a services company is directly related to the service level agreements (SLA) the company has with its customers. Your customers expect the services you provide always to be available, and their data kept safe. This security is something that Microsoft takes seriously. Azure provides tools you can use to manage availability, data security, and monitoring, so you know your services are always available for your customers.

Administration of an Azure VM isn't limited to managing the operating system, or software that runs on the VM. It helps to know which services Azure provides that ensure service availability and support automation. These services help you to plan your organization's business continuity and disaster recovery strategy.

Here, we cover an Azure service that helps you improve VM availability, streamlines VM management tasks, and keeps your VM data backed up and safe. Let's start by defining availability.

What is availability?

Availability is the percentage of time a service is available for use.

Let's assume you have a website, and you want your customers to be able to always access information. Your expectation is 100% availability concerning website access.

Why do I need to think about availability when using Azure?

Azure VMs run on physical servers hosted within the Azure datacenter. As with most physical devices, there's a chance that there could be a failure. If the physical server fails, the virtual machines hosted on that server also fail. If a failure happens, Azure moves the VM to a healthy host server automatically. However, this self-healing migration could take several minutes, during which the application(s) hosted on that VM aren't available.

Periodic updates initiated by Azure itself can also affect the VMs. These maintenance events range from software updates to hardware upgrades and are required to improve platform reliability and performance. These events usually are performed without impacting any guest VMs, but sometimes the virtual machines reboot to complete an update or upgrade.

Availability zones

[Availability zones](#) expands the level of control you have to maintain the availability of the applications and data on your VMs. An Availability Zone is a physically separate zone, within an Azure region. There are three Availability Zones per supported Azure region.

Each Availability Zone has a distinct power source, network, and cooling. By designing your solutions to use replicated VMs in zones, you can protect your apps and data from the loss of a data center. If one zone is compromised, then replicated apps and data are instantly available in another zone.

Virtual Machines Scale Sets

[Azure virtual machine scale sets](#) let you create and manage a group of load balanced VMs. The number of VM instances can automatically increase or decrease in response to demand or a defined schedule. Scale sets provide high availability to your applications, and allow you to centrally manage, configure, and update many VMs. There's no cost for the scale set itself, you only pay for each VM instance that you create.

Virtual machines in a scale set can also be deployed into multiple availability zones, a single availability zone, or regionally. Availability zone deployment options may differ based on the [orchestration mode](#).

Load balancer

Combine the [Azure Load Balancer](#) with an availability zone or availability set to get the most application resiliency. The Azure Load Balancer distributes traffic between multiple virtual machines. For our Standard tier virtual machines, the Azure Load Balancer is included. Not all virtual machine tiers include the Azure Load Balancer. For more information about load balancing your virtual machines, see **Load Balancing virtual machines** for [Linux](#) or [Windows](#).

Azure Storage redundancy

Azure Storage always stores multiple copies of your data so that it's protected from planned and unplanned events, including transient hardware failures, network or power outages, and massive natural disasters. Redundancy ensures that your storage account meets its availability and durability targets even in the face of failures.

When deciding which redundancy option is best for your scenario, consider the tradeoffs between lower costs and higher availability. The factors that help determine which redundancy option you should choose include:

- How your data is replicated in the primary region
- Whether your data is replicated to a second region that is geographically distant to the primary region, to protect against regional disasters
- Whether your application requires read access to the replicated data in the secondary region if the primary region becomes unavailable for any reason

For more information, see [Azure Storage redundancy](#).

Failover across locations

You can also replicate your infrastructure across sites to handle regional failover. **Azure Site Recovery** replicates workloads from a primary site to a secondary location. If an outage happens at your primary site, you can fail over to a secondary location. This failover enables users to continue to access your applications without interruption. You can then fail back to the primary location after it's up and running again. Azure Site Recovery is about replication of virtual or physical machines; it keeps your workloads available in an outage.

While there are many attractive technical features to Site Recovery, there are at least two significant business advantages:

- Site Recovery enables the use of Azure as a destination for recovery, thus eliminating the cost and complexity of maintaining a secondary physical datacenter.
- Site Recovery makes it incredibly simple to test failovers for recovery drills without impacting production environments. This feature makes it easy to test your planned or unplanned failovers. After all, you don't have a good disaster recovery plan if you've never tried to fail over.

The recovery plans you create with Site Recovery can be as simple or as complex as your scenario requires. They can include custom PowerShell scripts, Azure Automation runbooks, or manual intervention steps. You can use the recovery plans to replicate workloads to Azure, easily enabling new opportunities for migration, temporary bursts during surge periods, or development and testing of new applications.

Azure Site Recovery works with Azure resources, or Hyper-V, VMware, and physical servers in your on-premises infrastructure. It can be a key part of your organization's business continuity and disaster recovery (BCDR) strategy by orchestrating the replication, failover, and recovery of workloads and applications if the primary location fails.

6. Back up your virtual machines

<https://learn.microsoft.com/en-us/training/modules/intro-to-azure-virtual-machines/6-backup-services>

Back up your virtual machines

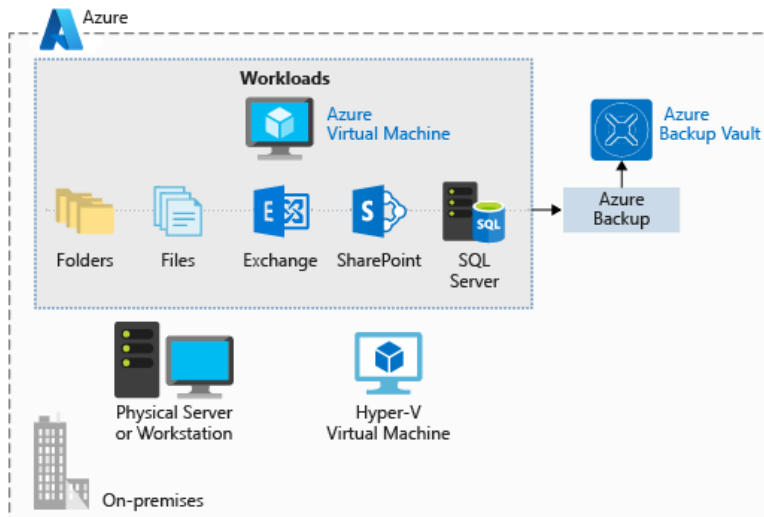
Completed

Data backup and recovery is a necessary piece of the planning for any good infrastructure. Assume a bug erases some company data, or maybe you need to retrieve some archived data for auditing purposes. Maintaining a good backup strategy ensures that you aren't scrambling when data or software needs to be restored.

Azure Backup is a *backup as a service* offering that protects physical or virtual machines no matter where they reside: on-premises or in the cloud.

Azure Backup can be used for a wide range of data backup scenarios, such as:

- Files and folders on Windows OS machines (physical or virtual, local or cloud)
- Application-aware snapshots (Volume Shadow Copy Service)
- Popular Microsoft server workloads such as Microsoft SQL Server, Microsoft SharePoint, and Microsoft Exchange
- Native support for Azure Virtual Machines, both Windows, and Linux
- Linux and Windows 10 client machines



Advantages of using Azure Backup

Traditional backup solutions don't always take full advantage of the underlying Azure platform. The result is a solution that tends to be expensive or inefficient. The solution either offers too much or too little storage, doesn't offer the correct types of storage, or has cumbersome and long-winded administrative tasks. Azure Backup was designed to work in tandem with other Azure services and provides several distinct benefits.

- **Automatic storage management.** Azure Backup automatically allocates and manages backup storage and uses a pay-as-you-use model. You only pay for what you use.
- **Unlimited scaling.** Azure Backup uses the power and scalability of Azure to deliver high availability.
- **Multiple storage options.** Azure Backup offers locally redundant storage where all copies of the data exist within the same region and geo-redundant storage where your data is replicated to a secondary region.
- **Unlimited data transfer.** Azure Backup doesn't limit the amount of inbound or outbound data you transfer. Azure Backup also doesn't charge for the data that is transferred.
- **Data encryption.** Data encryption allows for secure transmission and storage of your data in Azure.
- **Application-consistent backup.** An application-consistent backup means that a recovery point has all required data to restore the backup copy. Azure Backup provides application-consistent backups.
- **Long-term retention.** Azure doesn't limit the length of time you keep the backup data.

Use Azure Backup

Azure Backup uses several components that you download and deploy to each computer you want to back up. The component that you deploy depends on what you want to protect.

- Azure Backup agent

- System Center Data Protection Manager
- Azure Backup Server
- Azure Backup VM extension

Azure Backup uses a Recovery Services vault for storing the backup data. Azure Storage blobs back up a vault, making it an efficient and economical long-term storage medium. With the vault in place, you can select the machines to back up, and define a backup policy (when snapshots are taken and for how long they're stored).

7. Module assessment

<https://learn.microsoft.com/en-us/training/modules/intro-to-azure-virtual-machines/7-knowledge-check>

Module assessment

Completed

8. Summary

<https://learn.microsoft.com/en-us/training/modules/intro-to-azure-virtual-machines/8-summary>

Summary

Completed

In this module, you looked at the decisions you need to make before creating a virtual machine. These decisions include aspects such as the VM size, types of disks used, operating system image selected, and the types of resources created.

You also looked at the options to create and manage virtual machines in Azure. You saw how easy it is to create and manage VMs using the portal. You also learned when to use Resource Manager templates, PowerShell, the Azure CLI, and the Azure Client SDK.

Finally, you looked at the extensions and services available to more easily administer your VMs.

Important

In the optional exercises for this module, you created resources by using your own Azure subscription. Clean up these resources so that you won't continue to be charged for them.

Configure virtual machine availability

1. Introduction

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/1-introduction>

Introduction

Completed

Managing virtual machines at scale can be challenging, especially when usage patterns vary and demands on applications fluctuate. Azure Administrators need to be able to adjust their virtual machine resources to match changing demands. At the same time, they need to keep their virtual machine configuration consistent to ensure application stability. Achieving these goals means maintaining throughput and responsiveness while minimizing the costs of continually running a large collection of virtual machines.

Your company website uses virtual machines and manages large workloads. The IT department wants to ensure the virtual machines can dynamically adjust to increases and decreases in workloads. They also want to ensure there's a business continuity plan to provide for highly available machines. You're responsible for deploying highly available virtual machines. You decide to use Azure Virtual Machine Scale Sets and the autoscale feature.

In this module, you learn about scaling virtual machines. You learn about availability zones, availability sets, and update and fault domains. You also learn about scale sets and autoscale.

The goal of this module is to learn how to successfully respond to changing virtual machine workloads.

Learning objectives

In this module, you learn how to:

- Implement availability sets and availability zones.
- Implement update and fault domains.
- Implement Azure Virtual Machine Scale Sets.
- Autoscale virtual machines.

Skills measured

The content in the module helps you prepare for [Exam AZ-104: Microsoft Azure Administrator](#).

Prerequisites

- Familiarity with creating and managing Azure virtual machines.
- General knowledge of scaling infrastructure resources in fluctuating workloads.

2. Plan for maintenance and downtime

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/2-plan-for-maintenance-downtime>

Plan for maintenance and downtime

Completed

Azure Administrators prepare for planned and unplanned failures.

Things to know about maintenance planning

An availability plan for Azure virtual machines needs to include strategies for unplanned hardware maintenance, unexpected downtime, and planned maintenance.

- An **unplanned hardware maintenance** event occurs when the Azure platform predicts that the hardware or any platform component associated to a physical machine is about to fail. When the platform predicts a failure, it issues an unplanned hardware maintenance event. Azure uses Live Migration technology to migrate your virtual machines from the failing hardware to a healthy physical machine. Live Migration is a virtual machine preserving operation that only pauses the virtual machine for a short time, but performance might be reduced before or after the event.
- **Unexpected downtime** occurs when the hardware or the physical infrastructure for your virtual machine fails unexpectedly. Unexpected downtime can include local network failures, local disk failures, or other rack level failures. When detected, the Azure platform automatically migrates (heals) your virtual machine to a healthy physical machine in the same datacenter. During the healing procedure, virtual machines experience downtime (reboot) and in some cases loss of the temporary drive.
- **Planned maintenance** events are periodic updates made by Microsoft to the underlying Azure platform to improve overall reliability, performance, and security of the platform infrastructure that your virtual machines run on.

Note

Microsoft doesn't automatically update your virtual machine operating system or other software. You have complete control and responsibility for those updates. However, the underlying software host and hardware are periodically patched to ensure reliability and high performance.

3. Create availability sets

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/3-setup-availability-sets>

Create availability sets

Completed

An availability set is a logical feature you can use to ensure a group of related virtual machines are deployed together. The grouping helps to prevent a single point of failure from affecting all of your machines. The grouping ensures that not all of the machines are upgraded at the same time during a host operating system upgrade in the datacenter.

Things to know about availability sets

Let's review some characteristics of availability sets.

- All virtual machines in an availability set should perform the identical set of functionalities.
- All virtual machines in an availability set should have the same software installed.
- Azure ensures that virtual machines in an availability set run across multiple physical servers, compute racks, storage units, and network switches.

If a hardware or Azure software failure occurs, only a subset of the virtual machines in the availability set are affected. Your application stays up and continues to be available to your customers.

- You can create a virtual machine and an availability set at the same time.

A virtual machine can only be added to an availability set when the virtual machine is created. To change the availability set for a virtual machine, you need to delete and then recreate the virtual machine.

- You can build availability sets by using the Azure portal, Azure Resource Manager (ARM) templates, scripting, or API tools.

Note

Adding your virtual machines to an availability set doesn't protect your applications from operating system or application-specific failures. You need to explore other disaster recovery and backup techniques to provide application-level protection.

Things to consider when using availability sets

Availability sets are an essential capability when you want to build reliable cloud solutions. In your planning for availability sets, keep the following general principles in mind:

- **Consider redundancy.** To achieve redundancy in your configuration, place multiple virtual machines in an availability set.
- **Consider separation of application tiers.** Each application tier exercised in your configuration should be located in a separate availability set. The separation helps to mitigate single point of failure on all machines.
- **Consider load balancing.** For high availability and network performance, create a load-balanced availability set by using Azure Load Balancer. Load Balancer distributes incoming traffic across working instances of services that are defined in your load-balanced availability set.
- **Consider managed disks.** You can use Azure managed disks with your Azure virtual machines in availability sets for block-level storage.

4. Review update domains and fault domains

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/4-review-update-fault-domains>

Review update domains and fault domains

Completed

Azure Virtual Machine Availability Sets implements two node concepts to help Azure maintain high availability and fault tolerance when deploying and upgrading applications: *update domains* and *fault domains*. Each virtual machine in an availability set is placed in one update domain and one fault domain.

Things to know about update domains

An update domain is a group of nodes that are upgraded together during the process of a service upgrade (or *roll out*). An update domain allows Azure to perform incremental or rolling upgrades across a deployment. Here are some other characteristics of update domains.

- Each update domain contains a set of virtual machines and associated physical hardware that can be updated and rebooted at the same time.
- During planned maintenance, only one update domain is rebooted at a time.
- You can specify between 1 and 20 update domains when creating an availability set. If you don't specify a value, Azure defaults to five update domains.
- The update domain count is immutable after creation; to change it, you must delete and recreate the availability set.

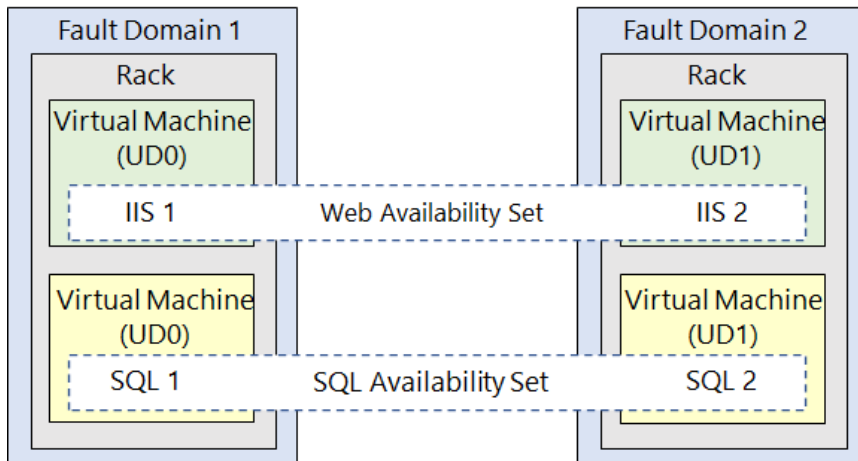
Things to know about fault domains

A fault domain is a group of nodes that represent a physical unit of failure. Think of a fault domain as nodes that belong to the same physical rack.

- A fault domain defines a group of virtual machines that share a common set of hardware (or *switches*) that share a single point of failure. An example is a server rack serviced by a set of power or networking switches.

- Two fault domains work together to mitigate against hardware failures, network outages, power interruptions, or software updates.

Let's look at a scenario with two fault domains that have two virtual machines each. The virtual machines in each fault domain are contained in different availability sets. The web availability set contains two virtual machines with one machine from each fault domain. The SQL availability set contains two different virtual machines with one from each fault domain.



5. Review availability zones

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/5-review-availability-zones>

Review availability zones

Completed

Availability zones are a high-availability offering that protects your applications and data from datacenter failures. You can use availability zones to build high-availability into your application architecture by colocating your compute, storage, networking, and data resources within a zone and replicating in other zones.

Consider a scenario where you create three or more virtual machines across three zones in an Azure region. Your virtual machines are effectively distributed across three fault domains and three update domains. The Azure platform recognizes this distribution across update domains to make sure that virtual machines in different zones aren't updated at the same time.

Things to know about availability zones

Review the following characteristics of availability zones.

- Availability zones are unique physical locations within an Azure region.
- Each zone is made up of one or more datacenters that are equipped with independent power, cooling, and networking.
- To ensure resiliency, there's a minimum of three separate zones in all enabled regions.
- The physical separation of availability zones within a region protects applications and data from datacenter failures.
- Zone-redundant services replicate your applications and data across availability zones to protect against single-points-of-failure.

Things to consider when using availability zones

Azure services that support availability zones are divided into two categories.

Category	Description	Examples
Zonal services	Azure <i>zonal</i> services pin each resource to a specific zone.	- Azure virtual machines - Azure managed disks
Zone-redundant services	For Azure services that are zone-redundant, the platform replicates automatically across all zones.	- Azure Storage that's zone-redundant - Azure SQL Database

Note

Note: Standard IP addresses can be configured as zone-redundant (recommended for high availability), zonal (pinned to a specific zone), or non-zonal (regional) depending on your deployment choice.

Tip

To achieve comprehensive business continuity on Azure, build your application architecture with a combination of availability zones and Azure [regional pairs](#).

6. Compare vertical and horizontal scaling

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/6-compare-vertical-horizontal-scaling>

Compare vertical and horizontal scaling

Completed

A robust virtual machine configuration includes support for scalability. Scalability allows throughput for a virtual machine in proportion to the availability of the associated hardware resources. A scalable virtual machine can handle increases in requests without adversely affecting response time and throughput. For most scaling operations, there are two implementation options: *vertical* and *horizontal*.

Things to know about vertical scaling

Vertical scaling, also known as *scale up and scale down*, involves increasing or decreasing the virtual machine **size** in response to a workload. Vertical scaling makes a virtual machine more (scale up) or less (scale down) powerful.

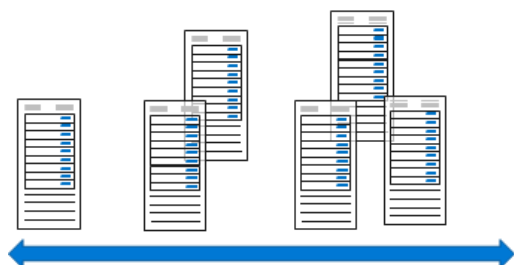


Here are some scenarios where using vertical scaling can be advantageous:

- If you have a service built on a virtual machine that's under-utilized such as on the weekend, you can use vertical scaling to decrease the virtual machine size and reduce your monthly costs.
- You can implement vertical scaling to increase your virtual machine size to support larger demand without having to create extra virtual machines.

Things to know about horizontal scaling

Horizontal scaling, also referred to as *scale out and scale in*, is used to adjust the **number** of virtual machines in your configuration to support the changing workload. When you implement horizontal scaling, there's an increase (scale out) or decrease (scale in) in the number of virtual machine instances.



Things to consider when using vertical and horizontal scaling

Review the following considerations regarding vertical and horizontal scaling. Think about which implementation might be required to support your company website.

- **Consider limitations.** Generally speaking, horizontal scaling has fewer limitations than vertical scaling. A vertical scaling implementation depends on the availability of larger hardware, which quickly hits an upper limit and can vary by region. Vertical scaling also usually requires a virtual machine to stop and restart, which can temporarily limit access to applications or data.
- **Consider flexibility.** When operating in the cloud, horizontal scaling is more flexible. A horizontal scaling implementation allows you to run potentially thousands of virtual machines to manage changes in workload and throughput.
- **Consider reprovisioning.** *Reprovisioning* is the process of removing an existing virtual machine and replacing it with a new machine. A robust availability plan considers where reprovisioning might be required and plans for interruptions to service. If reprovisioning might be required, determine if any data needs to be maintained and migrated to the new machine.

7. Implement Azure Virtual Machine Scale Sets

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/7-implement-scale-sets>

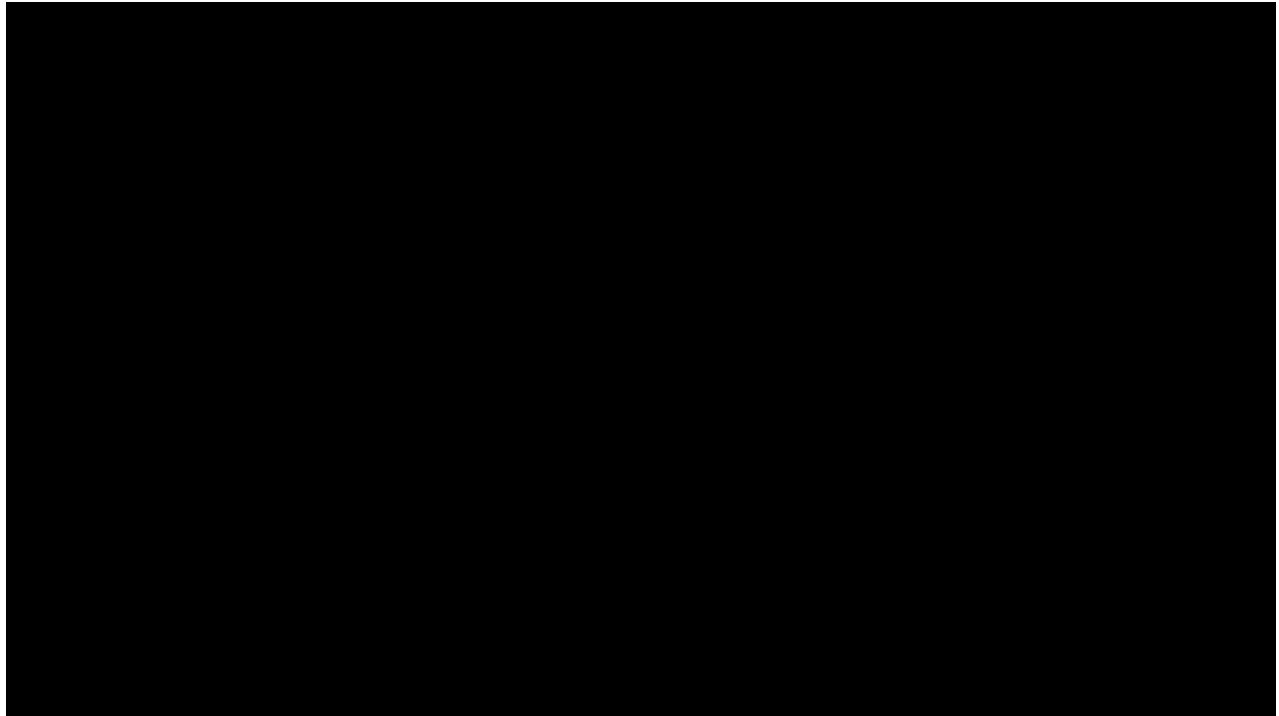
Implement Azure Virtual Machine Scale Sets

Completed

Azure Virtual Machine Scale Sets are an Azure Compute resource that you can use to deploy and manage a set of **identical** virtual machines. When you implement Virtual Machine Scale Sets and configure all your virtual machines in the same way, you gain true *autoscaling*. Virtual Machine Scale Sets automatically increases the number of your virtual machine instances as application demand increases, and reduces the number of machine instances as demand decreases.

With Virtual Machine Scale Sets, you don't need to pre-provision your virtual machines. It's easier to build large-scale services that target large compute, big data, and containerized workloads. As workloads increase, more virtual machine instances can be added. As workloads decrease, virtual machines instances can be removed. The process of adding and removing machines can be manual or automated, or a combination of both.

Increase app availability and scalability with Azure Virtual Machine Scale Sets



Things to know about Azure Virtual Machine Scale Sets

Review the following characteristics of Azure Virtual Machine Scale Sets.

- Virtual Machine Scale Sets support the use of Azure Load Balancer for basic layer-4 traffic distribution, and Azure Application Gateway for more advanced layer-7 traffic distribution and TLS/SSL termination.
- You can use Virtual Machine Scale Sets to run multiple instances of your application. If one of the virtual machine instances has a problem, customers continue to access your application through another virtual machine instance with minimal interruption.
- There are two types of orchestration modes available for Azure virtual machine scale sets: Uniform and Flexible. In **Uniform orchestration mode**, all virtual machine instances are created from the same base operating system image and configuration. In **Flexible orchestration mode**, VMs can use different images, sizes, or configurations within the same scale set. The orchestration mode must be chosen when the scale set is created.
- Customer demand for your application might change throughout the day or week. To meet customer demand, Virtual Machine Scale Sets implements autoscaling to automatically increase and decrease the number of virtual machines.

8. Create Virtual Machine Scale Sets

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/8-create-scale-sets>

Create Virtual Machine Scale Sets

Completed

You can implement Azure Virtual Machine Scale Sets in the Azure portal. You specify the number of virtual machines and their sizes, and indicate preferences for using Azure Spot instances, Azure managed disks, and allocation policies.

In the Azure portal, there are several settings to configure to create an Azure Virtual Machine Scale Sets implementation.

Create a virtual machine scale set ...

[Basics](#) [Disks](#) [Networking](#) [Scaling](#) [Management](#) [Health](#) [Advanced](#) [Tags](#) [Review + create](#)

Azure virtual machine scale sets let you create and manage a group of load balanced VMs. The number of VM instances can automatically increase or decrease in response to demand or a defined schedule. Scale sets provide high availability to your applications, and allow you to centrally manage, configure, and update a large number of VMs.

Orchestration

A scale set has a "scale set model" that defines the attributes of virtual machine instances (size, number of data disks, etc). As the number of instances in the scale set changes, new instances are added based on the scale set model.

Orchestration mode * ⓘ ☒ **Uniform:** optimized for large scale stateless workloads with identical instances
☐ **Flexible:** achieve high availability at scale with identical or multiple virtual

Instance details

Image * ⓘ

Ubuntu Server 20.04 LTS - x64 Gen2

[See all images](#) | [Configure VM generation](#)

VM architecture ⓘ ☐ Arm64
☒ x64

Run with Azure Spot discount ⓘ ☐

Size * ⓘ

Standard_D2s_v3 - 2 vcpus, 8 GiB memory

[See all sizes](#)

- **Orchestration mode:** Choose how the scale set manages the virtual machines. Flexible orchestration is the default and recommended mode for new deployments. For most new workloads, accept the Flexible default unless you have a specific requirement for identical instances.
- **Image:** Choose the base operating system or application for the VM.
- **VM Architecture:** Azure provides a choice of x64 or Arm64-based virtual machines to run your applications. x64-based VMs provide the most software compatibility. Arm64-based VMs provide up to 50% better price-performance than comparable x64 VMs.
- **Size:** Select a VM size to support the workload that you want to run. The size that you choose then determines factors such as processing power, memory, and storage capacity. Azure offers a wide variety of sizes to support many types of uses. Azure charges an hourly price based on the VM's size and operating system.

Under the **Advanced** tab, you can also select the following:

- **Spreading algorithm:** The spreading algorithm determines how VMs in the scale set are balanced across fault domains. With max spreading, VMs are spread across as many fault domains as possible in each zone. With fixed spreading, VMs are always spread across exactly five fault domains. In the case where fewer than five fault domains are available, a scale set using "Max spreading" completes, while a scale set using "Fixed spreading" fails. For this reason, Microsoft recommends using **Max spreading** for your implementation.

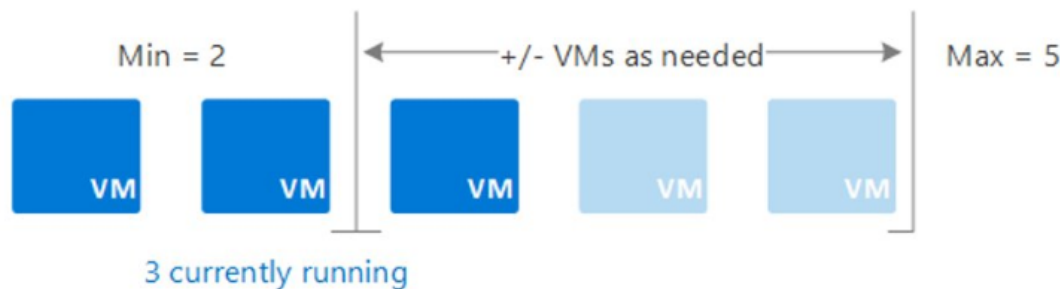
9. Implement autoscale

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/9-implement-autoscale>

Implement autoscale

Completed

An Azure Virtual Machine Scale Sets implementation can automatically increase or decrease the number of virtual machine instances that run your application. This process is known as *autoscaling*. Autoscaling allows you to dynamically scale your configuration to meet changing workload demands.



Autoscaling minimizes the number of unnecessary virtual machine instances that run your application when demand is low. Your customers continue to receive an acceptable level of performance as demand grows and more virtual machine instances are automatically added.

Things to consider when using autoscaling

Review the following considerations about autoscaling. Think about how this process can benefit your company website implementation.

- **Consider automatic adjusted capacity.** You can create autoscaling rules to define the acceptable performance for a positive customer experience. When the defined thresholds are met, the autoscale rules act to adjust the capacity of your Virtual Machine Scale Sets implementation.
- **Consider scale out.** If your application demand increases, the load on the virtual machine instances in your implementation increases. If the increased load is consistent, rather than a brief demand, you can configure autoscale rules to increase the number of virtual machine instances in your implementation.
- **Consider scale in.** On an evening or weekend, your application demand might decrease. If the decreased load is consistent over a period of time, you can configure autoscale rules to decrease the number of virtual machine instances in your implementation. The scale-in action reduces the cost to run your Virtual Machine Scale Sets implementation as you only run the number of instances required to meet the current demand.
- **Consider scheduled events.** You can implement autoscaling and schedule events to automatically increase or decrease the capacity of your implementation at fixed times.
- **Consider overhead.** Using Azure Virtual Machine Scale Sets with autoscaling reduces your management overhead to monitor and optimize the performance of your application.

10. Configure autoscale

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/10-configure-autoscale>

Configure autoscale

Completed

When you create an Azure Virtual Machine Scale Sets implementation in the Azure portal, you can enable manual or autoscaling. For optimal performance, you should define a minimum, maximum, and default number of virtual machine instances to use.

In the Azure portal, you can select the scaling mode.

Create a virtual machine scale set ...

Scaling

Scaling mode ⓘ

- ☒ **Manually update the capacity:** Maintain a fixed amount of instances.
- ☐ **Autoscaling:** Scaling based on a CPU metric, on any schedule.
- ☐ **No scaling profile:** manual attach virtual machines after deployment
-  Improve your availability by selecting multiple zones

Instance count * ⓘ

[Configure scaling options](#)

Scaling mode

- **Manually update the capacity:** Maintain a fixed instances count. Set the **Instance count** to the number of virtual machines in the scale set (0 - 1000). Configure the **Scale-in policy** which is the order virtual machines are selected for deletion. For example, you could balance across zones and then delete the virtual machine with the highest instance ID.
- **Autoscaling:** Scaling based on a CPU metric, or any schedule.

Configure autoscaling

Autoscaling is based on a scaling condition.

Add a scaling condition



Scale mode

- ☒ **Autoscaling:** Scaling based on a CPU metric, on any schedule

Default instance count * ⓘ

Instance limit

Minimum * ⓘ

Maximum * ⓘ

Scale out

CPU threshold greater than * ⓘ

Increase instance count by * ⓘ

Scale in

CPU threshold less than * ⓘ

Decrease instance count by * ⓘ

Query duration

Minutes * ⓘ

The engine will query CPU usage for the past 60 minutes before executing the scaling to avoid reacting to transient spikes.

Schedule

Schedule type *

- ☒ Specify start/end dates
- ☐ Repeat specific days

- **Default instance count.** The initial number of virtual machines deployed in this scale set (0-1000).
- **Instance limit.** The minimum instance count you want this condition to scale down to. The maximum instance count you want this condition to scale up to.

- **Scale out.** The CPU usage percentage threshold for triggering the scale-out autoscale rule. The number of instances to scale out by.
- **Scale in.** The CPU usage percentage threshold for triggering the scale-in autoscale rule. The number of instances to scale in by.
- **Query duration:** This duration is the time the Autoscale engine looks back for the metric usage average. This look back is to allow your metric to stabilize.
- **Schedule:** Specify the start and end dates. You can also repeat the schedule on specific days.

11. Module assessment

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/11-knowledge-check>

Module assessment

Completed

12. Summary and resources

<https://learn.microsoft.com/en-us/training/modules/configure-virtual-machine-availability/12-summary-resources>

Summary and resources

Completed

Azure provides several high availability options for virtual machines. You can achieve high availability by using availability sets, availability zones, and Azure Virtual Machine Scale Sets.

In this module, you learned how to configure virtual machine availability by using availability sets and availability zones with update and fault domains. You discovered how to autoscale virtual machines and configure vertical and horizontal scaling. You reviewed how to implement Virtual Machine Scale Sets, including storage resiliency and scalability options.

The main takeaways from this module are:

- Azure Virtual Machine Scale Sets allow for the deployment and management of a group of identical virtual machines, making it easier to build large-scale services.
- Autoscaling with Virtual Machine Scale Sets helps optimize performance by automatically adjusting the number of instances based on workload demands.
- Availability sets and availability zones are important features in Azure for achieving high availability and fault tolerance for virtual machines.

Learn more with Copilot

Copilot can assist you in configuring Azure infrastructure solutions. Copilot can compare, recommend, explain, and research products and services where you need more information. Open a Microsoft Edge browser and choose Copilot (top right) or navigate to copilot.microsoft.com. Take a few minutes to try these prompts and extend your learning with Copilot.

- How do virtual machine scale sets work with Azure availability zones and sets?
- What is the difference between manual and autoscaling of Azure virtual machines?

Learn more with documentation

- [Availability options for Azure Virtual Machines](#). This article provides an overview of the availability options for Azure virtual machines (VMs).
- [Autoscale with Azure Virtual Machine Scale Sets](#). This article reviews when use Virtual Machine Scale Sets.
- [Create virtual machines in a scale set using Azure portal](#). This article steps through using Azure portal to create a Virtual Machine Scale Set.

Learn more with self-paced training

- [Introduction to Azure Virtual Machines \(sandbox\)](#). Learn about the decisions you make before creating a virtual machine, the options to create and manage the VM, and the extensions and services you use to manage your VM.
- [Implement scale and high availability with Windows Server VM](#). You learn how to implement scaling for virtual machine scale sets and load-balanced VMs. You also learn how to implement Azure Site Recovery.
- [Introduction to Azure Virtual Machine Scale Sets](#). Learn about what Azure Virtual Machine Scale Sets do, how they work, and when you should use Azure Virtual Machine Scale Sets as a solution for your organization.

Configure Azure App Service plans

1. Introduction

<https://learn.microsoft.com/en-us/training/modules/configure-app-service-plans/1-introduction>

Introduction

Completed

Azure Administrators need to be able to scale a web application. Scaling enables an application to remain responsive during periods of high demand. Scaling also helps to save money by reducing the resources required when demand drops.

Suppose you work for a large chain of hotels. You're responsible for maintaining the hotel website. Customers visit the website to make new reservations and view details for their current bookings. At certain times of the year, the volume of website traffic grows because customers are browsing hotels for vacations during national/regional holidays. At other times, traffic declines. These website usage patterns are predictable.

In this module, you learn to implement Azure App Service plans. You learn how different App Service plans provide different pricing and scaling options. You learn how changing the plan affects performance.

The goal of this module is to ensure you can determine the best App Service plan for your application.

Learning objectives

In this module, you learn how to:

- Select an appropriate Azure App Service plan pricing tier.
- Scale an Azure App Service plan.

Skills measured

The content in the module helps you prepare for [Exam AZ-104: Microsoft Azure Administrator](#).

Prerequisites

- Basic knowledge of scaling and performance concepts.
- Familiarity with the Azure portal so you can configure the correct App Service plan.

2. Implement Azure App Service plans

<https://learn.microsoft.com/en-us/training/modules/configure-app-service-plans/2-implement-azure>

Implement Azure App Service plans

Completed

An App Service plan defines a set of compute resources for a web application to run. The compute resources are analogous to a server farm in conventional web hosting. One or more applications can be configured to run on the same computing resources (or in the same

App Service plan).

Things to know about App Service plans

Let's take a closer look at how to implement and use an App Service plan with your virtual machines.

- When you create an App Service plan in a region, a set of compute resources is created for the plan in the specified region. Any applications that you place into the plan run on the compute resources defined by the plan.
- Each App Service plan defines these settings:
 - **Operating system:** Linux or Windows.
 - **Region:** The region for the App Service plan, such as West US, Central India, North Europe, and so on.
 - **Pricing tier:** Determines what App Service features you get and how much you pay for the plan. The pricing tiers available to your App Service plan depend on the operating system selected at creation time.
 - **Number of VM instances:** Determined by your plan.
 - **Size of VM instances:** Defined by CPU, memory, and remote storage.
- You can continue to add new applications to an existing plan as long as the plan has enough resources to handle the increasing load.

Things to consider when using App Service plans

Review the following considerations about using Azure App Service plans to run and scale your applications. Think about what conditions might apply to running and scaling the hotel website.

- **Consider cost savings.** Because you pay for the computing resources that your App Service plan allocates, you can potentially save money by placing multiple applications into the same App Service plan.
- **Consider multiple applications in one plan.** Create a single plan to support multiple applications, to make it easier to configure and maintain shared virtual machine instances. Because the applications share the same virtual machine instances, you need to carefully manage your plan resources and capacity.
- **Consider plan capacity.** Before you add a new application to an existing plan, determine the resource requirements for the new application and identify the remaining capacity of your plan.

Important

Overloading an App Service plan can potentially cause downtime for new and existing applications.

- **Consider application isolation.** Isolate your application into a new App Service plan when:
 - The application is resource-intensive.
 - You want to scale the application independently from the other applications in the existing plan.
 - The application needs resource in a different geographical region.

3. Determine Azure App Service plan pricing

<https://learn.microsoft.com/en-us/training/modules/configure-app-service-plans/3-determine-plan-pricing>

Determine Azure App Service plan pricing

Completed

The pricing tier of an Azure App Service plan determines what App Service features you get and how much you pay for the plan. Pricing tier examples are: Free, Shared, Basic, Standard, Premium, PremiumV2, PremiumV3, Isolated, and IsolatedV2.

How applications run and scale in App Service plans

The Azure App Service plan is the scale unit of App Service applications. Depending on the pricing tier for your Azure App Service plan, your applications run and scale in a different manner. If your plan is configured to run five virtual machine instances, then all applications in the plan run on all five instances. If your plan is configured for autoscaling, then all applications in the plan are scaled out together based on the autoscale settings.

The pricing tiers are grouped into three categories:

- **Shared compute:**
 - Free and Shared, the two base tiers, run an app on the same Azure VM as other App Service apps, including apps of other customers.
 - These tiers allocate CPU quotas to each app that runs on the shared resources, and the resources can't scale out.
 - These tiers are intended to be used only for development and testing purposes.
- **Dedicated compute:**
 - The Basic, Standard, Premium, PremiumV2, and PremiumV3 tiers run apps on dedicated Azure VMs.
 - Only apps in the same App Service plan have the same compute resources. The higher the tier, the more VM instances that are available to you for scale-out.
- **Isolated:**
 - The Isolated and IsolatedV2 tiers run dedicated Azure VMs on dedicated Azure virtual networks.
 - This tier provides network isolation on top of compute isolation to your apps.
 - This tier provides the maximum scale-out capabilities.

Here's a sample of different [plan details](#).

Feature	Free F1	Basic B1	Standard S1	Premium P1V3	Isolated V2
Usage	Development, Testing	Development, Testing	Production workloads	Enhanced scale, performance	Network-isolated workloads
Staging slots	N/A	N/A	5	20	20
Auto scale	N/A	Manual	Rules	Rules, Elastic	Rules
Scale instances	N/A	3	10	30	200
Daily backups	N/A	N/A	10	50	50

Free and Shared

The Free and Shared service plans are base tiers that run on the same Azure virtual machines as other applications. Some applications might belong to other customers. These tiers are intended to be used for development and testing purposes only. No SLA is provided for the Free and Shared service plans. Free and Shared plans are metered on a per application basis.

Basic

The Basic service plan is designed for applications that have lower traffic requirements, and don't need advanced auto scale and traffic management features. Pricing is based on the size and number of instances you run. Built-in network load-balancing support automatically distributes traffic across instances. The Basic service plan with Linux runtime environments supports Web App for Containers.

Standard

The Standard service plan is designed for running production workloads. Pricing is based on the size and number of instances you run. Built-in network load-balancing support automatically distributes traffic across instances. The Standard plan includes auto scale that can automatically adjust the number of virtual machine instances running to match your traffic needs. The Standard service plan with Linux runtime environments supports Web App for Containers.

Premium

The Premium service plan is designed for production apps that need higher performance and scale. PremiumV3 is the current Premium tier, offering D4v4 and D4v4-series virtual machines and SSD storage. PremiumV3 supports standard compute SKUs and memory-optimized SKUs for high-memory workloads. PremiumV3 supports both rule-based autoscaling and automatic scaling. PremiumV3 is recommended for new deployments.

Isolated

The Isolated service plan supports mission-critical workloads needing network isolation. IsolatedV2 is the preferred tier offering newer hardware, up to 200 instances, private environments, and enhanced security. IsolatedV2 is recommended for new workloads due to better performance and simpler pricing.

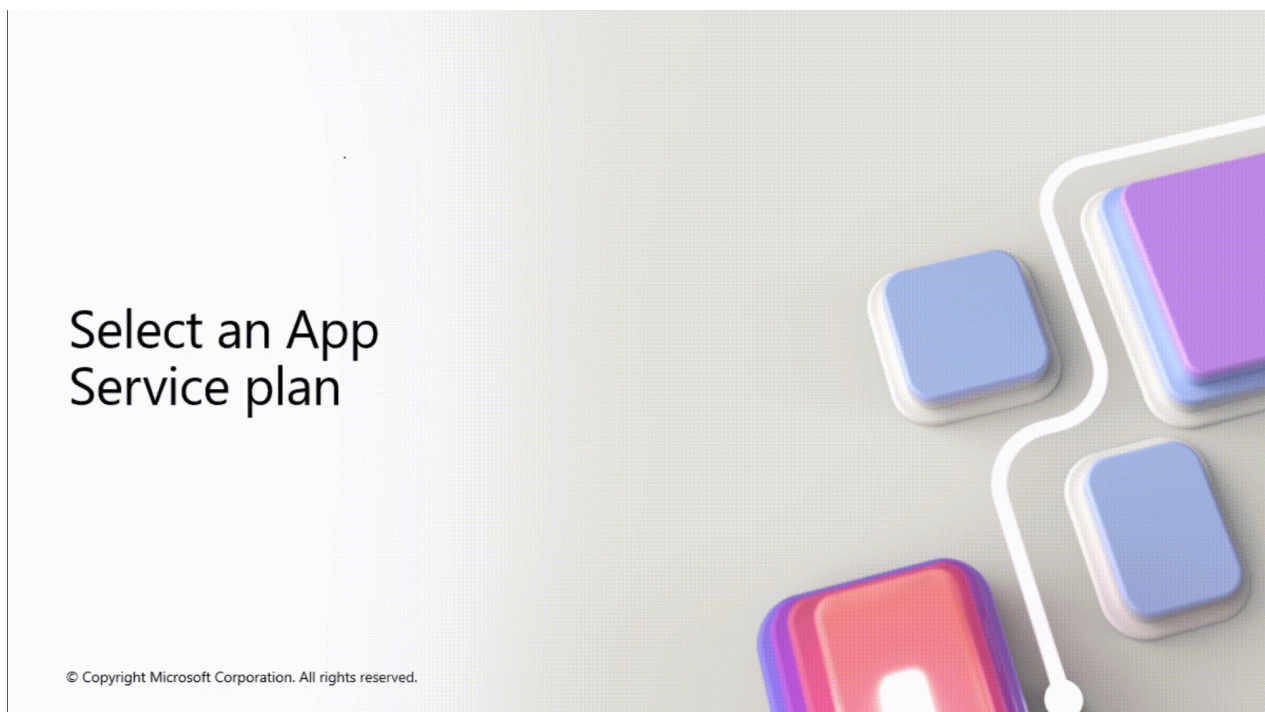
Task to be done: Select an App Service plan

You can view the available App Service plans in the Azure portal. You can make your choice based on hardware or feature requirements. Hardware considerations include CPU, memory, and scaling instances. Feature considerations include backups, staging slots, and zone redundancy.

Tip

When selecting a service plan, consider both hardware and feature requirements.

1. In the Azure portal search for and select **App Service plans**.
2. **Create** a new App Service plan.
3. Select **Explore pricing plans** to view the available plans.



4. Scale up and scale out Azure App Service

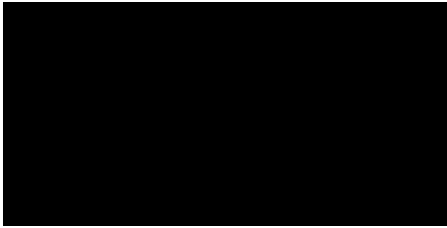
<https://learn.microsoft.com/en-us/training/modules/configure-app-service-plans/4-scale-up-scale-out>

Scale up and scale out Azure App Service

Completed

There are two methods for scaling your Azure App Service plan and applications: *scale up* and *scale out*. You can scale your applications manually or automatically, which is referred to as *autoscale*.

Watch the following video about how to implement automatic scaling for your Azure App Service plan and applications.



Things to know about Azure App Service scaling

Let's examine the details of scaling for your Azure App Service plan and App Service applications.

- The scale up method increases the amount of CPU, memory, and disk space. Scaling up gives you extra features like dedicated virtual machines, custom domains and certificates, staging slots, autoscaling, and more. You scale up by changing the pricing tier of the Azure App Service plan where your application is placed.
- The scale-out method increases the number of virtual machine instances that run your application. You can scale out to the maximum number of instances for your pricing tier. Take advantage of App Service Environments in the Isolated tier to further increase your scale-out count to 100 instances. The scale instance count can be configured manually or automatically (autoscale).
- With autoscale, you can automatically increase the scale instance count for the scale-out method. Autoscale is based on predefined rules and schedules.
- Your App Service plan can be scaled up and down at any time by changing the pricing tier of the plan.

Things to consider when using Azure App Service scaling

Review the following benefits of implementing scaling for your App Service plan and applications. Think about the scaling advantages for your hotel website.

- **Consider manually adjusting plan tiers.** Start your plan at a lower pricing tier and scale up as needed to acquire more App Service features. Scale down when features are no longer needed, and control your overall costs.

Consider a scenario where you start testing your web app by using the Azure App Service Free tier, where you pay nothing to use the service. After a while, you decide to add a custom DNS name to your web app, so you scale your plan up to the Shared tier. Next, you discover you need to create an SSL binding, so you scale your plan up to the Basic tier. Later, you determine a need for staging environments, so you scale up to the Standard tier. When you need more cores, memory, or storage, you can scale up to a bigger virtual machine size in the same tier.

The same scaling process works in reverse. If you decide you no longer need capabilities or features of a higher tier, scale your plan down to a lower tier and save money.

- **Consider autoscale to support users and reduce costs.** Keep serving your users when your application is experiencing high throughput. Implement autoscale to control how many features and support are offered at a given time based on your preference settings and rule conditions. Autoscale helps you save money when the load on your application decreases by automatically reducing your subscribed features.
- **Consider no redeployment.** When you change your scale settings, you don't need to change your code or redeploy your applications. Changing your plan scale settings takes only seconds to apply. Your changes affect all applications in your App Service plan.
- **Consider scaling for other Azure services.** If your App Service application depends on other Azure services, such as Azure SQL Database or Azure Storage, you can scale these resources separately. The App Service Plan doesn't manage these resources.

5. Configure Azure App Service autoscale

<https://learn.microsoft.com/en-us/training/modules/configure-app-service-plans/5-plan-scaling>

Configure Azure App Service autoscale

Completed

The autoscale process allows you to have the right amount of resources running to handle the load on your application. You can add resources to support increases in load and save money by removing idle resources.

Things to know about autoscale

Let's take a closer look at how to use autoscale for your Azure App Service plan and applications.

- To use autoscale, you specify the minimum, and maximum number of instances to run by using a set of rules and conditions.
- When your application runs under autoscale conditions, the number of virtual machine instances are automatically adjusted based on your rules. When rule conditions are met, one or more autoscale actions are triggered.
- An autoscale setting is used by the autoscale engine to determine whether to scale out or in. Autoscale settings are grouped into profiles.
- Autoscale rules include a trigger and a scale action (in or out). The trigger can be metric-based or time-based.

Settings

Scale out (App Service plan)

Choose how to scale your resource

Manual scale ☐ Maintain a fixed instance count

Custom autoscale ☒ Scale on any schedule, based on any metrics

Profile 1

Scale mode ☐ Scale based on a metric ☒ Scale to a specific instance count

Instance count *

Schedule ☒ Specify start/end dates ☐ Repeat specific days

Timezone

Start date

End date

- **Metric-based** rules measure application load and add or remove virtual machines based on the load, such as "do this action when CPU usage is above 50%." Example metrics include CPU time, Average response time, and Requests.
- **Time-based** rules (or, schedule-based) allow you to scale when you see time patterns in your load and want to scale before a possible load increase or decrease occurs. An example is "trigger a webhook every 8:00 AM on Saturday in a given time zone."
- The autoscale engine uses notification settings.

A notification setting defines what notifications should occur when an autoscale event occurs based on satisfying the criteria of an autoscale setting profile. Autoscale can notify one or more email addresses or make calls to one or more webhooks.

Things to consider when configuring autoscale

There are several considerations to keep in mind when you configure autoscale for your Azure App Service plan and applications.

- **Minimum instance count.** Set a minimum instance count to make sure your application is always running even when there's no load.
- **Maximum instance count.** Set a maximum instance count to limit your total possible hourly cost.
- **Adequate scale margin.** Make sure your maximum and minimum instance count values are different, and set an adequate margin between the two values. You can automatically scale between the minimum and maximum by using rules you create.
- **Scale rule combinations.** Always use a scale-out and scale-in rule combination that performs an increase and decrease. If you don't set a scale-out rule, your application might fail, or performance might degrade under increased loads. If you don't set a scale-in rule, you can experience unnecessary and extensive costs when the load decreases.

- **Metric statistics.** Carefully choose the appropriate statistic for your diagnostic metrics, including Average, Minimum, Maximum, and Total.
- **Default instance count.** Always select a safe default instance count. The default instance count is important because autoscale scales your service to the count you specify when metrics aren't available.
- **Notifications.** Always configure autoscale notifications. It's important to maintain awareness of how your application is performing as the load changes.

Things to consider when configuring automatic scaling

In addition to rule-based autoscale, Azure App Service offers Automatic scaling (also called Elastic scaling) for PremiumV2 and PremiumV3 tiers. This is a separate scaling feature that works differently from the autoscale rules.

- **HTTP traffic-based.** Automatic scaling responds directly to incoming HTTP requests without requiring you to configure scaling rules.
- **Platform-managed.** Azure automatically manages the scaling decisions based on traffic patterns, eliminating the need for rule configuration.
- **Always-ready instances.** Maintains warmed instances to handle traffic spikes immediately.
- **Tier availability.** Available only on PremiumV2 and PremiumV3 tiers.

Choose between Autoscale and Automatic scaling

- **Use rule-based Autoscale.** You need custom scaling logic, want to scale based on multiple metrics, or need schedule-based scaling.
- **Use Automatic scaling.** You want less management, can't predict load patterns, or need fast response to traffic changes without rule configuration.

6. Module assessment

<https://learn.microsoft.com/en-us/training/modules/configure-app-service-plans/6-knowledge-check>

Module assessment

Completed

7. Summary and resources

<https://learn.microsoft.com/en-us/training/modules/configure-app-service-plans/7-summary-resources>

Summary and resources

Completed

In this module, you learned about Azure App Service plans and how they're used to define the compute resources for running applications in Azure App Service. These plans can be configured with a specific region, number of virtual machine instances, and size of virtual machine instances. The pricing tier of the App Service plan determines the features and cost. Pricing tiers include Free and Shared plans for development and testing purposes. Pricing tiers also include Isolated plans for mission-critical workloads.

You learned about scaling in Azure App Service. Scale up involves increasing the CPU, memory, and disk space by changing the pricing tier. Scale out increases the number of virtual machine instances running the application. Autoscaling allows you to automatically adjust

the number of resources based on the load on your application. Autoscale can be configured with metric-based or time-based rules.

The main takeaways from this module are:

- Azure App Service plans are used to define the compute resources for running web applications in Azure App Service.
- The pricing tier of the App Service plan determines the features and cost, with options ranging from Free and Shared plans to Isolated plans.
- Scaling in Azure App Service can be done through scale up (changing the pricing tier) or scale out (increasing the number of virtual machine instances).
- Autoscaling allows for automatic adjustment of resources based on application load, with metric-based and time-based rules.

Learn more with Copilot

Copilot can assist you in configuring Azure infrastructure solutions. Copilot can compare, recommend, explain, and research products and services where you need more information. Open a Microsoft Edge browser and choose Copilot (top right) or navigate to copilot.microsoft.com. Take a few minutes to try these prompts and extend your learning with Copilot.

- In Microsoft Azure, what are app service pricing plans? Provide examples of when to use each plan.
- In Microsoft Azure, what does scale in and scale out mean? How do I determine when to scale an application?

Learn more with documentation

- [Azure App Service plans](#). This article provides an overview of App Service plans.
- [Manage an App Service plan in Azure](#). This guide shows how to create and manage an App Service plan.
- [Scale up an app in Azure App Service](#). This article shows you how to scale your app in Azure App Service.

Learn more with self-paced training

- [Scale apps in Azure App Service](#). Learn how autoscale operates in App Service. Learn to identify autoscale factors, enable autoscale, and create autoscale conditions.

Configure Azure App Service

1. Introduction

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/1-introduction>

Introduction

Completed

Azure Administrators are interested in solutions that make it easier to deploy and manage their web, mobile, and API applications. Azure Administrators are interested in solutions that are AI-ready.

Your company provides consumer research, and your team manages the on-premises servers. The servers you administer run the entire company infrastructure from web servers to databases. The hardware is aging and starting to struggle to keep up with some of the new data analysis applications. Rather than upgrade the hardware, the company decided to deploy Azure App Service.

In this module, you learn how to configure and manage Azure App Service. You learn about configuration settings, deployment slots, and custom domain names. You learn about application backup, recovery, and monitoring.

The goal of this module is to provide you with the knowledge and skills to effectively use Azure App Services.

Learning objectives

In this module, you learn how to:

- Identify features and usage cases for Azure App Service.
- Create an app with App Service.
- Configure deployment settings, specifically deployment slots.
- Secure your App Service app.
- Configure custom domain names.
- Back up and restore your App Service app.
- Configure Azure Application Insights.

Skills measured

The content in the module helps you prepare for [Exam AZ-104: Microsoft Azure Administrator](#).

Prerequisites

- Working knowledge of the Azure portal, so you can configure the service.
- Familiarity with cloud-based services, specifically web hosting services.

2. Implement Azure App Service

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/2-implement>

Implement Azure App Service

Completed

[Azure App Service](#) brings together everything you need to create websites, mobile backends, and web APIs for any platform or device. Applications run and scale with ease in both Windows and Linux-based environments.

App Service provides Quickstarts for programming languages. These languages include: ASP.NET, Java, Node.js, Python, and PHP.

App Service benefits

There are many advantages to using App Service to develop and deploy your web, mobile, and API apps. Review the following table and think about what features can help you host your App Service instances.

Benefit	Description
Multiple languages and frameworks	App Service has first-class support for ASP.NET, Java, Node.js, PHP, and Python. You can also run PowerShell and other scripts or executables as background services.
DevOps optimization	App Service supports continuous integration and deployment with Azure DevOps, GitHub, BitBucket, Docker Hub, and Azure Container Registry. You can promote updates through test and staging environments. Manage your apps in App Service by using Azure PowerShell or the cross-platform command-line interface (CLI).
Global scale with high availability	App Service helps you scale up or out manually or automatically. You can host your apps anywhere within the Microsoft global datacenter infrastructure, and the App Service SLA offers high availability.
Security and compliance	App Service is ISO, SOC, and PCI compliant. You can authenticate users with Microsoft Entra ID or with social logins via Google, Facebook, X, or Microsoft. Create IP address restrictions and manage service identities.
Application templates	Choose from an extensive list of application templates in Azure Marketplace, such as WordPress, Joomla, and Drupal.
Visual Studio integration	App Service offers dedicated tools in Visual Studio to help streamline the work of creating, deploying, and debugging.
API and mobile features	App Service provides turn-key CORS support for RESTful API scenarios. You can simplify your mobile app scenarios by enabling authentication, offline data sync, push notifications, and more.

3. Create an app with App Service

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/3-create-app-service>

Create an app with App Service

Completed

You can use the Web Apps, Mobile Apps, or API Apps features of Azure App Service, and create your own apps in the Azure portal.

Things to know about configuration settings

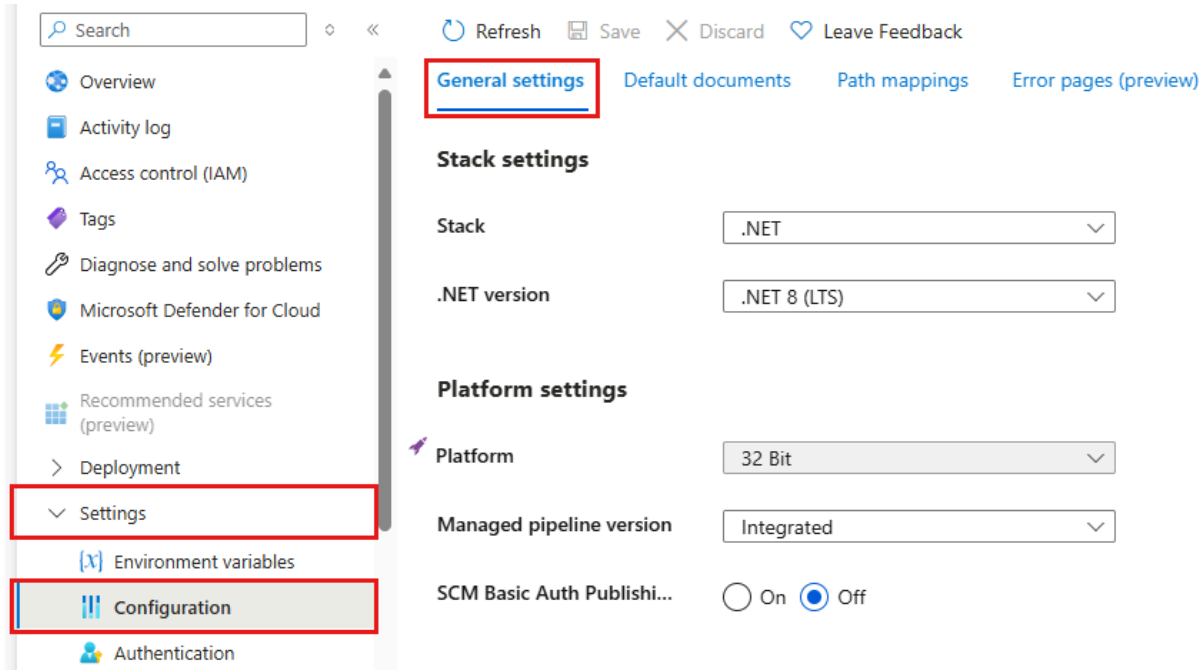
Let's examine some of the basic configuration settings you need to create an app with App Service.

- **Name:** The name for your app must be unique. The name identifies and locates your app in Azure. An example name is `webappces1.azurewebsites.net`. You can map a custom domain name, if you prefer to use that option instead.
- **Publish:** App Service hosts (publishes) your app as code or as a Docker Container.
- **Runtime stack:** App Service uses a software stack to run your app, including the language and SDK versions. For Linux apps and custom container apps, you can set an optional start-up command or file. Your choices for the stack include .NET Core, .NET Framework, Node.js, PHP, and Python. Various versions of each product are available for Linux and Windows.
- **Operating system:** The operating system for your app runtime stack can be Linux or Windows.
- **Region:** The region location that you choose for your app affects the App Service plans that are available.

- **Pricing plans:** Your app needs to be associated with an Azure App Service plan to establish available resources, features, and capacity. You can choose from pricing tiers that are available for the region location you selected.

Post-creation settings

After your app is created, other **Configuration** settings become available in the Azure portal, including app deployment options and path mapping.



Some of the extra configuration settings can be included in the developer's code, while others can be configured in your app. Here are a few of the extra application settings.

- **Always On:** You can keep your app loaded even when there's no traffic. This setting is required for continuous WebJobs or for WebJobs that are triggered by using a CRON expression.
- **Session affinity:** In a multi-instance deployment, you can ensure your app client is routed to the same instance for the life of the session.
- **HTTPS Only:** When enabled, all HTTP traffic is redirected to HTTPS.

Tip

Consider practicing on your own with the [Exercise - Create a web app in the Azure portal](#). This exercise provides a sandbox.

4. Explore continuous integration and deployment

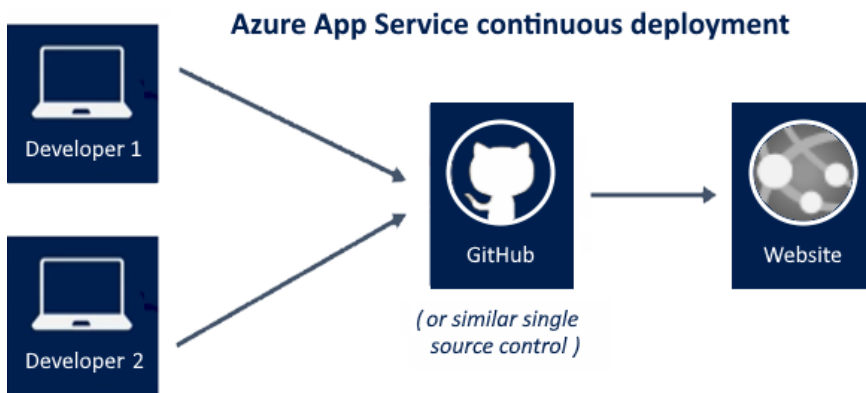
<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/4-explore-continuous-integration-deployment>

Explore continuous integration and deployment

Completed

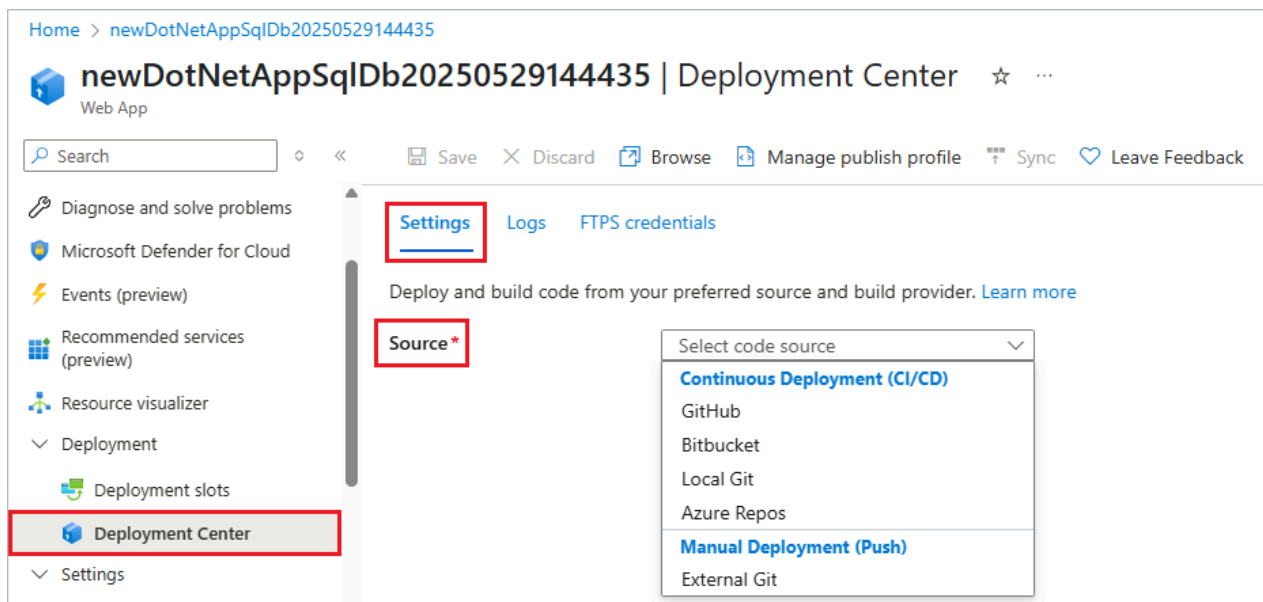
The Azure portal provides out-of-the-box continuous integration and deployment with Azure DevOps services, GitHub, Bitbucket, FTP, or a local Git repository on your development machine. You can connect your web app with any of the above sources and App Service handles the rest for you. App Service autosynchronizes your code and any future changes to the code into your web app. With Azure DevOps services, you can also define your own build and release process. Compile your source code, run tests, and build and deploy the

release into your web app every time you commit the code. All of the operations happen implicitly without any need for human administration.



Things to know about continuous and manual deployment

When you create your web app with App Service, you can choose continuous or manual deployment. As you review these options, consider which deployment method to implement for your App Service apps. These options are located in the Deployment Center.



Continuous deployment (CI/CD) is a process used to push out new features and bug fixes in a fast and repetitive pattern with minimal impact on end users. Azure supports automated deployment directly from several sources:

- **GitHub:** Azure supports automated deployment directly from GitHub. Azure supports automated deployment directly from GitHub using **two** build providers. When you connect your GitHub repository to Azure, you can choose between **GitHub Actions** (default) and **App Service Build Service**.
- **Bitbucket:** With its similarities to GitHub, you can configure an automated deployment with Bitbucket.
- **Local Git:** The App Service Web Apps feature offers a local URL that you can add as a repository.
- **Azure Repos:** Azure Repos is a set of version control tools that you can use to manage your code. Whether your software project is large or small, using version control as soon as possible is a good idea.

Manual deployment enables you to manually push your code to Azure.

- **Remote Git:** The App Service Web Apps feature offers a Git URL that you can add as a remote repository. Pushing to the remote repository deploys your app.

5. Create deployment slots

Create deployment slots

Completed

When you deploy your web app, web app on Linux, mobile backend, or API app to Azure App Service, you can use a separate deployment slot instead of the default production slot.

Things to know about deployment slots

Let's take a closer look at the characteristics of deployment slots.

- Deployment slots are live apps that have their own hostnames.
- Deployment slots are available in the Standard, Premium, and Isolated v2 App Service pricing tiers. Your app needs to be running in one of these tiers to use deployment slots.
- The Standard, Premium, and Isolated tiers offer different numbers of deployment slots.
- App content and configuration elements can be swapped between two deployment slots, including the production slot.

Save Discard Add Slot Swap Logs Refresh



Deployment Slots

Deployment slots are live apps with their own hostnames. App content and configurations elements can be swapped between two deployment slots, including the production slot.

NAME	STATUS	APP SERVICE PLAN	TRAFFIC %
webappces PRODUCTION	Running	ASP-webapprg-a247	100
webappces-Staging	Running	ASP-webapprg-a247	0

Things to consider when using deployment slots

There are several advantages to using deployment slots with your App Service app. Review the following benefits and think about how they can support your App Service implementation.

- **Consider validation.** You can validate changes to your app in a staging deployment slot before swapping the app changes with the content in the production slot.
- **Consider reductions in downtime.** Deploying an app to a slot first and swapping it into production ensures that all instances are ready. This option eliminates downtime when you deploy your app. The traffic redirection is seamless, and no requests are dropped because of swap operations. The entire workflow can be automated by configuring **Auto swap** when preswap validation isn't needed.
- **Consider restoring to last known good site.** After a swap, the slot with the previously staged app now has the previous production app. If the changes swapped into the production slot aren't as you expected, you can perform the same swap immediately to return to your "last known good site."

- **Consider Auto swap.** Auto swap streamlines Azure Pipeline scenarios where you want to deploy your app continuously with zero cold starts and zero downtime for app customers. When Auto swap is enabled from a slot into production, every time you push your code changes to that slot, App Service automatically swaps the app into production after it's warmed up in the source slot.

6. Add deployment slots

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/6-add-deployment-slots>

Add deployment slots

Completed

Deployment slots are configured in the Azure portal. You can swap your app content and configuration elements between deployment slots, including the production slot.

Things to know about creating deployment slots

Let's review some details about how deployment slots are configured.

- New deployment slots can be empty or cloned.
- Deployment slot settings fall into three categories:
 - Slot-specific app settings and connection strings (if applicable).
 - Continuous deployment settings (when enabled).
 - Azure App Service authentication settings (when enabled).
- When you clone a configuration from another deployment slot, the cloned configuration is editable. Some configuration elements follow the content across the swap. Other slot-specific configuration elements stay in the source slot after the swap.

Swapped settings versus slot-specific settings

The following table lists settings that are swapped between deployment slots. The table also lists settings that remain in the source slot (slot-specific). As you review these settings, consider which features are required for your App Service apps. Read more about [which settings are swapped](#).

Swapped settings	Slot-specific settings
Language stack and version, 32/64-bit	Custom domain names
App settings *	Nonpublic certificates and TLS/SSL settings
Connection strings *	Scale settings
Mounted storage accounts*	Always On
Public certificates	IP restrictions
WebJobs content	WebJobs schedulers
Hybrid connections **	Diagnostic settings
Service endpoints **	Cross-origin resource sharing (CORS)
Azure Content Delivery Network **	Virtual network integration
Path mapping	Managed identities

* Setting can be configured to be slot-specific.

** Feature isn't currently available.

7. Secure your App Service app

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/7-secure-app-service>

Secure your App Service app

Completed

Azure App Service provides built-in [authentication and authorization](#) support. You can sign in users and access data by writing minimal or no code in your web app, API, and mobile backend, and also your Azure Functions apps.

Secure authentication and authorization require deep understanding of security, including federation, encryption, JSON web tokens (JWT) management, grant types, and so on. App Service provides these utilities so you can spend more time and energy on providing business value to your customer.

Note

You aren't required to use Azure App Service for authentication and authorization. Many web frameworks are bundled with security features, and you can use your preferred service.

Things to know about app security with App Service

Let's take a closer look at how App Service helps you provide security for your app.

- The authentication and authorization security module in Azure App Service runs in the same environment as your application code, yet separately.
- The security module is configured by using app settings. No SDKs, specific languages, or changes to your application code are required.
- The security module handles several tasks for your app:
 - Authenticate users with the specified provider
 - Validate, store, and refresh tokens
 - Manage the authenticated session
 - Inject identity information into request headers

Things to consider when using App Service for app security

You configure authentication and authorization security in App Service by selecting features in the Azure portal. Review the following options and think about what security can benefit your App Service apps implementation.

- **Allow Anonymous requests (no action).** Defer authorization of unauthenticated traffic to your application code. For authenticated requests, App Service also passes along authentication information in the HTTP headers. This feature provides more flexibility for handling anonymous requests. With this feature, you can present multiple sign-in providers to your users.
- **Allow only authenticated requests.** Redirect all anonymous requests to `/.auth/Login/<provider>` for the provider you choose. The feature is equivalent to **Log in with <provider>**. If the anonymous request comes from a native mobile app, the

returned response is an `HTTP 401 Unauthorized` message. With this feature, you don't need to write any authentication code in your app.

Important

This feature restricts access to **all** calls to your app. Restricting access to all calls might not be desirable if your app requires a public home page, as is the case for many single-page apps.

- **Logging and tracing.** View authentication and authorization traces directly in your log files. If you see an authentication error that you didn't expect, you can conveniently find all the details by looking in your existing application logs. If you enable failed request tracing, you can see exactly how the security module participated in a failed request. In the trace logs, look for references to a module named `EasyAuthModule_32/64`.

8. Create custom domain names

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/8-create-custom-domain-names>

Create custom domain names

Completed

When you create a web app, Azure assigns the app to a subdomain of `azurewebsites.net`. Suppose your web app is named `contoso`. Azure creates a URL for your web app as `contoso.azurewebsites.net`. Azure also assigns a virtual IP address for your app. For a production web app, you might want users to see a custom domain name.

What is a custom domain?

A domain name is the address people type into a web browser to reach your website. A custom domain is a domain name that you own and configure to point to your Azure-hosted app, replacing the default Azure domain.

For example:

- Default Azure domain: `myapp-00000.westus.azurewebsites.net`
- Custom domain: `www.contoso.com`

Using a custom domain allows you to:

- Establish a branded, user-friendly web address.
- Improve trust and credibility with customers.
- Manage and secure traffic to your application.

Steps to configure a custom domain name for your app

Creating a custom domain name requires providers, security, and naming information.

Add custom domain



Domain provider * ⓘ

- ☐ App Service Domain
- ☒ All other domain services
- ☒ App Service Managed Certificate
- ☐ Add certificate later
- ☒ SNI SSL
- ☐ IP based SSL

TLS/SSL certificate * ⓘ

TLS/SSL type * ⓘ

Enter your custom domain. We'll give you hostname records to register with your domain provider, and we can validate and add your custom domain here. [Learn more](#)

Domain * ⓘ

www.contoso.com

Hostname record type * ⓘ

CNAME (www.example.com or any subd... ▼)

Domain validation

To validate your domain ownership, copy the hostname records below and enter them with your domain provider. [Learn more](#)

↓ Export to CSV

Type	Host	Value	Status
CNAME	www	webapp51451367.azurewebsites.net	
TXT	asuid.www	0DE62CB0A68623CB52D87E9A38105C8F8CDE5432741E4CD63DBA9E141B495DAE	

There are three steps to create a custom domain name.

- 1. Reserve your domain name.** The easiest way to set up a custom domain is to buy one directly in the Azure portal. (This name isn't the Azure assigned name of `*.azurewebsites.net`.) The registration process enables you to manage your web app's domain name directly in the Azure portal instead of going to a third-party site. Configuring the domain name in your web app is also a simple process in the Azure portal.
- 2. Create DNS records to map the domain to your Azure web app.** The Domain Name System (DNS) uses data records to map domain names to IP addresses. There are several types of DNS records.
 - For web apps, you create either an **A** (Address) record or a **CNAME** (Canonical Name) record.
 - An **A** record maps a domain name to an IP address.
 - A **CNAME** record maps a domain name to another domain name. DNS uses the second name to look up the address. Users still see the first domain name in their browser. As an example, you could map `contoso.com` to your `webapp.azurewebsites.net` URL.
 - If the IP address changes, a **CNAME** entry is still valid, whereas an **A** record must be updated.
 - Some domain registrars don't allow **CNAME** records for the root domain or for wildcard domains. In such cases, you must use an **A** record.
- 3. Enable the custom domain.** After you have your domain and create your DNS record, use the Azure portal to validate your custom domain and add it to your web app. Be sure to test your domain before publishing.

Important

App Service offers free managed TLS certificates. Certificates auto-renew 30 days before expiry. In the Azure portal, go to **Custom domains** → **Add binding** → **App Service Managed Certificate**.

9. Back up and restore your App Service app

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/9-backup-app-service>

Back up and restore your App Service app

Completed

The [Backup and Restore feature](#) in Azure App Service lets you easily create backups manually or on a schedule. You can configure the backups to be retained for a specific or indefinite amount of time. You can restore your app or site to a snapshot of a previous state by overwriting the existing content or restoring to another app or site.

The **Backups** page lists all the automatic and custom backups for your app and displays the status of each.

The screenshot shows the 'Backups' page for an App Service app named 'my-demo-app'. The left sidebar contains navigation links for 'Events (preview)', 'Deployment', 'Settings', and 'Backups' (which is highlighted with a red box). The main content area shows the 'Backups' page with a search bar, filters for 'Type', 'Status', and 'Time range', and a table of backup results. The table has columns for 'Backup time', 'Status', 'Type', and 'Restore'. The 'Status' column shows 'Succeeded' or 'Failed' with corresponding icons. The 'Type' column shows 'Automatic' or 'Custom'. The 'Restore' column shows a restore icon. The table shows 10 of 206 results.

Backup time ↓	Status ↑	Type ↑	Restore
09/05/2022, 14:52:19	✓ Succeeded	Automatic	↺
09/05/2022, 14:47:15	✓ Succeeded	Custom	↺
09/05/2022, 14:36:31	✗ Failed	Custom	↺
09/05/2022, 14:20:20	✗ Failed	Custom	↺
09/05/2022, 13:52:19	✓ Succeeded	Automatic	↺
09/05/2022, 12:52:19	✓ Succeeded	Automatic	↺
09/05/2022, 11:52:19	✓ Succeeded	Automatic	↺

Things to know about Backup and Restore

Examine the following details about the Backup and Restore feature. Think about how you can implement this feature for your App Service apps.

- Back up and restore is supported in the Basic, Standard, Premium, and Isolated tiers. For the Basic tier, you can only back up and restore the production slot.
- You need an Azure storage account and container in the same subscription as the app to back up.
- Azure App Service can back up the following information to the Azure storage account and container you configured for your app:
 - App configuration settings
 - File content
 - Any database connected to your app (SQL Database, Azure Database for MySQL, Azure Database for PostgreSQL, MySQL in-app)
- In your storage account, each backup consists of a Zip file and XML file:
 - The Zip file contains the back-up data for your app or site.

- The XML file contains a manifest of the Zip file contents.
- You can configure backups manually or on a schedule.
- Full backups are the default.
- Partial backups are supported. You can specify files and folders to exclude from a backup.
- You restore partial backups of your app or site the same way you restore a regular backup.
- Backups can hold up to 10 GB of app and database content.
- Backups for your app or site are visible on the **Containers** page of your storage account and app (or site) in the Azure portal.

Things to consider when creating backups and restoring backups

Let's review some considerations about creating a backup for your app or site, and restoring data and content from a backup.

- **Consider full backups.** Do a full backup to easily save all configuration settings, all file content, and all database content connected with your app or site.

When you restore a full backup, all content on the site is replaced with whatever is in the backup. If a file is on the site, but not in the backup, the file is deleted.

- **Consider partial backups.** Specify a partial backup so you can choose exactly which files to back up.

When you restore a partial backup, any content located in an excluded folder or file is left as-is.

- **Consider browsing back-up files.** Unzip and browse the Zip and XML files associated with your backup to access your backups. This option lets you view the content without actually performing an app or site restore.
- **Consider firewall on back-up destination.** If your storage account is enabled with a firewall, you can't use the storage account as the destination for your backups.

10. Use Azure Application Insights

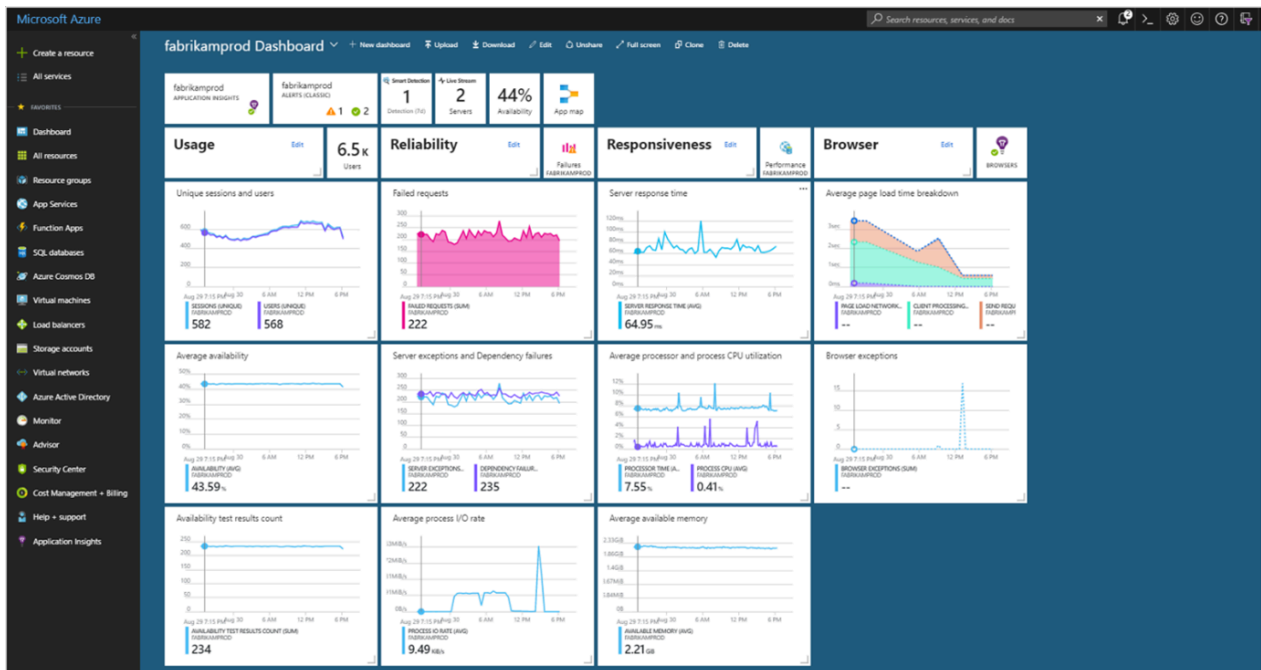
<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/10-use-application-insights>

Use Azure Application Insights

Completed

[Azure Application Insights](#) is a feature of Azure Monitor that lets you monitor your live applications. You can integrate Application Insights with your App Service configure to automatically detect performance anomalies in your apps.

Application Insights is designed to help you continuously improve the performance and usability of your apps. The feature offers powerful analytics tools to help you diagnose issues and understand what users actually do with your apps.



Things to know about Application Insights

Let's examine some characteristics of Application Insights for Azure Monitor.

- Application Insights works on various platforms including .NET, Node.js, and Java EE.
- The feature can be used for configurations that are hosted on-premises, in a hybrid environment, or in any public cloud.
- Application Insights integrates with your Azure Pipeline processes, and has connection points to many development tools.

Things to consider when using Application Insights

Application Insights is ideal for supporting your development team. The feature helps developers understand how your app is performing and how it's being used. Consider monitoring the following items in your App Service configuration scenario.

- **Consider Request rates, response times, and failure rates.** Find out which pages are most popular, at what times of day, and where your users are. See which pages perform best. If your response times and failure rates go high when there are more requests, then perhaps you have a resourcing problem.
- **Consider Dependency rates, response times, and failure rates.** Use Application Insights to discover if external services are degrading your app performance.
- **Consider Exceptions.** Analyze the aggregated statistics, or pick specific instances and drill into the stack trace and related requests. Both server and browser exceptions are reported.
- **Consider Page views and load performance.** Collect the number of page views reported by your users' browsers and analyze the load performance.
- **Consider User and session counts.** Application Insights can help you keep track of the number of users and sessions connected to your app.
- **Consider Performance counters.** Add Application Insights performance counters from your Windows or Linux server machines. Monitor performance output for the CPU, memory, network usage, and so on.
- **Consider Host diagnostics.** Integrate diagnostics from Docker or Azure into your app Application Insights.
- **Consider Diagnostic trace logs.** Implement trace logs from your app to help correlate trace events with requests and diagnose issues.
- **Consider Custom events and metrics.** Write your own custom events and metric tracking algorithms as client or server code. Track business events such as number of items sold, or number of games won.

Tip

Consider extending your learning with the [Troubleshoot solutions by using Application Insights](#) training module.

11. Exercise: Implement Web Apps

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/11-simulation-web-apps>

Exercise: Implement Web Apps

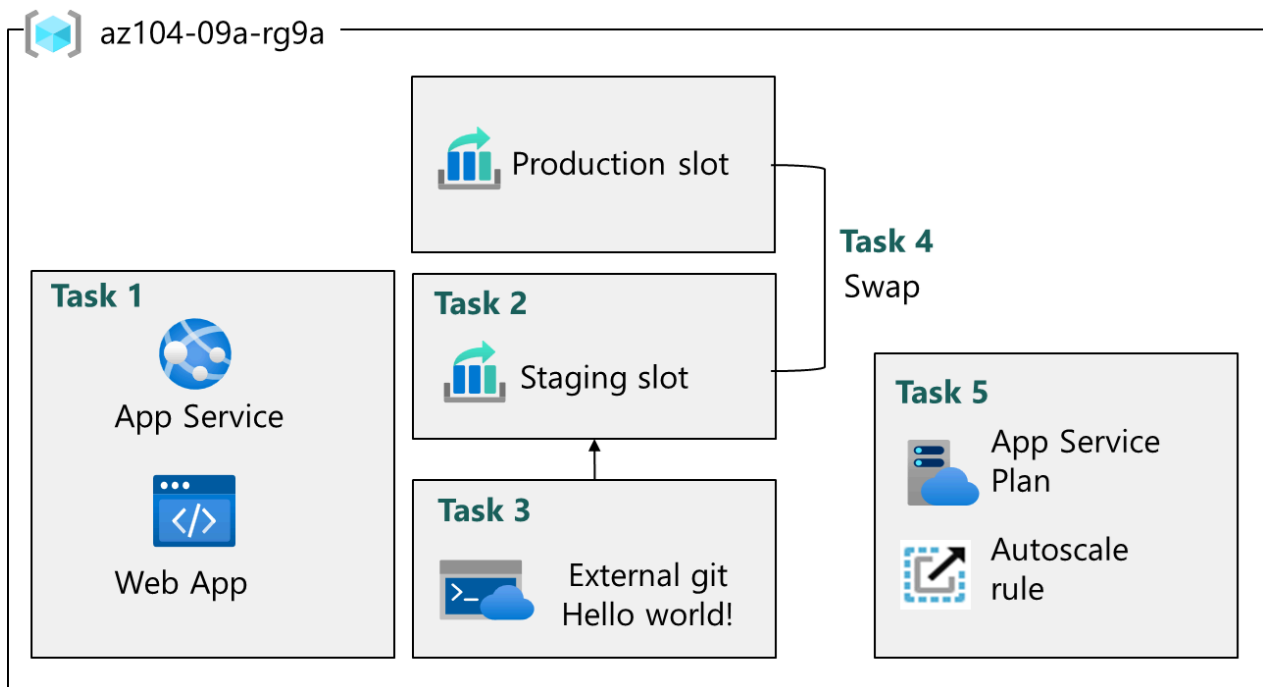
Completed

Lab scenario

Your organization is migrating on-premises web apps to Azure. As the Azure Administrator you need to:

- Host web sites running on Windows servers by using the PHP runtime stack.
- Use Azure Web Apps deployment slots.

Architecture diagram



Job skills

- Create an Azure web app.
- Create a staging deployment slot.
- Configure Web App deployment settings.
- Deploy code to the staging deployment slot.
- Swap the staging slots.
- Configure and test autoscaling of your Azure web app.

Note

Estimated time: 20 minutes. To complete this exercise, you need an [Azure subscription](#).

Launch the exercise, and follow the instructions. When finished, be sure to return to this page so you can continue learning.

[Launch Exercise](#)

12. Module assessment

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/12-knowledge-check>

Module assessment

Completed

13. Summary and resources

<https://learn.microsoft.com/en-us/training/modules/configure-azure-app-services/13-summary-resources>

Summary and resources

Completed

Azure App Service is an HTTP-based service for hosting web applications. With App Service, you can develop web apps in your favorite language. The service lets you easily run and scale your web apps on Windows and Linux-based environments.

In this module, you reviewed the features and usage cases for Azure App Service. You learned how to create, secure, and back up your web apps. You explored how to configure deployment settings, including deployment slots, and custom domain names for your web apps. You discovered how to use Azure Application Insights to monitor web apps.

The main takeaways for this module

- Azure App Service lets you develop and deploy web, mobile, and API apps.
- Azure App Service configuration settings include runtime stack, operating system, region, and App Service plan.
- Deployment slots help you manage different app stages. For example, development, test, stage, and production.
- The default Azure App Service domain name can be customized for your organization.
- Azure Application Insights is a feature of Azure Monitor that lets you monitor your live applications. You can integrate Application Insights with your App Service configuration to automatically detect performance anomalies in your apps.
- Application Insights lets you continuously monitor the performance and usability of your apps.

Learn more with Copilot

Copilot can assist you in configuring Azure infrastructure solutions. Copilot can compare, recommend, explain, and research products and services where you need more information. Open a Microsoft Edge browser and choose Copilot (top right) or navigate to copilot.microsoft.com. Take a few minutes to try these prompts and extend your learning with Copilot.

- What are the main tasks to configure an Azure App Service web app?
- What options are available for scaling an Azure App Service web app?

Learn more with documentation

- [App Service overview](#). This article provides an overview of the App Service and why you would use this service.
- [Configure an App Service app](#). This article explains how to configure common settings for web apps, mobile back end, or API app.
- [Set up staging environments in Azure App Service](#). The article covers deployment slots and swap operations.

Learn more with self-paced training

- [Stage a web app deployment for testing and rollback by using App Service deployment slots](#). Learn to use deployment slots to streamline deployment and roll back.
- [Explore Azure App Service deployment slots](#). Learn how slot swapping works and how to route traffic to different slots.
- [Host a web application with Azure App Service](#). Learn how to create a website through the hosted web app platform in Azure App Service.

Configure Azure Container Instances

1. Introduction

<https://learn.microsoft.com/en-us/training/modules/configure-azure-container-instances/1-introduction>

Introduction

Completed

Containers and virtual machines are both forms of virtualization, but there are some key differences between them.

To provide context, let's consider a scenario: You're an Azure Administrator responsible for deploying and managing applications in a cloud environment. Your organization is looking for a solution that offers fast startup times, easy management, and the ability to run applications in isolated containers. You want to understand the benefits of using Azure Container Instances and how it compares to virtual machines.

In this module, you learn when to use Azure Container Instances instead of virtual machines. You also get an overview of features and use cases.

The goal of this module is to introduce you to AI-ready Azure Container Instances.

Learning objectives

In this module, you learn how to:

- Identify when to use containers versus virtual machines.
- Identify the features and usage cases of Azure Container Instances.
- Implement Azure container groups.

Skills measured

The content in the module helps you prepare for [Exam AZ-104: Microsoft Azure Administrator](#).

Prerequisites

- Working knowledge of containerization concepts and terminology.
- Familiarity with cloud computing and experience with the Azure portal.

2. Compare containers to virtual machines

<https://learn.microsoft.com/en-us/training/modules/configure-azure-container-instances/2-compare-containers-to-virtual-machines>

Compare containers to virtual machines

Completed

Hardware virtualization makes it possible to run multiple isolated instances of operating systems concurrently on the same physical hardware. Containers represent the next stage in the virtualization of computing resources.

Container-based virtualization allows you to virtualize the operating system. This approach lets you run multiple applications within the same instance of an operating system, while maintaining isolation between the applications. The containers within a virtual machine provide functionality similar to that of virtual machines within a physical server.

Things to know about containers versus virtual machines

To better understand container-based virtualization, let's [compare containers and virtual machines](#).

Compare	Containers	Virtual machines
Isolation	A container typically provides lightweight isolation from the host and other containers, but a container doesn't provide as strong a security boundary as a virtual machine.	A virtual machine provides complete isolation from the host operating system and other virtual machines. This separation is useful when a strong security boundary is critical, such as hosting apps from competing companies on the same server or cluster.
Operating system	Containers run the user mode portion of an operating system and can be tailored to contain just the needed services for your app. This approach helps you use fewer system resources.	Virtual machines run a complete operating system including the kernel, which requires more system resources (CPU, memory, and storage).
Deployment	You can deploy individual containers by using Docker via the command line. You can deploy multiple containers by using an orchestrator such as Azure Kubernetes Service.	You can deploy individual virtual machines by using Windows Admin Center or Hyper-V Manager. You can deploy multiple virtual machines by using PowerShell or System Center Virtual Machine Manager.
Persistent storage	Containers use Azure Disks for local storage for a single node, or Azure Files (SMB shares) for storage shared by multiple nodes or servers.	Virtual machines use a virtual hard disk (VHD) for local storage for a single machine, or an SMB file share for storage shared by multiple servers.
Fault tolerance	If a cluster node fails, the orchestrator on another cluster node rapidly recreates any containers running on the node.	Virtual machines can fail over to another server in a cluster, where the virtual machine's operating system restarts on the new server.

Things to consider when using containers

Containers offer several advantages over physical and virtual machines. Review the following benefits and consider how you can implement containers for the internal apps for your company.

- **Consider flexibility and speed.** Gain increased flexibility and speed when developing and sharing your containerized application code.
- **Consider testing.** Choose containers for your configuration to allow for simplified testing of your apps.
- **Consider app deployment.** Implement containers to gain streamlined and accelerated deployment of your apps.
- **Consider workload density.** Support higher workload density and improve your resource utilization by working with containers.

3. Review Azure Container Instances

<https://learn.microsoft.com/en-us/training/modules/configure-azure-container-instances/3-review>

Review Azure Container Instances

Completed

Containers are becoming the preferred way to package, deploy, and manage cloud applications. There are many options for teams to build and deploy cloud native and containerized applications on Azure. In this unit, we review Azure Container Instances (ACI).

Azure Container Instances offers the fastest and simplest way to run a container in Azure, without having to manage any virtual machines and without having to adopt a higher-level service. Azure Container Instances is a great solution for any scenario that can operate in isolated containers.

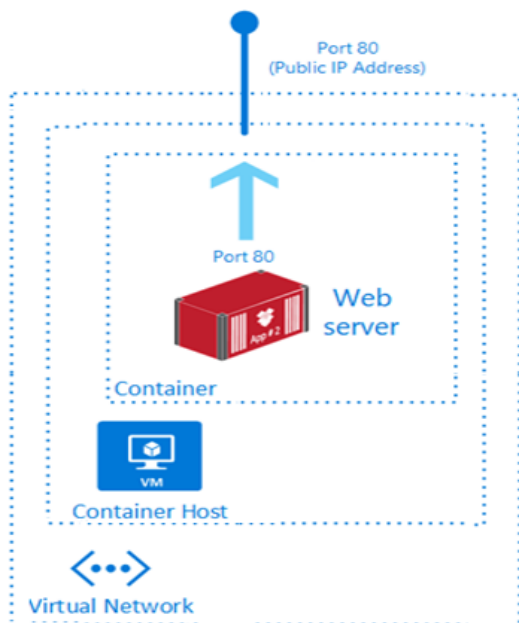
Understand container images

All containers are created from container images. A container image is a lightweight, standalone, executable package of software that encapsulates everything needed to run an application. It includes the following components:

- **Code:** The application's source code.
- **Runtime:** The environment required to execute the application.
- **System tools:** Utilities necessary for the application to function.
- **System libraries:** Shared libraries used by the application.
- **Settings:** Configuration parameters specific to the application.

When you create a container image, it becomes a portable unit that can run consistently across different computing environments. These images are the building blocks for containers, which are instances of these images running at runtime.

The following illustration shows a web server container built with Azure Container Instances. The container is running on a virtual machine in a virtual network.



Things to know about Azure Container Instances

Let's review some of the [benefits of using Azure Container Instances](#). As you review these points, think about how you can implement Container Instances for your internal applications.

- **Fast startup times.** Containers can start in seconds without the need to deploy and manage virtual machines.
- **Public IP connectivity and DNS names.** Containers can be directly exposed to the internet with an IP address and FQDN (fully qualified domain name).
- **Custom sizes.** You specify CPU cores (from 0.1 to 4 vCPU) and memory (from 0.1 to 16 GB) for each container at deployment time. Resource allocation is fixed for the lifetime of the container group.
- **Persistent storage.** Containers support direct mounting of Azure Files file shares.
- **Linux and Windows containers.** Container Instances can schedule both Windows and Linux containers. Specify the operating system type when you create your container groups.
- **Coscheduled groups.** Container Instances supports scheduling of multi-container groups that share host machine resources.

- **Virtual network deployment.** Linux container groups can be deployed into an Azure virtual network for private communication with other Azure resources. Virtual network deployed containers receive no public IP address and communicate only within the virtual network or peered networks.

4. Implement container groups

<https://learn.microsoft.com/en-us/training/modules/configure-azure-container-instances/4-implement-container-groups>

Implement container groups

Completed

The top-level resource in Azure Container Instances is the **container group**. A **container group** is a collection of containers that get scheduled on the same host machine. The containers share a lifecycle, resources, local network, and storage volumes.

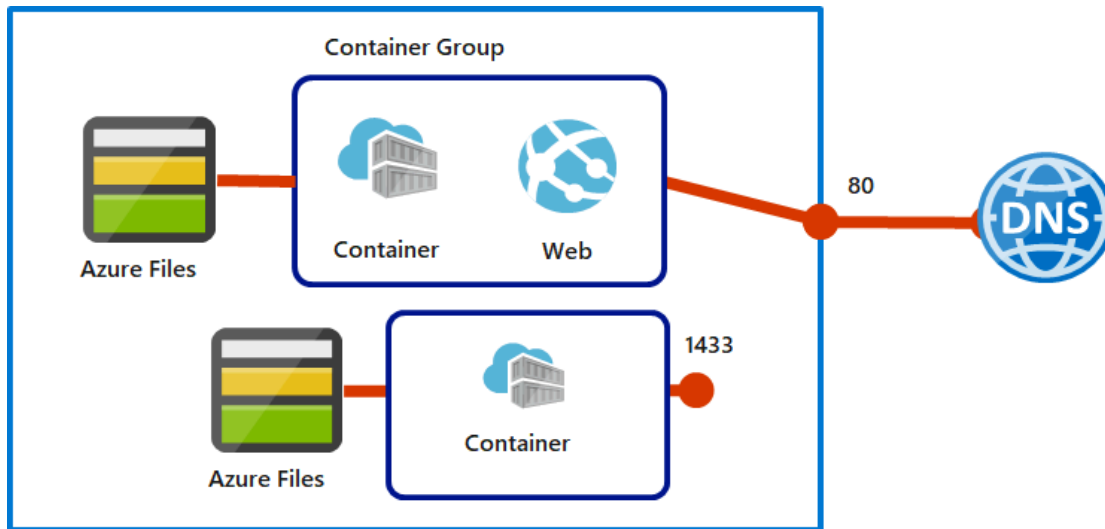
Things to know about container groups

Let's review some of details about container groups for Azure Container Instances.

- A container group is similar to a pod in Kubernetes. A pod typically has a 1:1 mapping with a container, but a pod can contain multiple containers. The containers in a multi-container pod can share related resources.
- Azure Container Instances allocates resources to a multi-container group by adding together the resource requests of all containers in the group. Resources can include items such as CPUs, memory, and GPUs.
- There are three common ways to deploy a multi-container group.
 - **Azure Resource Manager template.** JSON-based infrastructure as code, ideal when deploying alongside other Azure resources.
 - **Bicep.** Microsoft's recommended infrastructure as code language, more concise than Azure Resource Manager templates. Bicep includes full IntelliSense support.
 - **YAML files.** Container-focused format, ideal for deployments that include only container instances.
- Container groups can share an external-facing IP address, one or more ports on the IP address, and a DNS label with an FQDN.
 - **External client access.** You must expose the port on the IP address and from the container to enable external clients to reach a container in your group.
 - **Port mapping.** Port mapping isn't supported because containers in a group share a port namespace.
 - **Deleted groups.** When a container group is deleted, its IP address and FQDN are released.

Configuration example

Consider the following example of a multi-container group with two containers.



The multi-container group has the following characteristics and configuration:

- The container group is scheduled on a single host machine, and is assigned a DNS name label.
- The container group exposes a single public IP address with one exposed port.
- One container in the group listens on port 80. The other container listens on port 1433.
- The group includes two Azure Files file shares as volume mounts. Each container in the group mounts one of the file shares locally.

Things to consider when using container groups

Multi-container groups are useful when you want to divide a single functional task into a few container images. Different teams can deliver the images, and the images can have separate resource requirements.

Consider the following scenarios for working with multi-container groups. Think about what options can support your internal apps for the online retailer.

- **Consider web app updates.** Support updates to your web apps by implementing a multi-container group. One container in the group serves the web app and another container pulls the latest content from source control.
- **Consider log data collection.** Use a multi-container group to capture logging and metrics data about your app. Your application container outputs logs and metrics. A logging container collects the output data and writes the data to long-term storage.
- **Consider app monitoring.** Enable monitoring for your app with a multi-container group. A monitoring container periodically makes a request to your application container to ensure your app is running and responding correctly. The monitoring container raises an alert if it identifies possible issues with your app.
- **Consider front-end and back-end support.** Create a multi-container group to hold your front-end container and back-end container. The front-end container can serve a web app. The back-end container can run a service to retrieve data.

5. Review Azure Container Apps

<https://learn.microsoft.com/en-us/training/modules/configure-azure-container-instances/5-review-docker-platform>

Review Azure Container Apps

Completed

There are many options for teams to build and deploy cloud native and containerized applications on Azure. Let's understand which scenarios and use cases are best suited for Azure Container Apps and how it compares to other container options on Azure. Listen to a developer's view of Azure Container Instances.

Things to know about Azure Container Apps

[Azure Container Apps](#) is a serverless platform that allows you to maintain less infrastructure and save costs while running containerized applications. Instead of worrying about server configuration, container orchestration, and deployment details, Container Apps provides all the up-to-date server resources required to keep your applications stable and secure.

Common uses of Azure Container Apps include:

- Deploying API endpoints
- Hosting background processing jobs
- Handling event-driven processing
- Running microservices

Applications built on Azure Container Apps can dynamically scale based on the following characteristics:

- HTTP traffic
- Event-driven processing
- CPU or memory load
- Any KEDA-supported scaler

Things to consider when using Azure Container Apps

Azure Container Apps enables you to build serverless microservices and jobs based on containers. Distinctive features of Container Apps include:

- Optimized for running general purpose containers, especially for applications that span many microservices deployed in containers.
- Powered by Kubernetes and open-source technologies like Dapr, KEDA, and envoy.
- Supports Kubernetes-style apps and microservices with features like service discovery and traffic splitting.
- Enables event-driven application architectures by supporting scale based on traffic and pulling from event sources like queues, including scale to zero.
- Supports running on demand, scheduled, and event-driven jobs.

Azure Container Apps doesn't provide direct access to the underlying Kubernetes APIs. If you would like to build Kubernetes-style applications and don't require direct access to all the native Kubernetes APIs and cluster management, Container Apps provides a fully managed experience based on best-practices. For these reasons, many teams prefer to start building container microservices with Azure Container Apps.

Compare container management solutions

Azure offers several container platforms for different scenarios. Azure Container Instances (ACI) is best for isolated, short-lived tasks. Azure Container Apps (ACA) serves serverless microservices. Azure Kubernetes Service (AKS) provides full Kubernetes control for complex orchestration needs.

Feature	Azure Container Apps (ACA)	Azure Kubernetes Service (AKS)
Overview	ACA is a serverless container platform that simplifies the deployment and management of microservices-based applications by abstracting away the underlying infrastructure.	AKS simplifies deploying a managed Kubernetes cluster in Azure by offloading the operational overhead to Azure. It's suitable for complex applications that require orchestration.
Deployment	ACA provides a PaaS experience with quick deployment and management capabilities.	AKS offers more control and customization options for Kubernetes environments, making it suitable for complex applications and microservices.
Management	ACA builds upon AKS and offers a simplified PaaS experience for running containers.	AKS provides a more granular control over the Kubernetes environment, suitable for teams with Kubernetes expertise.

Feature	Azure Container Apps (ACA)	Azure Kubernetes Service (AKS)
Scalability	ACA supports both HTTP-based autoscaling and event-driven scaling, making it ideal for applications that need to respond quickly to changes in demand.	AKS offers horizontal pod autoscaling and cluster autoscaling, providing robust scalability options for containerized applications.
Use Cases	ACA is designed for microservices and serverless applications that benefit from rapid scaling and simplified management.	AKS is best for complex, long-running applications. These applications require full Kubernetes features and tight integration with other Azure services.

6. Exercise: Implement Container Instances

<https://learn.microsoft.com/en-us/training/modules/configure-azure-container-instances/6-simulation-containers>

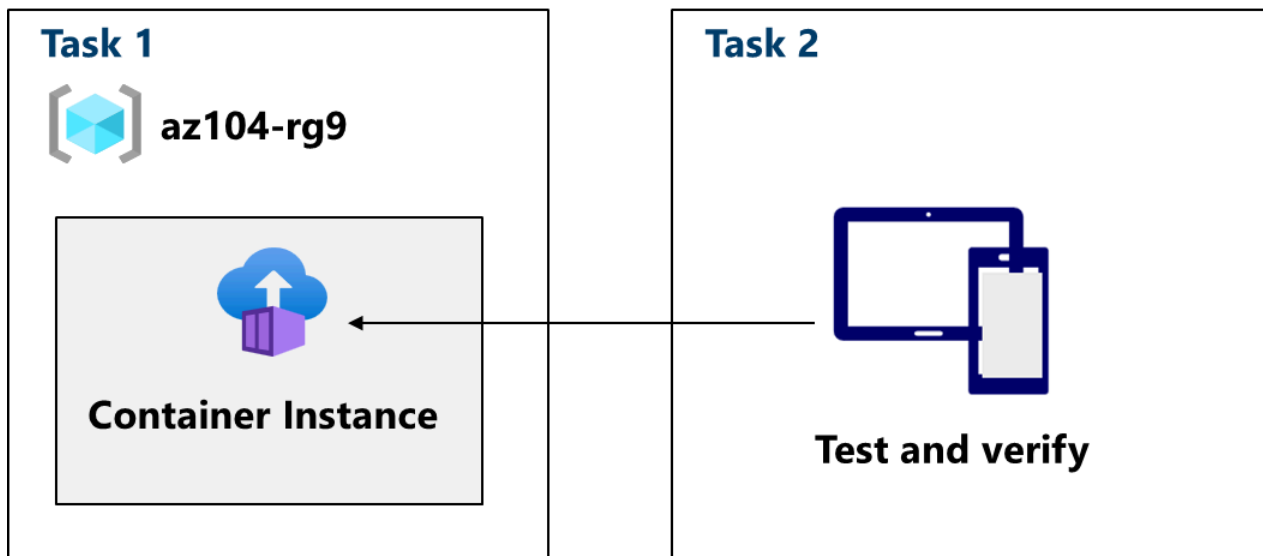
Exercise: Implement Container Instances

Completed

Lab scenario

Your organization has a web application that runs on a virtual machine in your on-premises data center. The organization wants to move all applications to the cloud but doesn't want to have a large number of servers to manage. You decide to evaluate Azure Container Instances and Docker.

Architecture diagram



Job skills

- Deploy an Azure Container Instance using a Docker image.
- Test and verify deployment of an Azure Container Instance.

Note

Estimated time: 15 minutes. To complete this exercise, you need an [Azure subscription](#).

Launch the exercise, and follow the instructions. When finished, be sure to return to this page so you can continue learning.

7. Module assessment

<https://learn.microsoft.com/en-us/training/modules/configure-azure-container-instances/7-knowledge-check>

Module assessment

Completed

8. Summary and resources

<https://learn.microsoft.com/en-us/training/modules/configure-azure-container-instances/8-summary-resources>

Summary and resources

Completed

In this module, you learned how to identify when to use Azure Container Instances versus Azure virtual machines. You explored the features and usage cases of Azure Container Instances. You discovered how to implement Azure container groups.

The main takeaways from this module are:

- Containers provide lightweight isolation and use fewer system resources compared to virtual machines.
- Containers can be deployed individually using Docker or with an orchestrator like Azure Container Apps.
- Containers use Azure Disks or Azure Files for storage.
- A container group is a collection of containers that get scheduled on the same host machine.
- Containers can be rapidly recreated on another cluster node if a node fails.

Learn more with Copilot

Copilot can assist you in configuring Azure infrastructure solutions. Copilot can compare, recommend, explain, and research products and services where you need more information. Open a Microsoft Edge browser and choose Copilot (top right) or navigate to copilot.microsoft.com. Take a few minutes to try these prompts and extend your learning with Copilot.

- Compare benefits and usage cases for containers and virtual machines.
- What are the best practices for configuring Azure Container Instances for task-based workloads? Explain the restart policies.
- How do I deploy a multi-container group in Azure Container Instances using Bicep? Show an example with environment variables.

Learn more with documentation

- [Containers versus virtual machines](#). This article reviews the key similarities and differences between containers and virtual machines (VMs), and when you might want to use each.
- [Quickstart: Deploy a container instance in Azure using the Azure portal](#). In this quickstart, you use the Azure portal to deploy an isolated Docker container and make its application available with a fully qualified domain name (FQDN). After configuring a few settings and deploying the container, you can browse to the running application:
- [Container groups in Azure Container Instances](#). This article describes what container groups are and the types of scenarios they enable.

Learn more with self-paced training

- [Run container images in Azure Container Instances](#). Learn how Azure Container Instances can help you quickly deploy containers, how to set environment variables, and specify container restart policies.
- [Implement Azure Container Apps](#). Learn how Azure Container Apps can help you deploy and manage microservices and containerized apps on a serverless platform that runs on top of Azure Kubernetes Service.
- [Introduction to Docker containers](#). Learn the benefits of using Docker containers as a containerization platform. Discuss the infrastructure provided by the Docker platform.