AZ-204

Certification details

CERTIFICATION EXAM

- Developing Solutions for Microsoft Azure

Skills measured

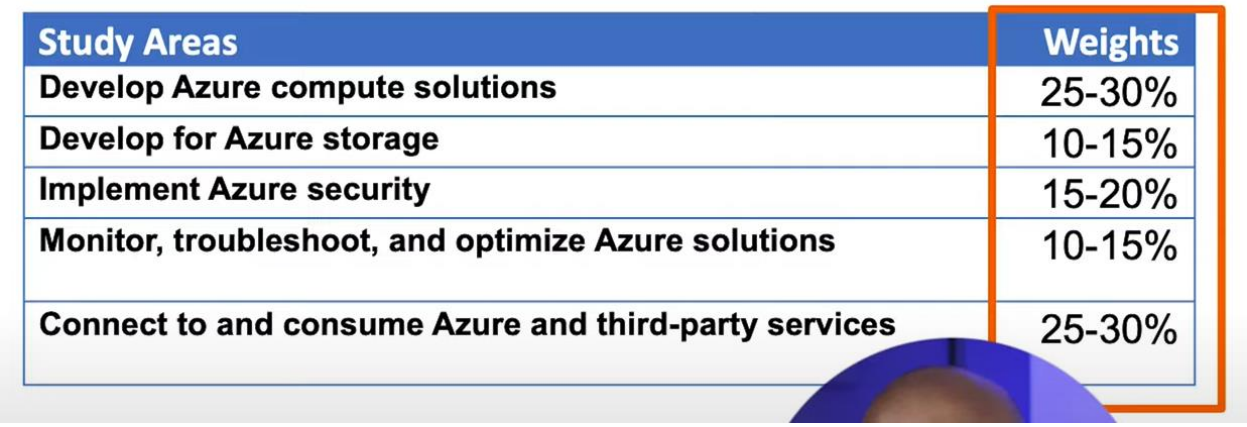
- This list contains the skills measured on the exam associated with this certification. For information about upcoming or recent changes, see the associated exam details page and download the exam skills outline.
- Develop Azure compute solutions
- Develop for Azure storage
- Implement Azure security
- Monitor, troubleshoot, and optimize Azure solutions
- Connect to and consume Azure services and third-party services

Syllabus

- **AZ-204: Create Azure App Service web apps**
- **AZ-204: Implement Azure Functions**
- **AZ-204: Develop solutions that use Blob storage**
- **AZ-204: Develop solutions that use Azure Cosmos DB**
- **AZ-204: Implement infrastructure as a service solution**
- **AZ-204: Implement user authentication and authorization**
- **AZ-204: Implement secure cloud solutions**
- **AZ-204: Implement API Management**
- **AZ-204: Develop event-based solutions**
- **AZ-204: Develop message-based solutions**
- **AZ-204: Instrument solutions to support monitoring and logging**
- **AZ-204: Integrate caching and content delivery within solutions**

- Azure App Service
- Azure Functions
- Azure Blob storage
- Azure Cosmos DB
- Infrastructure as a service solution
- Authentication and Authorization
- Secure Cloud solution
- API Management
- Event-based solutions
- Message-based solutions

- Monitoring and Logging
- caching and content delivery within solutions



Study Areas	Weights
Develop Azure compute solutions	25-30%
Develop for Azure storage	10-15%
Implement Azure security	15-20%
Monitor, troubleshoot, and optimize Azure solutions	10-15%
Connect to and consume Azure and third-party services	25-30%

https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE4oZ7B

https://github.com/MicrosoftLearning/AZ-204-DevelopingSolutionsforMicrosoftAzure

1. Create Azure App Service web apps

Azure App Service is an HTTP-based service for hosting web applications, REST APIs, and mobile back ends. You can develop in your favorite language, be it .NET, .NET Core, Java, Ruby, Node.js, PHP, or Python. Applications run and scale with ease on both Windows and Linux-based environments.

- Built-in auto scale support
- Continuous integration/deployment support
- Deployment slots
- Pricing tier (Free, Shared, Basic, Standard, Premium, PremiumV2, PremiumV3, Isolated)

Identity providers

App Service uses federated identity, in which a third-party identity provider manages the user identities and authentication flow for you. The following identity providers are available by default:

Provider	**Sign-in endpoint**	**How-To guidance**
Microsoft Identity Platform	`/.auth/login/aad`	App Service Microsoft Identity Platform login
Facebook	`/.auth/login/facebook`	App Service Facebook login
Google	`/.auth/login/google`	App Service Google login
Twitter	`/.auth/login/twitter`	App Service Twitter login
Any OpenID Connect provider	`/.auth/login/<providerName>`	App Service OpenID Connect login

Inbound features	**Outbound features**
App-assigned address	Hybrid Connections
Access restrictions	Gateway-required VNet Integration
Service endpoints	VNet Integration
Private endpoints	

Create Web App

```
az webapp up --location <myLocation> --name <myAppName> --html
```

Delete Web App

```
az group delete --name <resource_group> --no-wait
```

1. Which of the following App Service plans supports only function apps?

- ○ Dedicated
- ◉ Isolated

✗ That's incorrect. Isolated supports all of the app types available in App Service.

- ○ Consumption

✓ That's correct. The consumption tier is only available to function apps. It scales the functions dynamically depending on workload.

2. Which of the following networking features of App Service can be used to control outbound network traffic?

- ○ App-assigned address
- ◉ Hybrid Connections

✓ That's correct. Hybrid Connections are an outbound network feature.

- ○ Service endpoints

Logs

Type	Platform	Location	Description
Application logging	Windows, Linux	App Service file system and/or Azure Storage blobs	Logs messages generated by your application code. The messages can be generated by the web framework you choose, or from your application code directly using the standard logging pattern of your language. Each message is assigned one of the following categories: **Critical**, **Error**, **Warning**, **Info**, **Debug**, and **Trace**.
Web server logging	Windows	App Service file system or Azure Storage blobs	Raw HTTP request data in the W3C extended log file format. Each log message includes data like the HTTP method, resource URI, client IP, client port, user agent, response code, and so on.
Detailed error logging	Windows	App Service file system	Copies of the *.htm* error pages that would have been sent to the client browser. For security reasons, detailed error pages shouldn't be sent to clients in production, but App Service can save the error page each time an application error occurs that has HTTP code 400 or greater.
Failed request tracing	Windows	App Service file system	Detailed tracing information on failed requests, including a trace of the IIS components used to process the request and the time taken in each component. One folder is generated for each failed request, which contains the XML log file, and the XSL stylesheet to view the log file with.

Type	Platform	Location	Description
Deployment logging	Windows, Linux	App Service file system	Helps determine why a deployment failed. Deployment logging happens automatically and there are no configurable settings for deployment logging.

Ssl certificates

You have been asked to help secure information being transmitted between your companies app and the customer. Azure App Service has tools that let you create, upload, or import a private certificate or a public certificate into App Service.

The table below details the options you have for adding certificates in App Service:

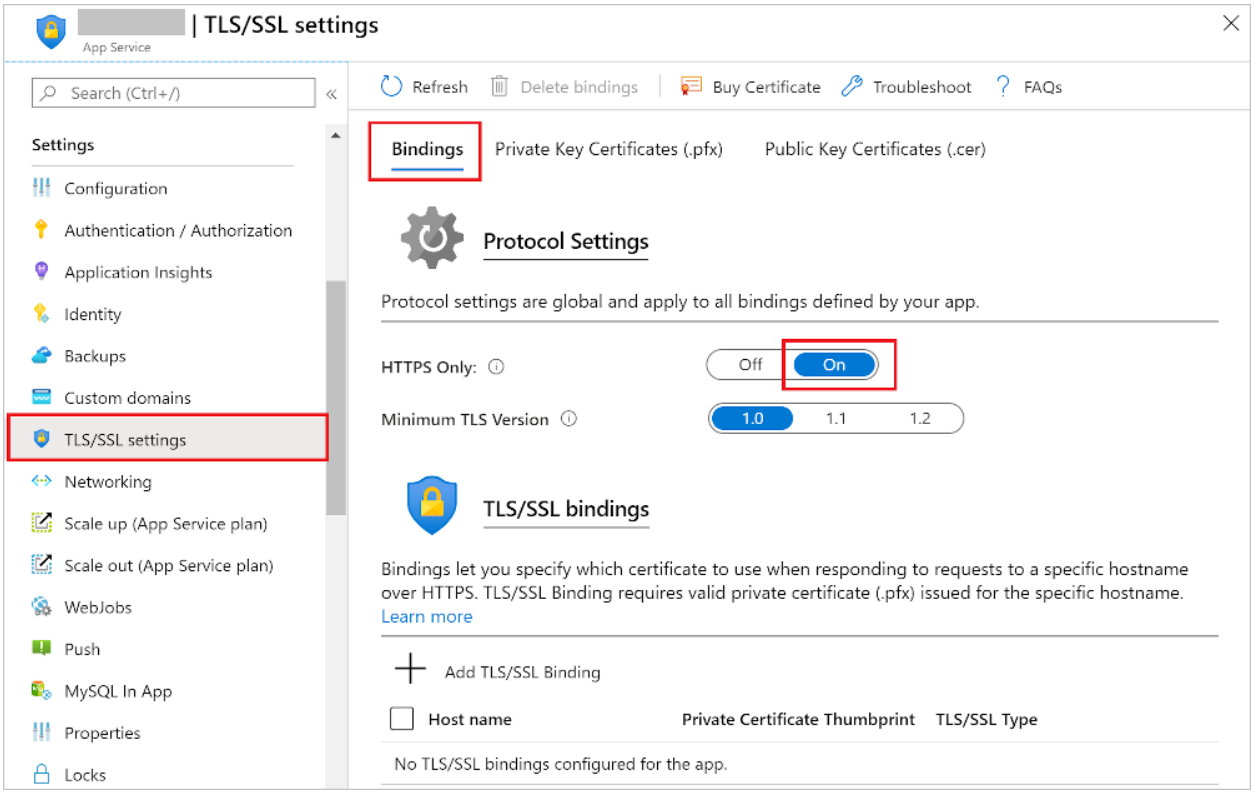
Option	Description
Create a free App Service managed certificate	A private certificate that's free of charge and easy to use if you just need to secure your custom domain in App Service.
Purchase an App Service certificate	A private certificate that's managed by Azure. It combines the simplicity of automated certificate management and the flexibility of renewal and export options.
Import a certificate from Key Vault	Useful if you use Azure Key Vault to manage your certificates.
Upload a private certificate	If you already have a private certificate from a third-party provider, you can upload it.
Upload a public certificate	Public certificates are not used to secure custom domains, but you can load them into your code if you need them to access remote resources.

Upload private certificate

```
openssl pkcs12 -export -out myserver.pfx -inkey <private-key-file> -in <merged-certificate-file>
```

Enforce HTTPS

By default, anyone can still access your app using HTTP. You can redirect all HTTP requests to the HTTPS port by navigating to your app page and, in the left navigation, select **TLS/SSL settings**. Then, in **HTTPS Only**, select **On**.


| TLS/SSL settings
App Service
Search (Ctrl+/)
Settings
Configuration
Authentication / Authorization
Application Insights
Identity
Backups
Custom domains
TLS/SSL settings
Networking
Scale up (App Service plan)
Scale out (App Service plan)
WebJobs
Push
MySQL In App
Properties
Locks
Refresh
Delete bindings
Buy Certificate
Troubleshoot
FAQs
Bindings
Private Key Certificates (.pfx)
Public Key Certificates (.cer)
Protocol Settings
Protocol settings are global and apply to all bindings defined by your app.
HTTPS Only:
Off
On
Minimum TLS Version
1.0
1.1
1.2
TLS/SSL bindings
Bindings let you specify which certificate to use when responding to requests to a specific hostname over HTTPS. TLS/SSL Binding requires valid private certificate (.pfx) issued for the specific hostname.
Learn more
Add TLS/SSL Binding
Host name
Private Certificate Thumbprint
TLS/SSL Type
No TLS/SSL bindings configured for the app.

1. In which of the app configuration settings categories below would you set the language and SDK version?

- (●) Application settings
  - ✗ That's incorrect. This category is used to configure settings that are passed to your app at runtime.
- ( ) Path mappings
- ( ) General settings
  - ✓ That's correct. This category is used to configure stack, platform, debugging, and incoming client certificate settings.

2. Which of the following types of application logging is supported on the Linux platform?

- ( ) Web server logging
- (●) Failed request tracing
  - ✗ That's incorrect. Failed request tracing is not supported on the Linux platform.
- ( ) Deployment logging
  - ✓ That's correct. Deployment logging is supported on the Linux platform.

3. Which of the following choices correctly lists the two parts of a feature flag?

- ( ) Name, App Settings
- (●) Name, one or more filters
  - ✓ That's correct. Each feature flag has two parts: a name and a list of one or more filters that are used to evaluate if a feature's state is *on*.
- ( ) Feature manager, one or more filters

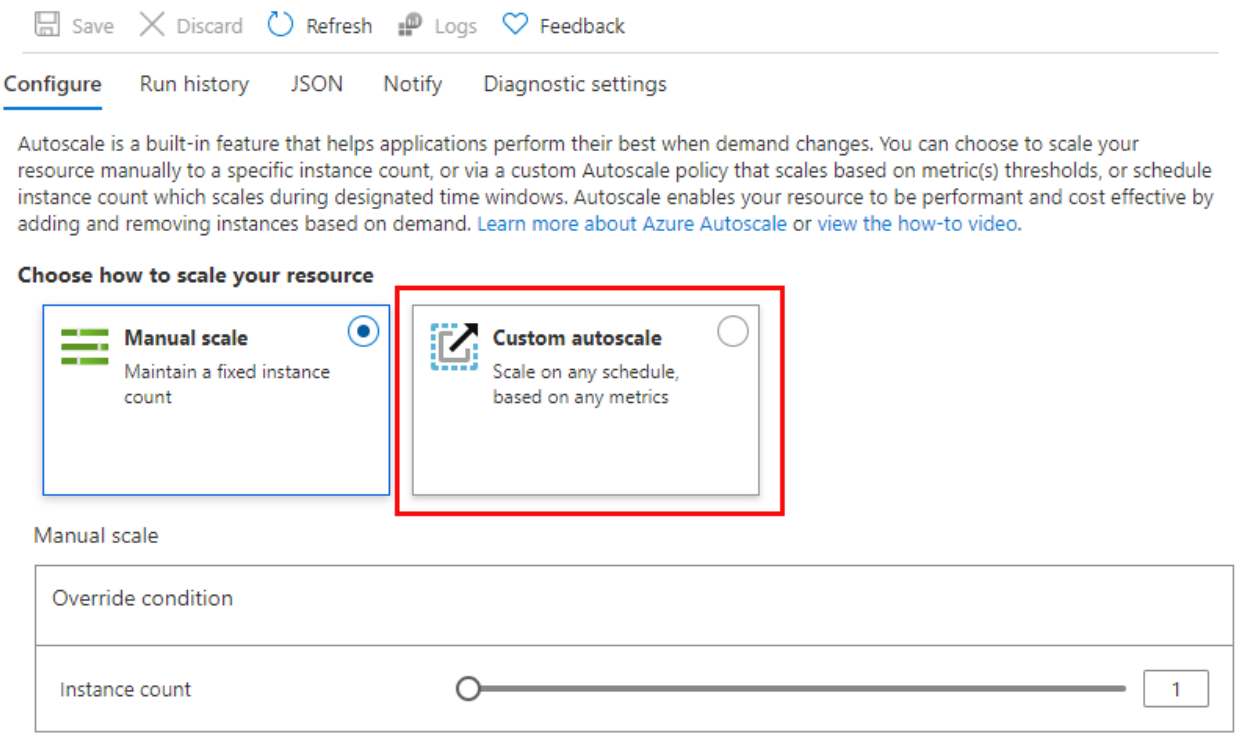
Auto Scaling

Autoscaling is a feature of the App Service Plan used by the web app. When the web app scales out, Azure starts new instances of the hardware defined by the App Service Plan to the app.

Not all pricing tiers support autoscaling. The development pricing tiers are either limited to a single instance (the **F1** and **D1** tiers), or they only provide manual scaling (the **B1** tier). If you've selected one of these tiers, you must first scale up to the **S1** or any of the **P** level production tiers.

Enable autoscaling

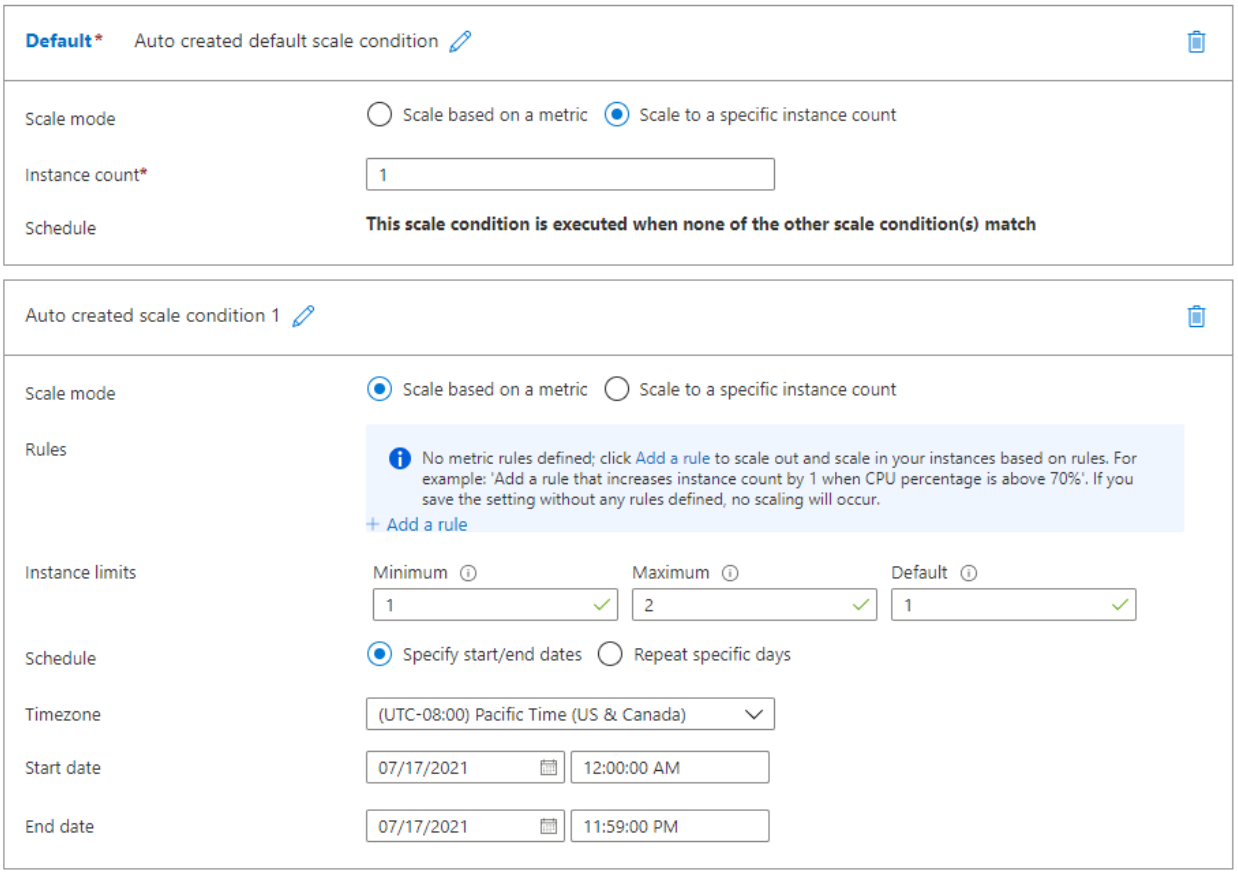
By default, an App Service Plan only implements manual scaling. Selecting **Custom autoscale** reveals condition groups you can use to manage your scale settings.

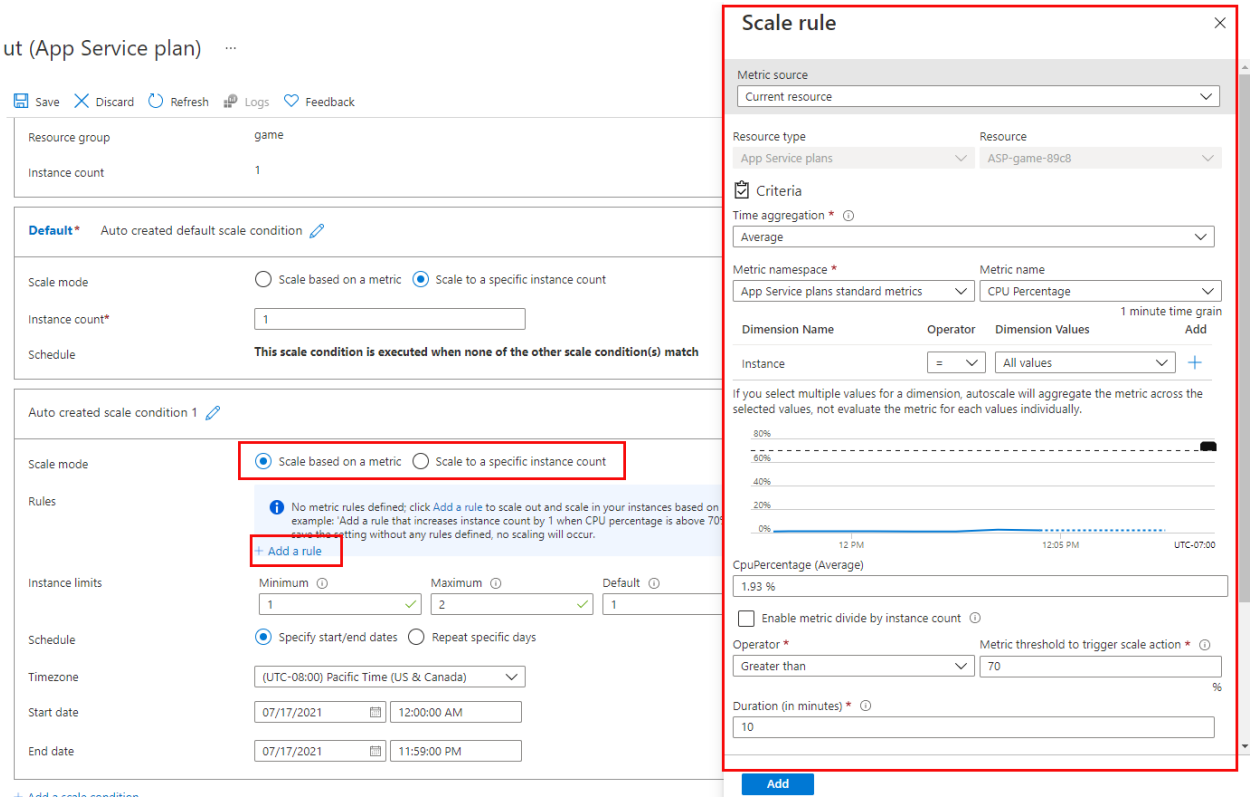


Add scale conditions

Once you enable autoscaling, you can edit the automatically created default scale condition, and you can add your own custom scale conditions. Remember that each scale condition can either scale based on a metric, or scale to a specific instance count.

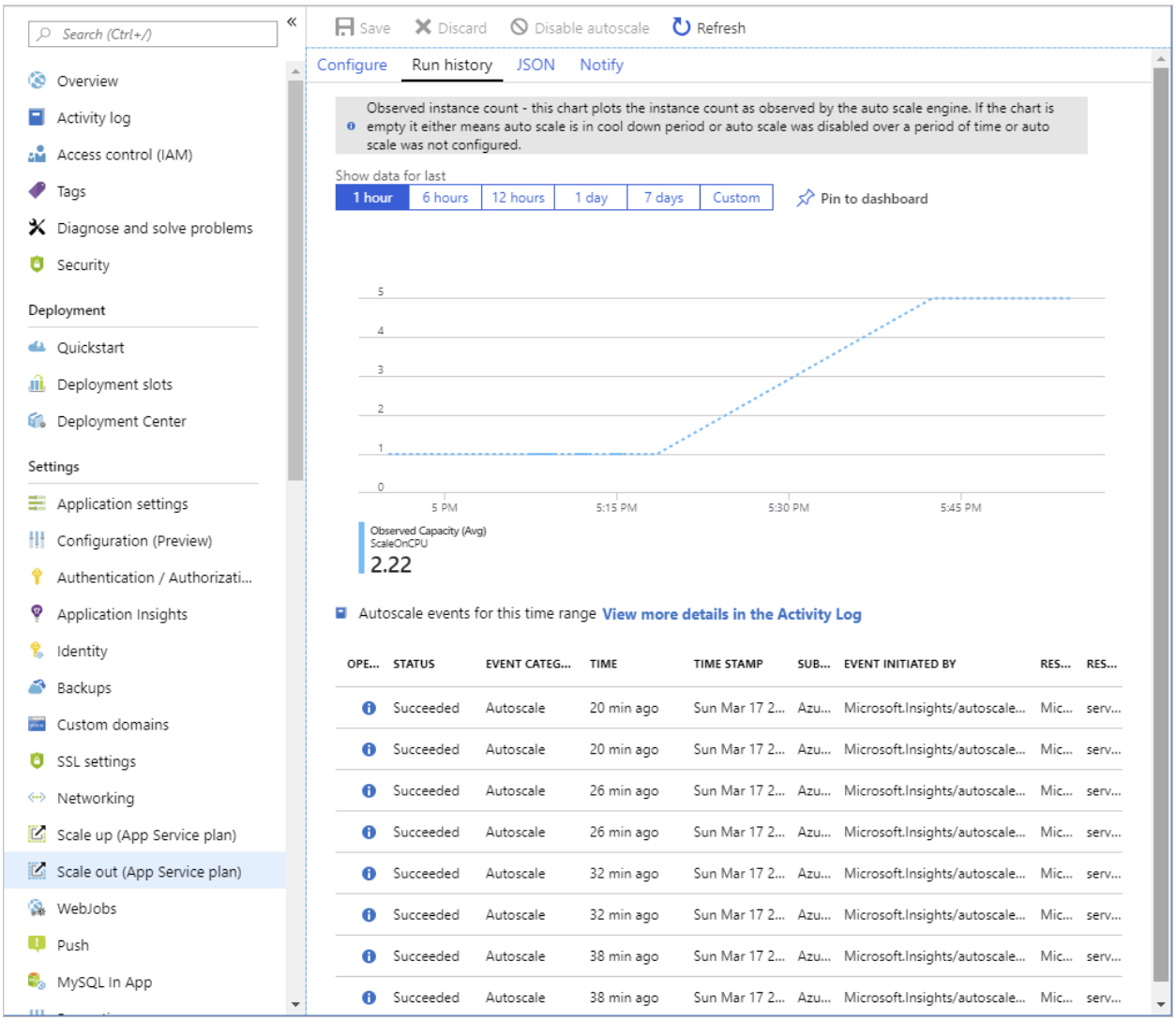
The Default scale condition is executed when none of the other scale conditions are active.


Default* Auto created default scale condition
Scale mode
Scale based on a metric
Scale to a specific instance count
Instance count*
1
Schedule
This scale condition is executed when none of the other scale condition(s) match
Auto created scale condition 1
Scale mode
Scale based on a metric
Scale to a specific instance count
Rules
No metric rules defined; click Add a rule to scale out and scale in your instances based on rules. For example: 'Add a rule that increases instance count by 1 when CPU percentage is above 70%'. If you save the setting without any rules defined, no scaling will occur.
Add a rule
Instance limits
Minimum
Maximum
Default
1
2
1
Schedule
Specify start/end dates
Repeat specific days
Timezone
(UTC-08:00) Pacific Time (US & Canada)
Start date
07/17/2021
12:00:00 AM
End date
07/17/2021
11:59:00 PM

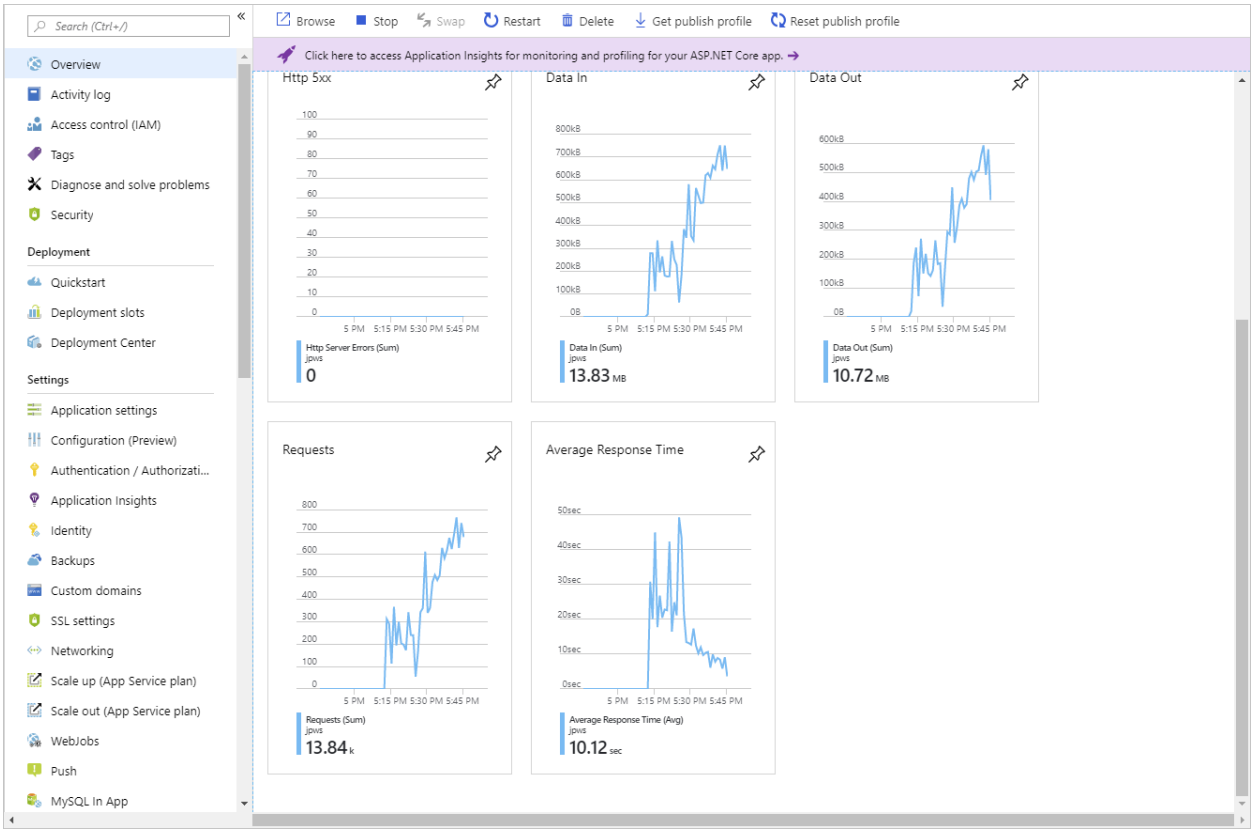

ut (App Service plan)
Save
Discard
Refresh
Logs
Feedback
Resource group
game
Instance count
1
Default* Auto created default scale condition
Scale mode
Scale based on a metric
Scale to a specific instance count
Instance count*
1
Schedule
This scale condition is executed when none of the other scale condition(s) match
Auto created scale condition 1
Scale mode
Scale based on a metric
Scale to a specific instance count
Rules
Add a rule
Instance limits
Minimum
Maximum
Default
1
2
1
Schedule
Specify start/end dates
Repeat specific days
Timezone
(UTC-08:00) Pacific Time (US & Canada)
Start date
07/17/2021
12:00:00 AM
End date
07/17/2021
11:59:00 PM
Scale rule
Metric source
Current resource
Resource type
App Service plans
Resource
ASP-game-89c8
Criteria
Time aggregation *
Average
Metric namespace *
App Service plans standard metrics
Metric name
CPU Percentage
1 minute time grain
Dimension Name
Operator
Dimension Values
Add
Instance
=
All values
If you select multiple values for a dimension, autoscale will aggregate the metric across the selected values, not evaluate the metric for each values individually.
80%
60%
40%
20%
0%
12 PM
12:05 PM
UTC-07:00
CpuPercentage (Average)
1.93 %
Enable metric divide by instance count
Operator *
Greater than
Metric threshold to trigger scale action *
70
%
Duration (in minutes) *
10
Add

Monitor autoscaling activity

The Azure portal enables you to track when autoscaling has occurred through the **Run history** chart. This chart shows how the number of instances varies over time, and which autoscale conditions caused each change.



You can use the **Run history** chart in conjunction with the metrics shown on the **Overview** page to correlate the autoscaling events with resource utilization.


Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Security
Deployment
Quickstart
Deployment slots
Deployment Center
Settings
Application settings
Configuration (Preview)
Authentication / Authorizati...
Application Insights
Identity
Backups
Custom domains
SSL settings
Networking
Scale up (App Service plan)
Scale out (App Service plan)
WebJobs
Push
MySQL In App
Browse
Stop
Swap
Restart
Delete
Get publish profile
Reset publish profile
Click here to access Application Insights for monitoring and profiling for your ASP.NET Core app.
Http 5xx
Http Server Errors (Sum)
jpws
0
Data In
Data In (Sum)
jpws
13.83 MB
Data Out
Data Out (Sum)
jpws
10.72 MB
Requests
Requests (Sum)
jpws
13.84 k
Average Response Time
Average Response Time (Avg)
jpws
10.12 sec
5 PM
5:15 PM
5:30 PM
5:45 PM

1. Which of these statements best describes autoscaling?

- ○ Autoscaling requires an administrator to actively monitor the workload on a system.
- ◉ Autoscaling is a scale out/scale in solution.

  ✔ That's correct. The system can scale out when specified resource metrics indicate increasing usage, and scale in when these metrics drop.
- ○ Scaling up/scale down provides better availability than autoscaling.

2. Which of these scenarios is a suitable candidate for autoscaling?

- ○ The number of users requiring access to an application varies according to a regular schedule. For example, more users use the system on a Friday than other days of the week.

  ✔ That's correct. Changes in application load that are predictable are good candidates for autoscaling.
- ○ The system is subject to a sudden influx of requests that grinds your system to a halt.
- ◉ Your organization is running a promotion and expects to see increased traffic to their web site for the next couple of weeks.

  ✘ That's incorrect. Manual scaling is a better option here since this is a one-off event with a known duration.

3. There are multiple rules in an autoscale profile. Which of the following scale operations will run if any of the rule conditions are met?

- ◉ scale-out

  ✔ That's correct. Scale-out operations will trigger if any of the rule conditions are met.
- ○ scale-in
- ○ scale-out/in

Azure App Service deployment slots

Settings that are swapped	**Settings that aren't swapped**
General settings, such as framework version, 32/64-bit, web sockets	Publishing endpoints
App settings (can be configured to stick to a slot)	Custom domain names
Connection strings (can be configured to stick to a slot)	Non-public certificates and TLS/SSL settings
Handler mappings	Scale settings
Public certificates	WebJobs schedulers

WebJobs content	IP restrictions
Hybrid connections *	Always On
Virtual network integration *	Diagnostic log settings
Service endpoints *	Cross-origin resource sharing (CORS)
Azure Content Delivery Network *	

1. By default, all client requests to the app's production URL (`http://<app_name>.azurewebsites.net`) are routed to the production slot. One can automatically route a portion of the traffic to another slot. What is the default routing rule applied to new deployment slots?

- (●) 0%
  - ✔ That's correct. By default, new slots are given a routing rule of `0%`.
- ( ) 10%
- ( ) 20%

2. Some configuration elements follow the content across a swap (not slot specific), whereas other configuration elements stay in the same slot after a swap (slot specific). Which of the settings below are swapped?

- ( ) Publishing endpoints
- (●) WebJobs content
  - ✔ That's correct. WebJobs content are swapped.
- ( ) WebJobs schedulers

Lab

2. Azure Functions

	Azure Functions	**Azure Logic Apps**
Development	Code-first (imperative)	Designer-first (declarative)
Connectivity	About a dozen built-in binding types, write code for custom bindings	Large collection of connectors, Enterprise Integration Pack for B2B scenarios, build custom connectors
Actions	Each activity is an Azure function; write code for activity functions	Large collection of ready-made actions
Monitoring	Azure Application Insights	Azure portal, Azure Monitor logs
Management	REST API, Visual Studio	Azure portal, REST API, PowerShell, Visual Studio
Execution context	Can run locally or in the cloud	Supports run-anywhere scenarios

There are three basic hosting plans available for Azure Functions:

- Consumption plan
- Functions Premium plan
- App service (Dedicated) plan.

All hosting plans are generally available (GA) on both Linux and Windows virtual machines.

Plan	**Benefits**
Consumption plan	This is the **default** hosting plan. It scales automatically and you only pay for compute resources when your functions are running. Instances of the Functions host are dynamically added and removed based on the number of incoming events.
Functions Premium plan	Automatically scales based on demand using pre-warmed workers which run applications with no delay after being idle, runs on more powerful instances, and connects to virtual networks.
App service plan	Run your functions within an App Service plan at regular App Service plan rates. Best for long-running scenarios where Durable Functions can't be used.

There are two other hosting options which provide the highest amount of control and isolation in which to run your function apps.

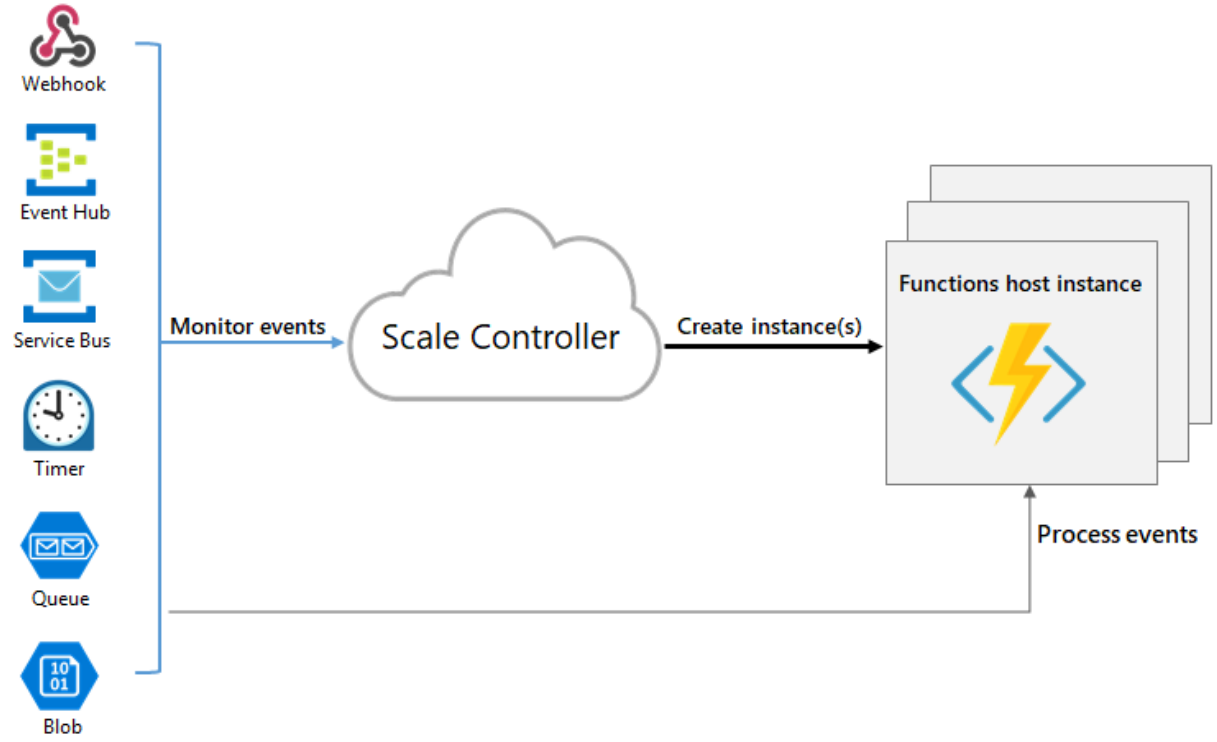
Hosting option	Details
ASE	App Service Environment (ASE) is an App Service feature that provides a fully isolated and dedicated environment for securely running App Service apps at high scale.
Kubernetes	Kubernetes provides a fully isolated and dedicated environment running on top of the Kubernetes platform. For more information visit Azure Functions on Kubernetes with KEDA.

- In the Consumption and Premium plans, Azure Functions scales CPU and memory resources by adding additional instances of the Functions host.

- Using an App Service plan, you can manually scale out by adding more VM instances. You can also enable autoscale, though autoscale will be slower than the elastic scale of the Premium plan

Runtime scaling

Azure Functions uses a component called the *scale controller* to monitor the rate of events and determine whether to scale out or scale in.



1. Which of the following Azure Functions hosting plans is best when predictive scaling and costs are required?

- ○ Functions Premium Plan
- ◉ App service plan

  ✔ That's correct. App service plans support setting autoscaling rules based on predictive usage.
- ○ Consumption plan

2. An organization wants to implement a serverless workflow to solve a business problem. One of the requirements is the solution needs to use a designer-first (declarative) development model. Which of the choices below meets the requirements?

- ○ Azure Functions
- ◉ Azure Logic Apps

  ✔ That's correct. Azure Logic Apps enables serverless workloads and uses a designer-first (declarative) development model.
- ○ WebJobs

Develop Azure Functions

A function contains two important pieces –

- **Code:** which can be written in a variety of languages,
- **Config file** – some configuration data in ***function.json*** file. For compiled languages, this config file is generated automatically from annotations in your code. For scripting languages, you must provide the config file yourself.

The *function.json* file defines the function's trigger, bindings, and other configuration settings. Every function has one and only one trigger. The runtime uses this config file to determine the events to monitor and how to pass data into and return data from a function execution. The following is an example *function.json* file.

```
{
    "disabled":false,
    "bindings":[
        // ... bindings here
        {
            "type": "bindingType",
            "direction": "in",
            "name": "myParamName",
            // ... more depending on binding
        }
    ]
}
```

1. Which of the following is required for a function to run?

- ○ Binding
- ○ Trigger
  - ✔ That's correct. A trigger defines how a function is invoked and a function must have exactly one trigger.
- ◉ Both triggers and bindings
  - ✘ That's incorrect. Only a trigger is required for a function to run.

2. Which of the following supports both the `in` and `out` direction settings?

- ◉ Bindings
  - ✔ That's correct. Input and output bindings use `in` and `out`.
- ○ Trigger
- ○ Connection value

Durable Functions

Durable Functions is an **extension of Azure Functions** that lets you write **stateful functions** in a serverless compute environment.

The *durable functions* extension lets you define stateful workflows by writing *orchestrator functions* and stateful entities by writing *entity functions* using the Azure Functions programming model. Behind the scenes, the extension manages state, checkpoints, and restarts for you, allowing you to focus on your business logic.

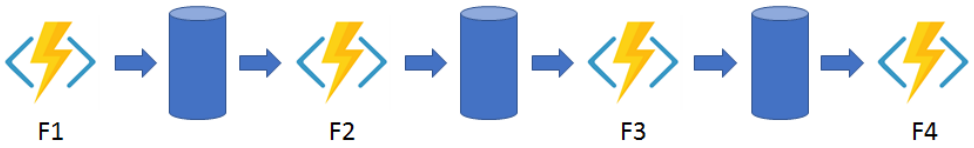
Application patterns

The primary use case for Durable Functions is simplifying complex, stateful coordination requirements in serverless applications. The following sections describe typical application patterns that can benefit from Durable Functions:

- Function chaining
- Fan-out/fan-in
- Async HTTP APIs
- Monitor
- Human interaction

Function chaining

In the function chaining pattern, a sequence of functions executes in a specific order. In this pattern, the output of one function is applied to the input of another function.

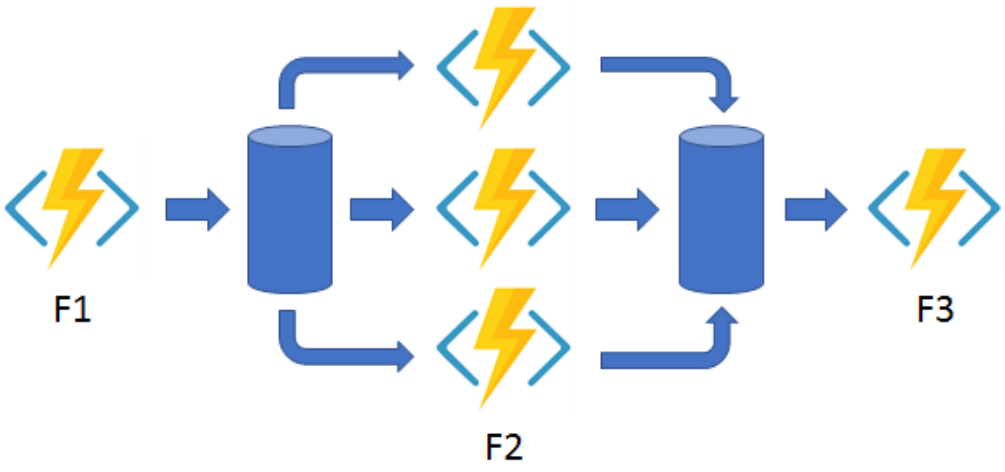


You can include error handling logic in **try/catch/finally** blocks.

```
[FunctionName("Chaining")]
public static async Task<object> Run(
    [OrchestrationTrigger] IDurableOrchestrationContext context)
{
    try
    {
        var x = await context.CallActivityAsync<object>("F1", null);
        var y = await context.CallActivityAsync<object>("F2", x);
        var z = await context.CallActivityAsync<object>("F3", y);
        return  await context.CallActivityAsync<object>("F4", z);
    }
    catch (Exception)
    {
        // Error handling or compensation goes here.
    }
}
```

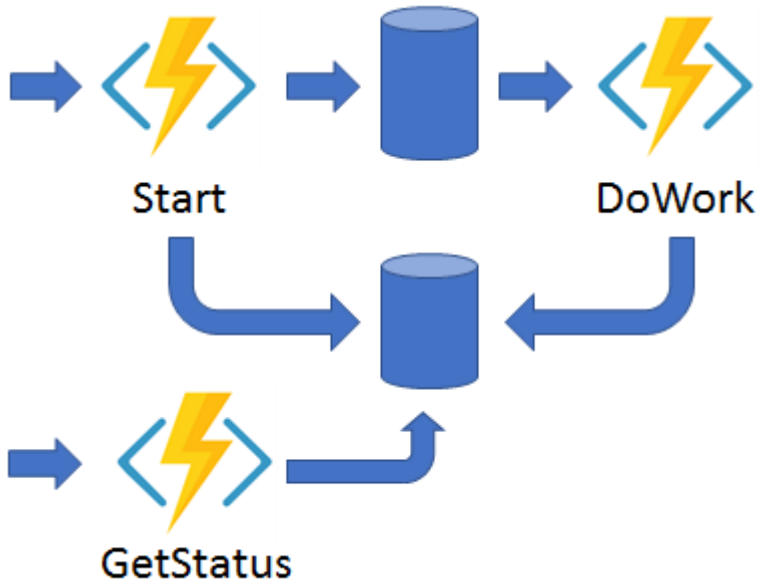
Fan out/fan in

In the fan out/fan in pattern, you execute multiple functions in parallel and then wait for all functions to finish. Often, some aggregation work is done on the results that are returned from the functions.



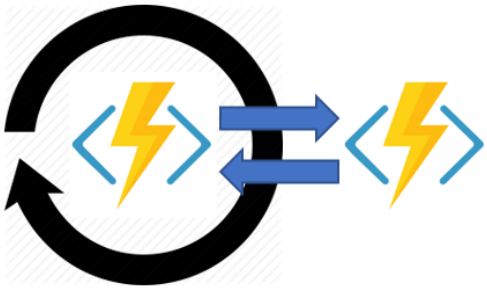
Async HTTP APIs

The async HTTP API pattern addresses the problem of coordinating the state of long-running operations with external clients. A common way to implement this pattern is by having an HTTP endpoint trigger the long-running action. Then, redirect the client to a status endpoint that the client polls to learn when the operation is finished.



Monitor

The monitor pattern refers to a flexible, recurring process in a workflow. An example is polling until specific conditions are met. You can use a regular timer trigger to address a basic scenario, such as a periodic cleanup job, but its interval is static and managing instance lifetimes becomes complex. You can use Durable Functions to create flexible recurrence intervals, manage task lifetimes, and create multiple monitor processes from a single orchestration.



Human interaction

Many automated processes involve some kind of human interaction. Involving humans in an automated process is tricky because people aren't as highly available and as responsive as cloud services. An automated process might allow for this interaction by using timeouts and compensation logic.

Durable Function types (Source [4])

1. Which of the following durable function types is used to read and update small pieces of state?

- ○ Orchestrator
- ○ Activity
- ◉ Entity

✔ That's correct. Entity functions define operations for reading and updating small pieces of state.

2. Which application pattern would you use for a durable function that is polling a resource until a specific condition is met?

- ○ Function chaining
- ○ Fan out/fan in
- ◉ Monitor

✔ That's correct. The monitor pattern refers to a flexible, recurring process in a workflow. An example is polling until specific conditions are met.

3. Azure Blob Storage

Azure Blob storage is Microsoft's object storage solution for the cloud. Blob storage is optimized for storing massive amounts of unstructured data.

Blob storage is designed for:

- Serving images or documents directly to a browser.
- Storing files for distributed access.
- Streaming video and audio.
- Writing to log files.
- Storing data for backup and restore, disaster recovery, and archiving.
- Storing data for analysis by an on-premises or Azure-hosted service.

Users or client applications can access objects in Blob storage via HTTP/HTTPS, from anywhere in the world. Objects in Blob storage are accessible via the Azure Storage REST API, Azure PowerShell, Azure CLI, or an Azure Storage client library.

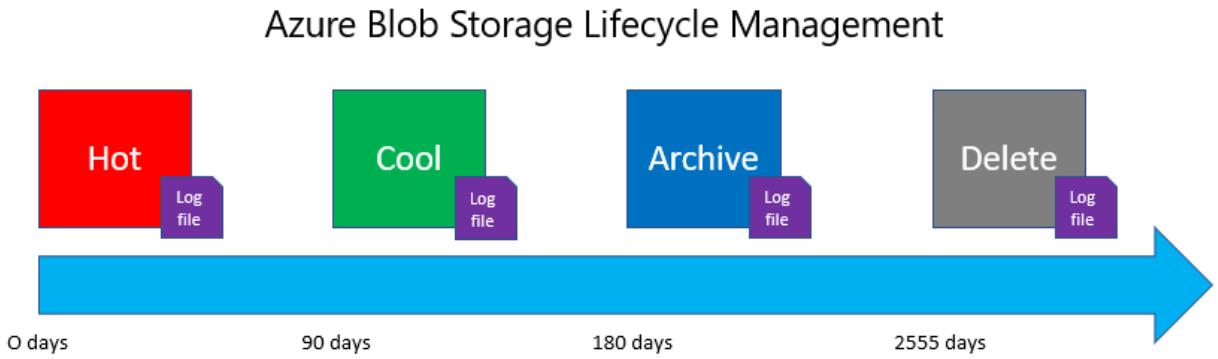
Types of storage accounts

Azure Storage offers two performance levels of storage accounts, standard and premium. Each performance level supports different features and has its own pricing model.

- **Standard:** This is the standard general-purpose v2 account and is recommended for most scenarios using Azure Storage.
- **Premium:** Premium accounts offer higher performance by using solid-state drives. If you create a premium account you can choose between three account types, block blobs, page blobs, or file shares.

The following table describes the types of storage accounts recommended by Microsoft for most scenarios using Blob storage.

Storage account type	**Supported storage services**	**Usage**
Standard general-purpose v2	Blob, Queue, and Table storage, Azure Files	Standard storage account type for blobs, file shares, queues, and tables. Recommended for most scenarios using Azure Storage. If you want support for NFS file shares in Azure Files, use the premium file shares account type.
Premium block blobs	Blob storage	Premium storage account type for block blobs and append blobs. Recommended for scenarios with high transactions rates, or scenarios that use smaller objects or require consistently low storage latency.
Premium page blobs	Page blobs only	Premium storage account type for page blobs only.
Premium file shares	Azure Files	Premium storage account type for file shares only.



Resource Types

Blob storage offers three types of resources:

- The **Storage account**.

- A **Container** in the storage account
- A **Blob** in a container

Create Storage Account

```
az storage account create --resource-group az204-blob-rg --name \
<myStorageAcct> --location <myLocation> \
--kind BlockBlobStorage --sku Premium_LRS
```

1. Which of the following types of blobs are used to store virtual hard drive files?

- (•) Block blobs
  - ✗ That's incorrect. Block blobs are made up of blocks of data that can be managed individually.
- ( ) Append blobs
- ( ) Page blobs
  - ✓ That's correct. Page blobs store random access files up to 8 TB in size, and are used to store virtual hard drive (VHD) files and serve as disks for Azure virtual machines.

2. Which of the following types of storage accounts is recommended for most scenarios using Azure Storage?

- (•) General-purpose v2
  - ✓ That's correct. This supports blobs, files, queues, and tables. It is recommended for most scenarios using Azure Storage.
- ( ) General-purpose v1
- ( ) FileStorage

1. Which access tier is considered to be offline and can't be read or modified?

- ( ) Cool
- (•) Archive
  - ✓ That's correct. Blobs in the archive tier must be rehydrated to either the hot or cool tears before it can be read or modified.
- ( ) Hot

2. Which of the following storage account types supports lifecycle policies?

- (•) General Purpose v1
  - ✗ That's incorrect. General Purpose v1 accounts need to be upgraded to v2 before lifecycle policies are supported.
- ( ) General Purpose v2
  - ✓ That's correct. Azure Blob storage lifecycle management offers a rich, rule-based policy for General Purpose v2 and Blob storage accounts.
- ( ) FileStorage

Azure Storage client libraries for .NET

The Azure Storage client libraries for .NET offer a convenient interface for making calls to Azure Storage. The latest version of the Azure Storage client library is version 12.x. Microsoft recommends using version 12.x for new applications.

Below are the classes in the **`Azure.Storage.Blobs`** namespace and their purpose:

Class	Description
`BlobClient`	The `BlobClient` allows you to manipulate Azure Storage blobs.
`BlobClientOptions`	Provides the client configuration options for connecting to Azure Blob Storage.
`BlobContainerClient`	The `BlobContainerClient` allows you to manipulate Azure Storage containers and their blobs.
`BlobServiceClient`	The `BlobServiceClient` allows you to manipulate Azure Storage service resources and blob co account provides the top-level namespace for the Blob service.
`BlobUriBuilder`	The `BlobUriBuilder` class provides a convenient way to modify the contents of a Uri instance t Azure Storage resources like an account, container, or blob.

Retrieve container properties

To retrieve container properties, call one of the following methods of the BlobContainerClient class:

```
GetProperties
GetPropertiesAsync
```

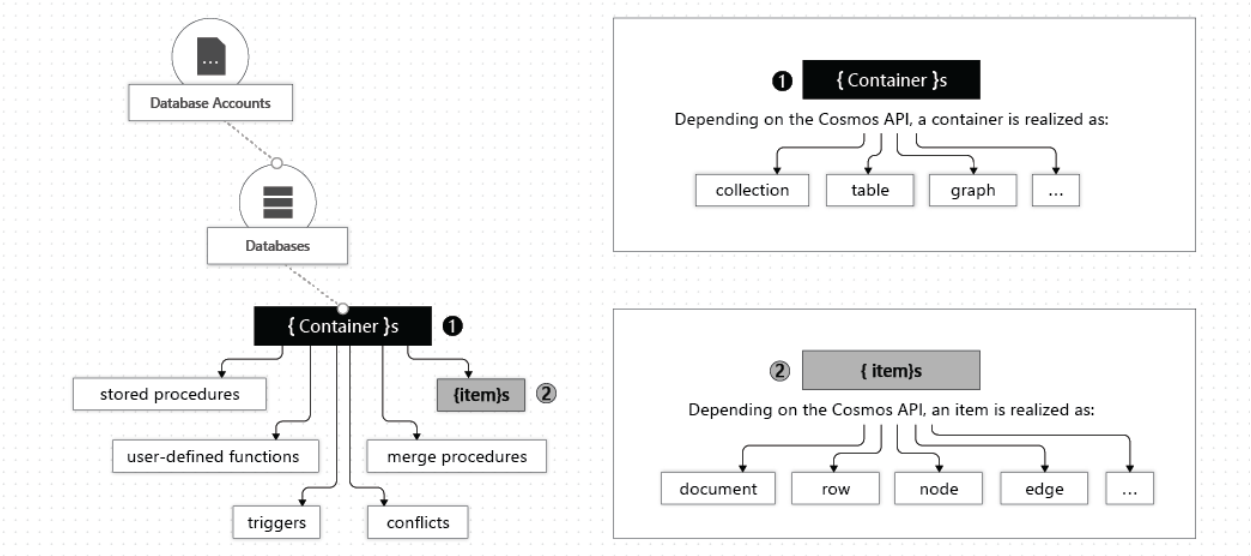
1. Which of the following standard HTTP headers are supported for both containers and blobs when setting properties by using REST?

- ○ Last-Modified
  - ✔ That's correct. Last-Modified is supported on both containers and blobs.
- ○ Content-Length
- ◉ Origin
  - ✘ That's incorrect. Origin is only supported on blobs.

2. Which of the following classes of the Azure Storage client library for .NET allows you to manipulate both Azure Storage containers and their blobs?

- ○ BlobClient
- ◉ BlobContainerClient
  - ✔ That's correct. The BlobContainerClient can be used to manipulate both containers and blobs.
- ○ BlobUriBuilder

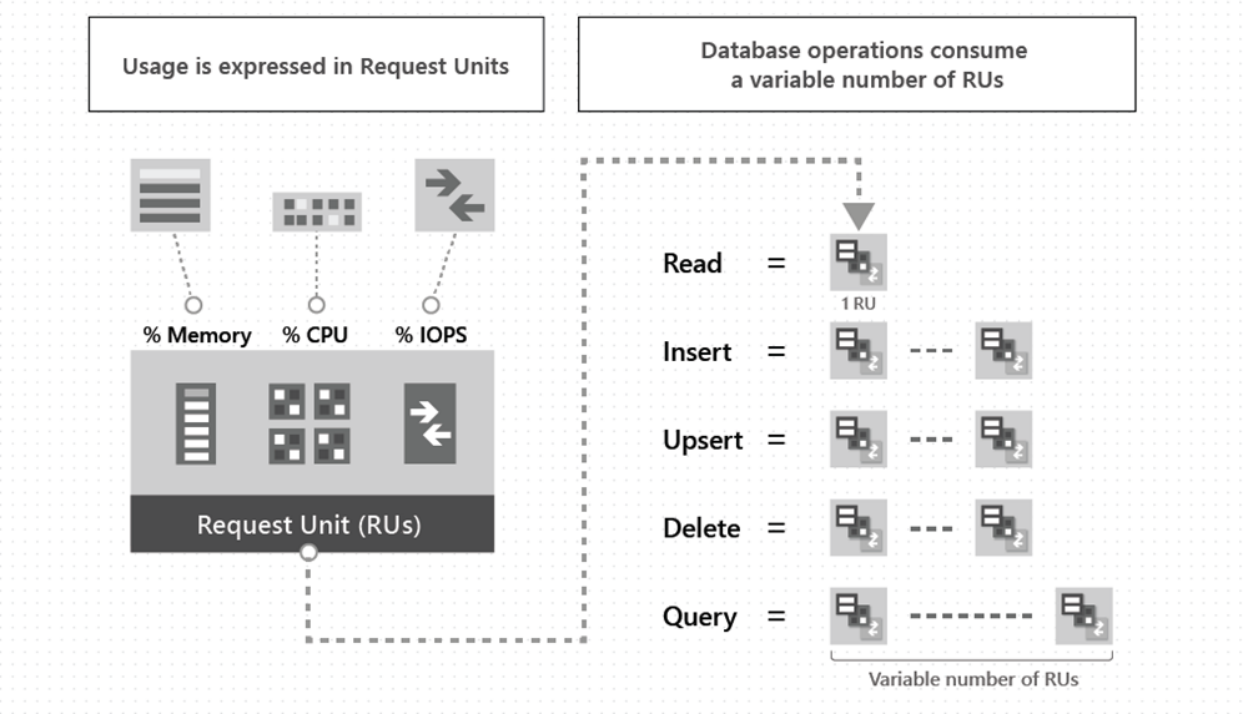
4. Azure Cosmos DB



Azure Cosmos items

Depending on which API you use, an Azure Cosmos item can represent either a document in a collection, a row in a table, or a node or edge in a graph. The following table shows the mapping of API-specific entities to an Azure Cosmos item:

Cosmos entity	SQL API	Cassandra API	Azure Cosmos DB API for MongoDB	Gremlin API	Table API
Azure Cosmos item	Item	Row	Document	Node or edge	Item



1. When setting up Azure Cosmos DB there are three account type options. Which of the account type options below used to specify the number of RUs for an application on a per-second basis?

- ○ Provisioned throughput

  ✔ That's correct. In this mode, you provision the number of RUs for your application on a per-second basis in increments of 100 RUs per second.

- ○ Serverless
- ◉ Autoscale

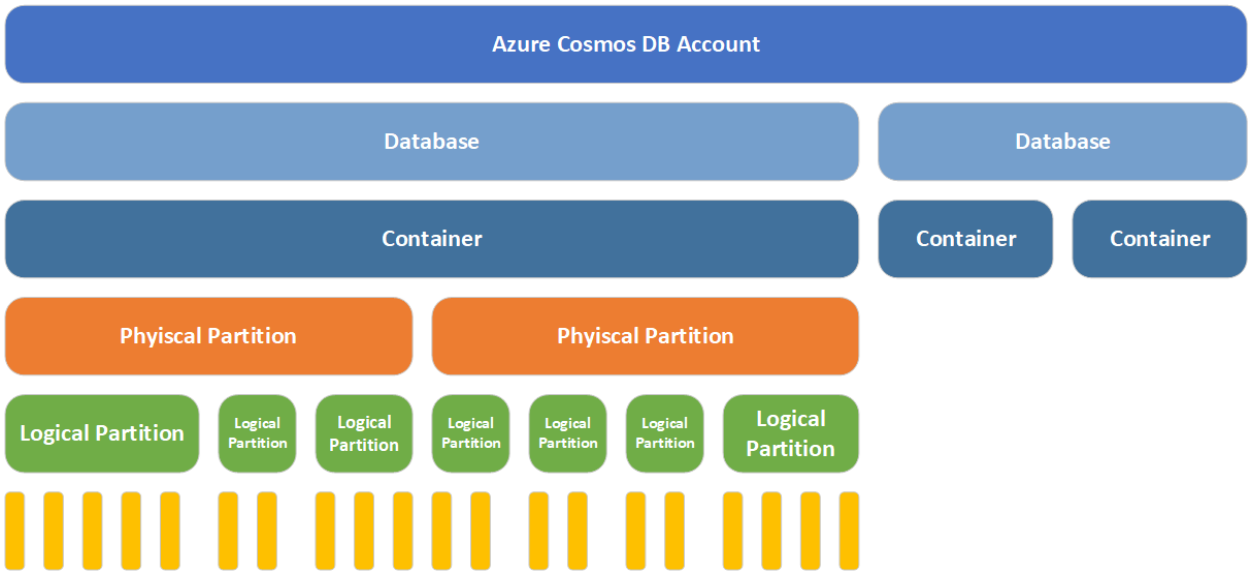
  ✘ That's incorrect. In this mode, you can automatically and instantly scale the throughput (RU/s).

2. Which of the following consistency levels below offers the greatest throughput?

- ○ Strong
- ○ Session
- ◉ Eventual

  ✔ That's correct. The eventual consistency level offers the greatest throughput at the cost of weaker consistency.

Azure Cosmos DB uses partitioning to scale individual containers in a database to meet the performance needs of your application.



- The top layer in the diagram, represents the Cosmos DB account. This is analagous to the ***server*** in SQL Azure
- Next is the database. Again, analgous to the ***database*** in SQL Azure
- Below the database, you can have many containers. Depending on the ***model*** you've selected your container will either be `Collection` or a `Graph` or a `Table`
- A container is served by many physical partitions. You can think of a physical partition as the assigned metal that serves your container. Each physical partition has a fixed amount of SSD backed storage and compute resources; and these things have physical limitations
- Each physical partition then hosts many logical partitions of your data
- And each logical partition, in turn holds many items (documents, nodes, rows)

Imagine a very simple document like so:

```
{
    "id":1
    "name":"Eoin"
    "city":"Dublin"
}
```

If you decided to specify `/city` as the partition key, then all the `Dublin` documents would be stored together, all the `London` documents together, and so on.

While logical partitions are somewhat nebulous groupings of like documents, physical partitions are very real and have 2 hard limits on them.

1. A physical partition can store a maximum of 10GB of data
2. A physical partition can facilitate at most 10,000 Request Units (RU)/s of throughput.

Physical Partition	Logical Partition	Current Size	Current Throughput	OK
P1	/city=Dublin	3GB	2,000 RU/s	✔
P1	/city=London	6GB	5,000 RU/s	✔

partition key

A partition key has two components: **partition key path** and the **partition key value**. For example, consider an item

```
{
  "userId": "Andrew",
  "worksFor": "Microsoft"
}
```

if you choose "`userId`" as the partition key, the following are the two partition key components:

- The partition key path (for example: "`userId`"). The partition key path accepts alphanumeric and underscore(_) characters. You can also use nested objects by using the standard path notation(/).
- The partition key value (for example: "`Andrew`"). The partition key value can be of string or numeric types.

1. Which of the options below best describes the relationship between logical and physical partitions?

- ○ Logical partitions are collections of physical partitions.
- ◉ Physical partitions are collections of logical partitions

  ✔ That's correct. One or more logical partitions are mapped to a single physical partition..
- ○ There is no relationship between physical and logical partitions.

2. Which of the below correctly lists the two components of a partition key?

- ○ Key path, synthetic key
- ◉ Key path, key value

  ✔ That's correct. A partition key has two components: partition key path and the partition key value.
- ○ Key value, item ID

Microsoft .NET SDK v3 for Azure Cosmos DB

Because Azure Cosmos DB supports multiple API models, version 3 of the .NET SDK uses the generic terms "container" and "item". A ***container*** can be a collection, graph, or table. An ***item*** can be a document, edge/vertex, or row, and is the content inside a container.

```
CosmosClient
Creates a new CosmosClient with a connection string
CosmosClient client = new CosmosClient(endpoint, key);

Create a database
DatabaseResponse databaseResponse = await
client.CreateDatabaseIfNotExistsAsync(databaseId, 10000);

Read a database by ID
DatabaseResponse readResponse = await database.ReadAsync();

Delete a database
await database.DeleteAsync();
```

Container Operations

```
Create a container
ContainerResponse simpleContainer = await database.CreateContainerIfNotExistsAsync(
    id: containerId,
    partitionKeyPath: partitionKey,
    throughput: 400);

Get a container by ID
Container container = database.GetContainer(containerId);
ContainerProperties containerProperties = await container.ReadContainerAsync();

Delete a container
await database.GetContainer(containerId).DeleteContainerAsync();
```

Item examples

```
Create an item
ItemResponse<SalesOrder> response = await container.CreateItemAsync(salesOrder, new
PartitionKey(salesOrder.AccountNumber));

Read an item
string id = "[id]";
string accountNumber = "[partition-key]";
ItemResponse<SalesOrder> response = await container.ReadItemAsync(id, new
PartitionKey(accountNumber));

Query an item
```

```
QueryDefinition query = new QueryDefinition(
    "select * from sales s where s.AccountNumber = @AccountInput ")
    .WithParameter("@AccountInput", "Account1");

FeedIterator<SalesOrder> resultSet = container.GetItemQueryIterator<SalesOrder>(
    query,
    requestOptions: new QueryRequestOptions()
    {
        PartitionKey = new PartitionKey("Account1"),
        MaxItemCount = 1
    });
```

1. When defining a stored procedure in the Azure portal input parameters are always sent as what type to the stored procedure?

- (●) String
  - ✔ That's correct. When defining a stored procedure in Azure portal, input parameters are always sent as a string to the stored procedure.
- ( ) Integer
- ( ) Boolean

2. Which of the following would one use to validate properties of an item being created?

- (●) Pre-trigger
  - ✔ That's correct. Pre-triggers can be used to conform data before it is added to the container.
- ( ) Post-trigger
- ( ) User-defined function

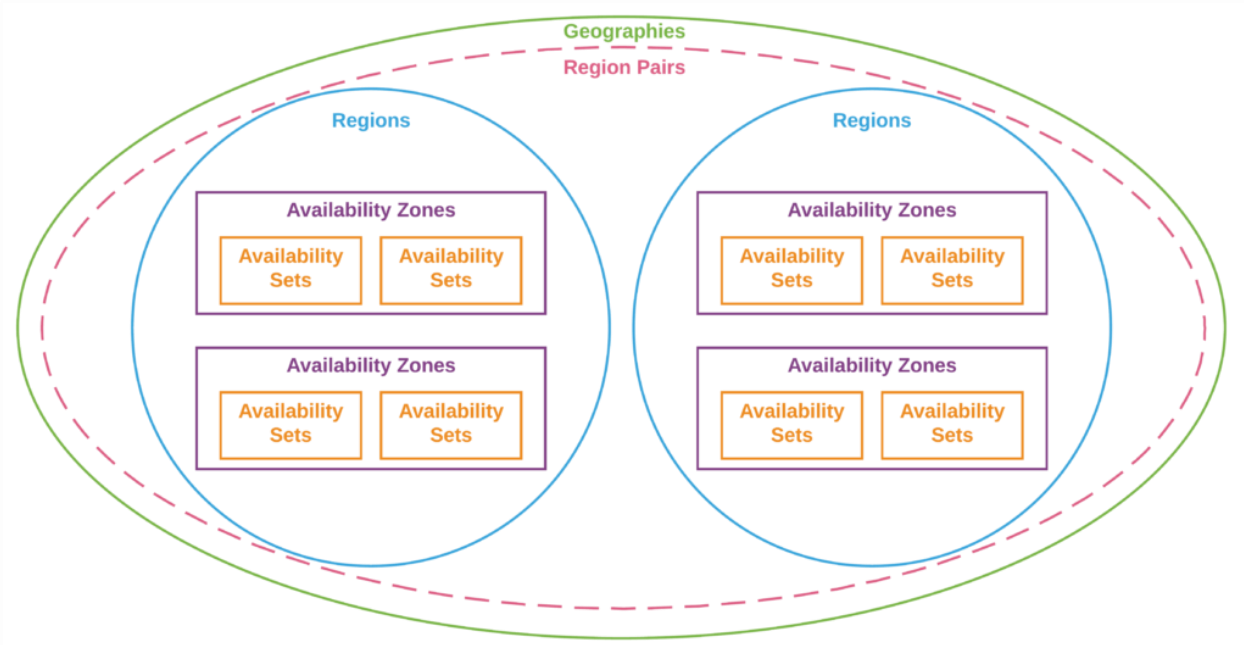
5. Implement infrastructure as a service solution

- Create Virtual machines
- Azure Resource Manager templates
- Manage containers.

Provision virtual machines in Azure

- Describe the design considerations for creating a virtual machine to support your apps needs
- Explain the different availability options for Azure VMs
- Describe the VM sizing options

- Create an Azure VM by using the Azure CLI



Availability sets

An availability set is a logical grouping of VMs that allows Azure to understand how your application is built to provide for redundancy and availability. An availability set is composed of two additional groupings that protect against hardware failures and allow updates to safely be applied - fault domains (FDs) and update domains (UDs).

Virtual machine 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.

Create a virtual machine by using the Azure CLI

Create a resource group with the `az group create` command. The command below creates a resource group named *az204-vm-rg*. Replace `<myLocation>` with a region near you.

```
az group create --name az204-vm-rg --location <myLocation>
```

Create a VM with the `az vm create` command. The command below creates a Linux VM named *az204vm* with an admin user named *azureuser*. After executing the command you will need to supply a password that meets the password requirements.

```
az vm create \
    --resource-group az204-vm-rg \
    --name az204vm \
    --image UbuntuLTS \
    --generate-ssh-keys \
    --admin-username azureuser
```

Delete

```
az group delete --name az204-vm-rg --no-wait
```

1. Which of the following Azure virtual machine types is most appropriate for testing and development?

- ○ Compute optimized
- ◉ General Purpose
  - ✔ That's correct. This type has a balanced CPU-to-memory ratio, and is ideal for testing and development.
- ○ Storage optimized

2. Which of the below represents a logical grouping of VMs that allows Azure to understand how your application is built to provide for redundancy and availability?

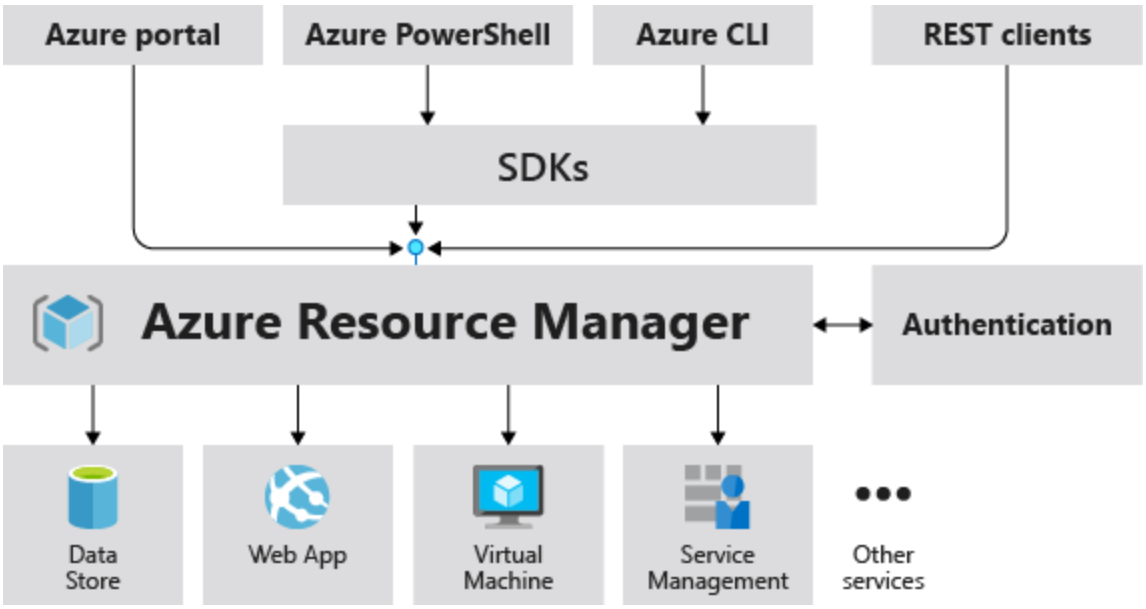
- ○ Load balancer
- ○ Availability zone
- ◉ Availability set
  - ✔ That's correct. An availability set is a logical grouping of VMs Reason.

Azure Resource Manager templates

Azure Resource Manager is the deployment and management service for Azure. It provides a management layer that enables you to create, update, and delete resources in your Azure subscription.

When a user sends a request from any of the Azure tools, APIs, or SDKs, Resource Manager receives the request. It authenticates and authorizes the request. Resource Manager sends the request to the Azure service, which takes the requested action. Because all requests are handled through the same API, you see consistent results and capabilities in all the different tools.

The following image shows the role Azure Resource Manager plays in handling Azure requests.



Why choose Azure Resource Manager templates

- **Declarative syntax**
- **Repeatable results**
- **Orchestration**

Template file

Within your template, you can write template expressions that extend the capabilities of JSON. The template has the following sections:

- Parameters - Provide values during deployment that allow the same template to be used with different environments.
- Variables - Define values that are reused in your templates. They can be constructed from parameter values.
- User-defined functions - Create customized functions that simplify your template.
- Resources - Specify the resources to deploy.
- Outputs - Return values from the deployed resources.

When you deploy a template, Resource Manager converts the template into REST API operations. For example, when Resource Manager receives a template with the following resource definition:

```
"resources": [
  {
    "type": "Microsoft.Storage/storageAccounts",
```

```
    "apiVersion": "2019-04-01",
    "name": "mystorageaccount",
    "location": "westus",
    "sku": {
      "name": "Standard_LRS"
    },
    "kind": "StorageV2",
    "properties": {}
  }
]
```

It converts the definition to the following REST API operation, which is sent to the `Microsoft.Storage` resource provider:

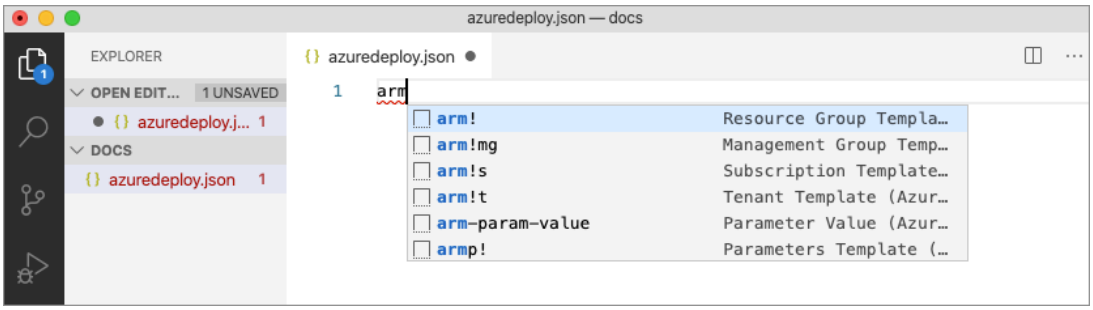
```
https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Storage/storageAccounts/mystorageaccount?api-version=2019-04-01
REQUEST BODY
{
  "location": "westus",
  "sku": {
    "name": "Standard_LRS"
  },
  "kind": "StorageV2",
  "properties": {}
}
```

You can deploy a template using any of the following options:

- Azure portal
- Azure CLI
- PowerShell
- REST API
- Button in GitHub repository
- Azure Cloud Shell

Create an Azure Resource Manager template

1. Create and open a new file named *azuredeploy.json* with Visual Studio Code.
2. Enter arm in the *azuredeploy.json* file and select arm! from the autocomplete options. This will insert a snippet with the basic building blocks for an Azure resource group deployment.



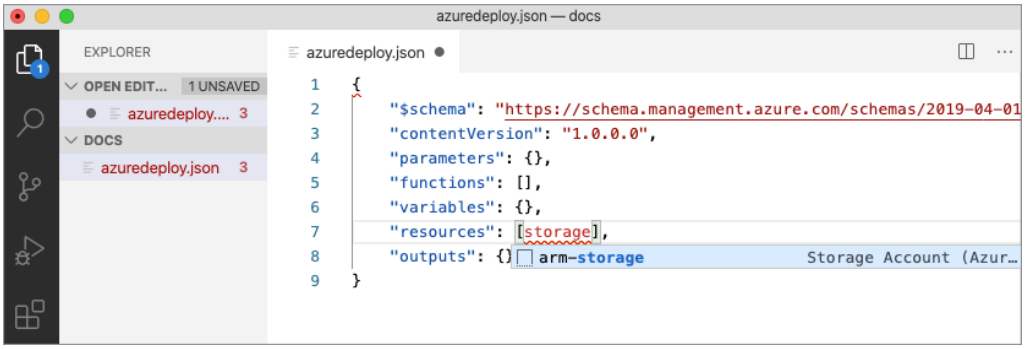
Your file should contain something similar to the example below.

```
{
    "$schema": "https://schema.management.azure.com/schemas/2019-04-
01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {},
    "functions": [],
    "variables": {},
    "resources": [],
    "outputs": {}
}
```

Add an Azure resource to the template

In this section you will add a snippet to support the creation of an Azure storage account to the template.

Place the cursor in the template `resources` block, type in `storage`, and select the arm-storage snippet.



The `resources` block should look similar to the example below.

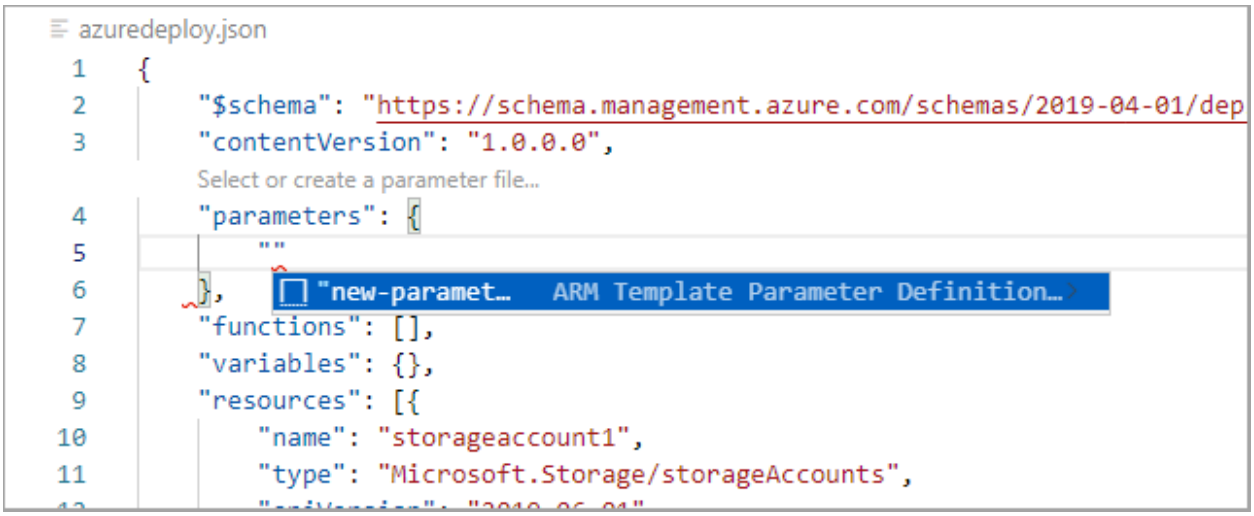
```
"resources": [{
    "name": "storageaccount1",
    "type": "Microsoft.Storage/storageAccounts",
    "apiVersion": "2019-06-01",
    "tags": {
        "displayName": "storageaccount1"
    },
    "location": "[resourceGroup().location]",
    "kind": "StorageV2",
```

```
    "sku": {
        "name": "Premium_LRS",
        "tier": "Premium"
    }
}],
```

Add parameters to the template

Now you will create and use a parameter to specify the storage account name.

Place your cursor in the parameters block, add a carriage return, type ", and then select the `new-parameter` snippet. This action adds a generic parameter to the template.



Make the following changes to the new parameter you just added:

1. Update the name of the parameter to `storageAccountName` and the description to `Storage Account Name`.
2. Azure storage account names have a minimum length of 3 characters and a maximum of 24. Add both `minLength` and `maxLength` to the parameter and provide appropriate values.

The `parameters` block should look similar to the example below.

JSONCopy

```
"parameters": {
    "storageAccountName": {
        "type": "string",
        "metadata": {
            "description": "Storage Account Name"
        },
        "minLength": 3,
        "maxLength": 24
    }
},
```

Follow the steps below to update the name property of the storage resource to use the parameter.

1. In the `resources` block, delete the current default name which is `storageaccount1` in the examples above. Leave the quotes (`""`) around the name in place.
2. Enter a square bracket `[`, which produces a list of Azure Resource Manager template functions. Select parameters from the list.
3. Add `()` at the end of parameters and select storageAccountName from the pop-up. If the list of parameters does not show up automatically you can enter a single quote `'` inside of the round brackets to display the list.

The resources block of the template should now be similar to the example below.

JSONCopy

```
"resources": [{
    "name": "[parameters('storageAccountName')]",
    "type": "Microsoft.Storage/storageAccounts",
    "apiVersion": "2019-06-01",
    "tags": {
        "displayName": "storageaccount1"
    },
    "location": "[resourceGroup().location]",
    "kind": "StorageV2",
    "sku": {
        "name": "Premium_LRS",
        "tier": "Premium"
    }
}],
```

Use the `az deployment group create` command to deploy your template.

```
az deployment group create
--resource-group az204-arm-rg
--template-file azuredeploy.json
--parameters azuredeploy.parameters.json
```

1. What purpose does the **outputs** section of an Azure Resource Manager template serve?

- ○ Specify the resources to deploy.
- ○ Return values from the deployed resources
  - ✔ That's correct. The "outputs" section returns values from the resource(s) that were deployed.
- ◉ Define values that are reused in your templates.
  - ✘ That's incorrect. The "variables" section of the file is used to define values.

2. Which Azure Resource Manager template deployment mode deletes resources in a resource group that aren't specified in the template?

- ○ Incremental
- ◉ Complete
  - ✔ That's correct. Complete mode will delete resources not specified in an Azure Resource Manager template deployment.
- ○ Both incremental and complete delete resources

Azure Container Registry

Azure Container Registry (ACR) is a managed, private Docker registry service based on the open-source Docker Registry 2.0. Create and maintain Azure container registries to store and manage your private Docker container images.

Tier	Description
Basic	A cost-optimized entry point for developers learning about Azure Container Registry. Basic registries have the same programmatic capabilities as Standard and Premium (such as Azure Active Directory authentication integration, image deletion, and webhooks). However, the included storage and image throughput are most appropriate for lower usage scenarios.
Standard	Standard registries offer the same capabilities as Basic, with increased included storage and image throughput. Standard registries should satisfy the needs of most production scenarios.
Premium	Premium registries provide the highest amount of included storage and concurrent operations, enabling high-volume scenarios. In addition to higher image throughput, Premium adds features such as geo-replication for managing a single registry across multiple regions, content trust for image tag signing, private link with private endpoints to restrict access to the registry.

For example, you can create a multi-step task that automates the following:

1. Build a web application image

2. Run the web application container
3. Build a web application test image
4. Run the web application test container, which performs tests against the running application container
5. If the tests pass, build a Helm chart archive package
6. Perform a `helm upgrade` using the new Helm chart archive package

Dockerfile

```
# Step 1: Specify the parent image for the new image
FROM ubuntu:18.04

# Step 2: Update OS packages and install additional software
RUN apt -y update &&  apt install -y wget nginx software-properties-common apt-transport-https \
      && wget -q https://packages.microsoft.com/config/ubuntu/18.04/packages-microsoft-prod.deb -O packages-microsoft-prod.deb \
      && dpkg -i packages-microsoft-prod.deb \
      && add-apt-repository universe \
      && apt -y update \
      && apt install -y dotnet-sdk-3.0

# Step 3: Configure Nginx environment
CMD service nginx start

# Step 4: Configure Nginx environment
COPY ./default /etc/nginx/sites-available/default

# STEP 5: Configure work directory
WORKDIR /app

# STEP 6: Copy website code to container
COPY ./website/. .

# STEP 7: Configure network requirements
EXPOSE 80:8080

# STEP 8: Define the entry point of the process that runs in the container
ENTRYPOINT ["dotnet", "website.dll"]
```

Create an Azure Container Registry

Create a resource group for the registry, replace `<myLocation>` in the command below with a location near you.

```
az group create --name az204-acr-rg --location <myLocation>
```

Create a basic container registry. The registry name must be unique within Azure, and contain 5-50 alphanumeric characters. Replace `<myContainerRegistry>` in the command below with a unique value.

```
az acr create --resource-group az204-acr-rg \
              --name <myContainerRegistry> --sku Basic
```

Build and push image from a Dockerfile

Now use Azure Container Registry to build and push an image based on a local Dockerfile.

```
echo FROM mcr.microsoft.com/hello-world > Dockerfile
```

Run the `az acr build` command, which builds the image and, after the image is successfully built, pushes it to your registry. Replace `<myContainerRegistry>` with the name you used earlier.

```
az acr build --image sample/hello-world:v1  \
             --registry <myContainerRegistry> \
             --file Dockerfile .
```

The command above will generate a lot of output, below is shortened sample of that output showing the last few lines with the final results. You can see in the `repository` field the `sample/hello-word` image is listed.

```
            - image:
                registry: <myContainerRegistry>.azurecr.io
                repository: sample/hello-world
                tag: v1
                digest:
            sha256:92c7f9c92844bbbb5d0a101b22f7c2a7949e40f8ea90c8b3bc396879d95e899a
              runtime-dependency:
                registry: mcr.microsoft.com
                repository: hello-world
                tag: latest
                digest:
            sha256:92c7f9c92844bbbb5d0a101b22f7c2a7949e40f8ea90c8b3bc396879d95e899a
              git: {}
```

Run ACR Image

```
az acr run --registry <myContainerRegistry> \
    --cmd '$Registry/sample/hello-world:v1' /dev/null
```

Clean up

```
az group delete --name az204-acr-rg --no-wait
```

1. Which of the following Azure Container Registry support geo-replication to manage a single registry across multiple regions?

- ○ Basic
- ○ Standard
- ◉ Premium

✔ That's correct. The premium tier adds geo-replication as a feature.

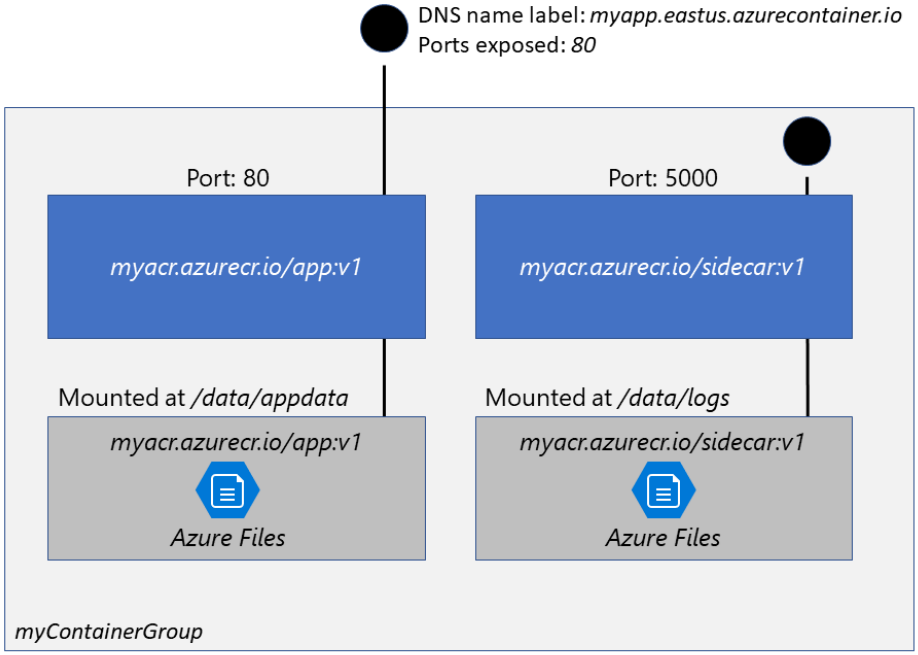
Azure Container Instances

Azure Container Instances (ACI) 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.

Container groups

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 in a container group share a lifecycle, resources, local network, and storage volumes. It's similar in concept to a *pod* in Kubernetes.



1. Which of the methods below is recommended when deploying a multi-container group that includes only containers?

- ○ Azure Resource Management template
- ◉ YAML file

  ✔ That's correct. Due to the YAML format's more concise nature, a YAML file is recommended when your deployment includes only container instances.

- ○ `az container create` command

6. Authentication and Authorization

Microsoft identity platform

There are several components that make up the Microsoft identity platform:

- **OAuth 2.0 and OpenID Connect standard-compliant authentication service** enabling developers to authenticate several identity types, including:
    - Work or school accounts, provisioned through Azure Active Directory
    - Personal Microsoft account, like Skype, Xbox, and Outlook.com
    - Social or local accounts, by using Azure Active Directory B2C
- **Open-source libraries**: Microsoft Authentication Libraries (MSAL) and support for other standards-compliant libraries
- **Application management portal**: A registration and configuration experience in the Azure portal, along with the other Azure management capabilities.
- **Application configuration API and PowerShell**: Programmatic configuration of your applications through the Microsoft Graph API and PowerShell so you can automate your DevOps tasks.

- **Single tenant**: only accessible in your tenant
- **Multi-tenant**: accessible in other tenants

1. Which of the types of permissions supported by the Microsoft identity platform is used by apps that have a signed-in user present?

- ○ Delegated permissions
  - ✔ That's correct. Delegated permissions are used by apps that have a signed-in user present. The app is delegated with the permission to act as a signed-in user when it makes calls to the target resource.
- ○ Application permissions
- ◉ Both delegated and application permissions
  - ✘ That's incorrect. Only delegated permissions are used by apps that have a signed-in user present.

2. Which of the following app scenarios require code to handle Conditional Access challenges?

- ○ Apps performing the device-code flow
- ◉ Apps performing the on-behalf-of flow
  - ✔ That's correct. Apps performing the on-behalf-of flow require code to handle Conditional Access challenges.
- ○ Apps performing the Integrated Windows authentication flow

Microsoft Authentication Library (MSAL)

Microsoft Authentication Library (MSAL) enables developers to acquire tokens from the Microsoft identity platform in order to authenticate users and access secured web APIs

Authentication flows

Below are some of the different authentication flows provided by Microsoft Authentication Library (MSAL). These flows can be used in a variety of different application scenarios.

Flow	**Description**
Authorization code	Native and web apps securely obtain tokens in the name of the user
Client credentials	Service applications run without user interaction
On-behalf-of	The application calls a service/web API, which in turns calls Microsoft Graph
Implicit	Used in browser-based applications
Device code	Enables sign-in to a device by using another device that has a browser
Integrated Windows	Windows computers silently acquire an access token when they are domain joined
Interactive	Mobile and desktops applications call Microsoft Graph in the name of a user
Username/password	The application signs in a user by using their username and password

1. Which of the following MSAL libraries supports single-page web apps?

- (●) MSAL Node
  - ✗ That's incorrect. MSAL Node does not support single-page web apps.
- ( ) MSAL.js
  - ✓ That's correct. MSAL.js supports single-page applications.
- ( ) MSAL.NET

Shared access signature (SAS)

A shared access signature (SAS) is a URI that grants restricted access rights to Azure Storage resources. You can provide a shared access signature to clients that you want grant delegate access to certain storage account resources.

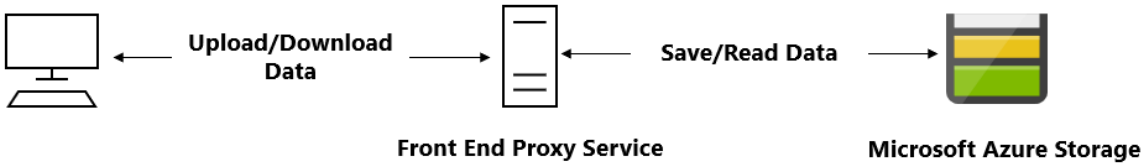
- **User delegation SAS**: A user delegation SAS is secured with Azure Active Directory credentials and also by the permissions specified for the SAS. A user delegation SAS applies to **Blob storage only.**
- **Service SAS**: A service SAS is secured with the storage account key. A service SAS delegates access to a resource in the following Azure Storage services: `Blob storage, Queue storage, Table storage, or Azure Files`.
- **Account SAS**: An account SAS is secured with the storage account key. An account SAS delegates access to resources in one or more of the storage services. `All of the operations available via a service or user delegation SAS are also available via an account SAS`.

URI	SAS token
`https://medicalrecords.blob.core.windows.net/patient-images/patient-116139-nq8z7f.jpg?`	`sp=r&st=2020-01-20T11:42:32Z&se=2020-01-20T19:42:32Z&spr=https&sv=2019-02-02&sr=b&sig=SrW1HZ5Nb6MbRzTbXCaPm%2BJiSEn15tC91Y4umMPwVZs%3D`

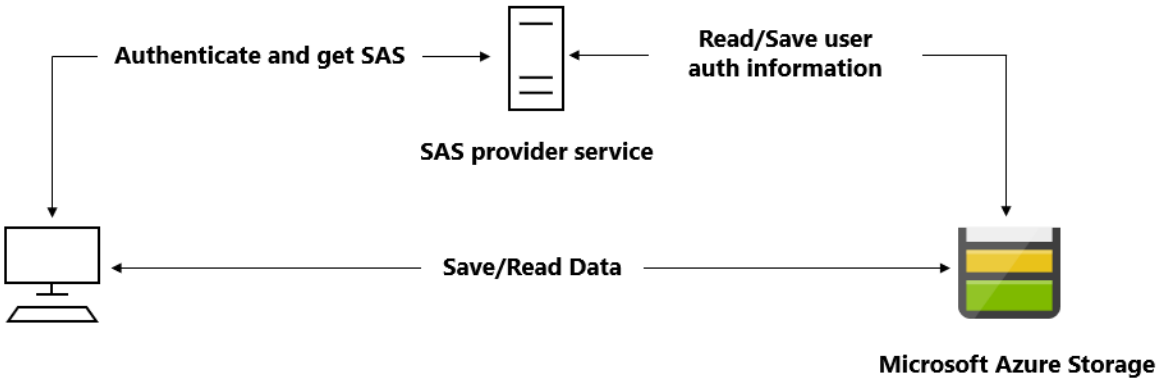
Use a SAS when you want to provide secure access to resources in your storage account to any client who does not otherwise have permissions to those resources.

A common scenario where a SAS is useful is a service where users read and write their own data to your storage account. In a scenario where a storage account stores user data, there are two typical design patterns:

- Clients upload and download data via a front-end proxy service, which performs authentication. This front-end proxy service has the advantage of allowing validation of business rules, but for large amounts of data or high-volume transactions, creating a service that can scale to match demand may be expensive or difficult.



- A lightweight service authenticates the client as needed and then generates a SAS. Once the client application receives the SAS, they can access storage account resources directly with the permissions defined by the SAS and for the interval allowed by the SAS. The SAS mitigates the need for routing all data through the front-end proxy service.



The following storage resources support stored access policies:

- Blob containers
- File shares
- Queues
- Tables

1. Which of the following types of shared access signatures (SAS) applies to Blob storage only?

- ○ Account SAS
- ○ Service SAS
- ◉ User delegation SAS

✔ That's correct. A user delegation SAS is secured with Azure Active Directory credentials and also by the permissions specified for the SAS. A user delegation SAS applies to Blob storage only.

2. Which of the following best practices provides the most flexible and secure way to use a service or account shared access signature (SAS)?

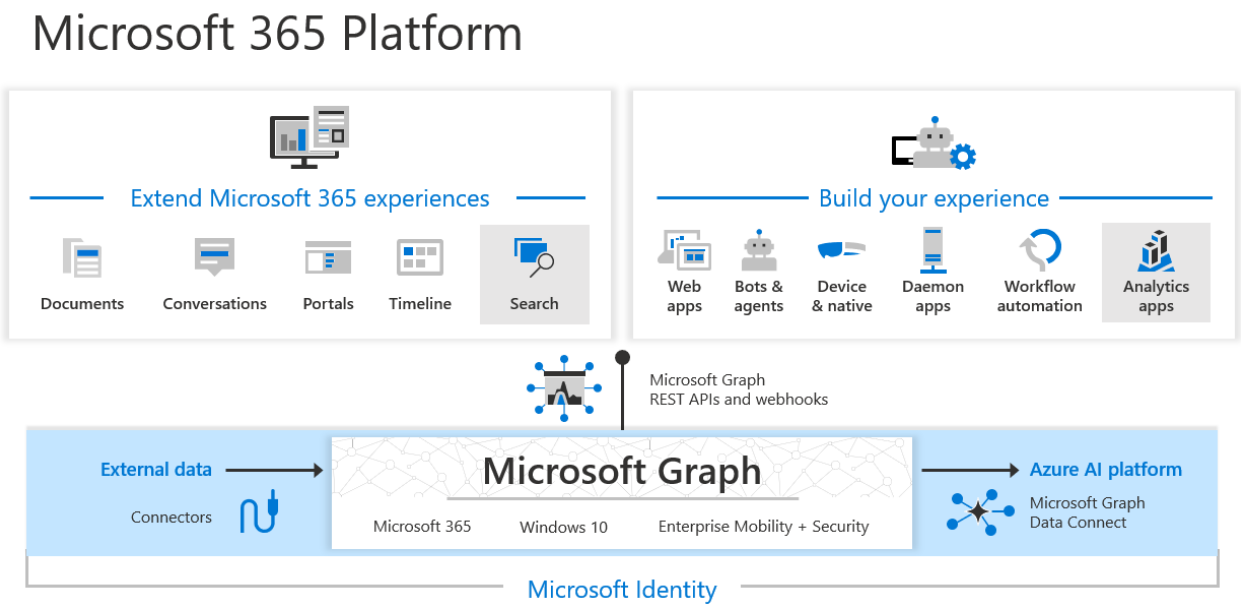
- ◉ Associate SAS tokens with a stored access policy.

✔ That's correct. The most flexible and secure way to use a service or account SAS is to associate the SAS tokens with a stored access policy.

- ○ Always use HTTPS

Microsoft Graph

Microsoft Graph is the gateway to data and intelligence in Microsoft 365. It provides a unified programmability model that you can use to access the tremendous amount of data in Microsoft 365, Windows 10, and Enterprise Mobility + Security.



In the Microsoft 365 platform, three main components facilitate the access and flow of data:

- The **Microsoft Graph API** offers a single endpoint, `https://graph.microsoft.com`. You can use REST APIs or SDKs to access the endpoint. Microsoft Graph also includes a powerful set of

services that manage user and device identity, access, compliance, security, and help protect organizations from data leakage or loss.

- **Microsoft Graph connectors** work in the incoming direction, **delivering data external to the Microsoft cloud into Microsoft Graph services and applications**, to enhance Microsoft 365 experiences such as Microsoft Search. Connectors exist for many commonly used data sources such as Box, Google Drive, Jira, and Salesforce.
- **Microsoft Graph Data Connect** provides a set of tools to streamline secure and scalable **delivery of Microsoft Graph data to popular Azure data stores**. The cached data serves as data sources for Azure development tools that you can use to build intelligent applications.

The Microsoft Graph .NET SDK is included in the following NuGet packages:

- `Microsoft.Graph` - Contains the models and request builders for accessing the v1.0 endpoint with the fluent API. Microsoft.Graph has a dependency on Microsoft.Graph.Core.
- `Microsoft.Graph.Beta` - Contains the models and request builders for accessing the beta endpoint with the fluent API. Microsoft.Graph.Beta has a dependency on Microsoft.Graph.Core.
- `Microsoft.Graph.Core` - The core library for making calls to Microsoft Graph.
- `Microsoft.Graph.Auth` - Provides an authentication scenario-based wrapper of the Microsoft Authentication Library (MSAL) for use with the Microsoft Graph SDK. Microsoft.Graph.Auth has a dependency on Microsoft.Graph.Core.

Microsoft Graph is a RESTful web API that enables you to access Microsoft Cloud service resources. After you register your app and get authentication tokens for a user or service, you can make requests to the Microsoft Graph API.

Call a REST API method

To read from or write to a resource such as a user or an email message, you construct a request that looks like the following:

```
{HTTP method} https://graph.microsoft.com/{version}/{resource}?{query-parameters}
```

```
GET https://graph.microsoft.com/v1.0/me/messages?filter=emailAddress eq
'jon@contoso.com'
```

```
// Build a client application.
IPublicClientApplication publicClientApplication = PublicClientApplicationBuilder
            .Create("INSERT-CLIENT-APP-ID")
            .Build();
```

```
// Create an authentication provider by passing in a client application and graph
scopes.
DeviceCodeProvider authProvider = new DeviceCodeProvider(publicClientApplication,
graphScopes);
```

```
// Create a new instance of GraphServiceClient with the authentication provider.
GraphServiceClient graphClient = new GraphServiceClient(authProvider);
```

```
// GET https://graph.microsoft.com/v1.0/me

var user = await graphClient.Me
    .Request()
    .GetAsync();
```

1. Which HTTP method below is used to update a resource with new values?

- ○ POST
- ◉ PATCH

  ✔ That's correct. The PATCH method does update a resource with a new value.
- ○ PUT

2. Which of the components of the Microsoft 365 platform is used to deliver data external to Azure into Microsoft Graph services and applications?

- ○ Microsoft Graph API
- ◉ Microsoft Graph connectors

  ✔ That's correct. Microsoft Graph connectors work in the incoming direction. Connectors exist for many commonly used data sources such as Box, Google Drive, Jira, and Salesforce.
- ○ Microsoft Graph Data Connect

3. Which of the following Microsoft Graph .NET SDK packages provides an authentication scenario-based wrapper of the Microsoft Authentication Library?

- ○ Microsoft.Graph
- ○ Microsoft.Graph.Core
- ◉ Microsoft.Graph.Auth

✔ That's correct. The Microsoft.Graph.Auth package provides an authentication scenario-based wrapper of the Microsoft Authentication Library for use with the Microsoft Graph SDK.

7. Implement secure cloud solutions

Azure Key Vault

Azure Key Vault is a cloud service for securely storing and accessing secrets. A secret is anything that you want to tightly control access to, such as API keys, passwords, certificates, or cryptographic keys.

Key Vault SDK is using Azure Identity client library, which allows seamless authentication to Key Vault across environments with same code. The table below provides information on the Azure Identity client libraries:

.NET	Python	Java	JavaScript
Azure Identity SDK .NET	Azure Identity SDK Python	Azure Identity SDK Java	Azure Identity SDK JavaScript

Authentication to Key Vault with REST

```
PUT /keys/MYKEY?api-version=<api_version>  HTTP/1.1
Authorization: Bearer <access_token>
```

Create a Key Vault by using the `az keyvault create` command.

```
az keyvault create
--name $myKeyVault
--resource-group az204-vault-rg
--location $myLocation
```

Add Sercrit to Azure Key Valut

```
az keyvault secret set
--vault-name $myKeyVault
--name "ExamplePassword"
--value "hVFkk965BuUv"
```

```
Delete Key Valut
az group delete --name az204-vault-rg --no-wait
```

1. Which of the below methods of authenticating to Azure Key Vault is recommended for most scenarios?

- ○ Service principal and certificate
- ○ Service principal and secret
- ◉ Managed identities

✔ That's correct. The benefit of this approach is that Azure automatically rotates the identity.

2. Azure Key Vault protects data when it is traveling between Azure Key Vault and clients. What protocol does it use for encryption?

- ○ Secure Sockets Layer
- ◉ Transport Layer Security

✔ That's correct. Azure Key Vault enforces Transport Layer Security protocol to protect data when it's traveling between Azure Key Vault and clients.

- ○ Presentation Layer

Managed Identities

A common challenge for developers is the management of secrets and credentials used to secure communication between different components making up a solution. Managed identities eliminate the need for developers to manage credentials.

Managed identities provide an identity for applications to use when connecting to resources that support Azure Active Directory (Azure AD) authentication. Applications may use the managed identity to obtain Azure AD tokens. For example, an application may use a managed identity to access resources like Azure Key Vault where developers can store credentials in a secure manner or to access storage accounts.

There are two types of managed identities:

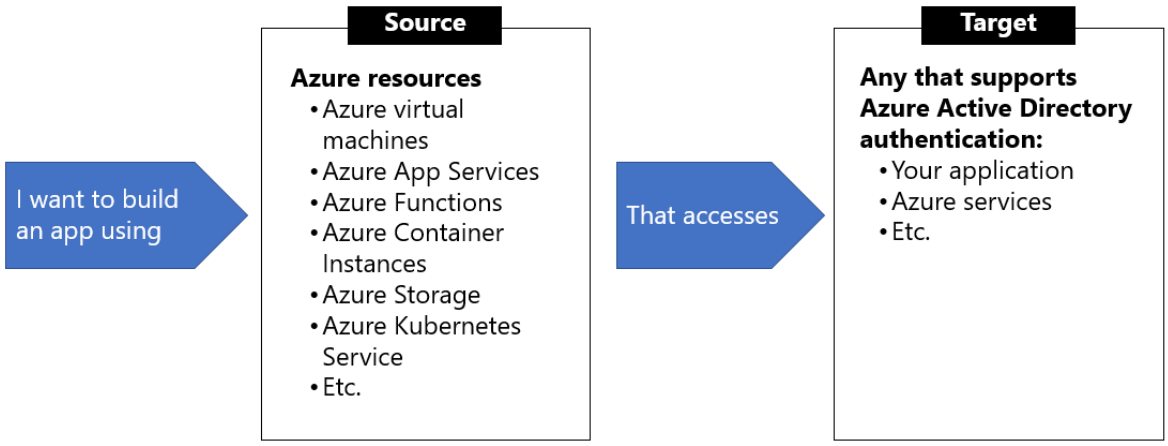
- A **system-assigned managed identity** is enabled directly on an Azure service instance. When the identity is enabled, Azure creates an identity for the instance in the Azure AD tenant that's trusted by the subscription of the instance. After the identity is created, the credentials are provisioned onto the instance. The lifecycle of a system-assigned identity is directly tied to the Azure service instance that it's enabled on. If the instance is deleted, Azure automatically cleans up the credentials and the identity in Azure AD.

- A **user-assigned managed identity** is created as a standalone Azure resource. Through a create process, Azure creates an identity in the Azure AD tenant that's trusted by the subscription in use. After the identity is created, the identity can be assigned to one or more Azure service instances. The lifecycle of a user-assigned identity is managed separately from the lifecycle of the Azure service instances to which it's assigned.

Characteristic	System-assigned managed identity	User-assigned managed identity
Creation	Created as part of an Azure resource (for example, an Azure virtual machine or Azure App Service)	Created as a stand-alone Azure resource
Lifecycle	Shared lifecycle with the Azure resource. When the parent resource is deleted, the managed identity is deleted as well.	Independent life-cycle. Must be explicitly deleted.
Sharing across Azure resources	Cannot be shared, it can only be associated with a single Azure resource.	Can be shared, the same user-assigned managed identity can be associated with more than one Azure resource.

When to use managed identities

The image below gives an overview the scenarios that support using managed identities. For example, you can use managed identities if you want to build an app using Azure App Services that accesses Azure Storage without having to manage any credentials.



managed identities authentication flow

How a system-assigned managed identity works with an Azure virtual machine

1. **Azure Resource Manager** `receives` a request to enable the system-assigned managed identity on a virtual machine.

2. **Azure Resource Manager** `creates` a service principal in Azure Active Directory for the identity of the virtual machine. The service principal is created in the Azure Active Directory tenant that's trusted by the subscription.

3. **Azure Resource Manager** `configures` the identity on the virtual machine by updating the Azure Instance Metadata Service identity endpoint with the service principal client ID and certificate.

4. After the virtual machine has an identity, use the service principal information to grant the virtual machine access to Azure resources

5. Your code that's running on the virtual machine can request a token from the Azure Instance Metadata service endpoint, accessible only from within the virtual machine: http://169.254.169.254/metadata/identity/oauth2/token

6. A call is made to Azure Active Directory to request an access token (as specified in step 5) by using the client ID and certificate configured in step 3. Azure Active Directory returns a JSON Web Token (JWT) access token.

7. Your code sends the access token on a call to a service that supports Azure Active Directory authentication.

How a user-assigned managed identity works with an Azure virtual machine

1. **Azure Resource Manager** `receives` a request to create a user-assigned managed identity.

2. **Azure Resource Manager** `creates` a service principal in Azure Active Directory for the user-assigned managed identity. The service principal is created in the Azure Active Directory tenant that's trusted by the subscription.

3. **Azure Resource Manager** `receives a request` to configure the user-assigned managed identity on a virtual machine and updates the Azure Instance Metadata Service identity endpoint with the user-assigned managed identity service principal client ID and certificate.

4. After the user-assigned managed identity is created, use the service principal information to grant the identity access to Azure resources. To call Azure Resource Manager, use role-based access control in Azure Active Directory to assign the appropriate role to the service principal of the user-assigned identity. To call Key Vault, grant your code access to the specific secret or key in Key Vault.

5. Your code that's running on the virtual machine can request a token from the Azure Instance Metadata Service identity endpoint, accessible only from within the virtual machine: `http://169.254.169.254/metadata/identity/oauth2/token`

6. A call is made to Azure Active Directory to request an access token (as specified in step 5) by using the client ID and certificate configured in step 3. Azure Active Directory returns a JSON Web Token (JWT) access token.

7. Your code sends the access token on a call to a service that supports Azure Active Directory authentication.

```
az vm create --resource-group myResourceGroup \
    --name myVM --image win2016datacenter \
    --generate-ssh-keys \
    --assign-identity \
    --admin-username azureuser \
    --admin-password myPassword12
```

Acquire Token

```
GET 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https://management.azure.com/' HTTP/1.1 Metadata: true
```

1. Which of the following characteristics is indicative of user-assigned identities?

- ○ Shared lifecycle with an Azure resource
- ◉ Independent life-cycle

  ✔ That's correct. User-assigned identities exist independently from the resources they are associated with and must be explicitly deleted.
- ○ Can only be associated with a single Azure resource

2. A client app requests managed identities for an access token for a given resource. Which of the below is the basis for the token?

- ◉ Oauth 2.0

  ✘ That's incorrect. Oauth 2.0 is a protocol that can be used to acquire a token, but is not the basis for the token.
- ○ Service principal

  ✔ That's correct. The token is based on the managed identities for Azure resources service principal.
- ○ Virtual machine

Azure App Configuration

Azure App Configuration provides a service to centrally manage application settings and feature flags.

Modern programs, especially programs running in a cloud, generally have many components that are distributed in nature. Spreading configuration settings across these components can lead to hard-to-troubleshoot errors during an application deployment. Use App Configuration to store all the settings for your application and secure their access in one place.

- **Feature flag**: A feature flag is a variable with a binary state of *on* or *off*. The feature flag also has an associated code block. The state of the feature flag triggers whether the code block runs or not.
- **Feature manager**: A feature manager is an application package that handles the lifecycle of all the feature flags in an application. The feature manager typically provides additional functionality, such as caching feature flags and updating their states.
- **Filter**: A filter is a rule for evaluating the state of a feature flag. A user group, a device or browser type, a geographic location, and a time window are all examples of what a filter can represent.

1. Which type of encryption does Azure App Configuration use to encrypt data at rest?

- ○ 64-bit AES
- ○ 128-bit AES
- ◉ 256-bit AES

✔ That's correct. Azure App Configuration encrypts sensitive information at rest using a 256-bit AES encryption key provided by Microsoft.

2. Which of the below evaluates the state of a feature flag?

- ○ Feature flag
- ◉ Feature manager

✘ That's incorrect. A feature manager is an application package that handles the lifecycle of all the feature flags in an application.

- ○ Filter

✔ That's correct. A filter is a rule for evaluating the state of a feature flag. A user group, a device or browser type, a geographic location, and a time window are all examples of what a filter can represent.

8. API Management

API Management helps organizations publish APIs to external, partner, and internal developers to unlock the potential of their data and services.

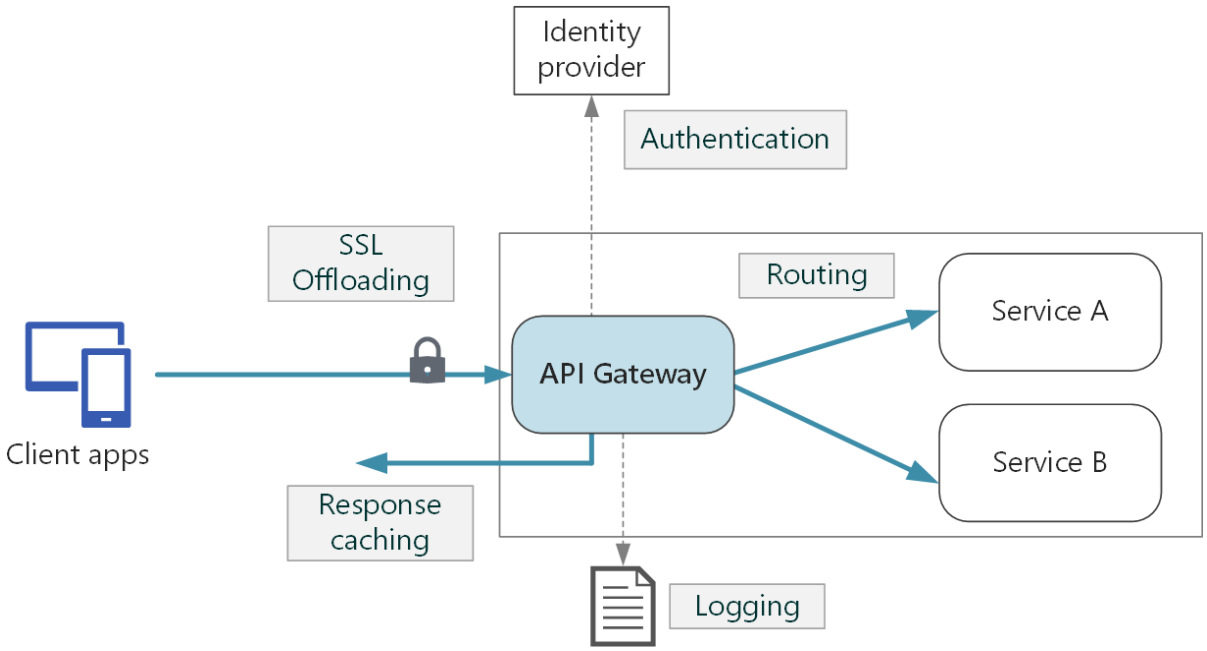
API Management provides the core functionality to ensure a successful API program through developer engagement, business insights, analytics, security, and protection. Each API consists of one or more operations, and each API can be added to one or more products. To use an API, developers subscribe to a product that contains that API, and then they can call the API's operation, subject to any usage policies that may be in effect.

The system is made up of the following components:

- The **API gateway** is the endpoint that:
    - Accepts API calls and routes them to your backend(s).
    - Verifies API keys, JWT tokens, certificates, and other credentials.
    - Enforces usage quotas and rate limits.
    - Transforms your API on the fly without code modifications.
    - Caches backend responses where set up.
    - Logs call metadata for analytics purposes.

- The **Azure portal** is the administrative interface where you set up your API program. Use it to:
    - Define or import API schema.
    - Package APIs into products.
    - Set up policies like quotas or transformations on the APIs.
    - Get insights from analytics.
    - Manage users.

- The **Developer portal** serves as the main web presence for developers, where they can:
    - Read API documentation.
    - Try out an API via the interactive console.
    - Create an account and subscribe to get API keys.
    - Access analytics on their own usage.



Explore API Management policies

The policy definition is a simple XML document that describes a sequence of **inbound** and **outbound** statements. The XML can be edited directly in the definition window.

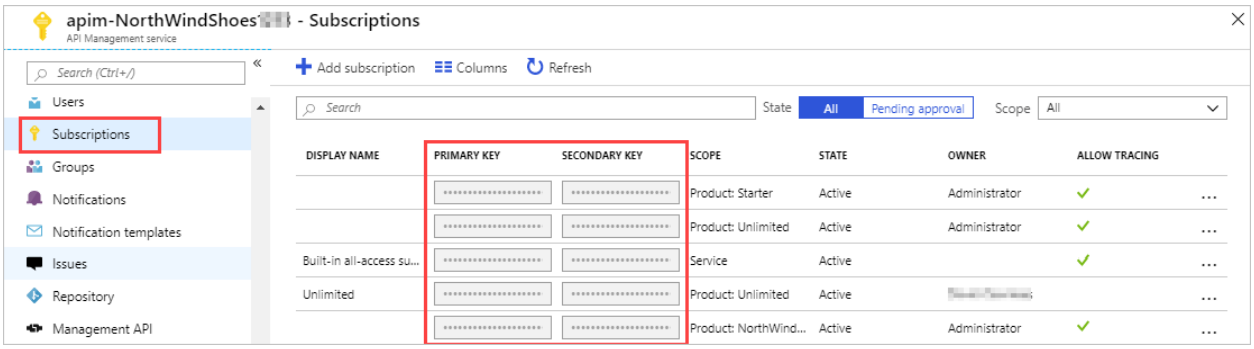
The configuration is divided into `inbound, backend, outbound, and on-error`. The series of specified policy statements is executing in order for a request and a response.

```
<policies>

 <inbound>
    <!-- statements to be applied to the request go here -->
  </inbound>

 <backend>
    <!-- statements to be applied before the request is forwarded to
         the backend service go here -->
 </backend>

 <outbound>
    <!-- statements to be applied to the response go here -->
  </outbound>

  <on-error>
    <!-- statements to be applied if there is an error condition go here -->
  </on-error>
</policies>
```

Subscriptions and Keys

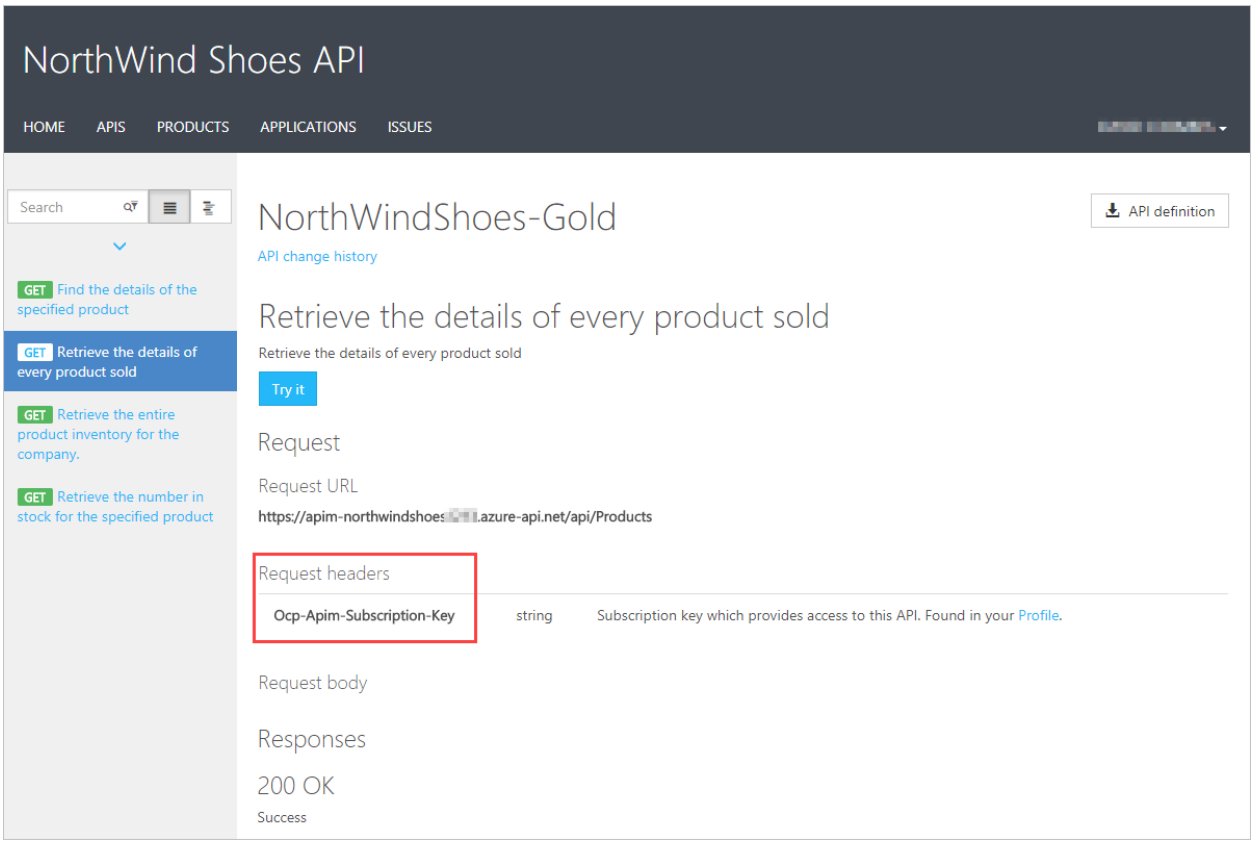
A subscription key is a unique auto-generated key that can be passed through in the headers of the client request or as a query string parameter. The three main subscription scopes are:

Scope	Details
All APIs	Applies to every API accessible from the gateway
Single API	This scope applies to a single imported API and all of its endpoints
Product	A product is a collection of one or more APIs that you configure in API Management. You can assign APIs to more than one product. Products can have different access rules, usage quotas, and terms of use.



Applications must include a valid key in all HTTP requests when they make calls to API endpoints that are protected by a subscription. Keys can be passed in the request header, or as a query string in the URL.

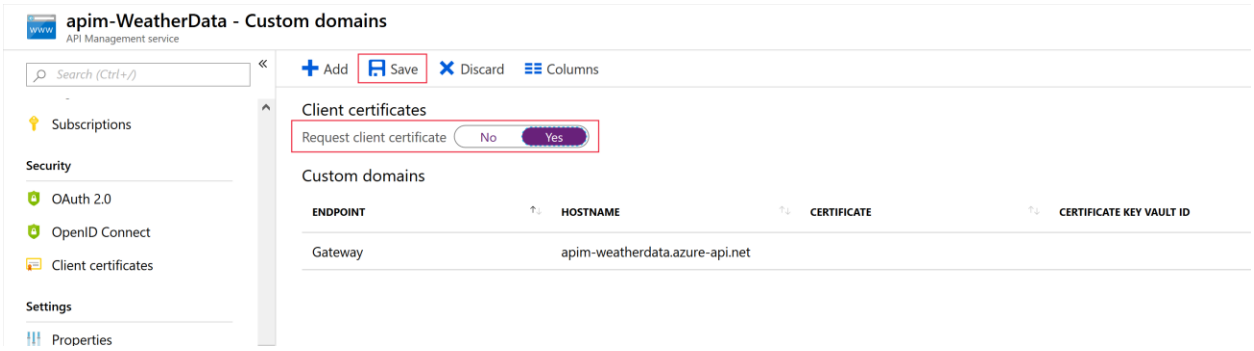
The default header name is **Ocp-Apim-Subscription-Key**, and the default query string is **subscription-key**.




```
curl --header "Ocp-Apim-Subscription-Key: <key string>" https://<apim gateway>.azure-api.net/api/path
```

Secure APIs by using certificates

Certificates can be used to provide Transport Layer Security (TLS) mutual authentication between the client and the API gateway



1. Which of the following components of the API Management service would a developer use if they need to create an account and subscribe to get API keys?

- ○ API gateway
- ○ Azure portal
- ◉ Developer portal

✔ **That's correct. The Developer portal serves as the main web presence for developers, and is where they can subscribe to get API keys.**

2. Which of the following API Management policies would one use if one wants to apply a policy based on a condition?

- ○ forward-request
- ◉ choose

✔ **That's correct. The choose policy applies enclosed policy statements based on the outcome of evaluation of boolean expressions.**

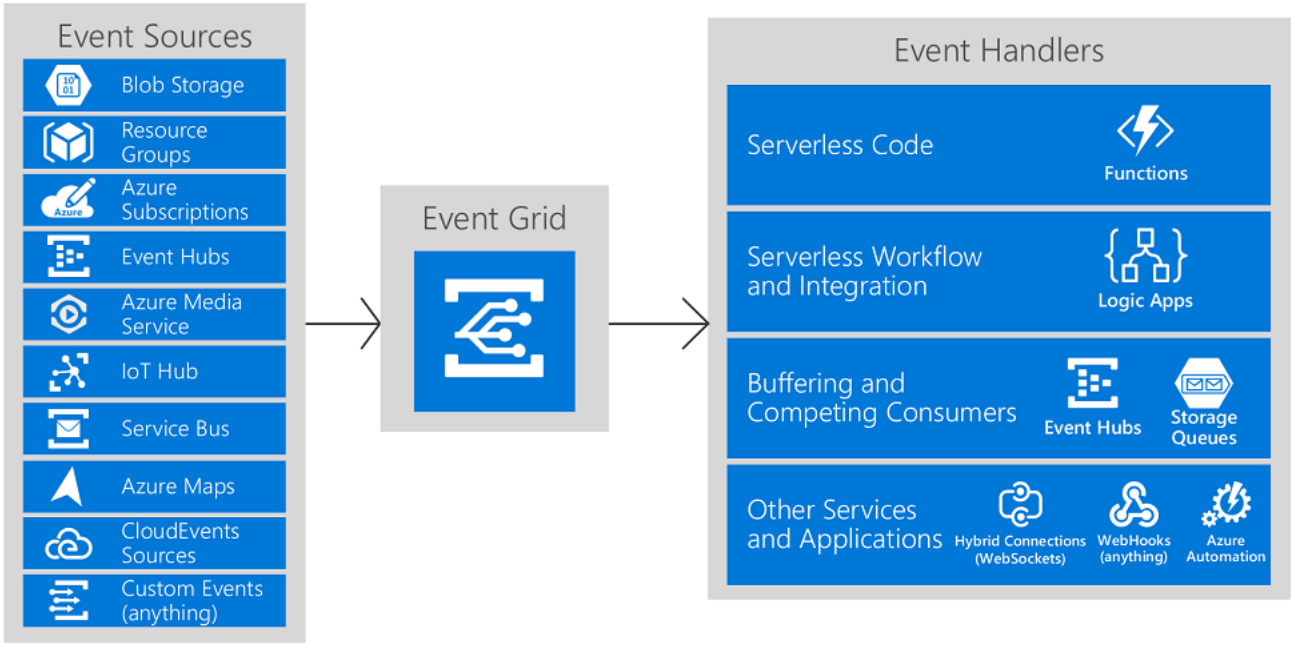
- ○ return-response

9. Event-based solutions

Azure Event Grid

Azure Event Grid is an eventing backplane that enables event-driven, reactive programming. It uses the publish-subscribe model. Publishers emit events, but have no expectation about how the events are handled. Subscribers decide on which events they want to handle

Event Grid allows you to easily build applications with event-based architectures. First, select the Azure resource you would like to subscribe to, and then give the event handler or WebHook endpoint to send the event to. Event Grid has built-in support for events coming from Azure services, like storage blobs and resource groups. Event Grid also has support for your own events, using custom topics.



- **Events** - What happened.
- **Event sources** - Where the event took place.
- **Topics** - The endpoint where publishers send events.
- **Event subscriptions** - The endpoint or built-in mechanism to route events, sometimes to more than one handler. Subscriptions are also used by handlers to intelligently filter incoming events.
- **Event handlers** - The app or service reacting to the event.

Event Grid sends the events to subscribers in an array that has a single event. You can find the JSON schema for the Event Grid event and each Azure publisher's data payload in the Event Schema store.

```
[
  {
    "topic": string,
    "subject": string,
    "id": string,
    "eventType": string,
    "eventTime": string,
    "data":{
      object-unique-to-each-publisher
    },
    "dataVersion": string,
    "metadataVersion": string
  }
]
```

CloudEvents v1.0 schema

In addition to its default event schema, Azure Event Grid natively supports events in the JSON implementation of CloudEvents v1.0 and HTTP protocol binding. CloudEvents is an open specification for describing event data.

Azure Blob Storage event in CloudEvents format

```
{
    "specversion": "1.0",
    "type": "Microsoft.Storage.BlobCreated",
    "source": "/subscriptions/{subscription-id}/resourceGroups/{resource-
group}/providers/Microsoft.Storage/storageAccounts/{storage-account}",
    "id": "9aeb0fdf-c01e-0131-0922-9eb54906e209",
    "time": "2019-11-18T15:13:39.4589254Z",
    "subject": "blobServices/default/containers/{storage-container}/blobs/{new-
file}",
    "dataschema": "#",
    "data": {
        "api": "PutBlockList",
        "clientRequestId": "4c5dd7fb-2c48-4a27-bb30-5361b5de920a",
        "requestId": "9aeb0fdf-c01e-0131-0922-9eb549000000",
        "eTag": "0x8D76C39E4407333",
        "contentType": "image/png",
        "contentLength": 30699,
        "blobType": "BlockBlob",
        "url": "https://gridtesting.blob.core.windows.net/testcontainer/{new-file}",
        "sequencer": "00000000000000000000000000009924000000000c41c18",
        "storageDiagnostics": {
            "batchId": "681fe319-3006-00a8-0022-9e7cde000000"
        }
    }
}
```

Event Grid provides durable delivery. It tries to deliver each event at least once for each matching subscription immediately. If a subscriber's endpoint doesn't acknowledge receipt of an event or if there is a failure, Event Grid retries delivery based on a fixed retry schedule and retry policy.

By default, Event Grid delivers one event at a time to the subscriber, and the payload is an array with a single event

Retry policy

- **Maximum number of attempts** - The value must be an integer between 1 and 30. The default value is 30.

- **Event time-to-live (TTL)** - The value must be an integer between 1 and 1440. The default value is 1440 minutes

The example below shows setting the maximum number of attempts to 18 by using the Azure CLI.

```
az eventgrid event-subscription create \
  -g gridResourceGroup \
  --topic-name <topic_name> \
  --name <event_subscription_name> \
  --endpoint <endpoint_URL> \
  --max-delivery-attempts 18
```

Built-in roles

Event Grid provides the following built-in roles:

Role	Description
Event Grid Subscription Reader	Lets you read Event Grid event subscriptions.
Event Grid Subscription Contributor	Lets you manage Event Grid event subscription operations.
Event Grid Contributor	Lets you create and manage Event Grid resources.
Event Grid Data Sender	Lets you send events to Event Grid topics.

You can set custom headers on the events that are delivered to the following destinations:

- Webhooks
- Azure Service Bus topics and queues
- Azure Event Hubs
- Relay Hybrid Connections

Filter events

When creating an event subscription, you have three options for filtering:

- Event types
- Subject begins with or ends with
- Advanced fields and operators

```
"filter": {
  "includedEventTypes": [
    "Microsoft.Resources.ResourceWriteFailure",
    "Microsoft.Resources.ResourceWriteSuccess"
  ]
}
```

1. Which of the following event schema properties requires a value?

- ○ Topic
- ◉ Data

  ✗ That's incorrect. A value is not required in this property.
- ○ Subject

  ✓ That's correct. The subject property specifies the publisher-defined path to the event subject and is required.

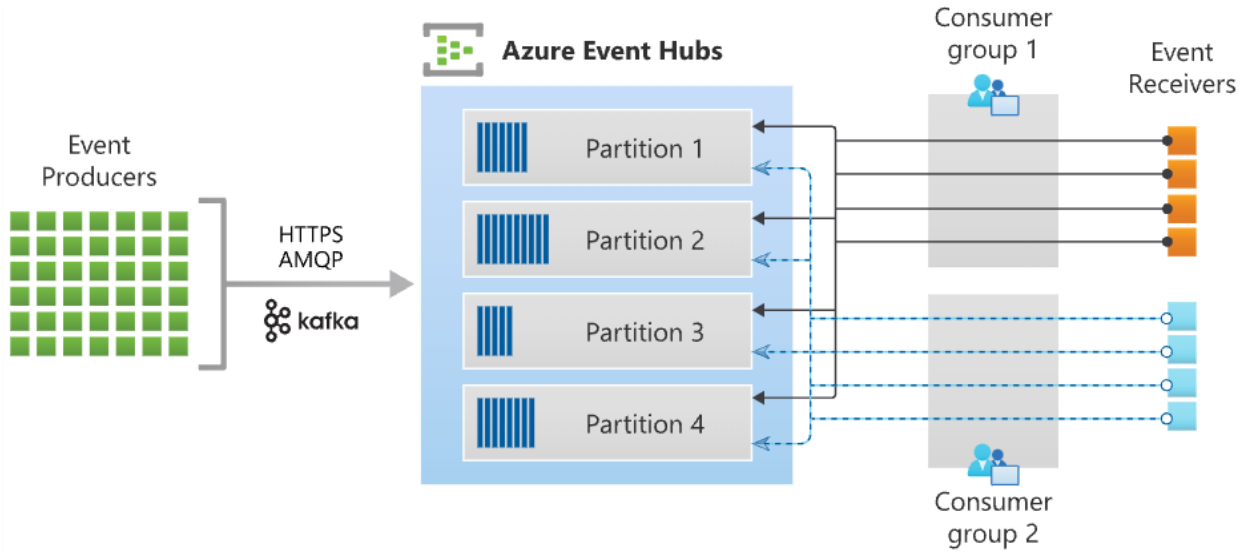
2. Which of the following Event Grid built-in roles is appropriate for managing Event Grid resources?

- ◉ Event Grid Contributor

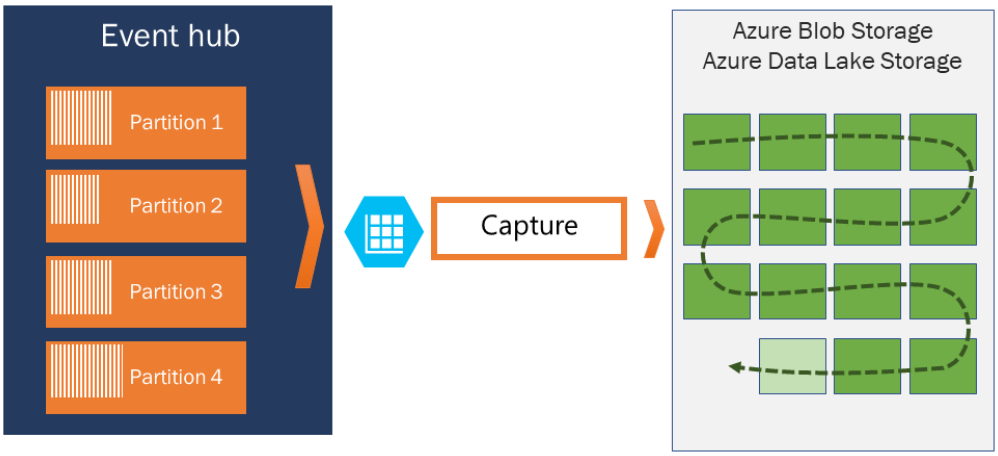
  ✓ That's correct. The Event Grid Contributor role has permissions to manage resources.
- ○ Event Grid Subscription Contributor
- ○ Event Grid Data Sender

Azure Event Hubs

Azure Event Hubs is a big data streaming platform and event ingestion service. It can receive and process millions of events per second. Data sent to an event hub can be transformed and stored by using any real-time analytics provider or batching/storage adapters.



Azure Event Hubs enables you to automatically capture the streaming data in Event Hubs in an Azure Blob storage or Azure Data Lake Storage account of your choice




```
{Namespace}/{EventHub}/{PartitionId}/{Year}/{Month}/{Day}/{Hour}/{Minute}/{Second}
```

```
https://mystorageaccount.blob.core.windows.net/mycontainer/mynamespace/myeventhub/0/2017/12/08/03/03/17.avro
```

- Azure Event Hubs Data Owner: Use this role to give *complete access* to Event Hubs resources.
- Azure Event Hubs Data Sender: Use this role to give *send access* to Event Hubs resources.
- Azure Event Hubs Data Receiver: Use this role to give *receiving access* to Event Hubs resources.

1. Which of the following Event Hubs concepts represents an ordered sequence of events that is held in an Event Hub?

- ○ Consumer group
- ◉ Partition

  ✔ That's correct. A partition is an ordered sequence of events that is held in an Event Hub.
- ○ Event Hub producer

2. Which of the following represents when an event processor marks or commits the position of the last successfully processed event within a partition?

- ◉ Checkpointing

  ✔ That's correct. Checkpointing is a process by which an event processor marks or commits the position of the last successfully processed event within a partition.
- ○ Scale
- ○ Load balance

10. Develop Message-based solutions

Azure supports two types of queue mechanisms:

- **Service Bus queues**
- **Storage queues**.

Service Bus queues are part of a broader Azure messaging infrastructure that supports queuing, publish/subscribe, and more advanced integration patterns. They're designed to integrate applications or application components that may span multiple communication protocols, data contracts, trust domains, or network environments.

Storage queues are part of the Azure Storage infrastructure. They allow you to store large numbers of messages. You access messages from anywhere in the world via authenticated calls using HTTP or HTTPS. A queue message can be up to 64 KB in size. A queue may contain millions of messages, up to the total capacity limit of a storage account. Queues are commonly used to create a backlog of work to process asynchronously.

Azure Service Bus

Microsoft Azure Service Bus is a fully managed enterprise integration message broker. Service Bus can decouple applications and services. Data is transferred between different applications and services using **messages**. A message is a container decorated with metadata, and contains data. The data can be any kind of information, including structured data encoded with the common formats such as the following ones: JSON, XML, Apache Avro, Plain Text.

Some common messaging scenarios are:

- ***Messaging***. Transfer business data, such as sales or purchase orders, journals, or inventory movements.
- ***Decouple applications.*** Improve reliability and scalability of applications and services. Client and service don't have to be online at the same time.
- ***Topics and subscriptions***. Enable 1:*n* relationships between publishers and subscribers.
- ***Message sessions***. Implement workflows that require message ordering or message deferral.

As a solution architect/developer, **you should consider using Service Bus queues** when:

- Your solution needs to receive messages without having to poll the queue. With Service Bus, you can achieve it by using a long-polling receive operation using the TCP-based protocols that Service Bus supports.
- Your solution requires the queue to provide a guaranteed first-in-first-out (FIFO) ordered delivery.
- Your solution needs to support automatic duplicate detection.
- You want your application to process messages as parallel long-running streams (messages are associated with a stream using the **session ID** property on the message). In this model, each node in the consuming application competes for streams, as opposed to messages. When a stream is given to a consuming node, the node can examine the state of the application stream state using transactions.
- Your solution requires transactional behavior and atomicity when sending or receiving multiple messages from a queue.
- Your application handles messages that can exceed 64 KB but won't likely approach the 256-KB limit.

Create a Service Bus messaging namespace. The command below will create a namespace using the variable you created earlier. The operation will take a few minutes to complete.

```
az servicebus namespace create \
    --resource-group az204-svcbus-rg \
    --name $myNameSpaceName \
    --location $myLocation
```

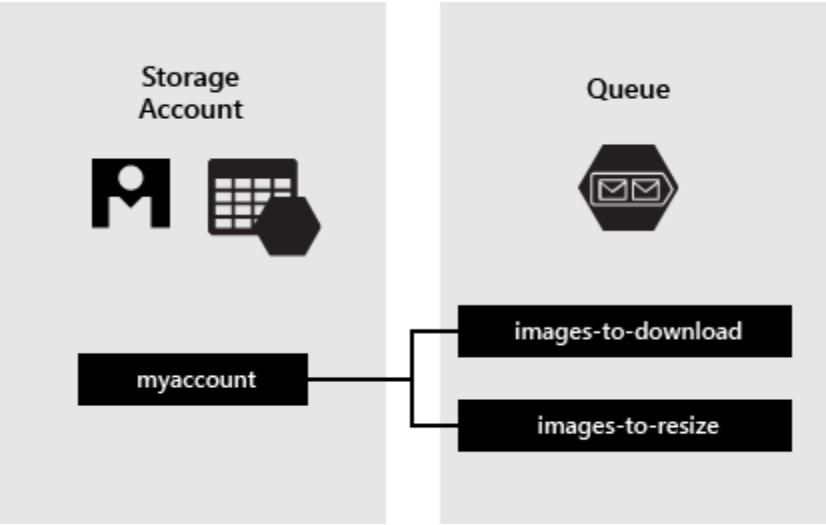
Create a Service Bus queue

```
az servicebus queue create --resource-group az204-svcbus-rg \
    --namespace-name $myNameSpaceName \
    --name az204-queue
```

Azure Storage Queue

Azure Queue Storage is a service for storing large numbers of messages. You access messages from anywhere in the world via authenticated calls using HTTP or HTTPS. A queue message can be up to 64 KB in size. A queue may contain millions of messages, up to the total capacity limit of a storage account. Queues are commonly used to create a backlog of work to process asynchronously.

The Queue service contains the following components:



- **URL format:** Queues are addressable using the URL format https://<storage account>.queue.core.windows.net/<queue>. For example, the following URL addresses a queue in the diagram above https://myaccount.queue.core.windows.net/images-to-download
- **Storage account:** All access to Azure Storage is done through a storage account.
- **Queue:** A queue contains a set of messages. All messages must be in a queue. Note that the queue name must be all lowercase.
- **Message:** A message, in any format, of up to 64 KB. For version 2017-07-29 or later, the maximum time-to-live can be any positive number, or -1 indicating that the message doesn't expire. If this parameter is omitted, the default time-to-live is seven days.

As a solution architect/developer, **you should consider using Storage queues** when:

- Your application must store over 80 gigabytes of messages in a queue.

- Your application wants to track progress for processing a message in the queue. It's useful if the worker processing a message crashes. Another worker can then use that information to continue from where the prior worker left off.
- You require server side logs of all of the transactions executed against your queues.

Create the Queue service client

The `QueueClient` class enables you to retrieve queues stored in Queue storage. Here's one way to create the service client:

```
QueueClient queueClient = new QueueClient(connectionString, queueName);
```

Create a queue

This example shows how to create a queue if it does not already exist:

```
// Get the connection string from app settings
string connectionString =
ConfigurationManager.AppSettings["StorageConnectionString"];

// Instantiate a QueueClient which will be used to create and manipulate the queue
QueueClient queueClient = new QueueClient(connectionString, queueName);

// Create the queue
queueClient.CreateIfNotExists();
```

Insert a message into a queue

To insert a message into an existing queue, call the `SendMessage` method. A message can be either a string (in UTF-8 format) or a byte array. The following code creates a queue (if it doesn't exist) and inserts a message:

```
// Get the connection string from app settings
string connectionString =
ConfigurationManager.AppSettings["StorageConnectionString"];

// Instantiate a QueueClient which will be used to create and manipulate the queue
QueueClient queueClient = new QueueClient(connectionString, queueName);

// Create the queue if it doesn't already exist
queueClient.CreateIfNotExists();

if (queueClient.Exists())
{
    // Send a message to the queue
    queueClient.SendMessage(message);
}
```

1. Which of the following advanced features of Azure Service Bus creates a first-in, first-out (FIFO) guarantee?

- ○ Transactions
- ○ Scheduled delivery
- ◉ Message sessions

✔ **That's correct. To create a first-in, first-out (FIFO) guarantee in Service Bus, use sessions. Message sessions enable joint and ordered handling of unbounded sequences of related messages.**

2. In Azure Service Bus messages are durably stored which enables a load-leveling benefit. Which of the below correctly describes the load-leveling benefit relative to a consuming application's performance?

- ◉ Performance needs to handle peak load

✘ **That's incorrect. Intermediating message producers and consumers with a queue means that the consuming application only has to be able to handle average load instead of peak load.**

- ○ Performance needs to handle average load

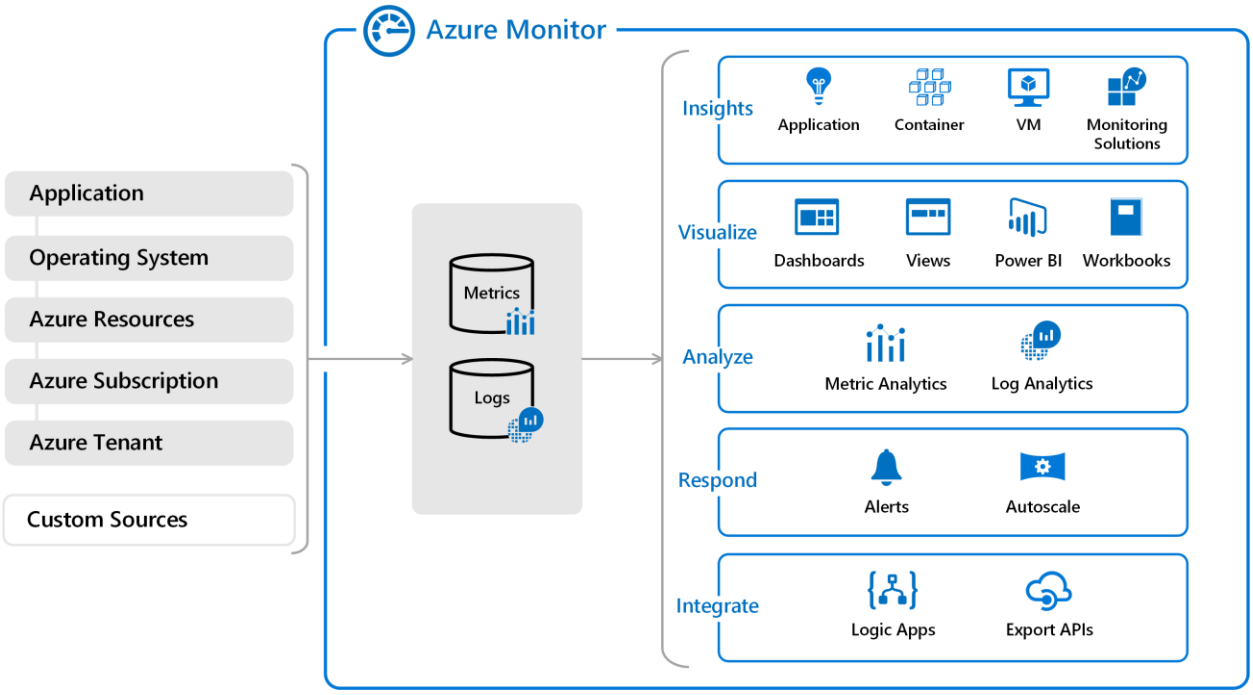
✔ **That's correct. Intermediating message producers and consumers with a queue means that the consuming application only has to be able to handle average load instead of peak load.**

- ○ Performance needs to handle low loads

11. Monitoring and Logging

Azure Monitor

Azure Monitor delivers a comprehensive solution for collecting, analyzing, and acting on telemetry from your cloud and on-premises environments. This information helps you understand how your applications are performing and proactively identify issues affecting them and the resources they depend on.



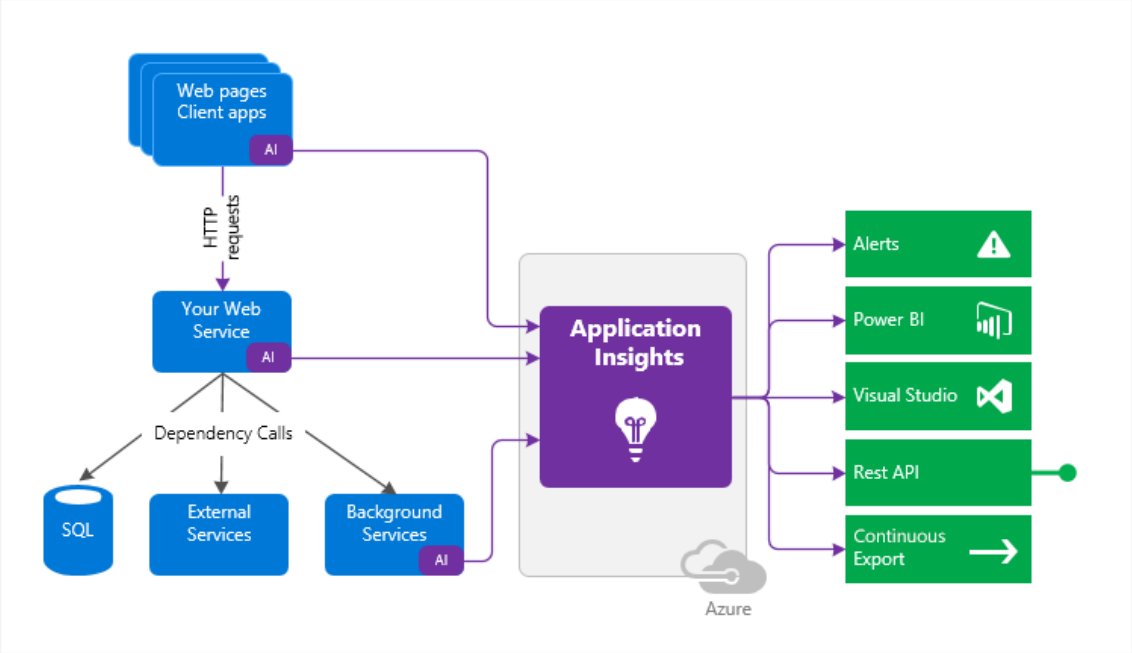
What data does Azure Monitor collect?

Azure Monitor can collect data from a variety of sources. This ranges from your application, any operating system and services it relies on, down to the platform itself. Azure Monitor collects data from each of the following tiers:

- **Application monitoring data**: Data about the performance and functionality of the code you have written, regardless of its platform.
- **Guest OS monitoring data**: Data about the operating system on which your application is running. This could be running in Azure, another cloud, or on-premises.
- **Azure resource monitoring data**: Data about the operation of an Azure resource. For a complete list of the resources that have metrics or logs, visit What can you monitor with Azure Monitor?
- **Azure subscription monitoring data**: Data about the operation and management of an Azure subscription, as well as data about the health and operation of Azure itself.
- **Azure tenant monitoring data**: Data about the operation of tenant-level Azure services, such as Azure Active Directory.

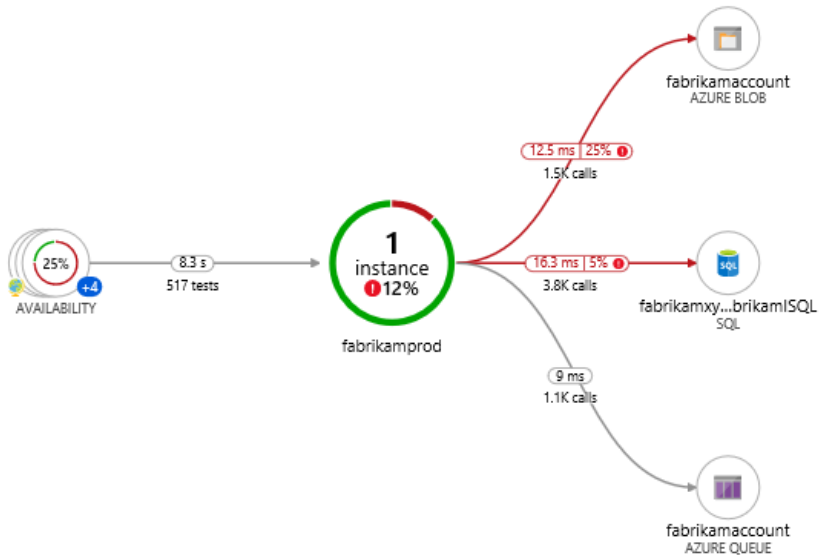
Application Insights

You install a small instrumentation package (SDK) in your application or enable Application Insights using the Application Insights Agent when supported. The instrumentation monitors your app and directs the telemetry data to an Azure Application Insights Resource using a unique GUID that we refer to as an Instrumentation Key.

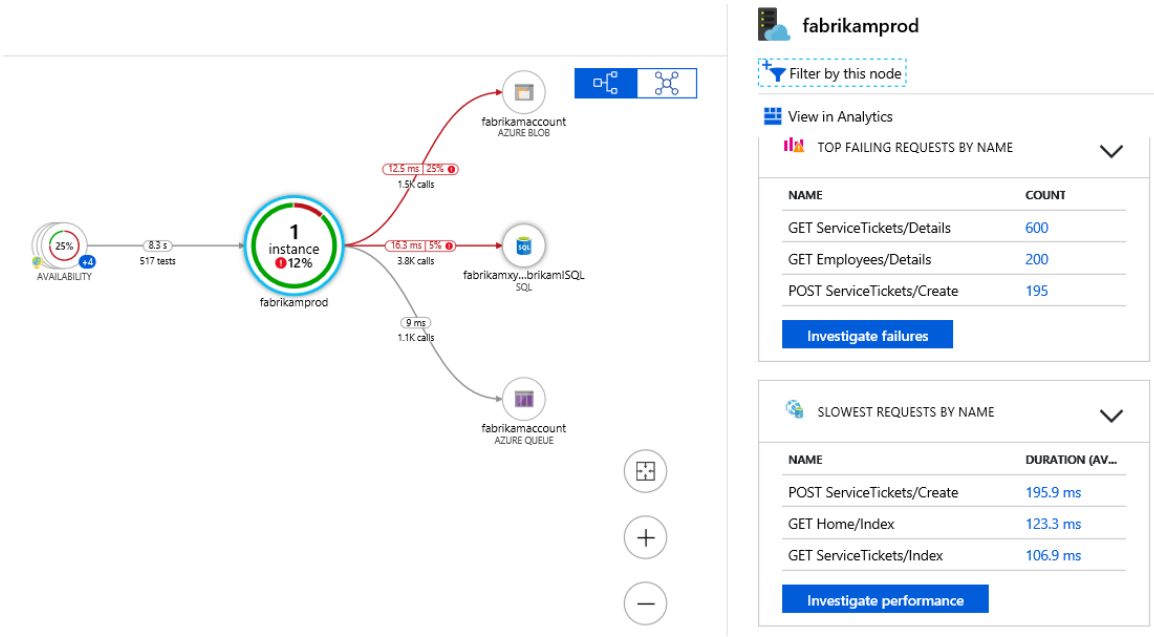


Application Map

Application Map helps you spot performance bottlenecks or failure hotspots across all components of your distributed application.



One of the key objectives with this experience is to be able to visualize complex topologies with hundreds of components. Click on any component to see related insights and go to the performance and failure triage experience for that component.



1. Which of the following availability tests is recommended for authentication tests?

- ○ URL ping
- ◉ Standard

  ✗ That's incorrect. This single request test is similar to the URL ping test, but it includes additional information.
- ○ Custom TrackAvailability

  ✓ That's correct. Custom TrackAvailability test is the long term supported solution for multi request or authentication test scenarios.

2. Which of the following metric collection types below provides near real-time querying and alerting on dimensions of metrics, and more responsive dashboards?

- ◉ Log-based

  ✗ That's incorrect. Log-based metrics are aggregated at query time and require more processing to produce results.
- ○ Pre-aggregated

  ✓ That's correct. Pre-aggregated metrics are stored as a time series and only with key dimensions which enables near real-time alerting on dimensions of metrics, more responsive dashboards.
- ○ Azure Service Bus

12. Caching and Content delivery within solutions

Azure Cache for Redis

Azure Cache for Redis provides an in-memory data store based on the Redis software.

It's able to process large volumes of application requests by keeping frequently accessed data in the server memory, which can be written to and read from quickly. Redis brings a critical low-latency and high-throughput data storage solution to modern applications.

Azure Cache for Redis improves application performance by supporting common application architecture patterns. Some of the most common include the following patterns:

Pattern	**Description**
Data cache	Databases are often too large to load directly into a cache. It's common to use the cache-aside pattern to load data into the cache only as needed. When the system makes changes to the data, the system can also update the cache, which is then distributed to other clients.
Content cache	Many web pages are generated from templates that use static content such as headers, footers, banners. These static items shouldn't change often. Using an in-memory cache provides quick access to static content compared to backend datastores.
Session store	This pattern is commonly used with shopping carts and other user history data that a web application might associate with user cookies. Storing too much in a cookie can have a negative effect on performance as the cookie size grows and is passed and validated with every request. A typical solution uses the cookie as a key to query the data in a database. Using an in-memory cache, like Azure Cache for Redis, to associate information with a user is much faster than interacting with a full relational database.
Job and message queuing	Applications often add tasks to a queue when the operations associated with the request take time to execute. Longer running operations are queued to be processed in sequence, often by another server. This method of deferring work is called task queuing.
Distributed transactions	Applications sometimes require a series of commands against a backend data-store to execute as a single atomic operation. All commands must succeed, or all must be rolled back to the initial state. Azure Cache for Redis supports executing a batch of commands as a single transaction.

Azure Cache for Redis is available in these tiers:

Tier	Description
Basic	An OSS Redis cache running on a single VM. This tier has no service-level agreement (SLA) and is ideal for development/test and non-critical workloads.
Standard	An OSS Redis cache running on two VMs in a replicated configuration.
Premium	High-performance OSS Redis caches. This tier offers higher throughput, lower latency, better availability, and more features. Premium caches are deployed on more powerful VMs compared to the VMs for Basic or Standard caches.
Enterprise	High-performance caches powered by Redis Labs' Redis Enterprise software. This tier supports Redis modules including RediSearch, RedisBloom, and RedisTimeSeries. Also, it offers even higher availability than the Premium tier.
Enterprise Flash	Cost-effective large caches powered by Redis Labs' Redis Enterprise software. This tier extends Redis data storage to non-volatile memory, which is cheaper than DRAM, on a VM. It reduces the overall per-GB memory cost.

```
static async Task Main(string[] args)
{
    // The connection to the Azure Cache for Redis is managed by the
ConnectionMultiplexer class.
    using (var cache = ConnectionMultiplexer.Connect(connectionString))
    {
        IDatabase db = cache.GetDatabase();

        // Snippet below executes a PING to test the server connection
        var result = await db.ExecuteAsync("ping");
        Console.WriteLine($"PING = {result.Type} : {result}");

        // Call StringSetAsync on the IDatabase object to set the key "test:key"
to the value "100"
        bool setValue = await db.StringSetAsync("test:key", "100");
        Console.WriteLine($"SET: {setValue}");

        // StringGetAsync takes the key to retrieve and return the value
        string getValue = await db.StringGetAsync("test:key");
        Console.WriteLine($"GET: {getValue}");

    }
}
```

1. Which of the Azure Cache for Redis service tiers is the lowest tier recommended for use in production scenarios?

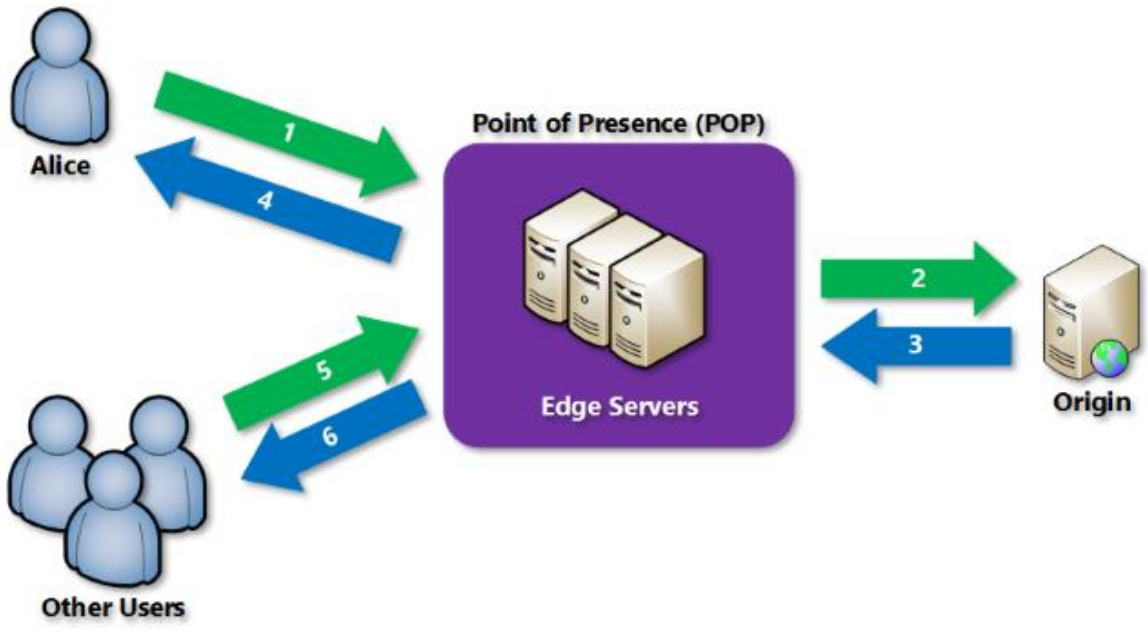
- (●) Basic
  - ✗ That's incorrect. The basic tier has no service-level agreement and is ideal for development/test and non-critical workloads.
- ( ) Standard
  - ✓ That's correct. The standard tier is the lowest tier that offers replication which is always recommended for production scenarios.
- ( ) Premium

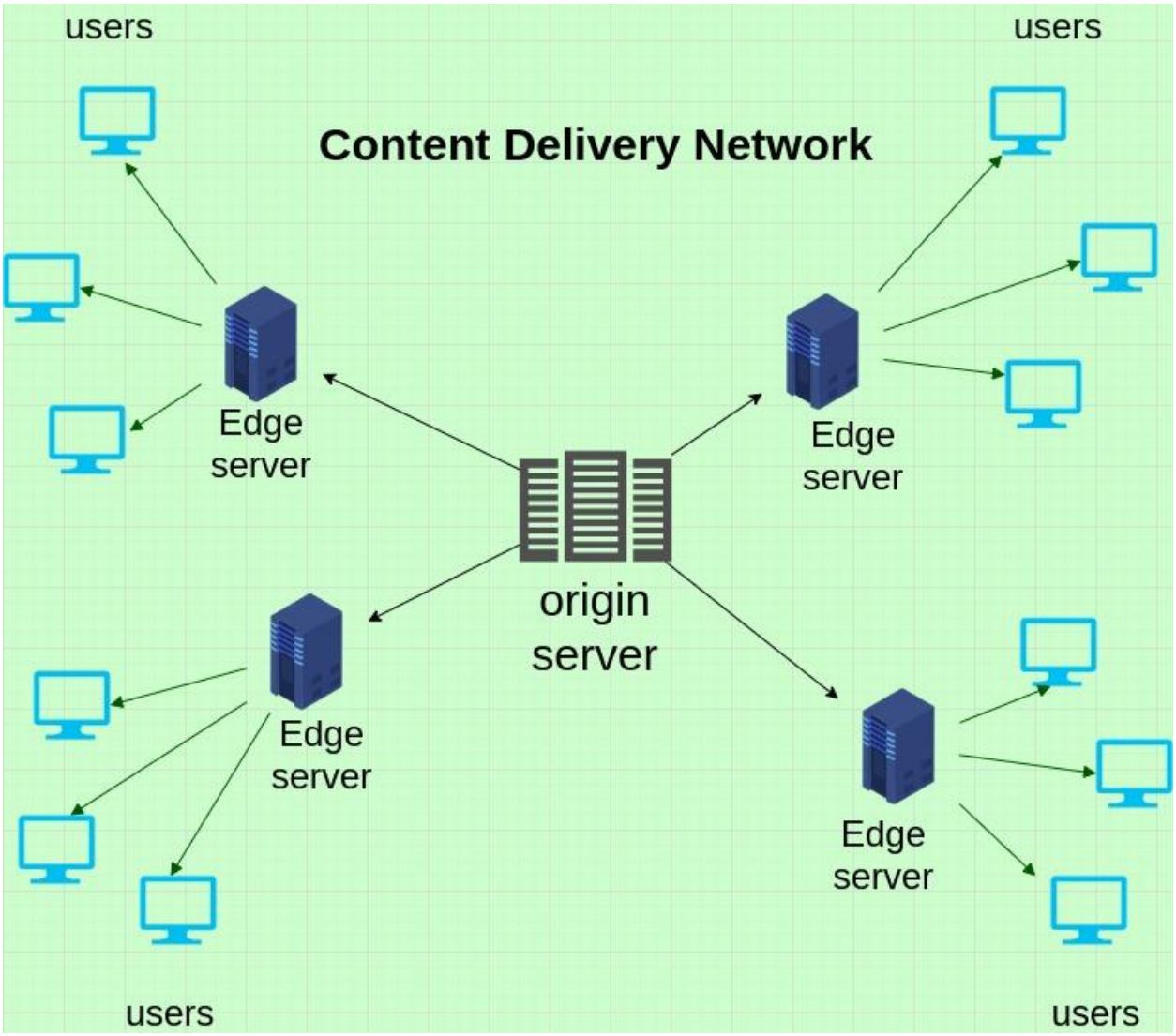
2. Caching is important because it allows us to store commonly used values in memory. However, we also need a way to expire values when they are stale. In Redis this is done by applying a time to live (TTL) to a key. Which value represents the expire time resolution?

- ( ) 1 millisecond
  - ✓ That's correct. The expire time resolution is always 1 millisecond.
- ( ) 10 milliseconds
- (●) seconds or milliseconds
  - ✗ That's incorrect. Expirations can be set using seconds or milliseconds precision, but the expire time resolution is always 1 millisecond.

Azure Content Delivery Network (CDN)

A content delivery network (CDN) is a distributed network of servers that can efficiently deliver web content to users. CDNs' store cached content on edge servers in point-of-presence (POP) locations that are close to end users, to minimize latency.





Azure Content Delivery Network (CDN) includes four products:

- Azure CDN Standard from Microsoft
- Azure CDN Standard from Akamai
- Azure CDN Standard from Verizon
- Azure CDN Premium from Verizon

```
az cdn endpoint load \
    --content-paths '/img/*' '/js/module.js' \
    --name ContosoEndpoint \
    --profile-name DemoProfile \
    --resource-group ExampleGroup
```

```
static void Main(string[] args)
{
    // Create CDN client
    CdnManagementClient cdn = new CdnManagementClient(new
TokenCredentials(authResult.AccessToken))
        { SubscriptionId = subscriptionId };
}
```

1. Each Azure subscription has default limits on resources needed for an Azure Content Delivery Network. Which of the following resources has subscription limitations that may impact your solution?

- (●) Resource group

  ✗ **That's incorrect. Resource groups are required by Azure CDN, but they are not limited by subscription level.**

- ( ) CDN profiles

  ✓ **That's correct. The number of CDN profiles that can be created is limited by the type of Azure subscription.**

- ( ) Storage account

2. When publishing a website through Azure CDN, the files on that site are cached until their time-to-live (TTL) expires. What is the default TTL for large file optimizations?

- (●) One day

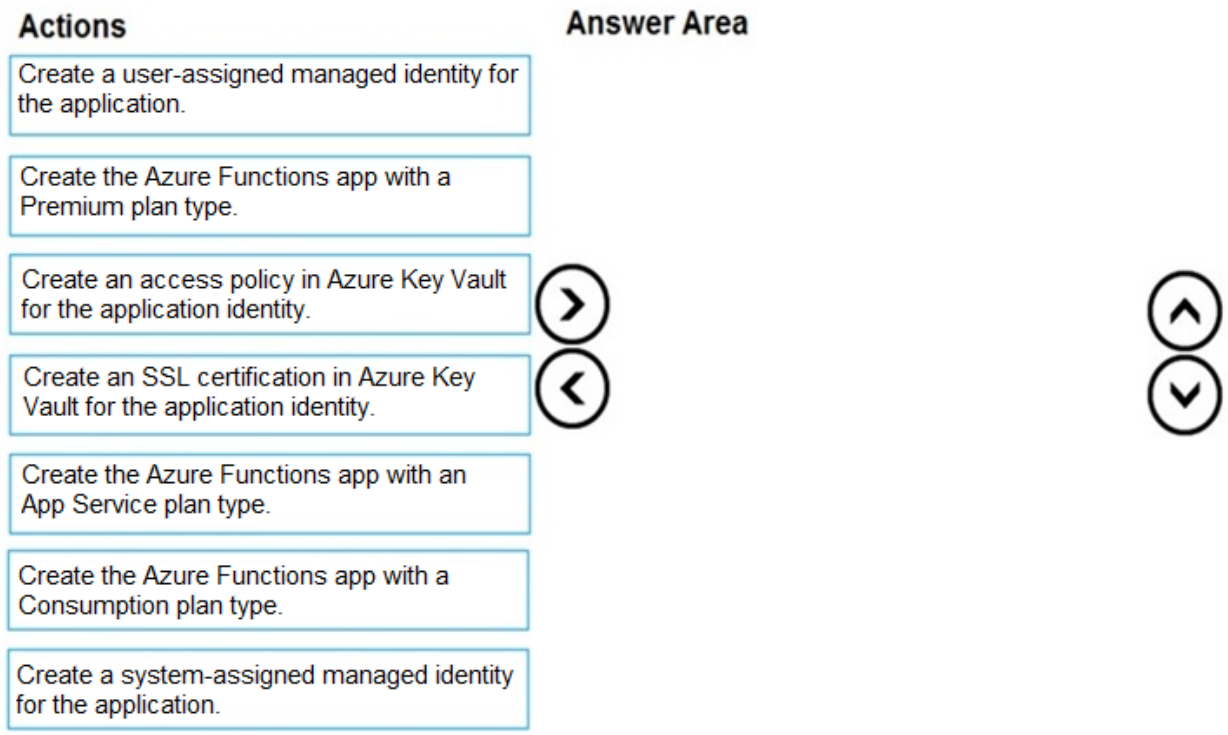
  ✓ **That's correct. The default TTL for large file optimizations is one day.**

- ( ) One week
- ( ) One year

Missing Questions

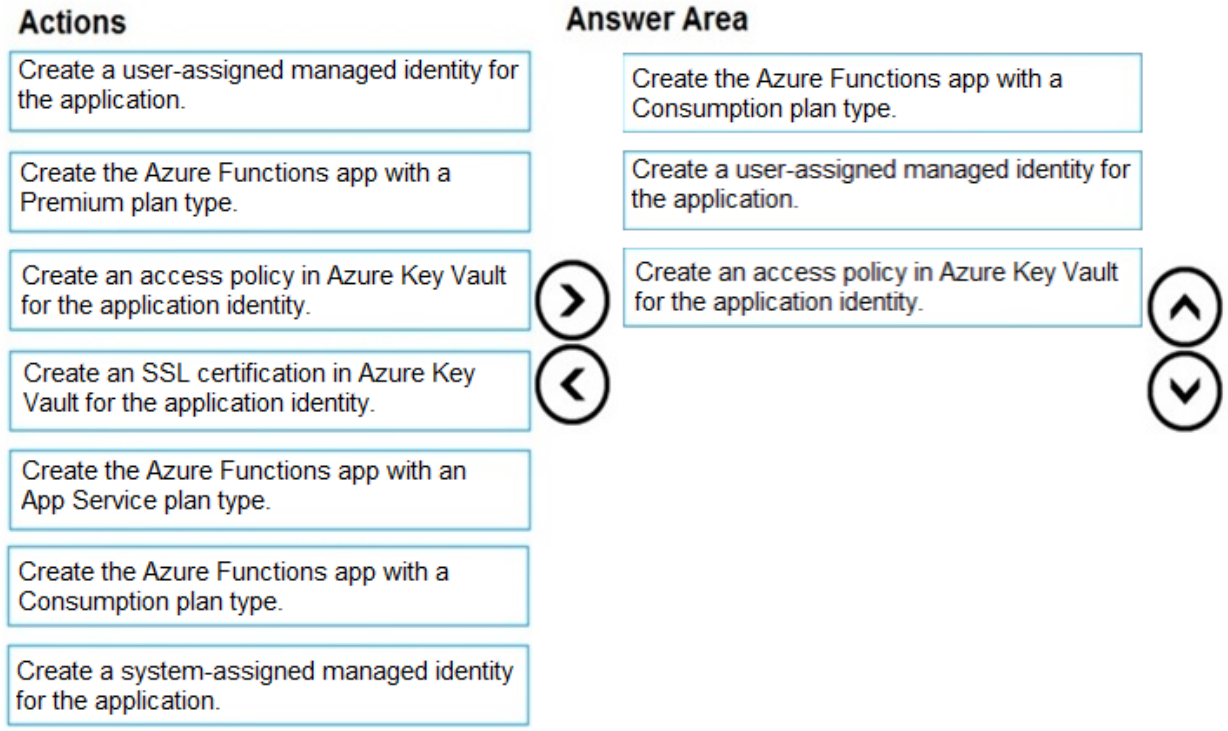
You are developing a serverless Java application on Azure. You create a new Azure Key Vault to work with secrets from a new Azure Functions application.
The application must meet the following requirements:
- Reference the Azure Key Vault without requiring any changes to the Java code.
- Dynamically add and remove instances of the Azure Functions host based on the number of incoming application events.
- Ensure that instances are perpetually warm to avoid any cold starts.
- Connect to a VNet.
- Authentication to the Azure Key Vault instance must be removed if the Azure Function application is deleted.

You need to grant the Azure Functions application access to the Azure Key Vault.
Which three actions should you perform in sequence? To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.
Select and Place:



Hide Answer

Suggested
Answer:



Step 1: Create the Azure Functions app with a Consumption plan type.

Use the Consumption plan for serverless.

Step 2: Create a system-assigned managed identity for the application.

Create a system-assigned managed identity for your application.

Key Vault references currently only support system-assigned managed identities. User-assigned identities cannot be used.

Step 3: Create an access policy in Key Vault for the application identity.

Create an access policy in Key Vault for the application identity you created earlier. Enable the "Get" secret permission on this policy. Do not configure the

"authorized application" or applicationId settings, as this is not compatible with a managed identity.

Reference:

https://docs.microsoft.com/en-us/azure/app-service/app-service-key-vault-references

Your company has an azure subscription that includes a storage account, a resource group, a blob container and a file share.

A fellow administrator named Jon Ross used an Azure Resource Manager template to deploy a virtual machine and an Azure Storage account.

You need to identify the Azure Resource Manager template the Jon Ross used.

Solution: You access the Virtual Machine blade.

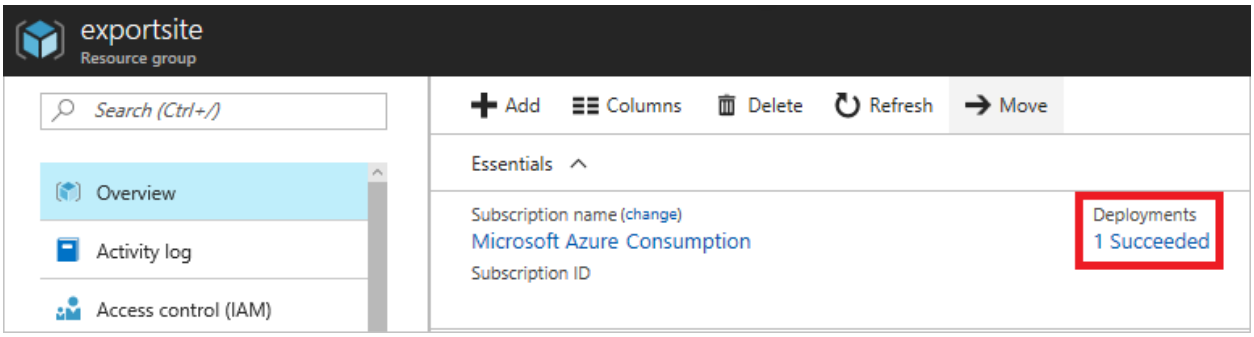
Does the solution meet the goal?

- A. Yes
- B. No

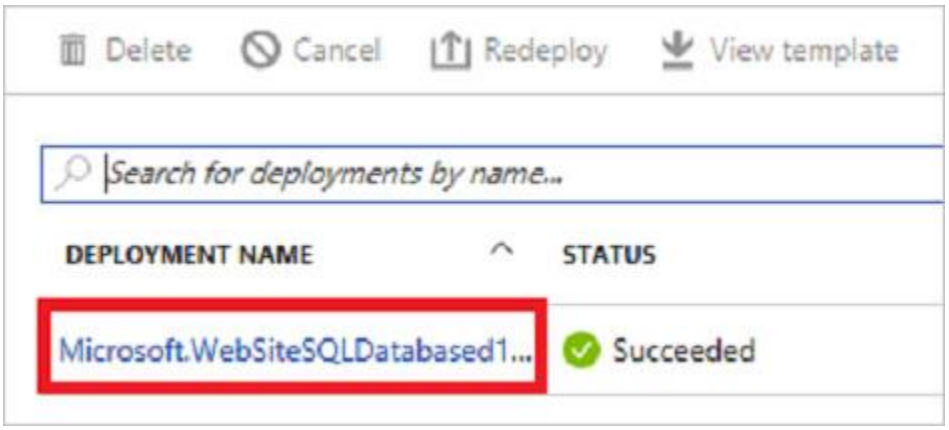
Suggested Answer: *B*

View template from deployment history

Go to the resource group for your new resource group. Notice that the portal shows the result of the last deployment. Select this link.

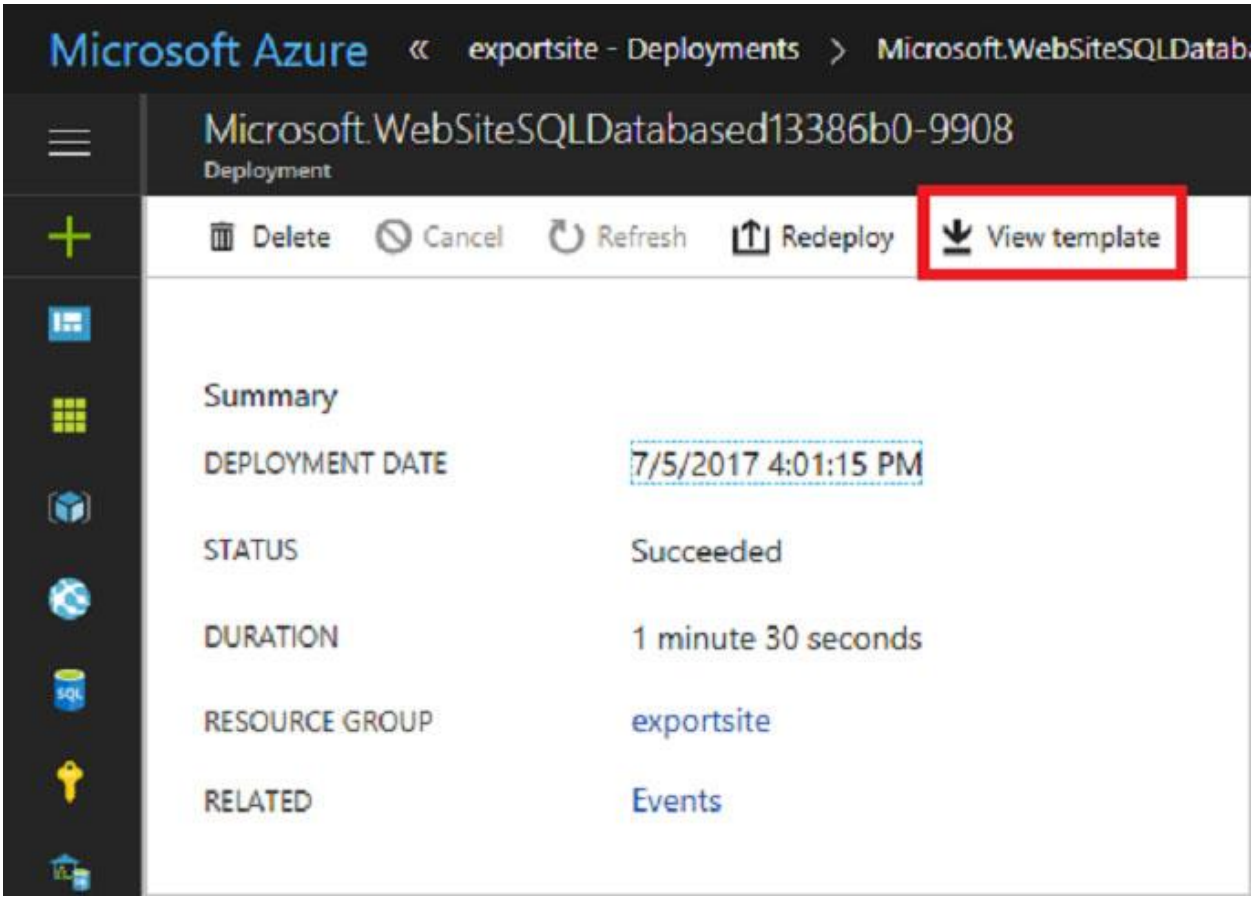


You see a history of deployments for the group. In your case, the portal probably lists only one deployment. Select this deployment.



The portal displays a summary of the deployment. The summary includes the status of the deployment and its operations and the values that you provided for parameters. To see the template that you used for the deployment,

select View template.



You have two Hyper-V hosts named Host1 and Host2. Host1 has an Azure virtual machine named VM1 that was deployed by using a custom Azure Resource
Manager template.
You need to move VM1 to Host2.
What should you do?

- A. From the Update management blade, click Enable.
- B. From the Overview blade, move VM1 to a different subscription.
- C. From the Redeploy blade, click Redeploy.
- D. From the Profile blade, modify the usage location.

Answer: C From the Redeploy blade, click Redeploy. Most Voted

Your company has an Azure Kubernetes Service (AKS) cluster that you manage from an Azure AD-joined device. The cluster is located in a resource group.
Developers have created an application named MyApp. MyApp was packaged into a container image.
You need to deploy the YAML manifest file for the application.
Solution: You install the Azure CLI on the device and run the kubectl apply `"f myapp.yaml command.
Does this meet the goal?

- A. Yes
- B. No

Your company has an Azure subscription.
You need to deploy a number of Azure virtual machines to the subscription by using Azure Resource Manager (ARM) templates. The virtual machines will be included in a single availability set.
You need to ensure that the ARM template allows for as many virtual machines as possible to remain accessible in the event of fabric failure or maintenance.
Which of the following is the value that you should configure for the **platformFaultDomainCount** property?

- A. 10
- B. 30
- C. Min Value
- D. Max Value

Suggested Answer: **D**

The number of fault domains for managed availability sets varies by region - either two or three per region.

Resource Manager deployments can then be increased to provide up to 20 update domains

- **platformFaultDomainCount** : **Max Value**
- **platformUpdateDomainCount** : **20**

You are designing an Azure WebJob that will run on the same instances as a web app.
You want to make use of a suitable WebJob type. The webjob type should also allow for the option to restrict the WebJob to a single instance.

Solution: You configure the use of the Continuous WebJob type.
Does the solution meet the goal?

- A. Yes
- B. No

`Answer : A – YES`

Yes, Continuous runs on all instances that the web app runs on. You can optionally restrict the WebJob to a single instance.

WebJob types

The following table describes the differences between *continuous* and *triggered* WebJobs.

Continuous	Triggered
Starts immediately when the WebJob is created. To keep the job from ending, the program or script typically does its work inside an endless loop. If the job does end, you can restart it. Typically used with WebJobs SDK.	Starts only when triggered manually or on a schedule.
Runs on all instances that the web app runs on. You can optionally restrict the WebJob to a single instance.	Runs on a single instance that Azure selects for load balancing.

This question requires that you evaluate the underlined text to determine if it is correct.
You company has an on-premises deployment of MongoDB, and an Azure Cosmos DB account that makes use of the MongoDB API.
You need to devise a strategy to migrate MongoDB to the Azure Cosmos DB account.
You include the Data Management Gateway tool in your migration strategy.
Instructions: Review the underlined text. If it makes the statement correct, select ‚€No change required.‚€ If the statement is incorrect, select the answer choice that makes the statement correct.

- A. No change required
- B. mongorestore
- C. Azure Storage Explorer

- D. AzCopy

You are developing a mobile app that uses an Azure SQL Database named Weyland.
The database contains a table names Customers that has a field named email_address.
You want to implement dynamic data masking to hide the data in the email_address field.
Solution: You run the follows transact-SQL statement:
ALTER TABLE [dbo].[Weyland].[Customers]
ALTER COLUMN [email_address]
ADD MASKED WITH (FUNCTION = 'email()')
Does the solution meet the goal?

- A. Yes
- B. No

You are developing an e-Commerce Web App.
You want to use Azure Key Vault to ensure that sign-ins to the e-Commerce Web App are secured by using Azure App Service authentication and Azure Active
Directory (AAD).
What should you do on the e-Commerce Web App?

- A. Run the az keyvault secret command.
- B. Enable Azure AD Connect.
- C. Enable Managed Service Identity (MSI).
- D. Create an Azure AD service principal.

A managed identity from Azure Active Directory allows your app to easily access other AAD-protected resources such as Azure Key Vault.

Your Azure Active Directory Azure (Azure AD) tenant has an Azure subscription linked to it.
Your developer has created a mobile application that obtains Azure AD access tokens using the OAuth 2 implicit grant type.
The mobile application must be registered in Azure AD.

You require a redirect URI from the developer for registration purposes.
Instructions: Review the underlined text. If it makes the statement correct, select ‚€No change is needed.‚€ If the statement is incorrect, select the answer choice that makes the statement correct.

- A. No change required.
- B. a secret
- C. a login hint
- D. a client ID

Your company has an Azure Active Directory (Azure AD) environment. Users occasionally connect to Azure AD via the Internet.
You have been tasked with making sure that users who connect to Azure AD via the internet from an unidentified IP address, are automatically encouraged to change passwords.
Solution: You configure the use of **Azure AD Privileged Identity Management**.
Does the solution meet the goal?

- A. Yes
- B. No

Hide Answer

- **Azure AD Privileged Identity Management –** **Wrong**
- Azure AD Identity **Protection** - **Correct**

You manage an Azure SQL database that allows for Azure AD authentication.
You need to make sure that database developers can connect to the SQL database via Microsoft SQL Server Management Studio (SSMS). You also need to make sure the developers use their on-premises Active Directory account for authentication. Your strategy should allow for authentication prompts to be kept to a minimum.
Which of the following should you implement?

- A. Azure AD token.
- B. Azure Multi-Factor authentication.
- C. Active Directory integrated authentication.
- D. OATH software tokens.

Answer: *C*

Azure AD can be the initial Azure AD managed domain. Azure AD can also be an on-premises Active Directory Domain Services that is federated with the Azure AD. Using an Azure AD identity to connect using SSMS or SSDT

You are configuring a web app that delivers streaming video to users. The application makes use of continuous integration and deployment.
You need to ensure that the application is highly available and that the users' streaming experience is constant. You also want to configure the application to store data in a geographic location that is nearest to the user.
Solution: You include the use of an Azure Content Delivery Network (CDN) in your design.
Does the solution meet the goal?

- A. Yes
- B. No

You develop a Web App on a tier D1 app service plan.
You notice that page load times increase during periods of peak traffic.
You want to implement automatic scaling when CPU load is above 80 percent. Your solution must minimize costs.
What should you do first?

- A. Enable autoscaling on the Web App.
- B. Switch to the Premium App Service tier plan.
- C. Switch to the Standard App Service tier plan. Most Voted
- D. Switch to the Azure App Services consumption plan.

Your company's Azure subscription includes an Azure Log Analytics workspace.
Your company has a hundred on-premises servers that run either Windows Server 2012 R2 or Windows Server 2016, and is linked to the Azure Log Analytics workspace. The Azure Log Analytics workspace is set up to gather performance counters associated with security from these linked servers.
You must configure alerts based on the information gathered by the Azure Log Analytics workspace.

You have to make sure that alert rules allow for dimensions, and that alert creation time should be kept to a minimum. Furthermore, a single alert notification must be created when the alert is created and when the alert is resolved.
You need to make use of the necessary signal type when creating the alert rules.
Which of the following is the option you should use?

- A. The Activity log signal type.
- B. The Application Log signal type.
- C. The Metric signal type.
- D. The Audit Log signal type.

You are developing a .NET Core MVC application that allows customers to research independent holiday accommodation providers.
You want to implement Azure Search to allow the application to search the index by using various criteria to locate documents related to accommodation.
You want the application to allow customers to search the index by **using regular expressions**.
What should you do?

- A. Configure the SearchMode property of the SearchParameters class.
- B. Configure the QueryType property of the SearchParameters class.
- C. Configure the Facets property of the SearchParameters class.
- D. Configure the Filter property of the SearchParameters class.

Hide Answer

Suggested Answer: *B*

The SearchParameters.QueryType Property gets or sets a value that specifies the syntax of the search query. The default is 'simple'. Use 'full' if your query uses the Lucene query syntax.
You can write queries against Azure Search based on the rich Lucene Query Parser syntax for specialized query forms: wildcard, fuzzy search, proximity search, regular expressions are a few examples.

You are developing a .NET Core MVC application that allows customers to research independent holiday accommodation providers.
You want to implement Azure Search to allow the application to search the index by using various criteria to locate documents related to accommodation venues.
You want the application to list holiday accommodation venues that fall within a specific price range

and are within a specified distance to an airport.
What should you do?

- A. Configure the SearchMode property of the SearchParameters class.
- B. Configure the QueryType property of the SearchParameters class.
- C. Configure the Facets property of the SearchParameters class.
- D. Configure the Filter property of the SearchParameters class.

The Filter property gets or sets the OData $filter expression to apply to the search query.

You are a developer at your company.
You need to update the definitions for an existing Logic App.
What should you use?

- A. the Enterprise Integration Pack (EIP)
- B. the Logic App Code View
- C. the API Connections
- D. the Logic Apps Designer

You are a developer at your company.
You need to edit the workflows for an existing Logic App.
What should you use?

- A. the Enterprise Integration Pack (EIP)
- B. the Logic App Code View
- C. the API Connections
- D. the Logic Apps Designer

The API back end is hosted in an Azure App Service instance. You have implemented a RESTful service for the API back end.
You must configure back-end authentication for the API Management service instance.
Solution: You configure Client cert gateway credentials for the Azure resource.
Does the solution meet the goal?

- A. Yes
- B. No

- You con!gure Basic gateway credentials for the HTTP(s) endpoint
- You con!gure Client cert gateway credentials for the HTTP(s) endpoint
- You configure Client cert gateway credentials for the Azure resource

You develop a stateful ASP.NET Core 2.1 web application named PolicyApp and deploy it to an Azure App Service Web App. The PolicyApp reacts to events from
Azure Event Grid and performs policy actions based on those events.
You have the following requirements:
☞ Authentication events must be used to monitor users when they sign in and sign out.
☞ All authentication events must be processed by PolicyApp.
☞ Sign outs must be processed as fast as possible.
What should you do?

- A. Create a new Azure Event Grid subscription for all authentication events. Use the subscription to process sign-out events.
- B. Create a separate Azure Event Grid handler for sign-in and sign-out events.
- C. Create separate Azure Event Grid topics and subscriptions for sign-in and sign-out events.
- D. Add a subject prefix to sign-out events. Create an Azure Event Grid subscription. Configure the subscription to use the subjectBeginsWith filter.

Hide Answer

Q's

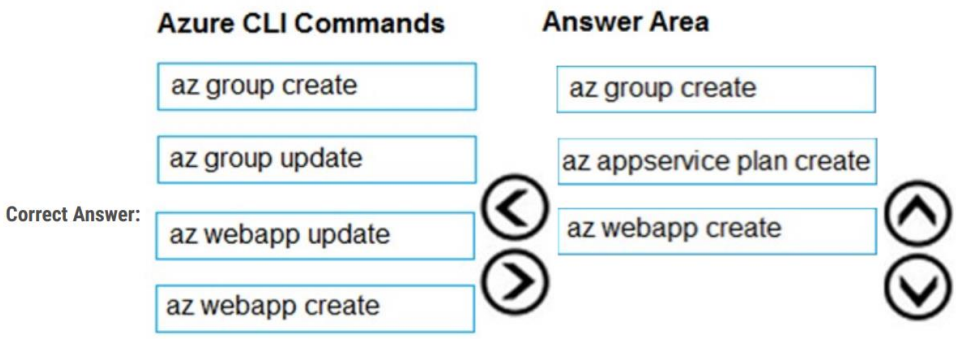
When photos are uploaded, they must be processed to produce and save a mobile-friendly version of the image

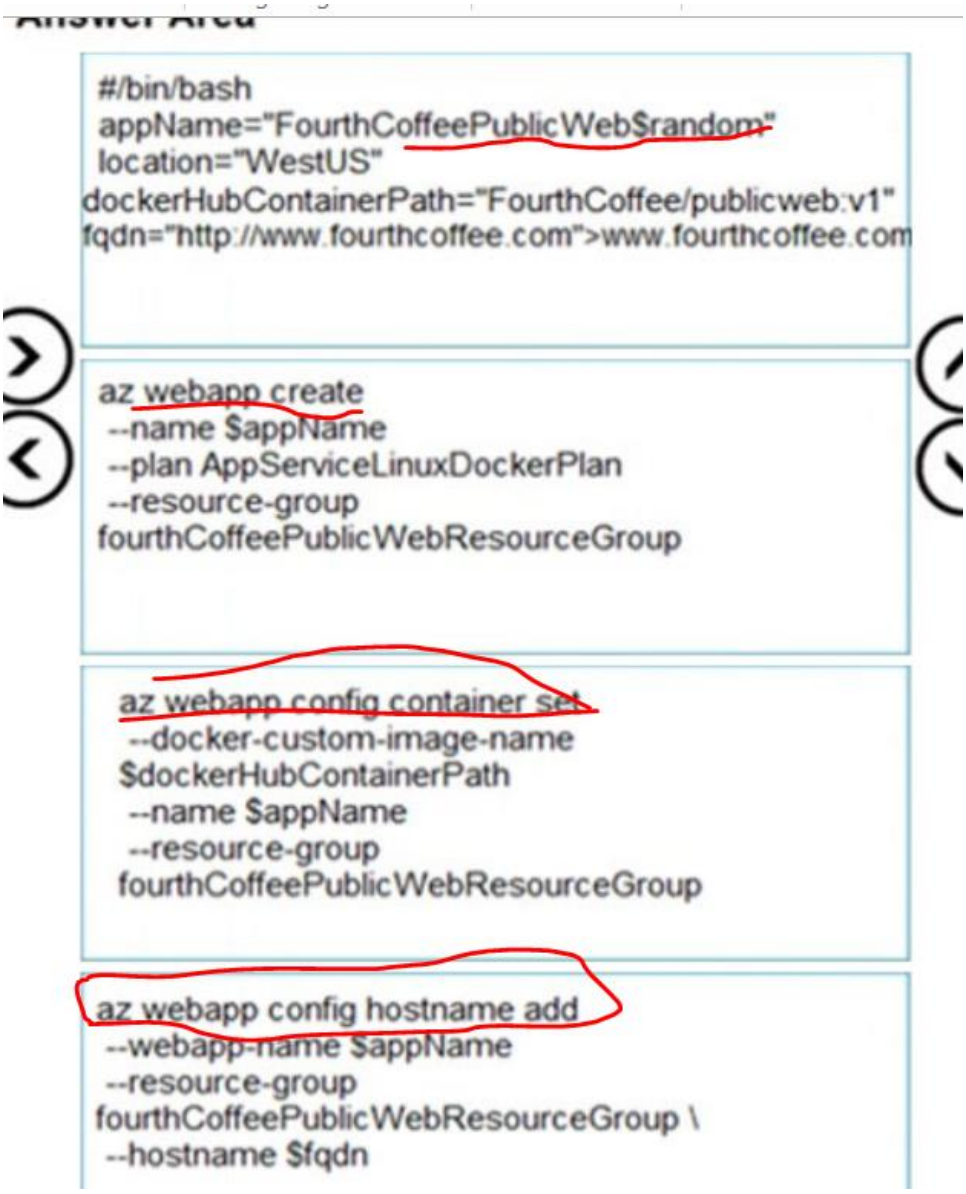
Azure Blob upload + **Azure Function Trigger with Consuption plan**

Azure App Service API , deployment slots named Testing and Production. You enable auto swap on the Production deployment slot

- **Enable auto swap** for the Testing slot. Deploy the app to the Testing slot
- **Disable auto swap**. Re-enable auto swap and deploy the app to the Production slot

---






```
Azure Function with Consumption Plan

User-Assigned Managed Identity

Create Access Policy + Azure Key Vault
```

Create Java web app using GitHub repository & deployment slot named staging.

```
az group
```

```
az appservice plan

az webapp

az webapp deployment slot: staging/Production

az webapp deployment source: Github
```

Trigger CosmosDB, Tip Property

```
getRequest()

isNaN==null

setBody()
```

HTTP triggered Azure Function app to process Azure Storage blob data. The app continues to time out after four minutes. The app must process the blob data.

HTTP trigger payload into an **Azure Service Bus** queue

The VMSS must not be created until the storage accounts have been successfully created and an associated load balancer and virtual network is configured.

```
CopyIndex

Copy

DependsOn
```

Steps

Actions

Answer Area

Register the application with the Microsoft identity platform.

Build a client by using the client app ID.

Create an authentication provider.

Create a new instance of the GraphServiceClient.

Invoke the request to the Microsoft Graph API.

Actions

Send console logs.

Answer Area

Create a Log Analytics workspace.

Add a VMInsights solution.

Install agents on the VM and VM scale set to be monitored.

Create an Application Insights resource.

Actions

Create action groups and alert rules.

Add a diagnostic setting to the Azure Function App.

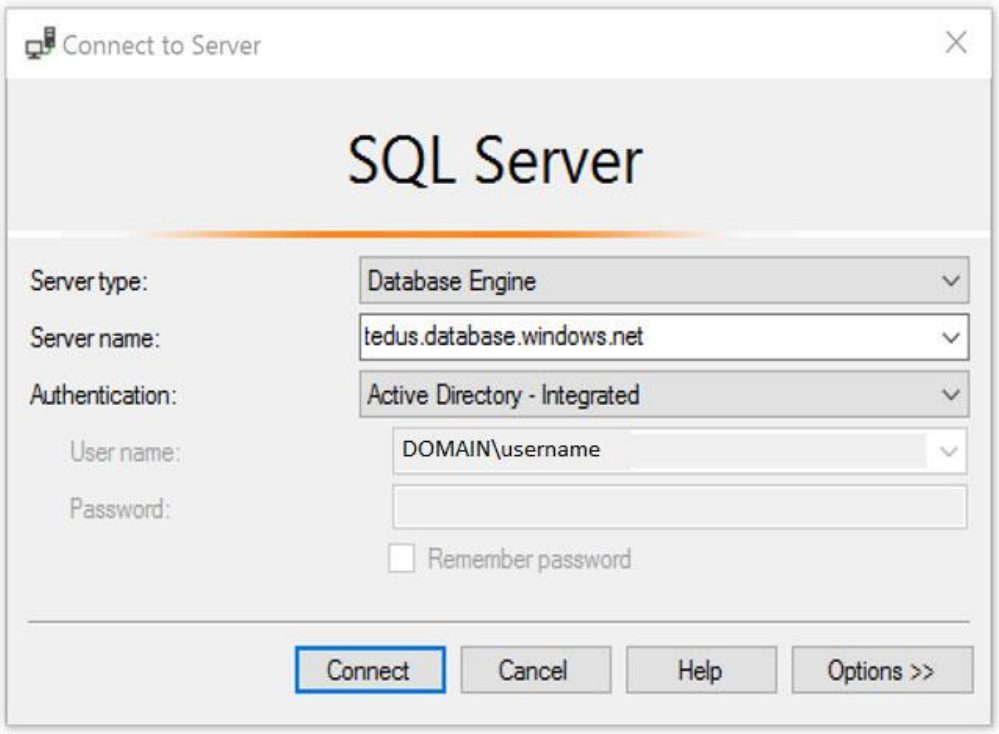
Create an Azure storage account.

Answer Area

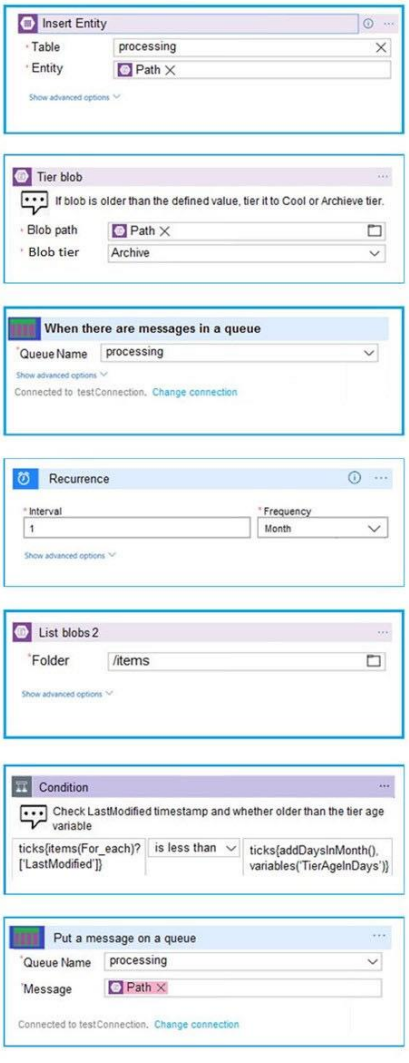
Create a Log Analytics workspace.

Install the Logic Apps Management solution.

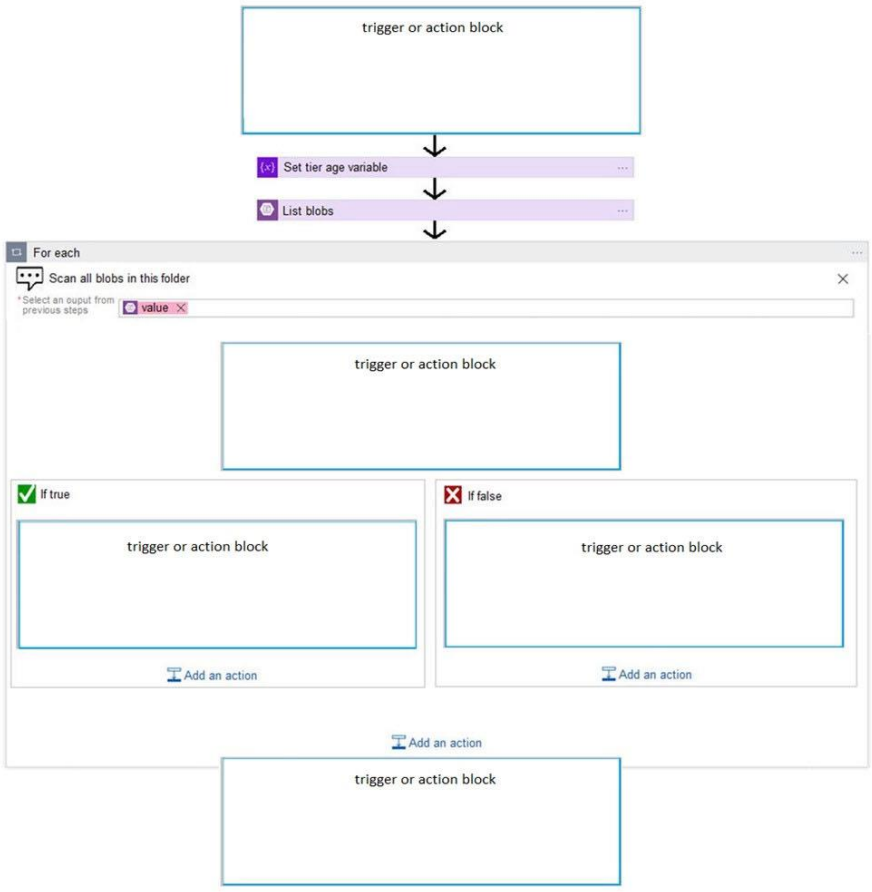
Add a diagnostic setting to the Azure Logic App.

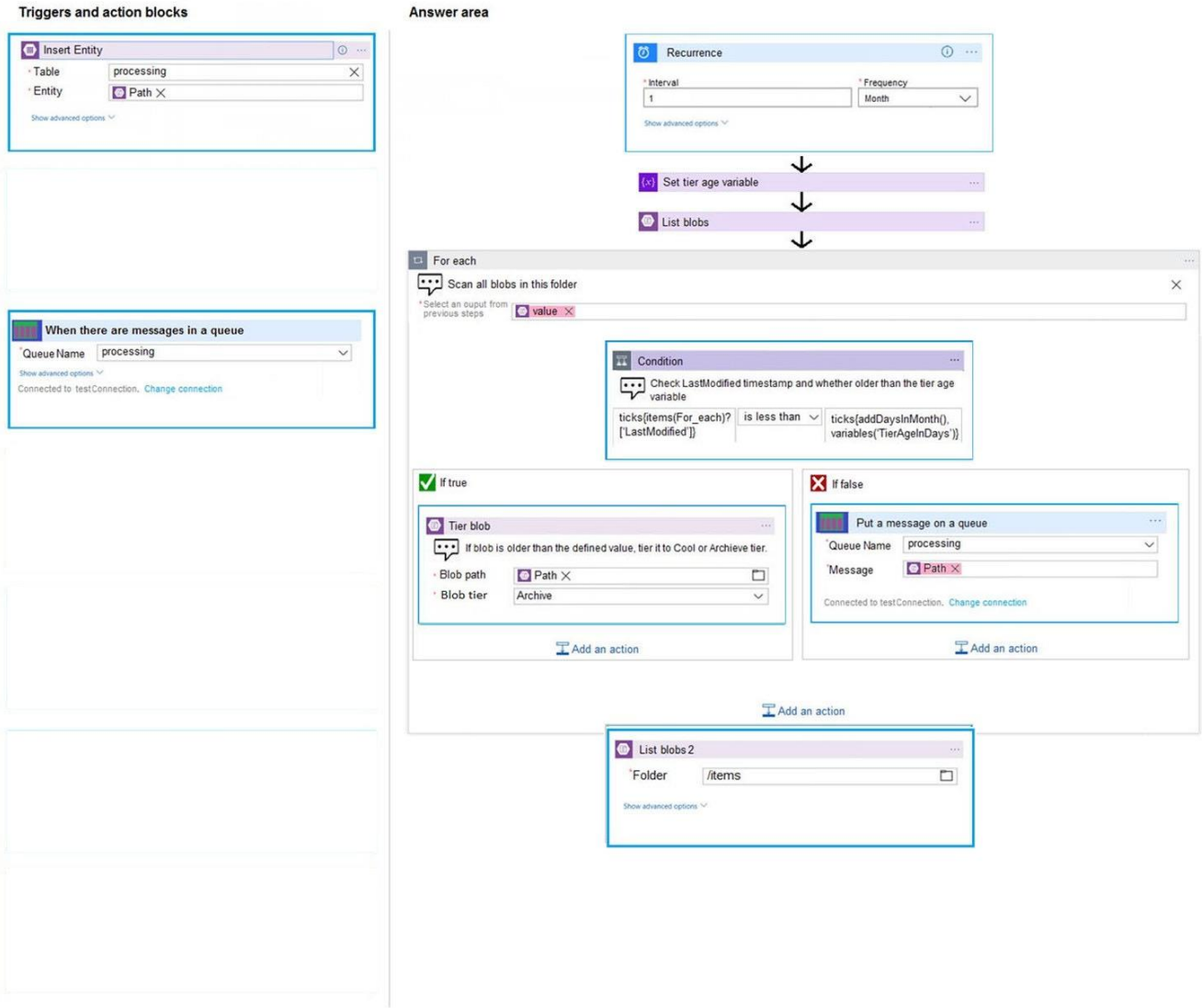

Connect to Server
SQL Server
Server type:
Database Engine
Server name:
tedus.database.windows.net
Authentication:
Active Directory - Integrated
User name:
DOMAIN\username
Password:
Remember password
Connect
Cancel
Help
Options >>

Triggers and action blocks



Answer area





Actions	Answer Area
Create a Service Bus Namespace for each restaurat for which a driver can receive messages.	Create s single Service Bus Namespace.
Create a single Service Bus subscription.	Create a single Service Bus topic.
Create a Service Bus topic for each restaurant for which a driver can receive messages.	Create a Service Bus subscription for each restaurant for which a driver can receive orders.

Answer Area

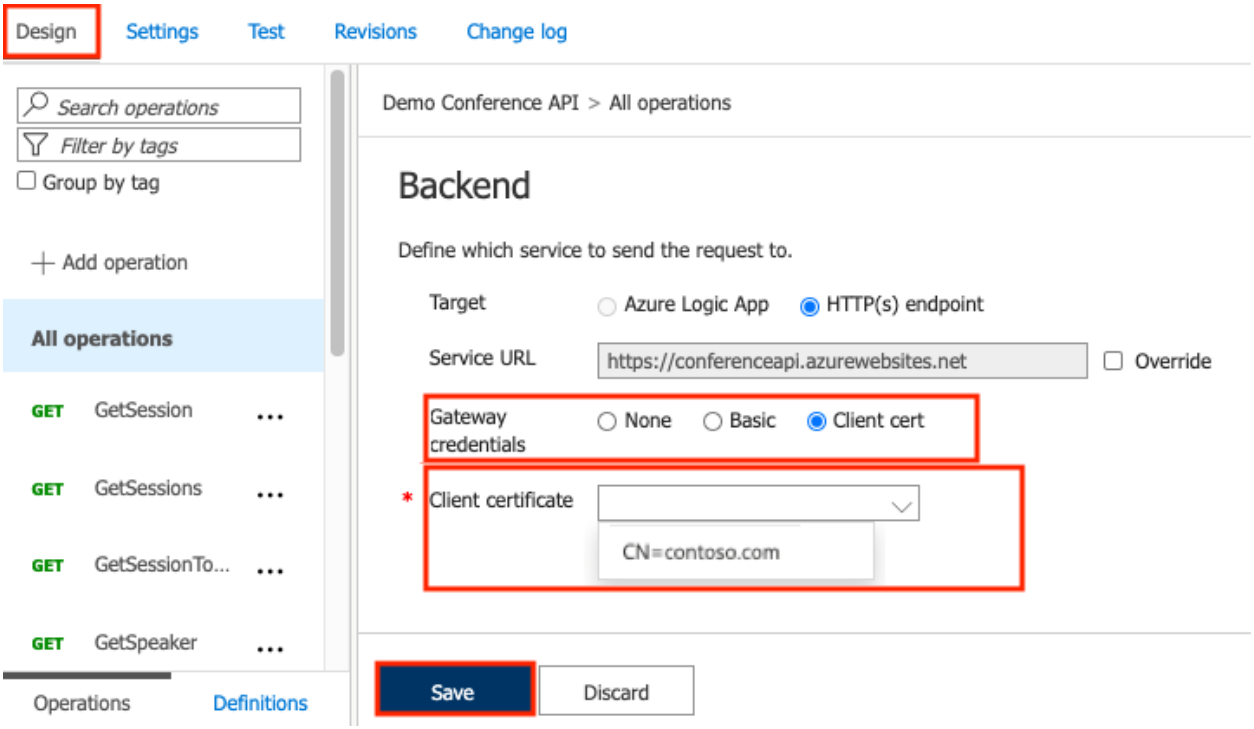
Option	Value
WebHook event delivery	ValidationCode handshake ValidationURL handshake JWT token
Topic publishing	SAS tokens Key authentication Management Access Control

Technologies

- Azure Event Hub
- Azure Service Bus
- Azure App Service

Answer Area

Object	Technology
Event Source	Azure Blob Storage
Event Receiver	Azure Event Grid
Event Handler	Azure Logic App



Actions

Create a custom connector for the Logic App.

Link the custom connector to the Logic App.

Answer Area

Create an integration account in the Azure portal.

Add partners, schemas, certificates, maps, and agreements.

Link the Logic App to the integration account.

Update the Logic App to use the partners, schemas, certificates, maps, and agreements.

Actions

Update line SC05 of Security.cs to include error handling and then redeploy the code.

Update line SC03 of Security.cs to include a using statement and then re-deploy the code.

Upload the certificate to source control.

Answer Area

Generate a certificate.

Upload the certificate to Azure Key Vault.

Import the certificate to Azure App Service.

Add the certificate thumbprint to the WEBSITE_LOAD_CERTIFICATES app setting.

Answer Area

```
var resolver = new KeyVaultKeyResolver(_keyVaultClient);
var keyBundle = await _keyVaultClient.GetKeyAsync("...", "...");
```

```
var key = keyBundle.Key;
var key = keyBundle.KeyIdentifier.Identifier;
var key = await resolver.ResolveKeyAsync("encrypt", null);
var key = await resolver.ResolveKeyAsync(keyBundle.KeyIdentifier.Identifier, CancellationToken.None);
```

```
var x = keyBundle.Managed;
var x = AuthenticationScheme.SharedKey;
var x = new BlobEncryptionPolicy(key, resolver);
var x = new DeleteRetentionPolicy {Enabled = key.Kid != null};
```

```
cloudBlobClient.AuthenticationScheme = x;
cloudBlobClient.DefaultRequestOptions.RequireEncryption = x;
cloudBlobClient.DefaultRequestOptions.EncryptionPolicy = x;
cloudBlobClient.SetServiceProperties(new ServiceProperties(deleteRetentionPolicy:x));
```

Actions	Answer area
	Build a new application image by using dockerfile.
Create an alias of the image with a new build number.	Create an alias of the image with the fully qualified path to the registry.
	Log in to the registry and push image.
Download the image to your local computer.	

Content

https://coosmiin.github.io/az-204/azure/certification/2021/02/21/notes-on-az-204-developing-solutions-for-azure-certification.html#implement-iaas-solutions

Codes

Azure Storage Account

```
public static async Task Main(string[] args)
{
    StorageSharedKeyCredential accountCredentials = new
StorageSharedKeyCredential(storageAccountName, storageAccountKey);

    BlobServiceClient serviceClient = new BlobServiceClient(new
Uri(blobServiceEndpoint), accountCredentials);

    AccountInfo info = await serviceClient.GetAccountInfoAsync();

    await Console.Out.WriteLineAsync($"Connected to Azure Storage Account");
    await Console.Out.WriteLineAsync($"Account name:\t{storageAccountName}");
    await Console.Out.WriteLineAsync($"Account kind:\t{info?.AccountKind}");
    await Console.Out.WriteLineAsync($"Account sku:\t{info?.SkuName}");
}
```

Cosmos DB storage

Azure VM & Containers

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

Labs

https://microsoftlearning.github.io/AZ-204-DevelopingSolutionsforMicrosoftAzure/

DUMPS

https://cloudtech.how/microsoft/exam-az-204-developing-solutions-for-microsoft-azure/az-204-question-5/ **(BEST OF ALL)**

https://free-braindumps.com/microsoft/free-az-204-braindumps.html?p=2

https://www.examtopics.com/exams/microsoft/az-204/

https://www.certlibrary.com/exam/AZ-204

REF.

https://www.aguidetocloud.com/courses/az-204

https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE4oZ7B

https://github.com/MicrosoftLearning/AZ-204-DevelopingSolutionsforMicrosoftAzure

https://parveensingh.com/how-to-pass-az-204-in-30-days/

https://docs.microsoft.com/en-us/learn/certifications/exams/az-204

https://docs.microsoft.com/en-us/learn/certifications/azure-developer/?wt.mc_id=esi_lxp_webpage_wwl

https://www.thomasmaurer.ch/2020/03/az-204-study-guide-developing-solutions-for-microsoft-azure/

https://www.youtube.com/watch?v=tB_tBPfQJMI&list=PLhLKc18P9YODdrbyuA52Zn9-kwboIOz2W