AZ 204 Final Exam 4

Done by Ahmed Fouad
Ahmed Fouad | LinkedIn

Nokhba Academy FB page

https://shorturl.at/UUDf2

1. Your company recently developed an innovative mobile game that is expected to attract massive user engagement right after launch. You decide to utilize Azure as the hosting platform and wish to minimize costs while maximizing reliability and scalability. What should you do to deploy the web backend supporting the game?

A. Host the backend on an App Service with the Free tier

B. Utilize a virtual machine with automatic scaling enabled

C. Launch the backend on a dedicated App Service environment

D. Distribute the workload with Azure Container Instances

Answer: B

Feedback (if correct):

A virtual machine with automatic scaling meets the requirement for minimizing costs while preserving reliability and scalability. With this setup, you can accommodate varying traffic levels dynamically, preventing wasteful expenses and downtime.

Feedback (if wrong):

Choosing the Free tier of Azure App Service limits scalability and fails to deliver the SLAs necessary for reliable operation. Leveraging solely a virtual machine constrains expansion possibilities and neglects automatic scaling features offered by Azure. Preferring a dedicated App Service environment entails higher costs, potentially exceeding budget allocations. Azure Container Instances might impair operational visibility and granular controls, rendering them less desirable for mission-critical systems.

Skill Mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop Azure compute solutions

Competency: Understand Azure hosting options and scalability

Bloom's Taxonomy Level: Application

Difficulty Level: Intermediate

2. You are developing an ASP.NET Core website to manage photographs stored in Azure Blob Storage containers. Users authenticate using their Azure Active Directory (Azure AD) credentials. Role-based access control (RBAC) role permissions are implemented on the containers storing photographs, and users are assigned to RBAC roles.

You need to ensure that the website's Azure AD Application properly handles the "tip" property in Cosmos DB. The "tip" property should exist on the document and contain a numeric value. However, there are existing websites and mobile apps that will not immediately set the "tip" property.

Which steps should you take to complete the trigger function in Azure Cosmos DB and ensure the proper handling of the "tip" property? (Select two or more options.)

A) Check if the "tip" property exists in the document, and if not, set it to null.

B) Validate if the "tip" property is not present or is null, and if so, set it to a default value of 0.

C) Use the getContext().getRequest() function to retrieve the request context.

D) Use the isNaN() function to check if the "tip" property exists and has a non-numeric value. If so, set it to a chosen default numeric value (often 0) to ensure data consistency.

Answer: A,D

Feedback(if correct):-

A) Check if the "tip" property exists in the document, and if not, set it to null.

This addresses the scenario where older applications might not include the "tip" property at all. Setting it to `null` can be appropriate if your data model and downstream processing require differentiating between a missing property and a zero value.

D) Use the isNaN() function to check if the tip property exists and has a non-numeric value. If so, set it to a chosen default numeric value (often 0) to ensure data consistency.

This ensures data integrity by verifying that the "tip" property holds a valid numeric value. While setting it to `null` might be an option in some cases, choosing a default value (like 0) provides a clear and consistent representation of the tip amount, even if received from applications sending invalid data.

Feedback(if wrong):-

- B) Validate if the "tip" property is not present or is null, and if so, set it to a default value of 0.

This effectively combines aspects of A and D but explicitly mentions setting a default value (0) which might not be necessary if null differentiation is crucial downstream. However, it's still a valid approach depending on your specific requirements.

- C) Use the getContext().getRequest() function to retrieve the request context.

This function, while important for potentially accessing user information or other request details, is not directly related to handling the "tip" property logic within the trigger function.

skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Develop Azure compute solutions

Competencies: Implement Azure Cosmos DB, Develop for Azure Storage

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

3. You are developing an ASP.NET Core website that manages photographs stored in Azure Blob Storage containers. Users authenticate using their Azure Active Directory (Azure AD) credentials. Role-based access control (RBAC) role permissions are implemented on the containers storing photographs, and users are assigned to RBAC roles.

You need to configure the website's Azure AD Application to ensure that user permissions can be utilized with the Azure Blob containers. Which steps should you take to configure the Azure AD Application? (Select one or more options.)

A) Configure the application to use the "user_impersonation" permission.

B) Configure the application to use the "delegated" permission.

C) Use the getContext().getRequest() function to retrieve the request context.

D) Implement the isNaN() function to check if the "tip" property is not a number, and if so, set it to null.

Answer: B

Feedback(if correct):

The correct answer is B) Configure the application to use the "delegated" permission. This is because when users authenticate using their Azure AD credentials, the application needs to use delegated permissions to access Azure Blob Storage on behalf of the signed-in user, allowing the website to utilize user permissions with the Azure Blob containers.

Feedback(if wrong):

Option A is incorrect because "user_impersonation" permission is typically used for scenarios where the application needs to access resources on behalf of the signed-in user but not necessarily within the context of Azure Blob Storage permissions.

Option C is incorrect as it refers to a JavaScript function getContext().getRequest(), which is unrelated to configuring the Azure AD Application.

Option D is incorrect as it pertains to implementing logic within the application's code to handle a specific property in Cosmos DB, which is not related to configuring the Azure AD Application for Azure Blob Storage permissions.

Skill Mapping:

Skill: Azure Developer Certification (AZ-204)

Subskills: Develop Azure compute solutions

Competencies: Choose the appropriate Azure services to meet scalability, availability, and cost requirements for web applications

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

4. You are tasked with securing the Shipping Function app, and ensuring secure function endpoints using app-level security with Azure Active Directory (Azure AD). How should you configure the app?

A) Authorization level: Anonymous, User claims: API Key, Trigger type: Blob

B) Authorization level: Admin, User claims: Shared Access Signature (SAS) token, Trigger type: Queue

C) Authorization level: Function, User claims: JSON Web Token (JWT), Trigger type: HTTP

D) Authorization level: Admin, User claims: JSON Web Token (JWT), Trigger type: Timer

Answer: C

Feedback(if correct): Option C is correct because it configures the Shipping Function app to use app-level security with Azure AD, ensuring that function endpoints are secured, user claims are authenticated using JSON Web Tokens (JWTs), and the trigger type is set to HTTP, allowing authentication via Azure AD and token acquisition through standard HTTP protocol messages.

Feedback(if wrong):

- Option A is incorrect because it sets the authorization level to Anonymous, which does not provide the required security for the Shipping Function app.

- Option B is incorrect because it uses Shared Access Signature (SAS) tokens for user claims, which is not suitable for Azure AD authentication.

- Option D is incorrect because it sets the trigger type to Timer, which is not appropriate for securing function endpoints with Azure AD authentication.

Skill mapping:

- Skill: Azure Developer Certification (AZ-204)

- Subskill: Develop Azure compute solutions

- Competencies: Choose the appropriate Azure services to meet scalability, availability, and cost requirements for web applications

- Difficulty Level: Intermediate

- Bloom's Taxonomy Level: Application

5. You are developing a secure Shipping Function app that integrates with Azure Active Directory (Azure AD) for authentication. How should you configure the app? Choose the appropriate configuration settings from the options below:

A) Authorization level: Anonymous, User claims: API Key, Trigger type: Blob

B) Authorization level: Function, User claims: JSON Web Token (JWT), Trigger type: HTTP

C) Authorization level: Admin, User claims: Shared Access Signature (SAS) token, Trigger type: Queue

D) Authorization level: Admin, User claims: JSON Web Token (JWT), Trigger type: Timer

Answer: B

Feedback(if correct):

Option B is the correct configuration for the Shipping Function app. Configuring the authorization level as "Function" ensures that only authenticated users can access the function, using JSON Web Token (JWT) for user claims ensures secure authentication through Azure AD, and HTTP is a suitable trigger type for web-based function endpoints.

Feedback(if wrong):

- Option A sets the authorization level to "Anonymous," which allows unauthenticated access, which is not suitable for a secure function app. Additionally, using API Key for user claims is not recommended for Azure AD authentication, and Blob is not typically used as a trigger type for function apps.

- Option C uses a Shared Access Signature (SAS) token for user claims, which is not typically used for Azure AD authentication. Additionally, Queue is not a suitable trigger type for HTTP-based function endpoints.

- Option D sets the authorization level to "Admin," which is not recommended for user-level authentication in an Azure AD-integrated scenario. While using JSON Web Token (JWT) for user claims is appropriate, Timer is not a suitable trigger type for web-based function endpoints.

Skill mapping:

- Skills: Azure Developer Certification (AZ-204)

- Subskills: Develop Azure compute solutions

- Competencies: Secure Azure solutions, Azure Active Directory integration

- Difficulty Level: Intermediate

- Bloom's Taxonomy Level: Application

6. You are developing an Azure Batch project that processes and converts files, storing them in Azure storage. You are tasked with creating a function to initiate the batch job. You need to ensure that the processed files are correctly stored in the designated output container and that failed files are stored separately. Which options should you select to complete the code segment?
   Options:

A) ValidateJob

B) CreateJob

C) StartJob

D) ExecuteJob

Answer: B

Feedback(if correct): CreateJob is correct because it is used to create an Azure Batch job, which is necessary to initiate the batch processing of files.

Feedback(if wrong):

Option A) ValidateJob is incorrect because it is not used to initiate the execution of a job; it is typically used to check the validity of a job's configuration before creation.

Option C) StartJob is incorrect because it is not a standard Azure Batch operation; typically, you would use "SubmitJob" or "AddTask" to initiate the job execution.

Option D) ExecuteJob is incorrect because it is not a standard Azure Batch operation; typically, you would use "SubmitJob" or "AddTask" to initiate the job execution.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskill: Develop Azure Compute Solutions

Competency: Azure Batch

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

7. In an Azure Batch project for file processing and conversion, you are developing a function to start the batch job. Your task is to ensure that the converted files are stored in the output container specified and the failed files are stored in a separate container. What should be the appropriate options to complete the code segment?

A) GetJob

B) CreateJob

C) RunJob

D) CompleteJob

Answer: B

Feedback(if correct): CreateJob is correct. This option is appropriate for creating a new Azure Batch job, which is necessary for initiating the batch job for file processing and conversion.

Feedback(if wrong):

A) GetJob: This option is incorrect because it is used to retrieve information about an existing Azure Batch job, not to initiate a new job.

C) RunJob: This option is incorrect because there is no direct method called RunJob in Azure Batch. Initiating a job typically involves creating a new job, not running an existing one.

D) CompleteJob: This option is incorrect because it is used to mark a job as completed in Azure Batch after all tasks associated with the job have finished execution. It is not used for initiating a new job.

Skill Mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop Azure Compute Solutions

Competency: Azure Compute Solutions

Bloom's Taxonomy Level: Application

8. What is the first step to take when developing the function responsible for initiating the Azure Batch job while ensuring accurate file processing?

A) Create the Batch client object using the provided BatchSharedKeyCredentials.

B) Iterate through the file task inventory, creating CloudTask instances for each task, and assigning unique task IDs along with command lines.

C) Establish a CloudJob instance and configure its attributes, such as the job ID and pool details.

D) Define the collection of output files for successful task execution and specify the destination container for the output.

Answer: C

Feedback(if correct): The correct answer is C) Establish a CloudJob instance and configure its attributes, such as the job ID and pool details. This is the first step in developing the function responsible for initiating the Azure Batch job because it involves setting up the job parameters and defining its properties, including the job ID and pool details, which are essential for executing the job on the Azure Batch service.

Feedback(if wrong):

A) Create the Batch client object using the provided BatchSharedKeyCredentials. This option is incorrect because establishing the Batch client object is not the initial step; instead, it comes after configuring the CloudJob instance.

B) Iterating through the file task inventory and creating CloudTask instances for each task is not the first step. It follows after configuring the CloudJob instance and defining its attributes.

D) Defining the collection of output files and specifying the destination container is not the first step in initiating an Azure Batch job. It is part of the job configuration but comes after establishing the CloudJob instance and configuring its attributes.

Skill Mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop Azure Compute Solutions

Competency: Azure Compute Solutions

Bloom's Taxonomy Level: Application

Difficulty Level: Intermediate

9. You are developing a function to manage Azure Batch jobs for file processing. After submitting the job to the Azure Batch service, what action should you take next to ensure proper handling of the processed files?

```csharp
// Submitting the job to Azure Batch service

BatchClient batchClient = BatchClient.Open(sharedKeyCredentials);

CloudJob job = batchClient.JobOperations.CreateJob();

// Code to submit tasks and define job properties goes here...

job.Commit();
```

What should be the next step after committing the job to the Batch service?

A) Define the storage location for successfully processed files.

B) Determine the storage location for files that failed to process.

C) Attach the list of output files to the corresponding task objects.

D) Return the list of task objects from the function for further processing.

Answer: C

Feedback(if correct): After committing the job to the Batch service, the next step is to attach the list of output files to the corresponding task objects. This ensures that the processed files are correctly associated with their respective tasks and can be managed accordingly.

Feedback(if wrong):

A) Incorrect. Defining the storage location for successfully processed files should be done before committing the job, not after.

B) Incorrect. Determining the storage location for files that failed to process should also be handled before committing the job, as part of the job configuration.

D) Incorrect. Returning the list of task objects from the function for further processing is not necessary after committing the job. The focus should be on managing the output files within the job execution context.

Skill Mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop Azure Compute Solutions

Competency: Azure Compute Solutions

Bloom's Taxonomy Level: Application

Difficulty Level: Intermediate

10. You are working on an Azure Batch project for processing large volumes of image data. Your task is to develop a function responsible for initiating batch jobs to perform image processing tasks. The function needs to ensure that the processed images are stored correctly and that any failed processing tasks are handled appropriately. As part of the function development, you have written code to submit the job to the Azure Batch service. After submitting the job, you need to determine the next steps to ensure the proper handling of the processed files.

Code Snippet:

```csharp
// Submitting the job to Azure Batch service

BatchClient batchClient = BatchClient.Open(sharedKeyCredentials);

CloudJob job = batchClient.JobOperations.CreateJob();

// Code to submit tasks and define job properties goes here...

job.Commit();
```

Your task is to identify the subsequent action that should be taken after committing the job to the Batch service to guarantee the correct handling of the processed files.

A) Configure the storage location for successfully processed images.

B) Identify the storage location for images that failed processing.

C) Attach the list of output files to the corresponding task objects.

D) Retrieve the status of the submitted job for monitoring purposes.

Answer: D

Feedback(if correct):-

After committing the job to the Batch service, the next step should be to retrieve the status of the submitted job for monitoring purposes. This ensures that you can track the progress of the job and handle any issues that may arise during processing.

Feedback(if wrong):-
A) Configure the storage location for successfully processed images.

- This step typically occurs before submitting the tasks to Azure Batch, not after committing the job.

B) Identify the storage location for images that failed processing.

- While identifying the storage location for failed images is important, it is not the immediate next step after committing the job.

C) Attach the list of output files to the corresponding task objects.

- Attaching the list of output files to task objects is typically done before submitting the tasks, not after committing the job.

Skill Mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop Azure Compute Solutions

Competency: Azure Compute Solutions

Bloom's Taxonomy Level: Application

Difficulty Level: Intermediate

11. You are working on an Azure Batch project aimed at processing large volumes of image data. Your current task involves developing a function responsible for initiating batch jobs to execute image processing tasks. It's imperative that the function guarantees the correct storage of processed images and appropriately handles any failed processing tasks.

Code Snippet:

```csharp

// Submitting the job to Azure Batch service

BatchClient batchClient = BatchClient.Open(sharedKeyCredentials);

CloudJob job = batchClient.JobOperations.CreateJob();

// Additional code to submit tasks and define job properties goes here...
```

job.Commit();

```

After defining the list of output files for successful task completion and specifying the output container destination, what should be the next step in implementing the function to ensure correct file processing?

A) Define the list of output files for failed task completion and specify the failed container destination.

B) Add the output file lists to the corresponding CloudTask objects.

C) Return the list of CloudTask objects.

D) Create the Batch client object using the provided BatchSharedKeyCredentials.

Answer: B

Feedback(if correct): The correct answer is B) Add the output file lists to the corresponding CloudTask objects. After specifying the output container destination for successful task completion, the next step is to associate the output file lists with their respective CloudTask objects, ensuring that each task's output is properly handled.

Feedback(if wrong):

A) Incorrect. Defining the list of output files for failed task completion should be handled separately and is not the immediate next step after specifying the output container destination for successful task completion.

C) Incorrect. Returning the list of CloudTask objects is not the immediate next step after specifying the output container destination for successful task completion. This step would be more relevant before submitting the job to the Azure Batch service.

D) Incorrect. Creating the Batch client object using the provided BatchSharedKeyCredentials has already been done before submitting the job to the Azure Batch service and is not relevant to the subsequent steps after specifying the output container destination.

Skill Mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop Azure Compute Solutions

Competency: Azure Compute Solutions

Bloom's Taxonomy Level: Application

Difficulty Level: Intermediate
```

12. You are working on an Azure Batch project aimed at processing large volumes of image data. Your current task involves developing a function responsible for initiating batch jobs to execute image processing tasks. It's imperative that the function guarantees the correct storage of processed images and appropriately handles any failed processing tasks.

Code Snippet:

```csharp
// Submitting the job to Azure Batch service

BatchClient batchClient = BatchClient.Open(sharedKeyCredentials);

CloudJob job = batchClient.JobOperations.CreateJob();

// Additional code to submit tasks and define job properties goes here...

job.Commit();
```

After adding the output file lists to the corresponding CloudTask objects, what should be the last step in implementing the function that starts the Azure Batch job and ensures correct file processing?

A) Create a CloudJob object and define its properties, including the job ID and pool information.

B) Commit the job to the Batch service.

C) Return the list of CloudTask objects.

D) Define the list of output files for successful task completion and specify the output container destination.

Answer: B

Feedback(if correct):-

The correct answer is B) Commit the job to the Batch service. Once the output file lists are added to the corresponding CloudTask objects, the final step is to commit the job to the Batch service, ensuring that the tasks are executed according to the defined properties and the processed files are stored appropriately.

Feedback(if wrong):-

Option A is incorrect because creating a CloudJob object and defining its properties should be done before committing the job to the Batch service, not after adding the output file lists to the CloudTask objects.

Option C is incorrect because returning the list of CloudTask objects is not the final step in ensuring correct file processing. The CloudTask objects have already been configured with output file lists, and the job needs to be committed to the Batch service for execution.

Option D is incorrect because defining the list of output files for successful task completion and specifying the output container destination should be done before adding the output file lists to the CloudTask objects.

Skill Mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop Azure Compute Solutions

Competency: Azure Compute Solutions

Bloom's Taxonomy Level: Application

Difficulty Level: Intermediate

13. You are architecting a cloud-based system for a media streaming service. The system needs to support real-time notifications to users whenever new content is available without continuously querying the server. Which Azure services should you consider to fulfill this requirement?

A) Azure Kubernetes Service (AKS)

B) Azure Queue Storage

C) Azure Event Grid

D) Azure Virtual Machines (VMs)

Answer: C

Feedback(if correct): Azure Event Grid is the optimal choice for real-time notifications as it supports the publish-subscribe pattern, enabling efficient event-driven architectures without the need for continuous polling.

Azure Event Grid is a fully managed event routing service that enables you to react to events across many Azure services and third-party sources. It supports the publish-subscribe pattern, making it suitable for real-time notifications whenever new content is available.

Feedback(if wrong):

A) Azure Kubernetes Service (AKS): While AKS is a managed Kubernetes service that enables you to deploy, manage, and scale containerized applications, it does not inherently support real-time notifications or event-driven architectures.

B) Azure Queue Storage: Azure Queue Storage is a service for storing large numbers of messages that can be accessed from anywhere in the world via authenticated calls. However, it does not provide real-time notifications or support the publish-subscribe pattern required for this scenario.

D) Azure Virtual Machines (VMs): Azure VMs provide on-demand scalable computing resources, but they do not offer built-in capabilities for real-time event processing or notifications. They are more suited for traditional computing tasks rather than event-driven architectures.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Develop Azure compute solutions

Competencies: Event-driven architectures, Azure messaging services

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

14. A company plans to deploy a new application on an Azure Linux virtual machine (VM) and requires the entire VM to be encrypted at rest to comply with organizational security standards. Which commands should the company use to achieve this?

A) Register the Microsoft.Keyvault provider

B) Create an Azure Key Vault with disk encryption enabled

C) Create a new Azure resource group

D) Provision a new Azure virtual machine with an Ubuntu image

Answer: A, B

Feedback(if correct):

Option A: Registering the Microsoft.Keyvault provider is necessary to enable the use of Azure Key Vault, which is essential for managing encryption keys.

Option B: Creating an Azure Key Vault with disk encryption enabled is crucial for storing encryption keys securely and enabling disk encryption for the VM.

Feedback(if wrong):

Option C: Creating a new Azure resource group is not directly related to enabling encryption on the VM. While resource groups are used to logically organize Azure resources, they do not play a direct role in enabling encryption.

Option D: Provisioning a new Azure virtual machine with the Ubuntu image is essential for deploying the VM, but it is not directly related to enabling disk encryption. Disk encryption can be configured during or after VM provisioning, but the act of provisioning the VM itself does not directly involve enabling encryption.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Develop Azure compute solutions

Competencies: Virtual Machines, Azure Disk Encryption:

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

15. A company plans to deploy a new application on an Azure Linux virtual machine (VM) and requires the entire VM to be encrypted at rest to comply with organizational security standards. Which additional commands should the company use to achieve this? Select the appropriate options from the following.

A) Generate SSH keys for secure access to the VM

B) Create a cryptographic key within the Key Vault

C) Provision the Ubuntu 16.04 LTS image for the VM

D) Enable encryption for the VM's disks

Answer: B, D

Feedback(if correct):-

B) Create a cryptographic key within the Key Vault: This is necessary as Azure Disk Encryption for Linux relies on encryption keys stored in Azure Key Vault to encrypt the VM disks. Thus, creating a cryptographic key within the Key Vault is a crucial step.

D) Enable encryption for the VM's disks:** This is essential as it activates the disk encryption feature for the VM, ensuring that the entire VM's disks are encrypted at rest to comply with organizational security standards.

Feedback(if wrong):-

A) Generate SSH keys for secure access to the VM:** While SSH keys are important for secure access to the VM, they are not directly related to encrypting the VM's disks at rest. SSH keys are used for authentication purposes and do not affect disk encryption.

C) Provision the Ubuntu 16.04 LTS image for the VM:** The choice of the operating system image does not directly impact disk encryption. While Azure Disk Encryption for Linux supports specific Linux distributions, provisioning the Ubuntu 16.04 LTS image does not directly contribute to encrypting the VM's disks at rest.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Develop Azure compute solutions

Competencies: Virtual Machines, Azure Disk Encryption:

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

16. A company plans to deploy a new application on an Azure Linux virtual machine (VM) and requires the entire VM to be encrypted at rest to comply with organizational security standards. Which final commands should the company use to achieve this? Select the appropriate options from the following.

A) Create a new Azure resource group

B) Enable encryption for the virtual machine's disks

C) Define the Ubuntu 16.04 LTS image for the VM

D) Register the Microsoft.Keyvault provider

Answer: B, D

Feedback(if correct):

Option B (Enable encryption for the virtual machine's disks) is necessary to activate disk encryption for the VM, ensuring that all disks associated with the VM are encrypted at rest to comply with organizational security standards.

Option D (Register the Microsoft.Keyvault provider) is crucial for accessing and utilizing Azure Key Vault, which is required for storing the encryption keys used by Azure Disk Encryption to encrypt the VM's disks.

Feedback(if wrong):

Option A (Create a new Azure resource group) is unrelated to enabling encryption for the VM's disks. While creating a new resource group may be necessary for organizing Azure resources, it does not directly impact the encryption process.

Option C (Define the Ubuntu 16.04 LTS image for the VM) specifies the operating system image to be used for the VM but does not directly affect the encryption of the VM's disks. Choosing the Ubuntu 16.04 LTS image is important for the VM configuration but not for enabling disk encryption.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Develop Azure compute solutions

Competencies: Virtual Machines, Azure Disk Encryption:

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

17. You are designing a healthcare application with specific consistency requirements. You have provisioned an Azure Cosmos DB NoSQL database with a default consistency level set to Strong and Indexing Mode set to Consistent. To meet the application requirements while minimizing latency and ensuring availability, you need to override the default consistency level at the query level. Match the appropriate consistency levels to the corresponding requirements:

Provides a consistency level that is slightly less strict than Strong, allowing the application to get data that is no more than a certain amount of time or versions behind the most recent data.

A) Strong

B) Bounded Staleness:

C) Eventual

D) Consistent Prefix

Answer: B

Feedback(if correct):

Option B (Bounded Staleness) is correct. Bounded Staleness provides a consistency level slightly less strict than Strong by allowing the application to retrieve data that is no more than a certain amount of time or versions behind the most recent data, which aligns with the specified requirements.

Feedback(if wrong):

A) Strong: Strong consistency provides the strongest consistency level, ensuring that the application always receives the most recent data, but it doesn't allow for a certain amount of lag in data retrieval.

C) Eventual: Eventual consistency does not provide strong guarantees about the recency of the data and may result in receiving outdated information, which doesn't meet the requirement.

D) Consistent Prefix: While Consistent Prefix guarantees that the reads honor the order of writes, it doesn't ensure receiving the most recent data within a certain time frame or versions behind, which doesn't align with the requirement.

Skill mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop for Azure storage.

Competency: Develop Azure Cosmos DB solutions, Choose appropriate consistency levels for Azure Cosmos DB

Difficulty Level: Intermediate

Blooms Taxonomy Level: Application

18. You are designing a healthcare application with specific consistency requirements. You have provisioned an Azure Cosmos DB NoSQL database with a default consistency level set to Strong and Indexing Mode set to Consistent. To meet the application requirements while minimizing latency and ensuring availability, you need to override the default consistency level at the query level. Match the appropriate consistency levels to the corresponding requirements: Provides the least strict consistency level and ensures that the application will eventually get the most recent data, but there may be a delay in getting that data.

A) Strong

B) Bounded Staleness:

C) Eventual

D) Consistent Prefix

Answer: C

Feedback(if correct):

Eventual consistency provides the least strict consistency level, ensuring that the application eventually gets the most recent data, albeit with a potential delay, which aligns with the specified requirements.

Feedback(if wrong):

A) Strong: Strong consistency provides the strongest consistency level, ensuring that the application always receives the most recent data, which doesn't align with the requirement for potential delays.

B) Bounded Staleness: Bounded Staleness provides a consistency level slightly less strict than Strong by allowing the application to retrieve data within a specified lag limit, which doesn't align with the requirement for eventual consistency and potential delays.

D) Consistent Prefix: Consistent Prefix guarantees that read honor the order of writes but does not guarantee eventual consistency or potential delays in data retrieval, which doesn't align with the requirement for eventual consistency.

Skill mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop for Azure storage.

Competency: Develop Azure Cosmos DB solutions, Choose appropriate consistency levels for Azure Cosmos DB

Difficulty Level: Intermediate

Blooms Taxonomy Level: Application

19. You are tasked with deploying a medical records application to an Azure virtual machine (VM). The application's virtual hard disk (VHD) is generated by an on-premises build server. It's crucial to ensure that both the application and its related data are encrypted both during and after deployment to Azure. Which sequence of actions should you perform to achieve this?

A. Encrypt the on-premises VHD by using BitLocker without a TPM. Upload the VM to Azure Storage. Run the Azure PowerShell command Set-AzureRMVMOSDisk.

B. Encrypt the on-premises VHD by using BitLocker without a TPM. Upload the VM to Azure Storage. Run the Azure PowerShell command Set-AzureRmVMDiskEncryptionExtension.

C. Encrypt the on-premises VHD by using BitLocker without a TPM. Upload the VM to Azure Storage. Run the Azure PowerShell command Set-AzureNewDiskEncryptionExtension.

D. Encrypt the on-premises VHD by using BitLocker without a TPM. Upload the VM to Azure Storage. Run the Azure PowerShell command Set-AzureNewVMOSDisk.

Answer: B

Feedback(if correct):-

The correct sequence of actions to achieve encryption for the medical records application in Azure VM is:

B. Encrypt the on-premises VHD by using BitLocker without a TPM. Upload the VM to Azure Storage. Run the Azure PowerShell command Set-AzureRmVMDiskEncryptionExtension.

This sequence ensures that the on-premises VHD is encrypted using BitLocker without a TPM, and then the VM is uploaded to Azure Storage. Finally, the Azure PowerShell command Set-AzureRmVMDiskEncryptionExtension is run to enable encryption for the VM disk in Azure.

Step 1: Encrypt the on-premises VHD by using BitLocker without a TPM. Upload the VM to Azure Storage.

Step 2: Run the Azure PowerShell command Set-AzureRMVMOSDisk to configure the OS disk for the VM.

Step 3: Run the Azure PowerShell command Set-AzureRmVMDiskEncryptionExtension to enable encryption for the VM disk.

Feedback(if wrong):-

Option A is incorrect because the command Set-AzureRMVMOSDisk is used to set the operating system disk for an Azure VM, not to enable encryption.

Option C is incorrect because there is no command named Set-AzureNewDiskEncryptionExtension in Azure PowerShell.

Option D is incorrect because there is no command named Set-AzureNewVMOSDisk in Azure PowerShell.

Skill mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop for Azure storage.

Competency: Develop for Azure storage

Difficulty Level: Intermediate

Blooms Taxonomy Level: Application

20. You are developing an Azure Cosmos DB solution using the Azure Cosmos DB SQL API. The data consists of millions of documents, each potentially containing hundreds of properties. These properties do not have distinct values for partitioning, and the workload needs to be evenly spread across partitions to meet performance requirements. What should you consider when selecting a partition key?

A. Even distribution
B. No distinct values
C. Include a value containing the collection name in the partition key
D. Include a value containing the collection name in the data key

Answer: A, B

Feedback(if correct):

When selecting a partition key for an Azure Cosmos DB solution using the Azure Cosmos DB SQL API, several factors need consideration to ensure optimal performance and scalability. In this scenario, where the data consists of millions of documents with potentially hundreds of properties and no distinct values for partitioning, the following considerations should be taken into account:

A. Even distribution: Ensure that the partition key allows for even distribution of data across partitions. This helps distribute the workload evenly and prevents hot partitions, which can degrade performance.

B. No distinct values: Since there are no distinct values available for partitioning, it's essential to choose a partition key that provides a good distribution of data without creating hot partitions. Avoiding distinct values helps evenly distribute the workload across partitions.

Feedback(if wrong):

Option C: Including a value containing the collection name in the partition key is not necessary for achieving an even distribution of data across partitions. While it might be beneficial in some scenarios for organizational purposes or query optimization, it is not a primary consideration when selecting a partition key based on the scenario provided. This option does not directly address the requirement of evenly spreading the workload across partitions.

Option D: Including a value containing the collection name in the data key does not directly relate to the selection of a partition key. The partition key is used to distribute data across physical partitions in Azure Cosmos DB, while the data key typically refers to a unique identifier or field within the document itself. This option does not align with the requirement of selecting an appropriate partition key for achieving an even distribution of workload across partitions.

Skill mapping:

Skill: Azure Developer Certification (AZ-204)

Subskill: Develop for Azure storage.

Competency: Develop for Azure storage

Difficulty Level: Intermediate

Blooms Taxonomy Level: Application

21. You are developing an application that utilizes Azure Blob Storage to store application data, including snapshots for reverting to earlier states. The Azure storage account is configured with soft delete enabled. After performing a sequence of operations on the blob and its snapshots, a system error results in the deletion of the data blob and all snapshots. You need to determine which application states can be restored. Which of the following application states can be restored? Choose the appropriate options from the answer choices below:

A) Data blob: Can be restored

B) Snapshot 1: Cannot be restored

C) Snapshot 2: Can be restored

D) Data blob: Cannot be restored

Answer: A, C

Feedback(if correct):

Both options A and C are correct because they were not explicitly deleted at the time of the system error, and soft delete allows for the recovery of data when blobs or blob snapshots are deleted.

Feedback(if wrong):

Option B is incorrect because Snapshot 1 was deleted before the system error occurred, and soft delete does not apply retroactively to recover deleted items. Option D is incorrect because the data blob can be restored even though it was deleted due to the soft delete feature.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskill: Develop for Azure storage

Competency: Azure Storage

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Analysis

22. Your organization is migrating from an on-premises search solution to Azure Search. After the migration, users report that they cannot search for documents by metadata properties. What adjustments can you make to enable searching by metadata in Azure Search?

A) Ensure that the metadata properties are correctly defined in the index schema, including their data types and attributes.

B) Implement a custom indexer that retrieves metadata from the source documents and maps it to fields in the Azure Search index.

C) Use the Azure Search REST API to configure field mappings and boost the relevance of search results based on metadata properties.

D) Enable the "Retrieve Metadata" option in the Azure Search indexer configuration to ensure that metadata properties are included in the indexed documents for search queries.

Answer: A

Feedback(if correct): Option A is correct. It is essential to ensure that the metadata properties are correctly defined in the index schema, including their data types and attributes. This ensures that the search engine recognizes and indexes these properties, allowing users to search for documents based on metadata.

Feedback(if wrong):

B) Implementing a custom indexer may be a solution, but it is not directly related to enabling searching by metadata. This option introduces additional complexity and may not address the core issue.

C) While using the Azure Search REST API can be a valid approach for configuring field mappings and boosting search relevance, it does not directly address the problem of enabling searching by metadata.

D) Enabling the "Retrieve Metadata" option in the Azure Search indexer configuration retrieves metadata from the source documents but does not ensure that users can search for documents by metadata properties. This option may help gather metadata but does not directly enable searching by metadata.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Develop for Azure storage

Competencies: Blob storage, Table storage, Queue storage, File storage

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

23. You are tasked with securing the Shipping Function app, and ensuring secure function endpoints using app-level security with Azure Active Directory (Azure AD). How should you configure the app? Select the appropriate options from the answer choices below:

A) Authorization level: Anonymous, User claims: API Key, Trigger type: Blob

B) Authorization level: Function, User claims: JSON Web Token (JWT), Trigger type: HTTP

C) Authorization level: Admin, User claims: Shared Access Signature (SAS) token, Trigger type: Queue

D) Authorization level: Admin, User claims: JSON Web Token (JWT), Trigger type: Timer

Answer: B

Feedback(if correct): This configuration ensures that the Shipping Function app is secured with Azure AD app-level security, using JSON Web Tokens (JWTs) for user claims and HTTP as the trigger type. This aligns with the scenario requirement of implementing secure function endpoints.

Feedback(if wrong):

Option A (Authorization level: Anonymous, User claims: API Key, Trigger type: Blob) is incorrect because it does not utilize Azure AD for authentication and lacks the necessary security measures for the scenario.

Option C (Authorization level: Admin, User claims: Shared Access Signature (SAS) token, Trigger type: Queue) is incorrect because it does not involve Azure AD authentication and is not suitable for securing function endpoints with app-level security.

Option D (Authorization level: Admin, User claims: JSON Web Token (JWT), Trigger type: Timer) is incorrect because although it uses JWT for user claims, it does not specify the correct authorization level or trigger type for securing function endpoints with Azure AD app-level security.

Skill Mapping:

- Skills: Azure Developer Certification (AZ-204)

- Subskills: Implement Azure security

- Competencies: Azure Active Directory (Azure AD), Multifactor Authentication (MFA), Conditional Access Policies

- Difficulty Level: Intermediate

- Bloom's Taxonomy Level: Application

24. You are developing an Azure Function app that integrates with Azure Active Directory (Azure AD) for user authentication. The app will have different access levels based on the user's Azure AD group membership: admin, member, and guest. You configure the Azure AD application manifest to include group membership claims in the authentication token. Additionally, you use the group claims from the JWT (JSON Web Token) to determine user permissions within the function app. Does the solution meet the goal?

A. Yes

B. No

Answer: A

Feedback(if correct):-

The provided description suggests that you have developed an Azure Function app that integrates with Azure Active Directory (Azure AD) for user authentication and incorporated group membership claims in the authentication token. By using the group claims from the JWT (JSON Web Token) to determine user permissions within the function app, you ensure that the app adheres to the appropriate access levels (admin, member, and guest) based on the user's Azure AD group membership. Overall, the solution meets the stated goal.

Feedback(if wrong):-

The solution presented in the scenario does meet the goal because the developer has covered all the necessary steps for implementing access restrictions based on Azure AD group membership. The Azure AD application manifest has been updated to include group membership claims in the authentication token, and the Azure AD group claims are being used within the function app to determine user permissions. Allowing or denying access based on the received group claims will ensure appropriate separation of duties without needing frequent manual intervention.

Skill Mapping:

- Skills: Azure Developer Certification (AZ-204)

- Subskills: Implement Azure security

- Competencies: Azure Active Directory (Azure AD), Multifactor Authentication (MFA), Conditional Access Policies

- Difficulty Level: Intermediate

- Bloom's Taxonomy Level: Application

25. You are designing a web application hosted on Azure that requires user authentication through Azure Active Directory (Azure AD). The application will grant different levels of access to users based on their Azure AD group membership: admin, user, and guest. To implement this, you create an Azure AD application and configure its manifest to include group membership claims in the authentication token. Subsequently, you use the group claims from the JWT (JSON Web Token) to determine user permissions within the application. Does the solution meet the goal?

A. Yes

B. No

Answer: B

Feedback(if correct):-

the Azure AD application is configured to include group membership claims, there is no mention of using these claims within the application to determine user permissions. Therefore, the solution does not meet the goal.

Without integrating the group claims from the JWT into the application for assigning user permissions, users cannot be granted different access levels based on their Azure AD group membership.

Feedback(if wrong):-

While the initial steps (Azure AD authentication and group claims in JWT) are a good foundation, there's no mention of utilizing these claims for authorization. Without leveraging the group membership information to control access levels, the application wouldn't differentiate user permissions based on their Azure AD groups. The group claims would be present in the JWT but not used for authorization decisions.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Implement Azure security

Competencies: Azure Active Directory (Azure AD), Multifactor Authentication (MFA), Conditional Access Policies

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

26. You are configuring an Azure AD Application for an ASP.NET Core website that manages photographs stored in Azure Blob Storage containers. Users authenticate using Azure AD credentials, and RBAC role permissions are implemented on the containers to control access. You need to grant the website the minimum necessary permissions to interact with Azure Blob containers based on user roles. Which Azure AD Application permission(s) should you configure to achieve this?

A) user_impersonation + Azure Storage (Delegated) - Full access to Storage.Blobs API

B) Microsoft Graph (Delegated) - Read user roles + Azure Storage (Delegated) - Read access to Storage.Blobs API

C) profile + Azure Storage (Delegated) - Write access to Storage.Blobs API

D) API - Azure Storage + Type - Web App

Answer: B

Feedback(if correct):-

Microsoft Graph (Delegated) - Read user roles: This grants the website access to read the user's assigned roles within your Azure AD tenant. This information is crucial for determining the user's permissions for accessing specific Blob containers based on RBAC assignments.

Azure Storage (Delegated) - Read access to Storage.Blobs API: This grants the website the minimum necessary permission to interact with Blob containers, allowing it to read, list, and download photos based on the user's RBAC role.

Feedback(if wrong):-

A) user_impersonation + Azure Storage (Delegated) - Full access to Storage.Blobs API:

While "user_impersonation" allows acting on the user's behalf, it's not necessary here. "Delegated" permissions are sufficient for reading user roles and accessing blobs based on those roles.

"Full access" to Storage. Blobs API grants excessive permissions, exceeding the requirement of minimum necessary access.

C) profile + Azure Storage (Delegated) - Write access to Storage.Blobs API:

"profile" permission doesn't provide group membership information needed for RBAC.

"Write" access to Storage. Blobs API might not be necessary depending on the website's functionality (e.g. if it only needs to display photos).

D) API - Azure Storage + Type - Web App:

This option doesn't specify the actual permissions. "API - Azure Storage" is a broad category, and "Type - Web App" doesn't translate to specific permissions.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Implement Azure security

Competencies: Azure Active Directory (Azure AD), Multifactor Authentication (MFA), Conditional Access Policies, Azure Services Integration, Azure Blob Storage, Azure Active Directory (Azure AD), Azure AD Application Configuration, Azure RBAC (Role-Based Access Control), Azure API Permissions

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

27. A weather monitoring system generates alerts based on temperature, humidity, and wind speed measurements collected from IoT devices. Alert notifications are sent using Azure Event Grid, and the notification service resides in an Azure App Service. To ensure high availability and responsiveness during spikes in events, you need to create a scaling strategy for the Azure App Service. Given the scenario, how should you configure the Scale rule to support continuous scaling down once the scaling up condition is no longer met?

A) Scale Rule

Trigger: Service Bus Namespace

Metric: Active Message Count

Operator: Less Than or Equal To

Threshold Value: 0

Time aggregation: Average

Duration: 1 minute

Action: Decrease count by 1

Cool Down: 5 minutes

B) Scale Rule

Trigger: Storage queue

Metric: Message Count

Operator: Less Than or Equal To

Threshold Value: 0

Time aggregation: Total

Duration: 5 minutes

Action: Decrease count by 1

Cool Down: 5 minutes

C) Scale Rule

Trigger: Service Bus queue

Metric: Active Message Count

Operator: Less Than or Equal To

Threshold Value: 0

Time aggregation: Average

Duration: 1 minute

Action: Decrease count by 1

Cool Down: 5 minutes

D) Scale Rule

Trigger: Storage queue (classic)

Metric: Message Count

Operator: Less Than or Equal To

Threshold Value: 0

Time aggregation: Total

Duration: 5 minutes

Action: Decrease count by 1

Cool Down: 5 minutes

Answer: C

Feedback(if correct):-

Answer: C) Scale Rule

Trigger: Service Bus queue

Metric: Active Message Count

Operator: Less Than or Equal To

Threshold Value: 0

Time aggregation: Average

Duration: 1 minute

Action: Decrease count by 1

Cool Down: 5 minutes

Among the four options provided, option C seems to be the most fitting choice for scaling down the Azure App Service once the scaling-up condition is no longer met. Using the Service Bus queue as a trigger along with the Active Message Count metric would allow fine-grained control over scaling decisions based on the number of messages actively available in the queue. The Less Than or Equal To operator, paired with a threshold value of zero, indicates that scaling down should commence when there are no active messages left in the queue.

Using an averaged time aggregation and duration of 1 minute would help react swiftly to changes in the queue length, reducing the chance of overprovisioning compute resources. Lastly, having a cool-down period of 5 minutes would prevent abrupt fluctuations in the number of instances and promote gradual scaling adjustments, contributing to financial savings and enhanced system stability.

Feedback(if wrong):-

A) Scale Rule

Trigger: Service Bus Namespace

Metric: Active Message Count

Operator: Less Than or Equal To

Threshold Value: 0

Time aggregation: Average

Duration: 1 minute

Action: Decrease count by 1

Cool Down: 5 minutes

This option is incorrect since the Service Bus Namespace isn't representative of the queue length but rather represents the entire namespace housing several entities, including queues, topics, and subscriptions.

B) Scale Rule

Trigger: Storage queue

Metric: Message Count

Operator: Less Than or Equal To

Threshold Value: 0

Time aggregation: Total

Duration: 5 minutes

Action: Decrease count by 1

Cool Down: 5 minutes

This option is incorrect as it uses the Storage queue as a trigger instead of the Service Bus queue, which is the primary source of event data in our scenario. Moreover, the Message Count metric is less specific than the Active Message Count metric, which is more suited to capturing the currently active messages.

D) Scale Rule

Trigger: Storage queue (classic)

Metric: Message Count

Operator: Less Than or Equal To

Threshold Value: 0

Time aggregation: Total

Duration: 5 minutes

Action: Decrease count by 1

Cool Down: 5 minutes

Similar to option B, this choice uses the Storage queue (classic) as a trigger instead of the Service Bus queue, making it ill-suited for our scenario. Plus, it opts for the classic Storage queue, which is gradually being phased out in favor of newer technologies.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Implement Azure security

Competencies: Azure Key Vault, Azure Active Directory (Azure AD), Access control

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

28. You are building a web application that stores confidential data in Azure Key Vault and deploys to Azure App Service. The application must authenticate using Azure Active Directory. To ensure secure connections, you need to configure the Key Vault access policies. Which options should you choose?

A) Principal: Azure App Service | Key Permissions: Get, List, Update, Create, Delete, | Secret Permissions: Set

B) Principal: Azure Active Directory | Key Permissions: Get, List | Secret Permissions: Set

C) Principal: Azure App Service | Key Permissions: Get, List, Update, Create, Delete, Import, Backup, Restore | Secret Permissions: Get, List, Set

D) Principal: Azure Key Vault | Key Permissions: Get, List, Update, Create, Delete, Import, Backup, Restore | Secret Permissions: Get, List, Set

Answer: B

Feedback(if correct):-

Option B specifies Azure Active Directory as the principal, aligning with the requirement of authenticating using Azure AD.

It grants the necessary permissions for the web application to retrieve keys and secrets from Azure Key Vault.

The permissions provided (Get, List for keys and secrets, Set for secrets) ensure secure access to the confidential data stored in Azure Key Vault while limiting unnecessary access.

Feedback(if wrong):-

Option A grants permissions directly to Azure App Service, which is not the recommended approach for authenticating the application.

Option C provides overly permissive permissions, including update, create, delete, import, backup, and restore operations, which are unnecessary and potentially risky for an application accessing confidential data.

Option D specifies Azure Key Vault itself as the principal, which is not appropriate for granting access to a specific application or service.

Skill mapping

Skills: Azure Developer Certification (AZ-204)

Subskills: Implement Azure security

Competencies: Azure Key Vault, Azure Active Directory (Azure AD), Access control

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

29. You are developing a real-time analytics solution using Azure Event Hubs to ingest streaming data. The solution must dynamically scale based on the volume of incoming events to ensure optimal performance and cost-efficiency. What is the best approach to configure autoscaling for Azure Event Hubs?

A) Increase the number of partitions in Azure Event Hubs based on the rate of incoming events.

B) Use Azure Monitor to track the ingress and egress rates of Azure Event Hubs and manually adjust the throughput units.

C) Implement Azure Stream Analytics to monitor the event ingestion rate and automatically adjust the number of consumer instances.

D) Configure Azure Functions with a trigger from Azure Event Hubs to dynamically scale compute resources based on incoming events.

Answer: C, D

Feedback(if correct):

Option C) Implementing Azure Stream Analytics to monitor event ingestion rates and automatically adjust the number of consumer instances is a suitable approach for autoscaling. Azure Stream Analytics offers built-in functionality for dynamically scaling based on workload changes, optimizing both performance and cost-effectiveness.

Option D) Configuring Azure Functions with a trigger from Azure Event Hubs to dynamically scale compute resources based on incoming events is another viable approach. Azure Functions provide serverless computing with automatic scaling based on trigger events, making it suitable for handling varying workloads.

Feedback(if wrong):

Option A) Increasing the number of partitions in Azure Event Hubs based on the rate of incoming events is not a form of autoscaling. Partition numbers in Azure Event Hubs are fixed and predefined, and they cannot be adjusted dynamically based on workload changes.

Option B) Using Azure Monitor to track ingress and egress rates and manually adjusting throughput units lack automation and dynamic scaling based on real-time workload changes. It requires manual intervention and does not provide an efficient autoscaling solution.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Implement Azure security

Competencies: Azure Key Vault, Azure Active Directory (Azure AD), Access control

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

30. A company stores highly confidential financial reports in Azure Blob Storage. They need to ensure that only authorized personnel can access these reports. Which of the following security configurations would BEST achieve this goal?

A)

1. Configure the storage account with public read access.
2. Share the blob container URL publicly.

B)

1. Grant the "Storage Blob Contributor" role to a specific Azure Active Directory (AAD) group containing authorized employees.
2. Enable customer-managed keys for the storage account.

C)

1. Encrypt the financial reports at rest and in transit using Azure Key Vault.
2. Apply Azure Active Directory (AAD) integration for blob container access control.

D)

1. Configure static website hosting for the blob container containing the reports.
2. Grant read access to a public service principal.

Answer: C

Feedback(if correct):

Option C is the correct choice. It ensures that the financial reports are encrypted both at rest and in transit using Azure Key Vault, which provides robust security for sensitive data. Additionally, applying Azure Active Directory integration for blob container access control ensures that only authorized personnel with appropriate permissions can access the reports.

Feedback(if wrong):

Option A is incorrect because configuring the storage account with public read access and sharing the blob container URL publicly would make the financial reports accessible to anyone with the URL, posing a significant security risk.

Option B is incorrect because granting the "Storage Blob Contributor" role to an Azure Active Directory (AAD) group does not provide fine-grained access control based on individual user permissions. Additionally, enabling customer-managed keys for the storage account does not directly address access control for the financial reports.

Option D is incorrect because configuring static website hosting for the blob container and granting read access to a public service principal would make the financial reports publicly accessible, compromising their confidentiality.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Implement Azure security

Competencies: Azure SQL Database, Azure Active Directory (Azure AD), Row-Level Security

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

31. You seek to secure financially sensitive information within an Azure SQL Database table ('financial'), accessible exclusively by authorized employees. Address the challenge by performing the following steps:

1. Create an Azure Active Directory Group titled 'finance'.

2. Populate the group with eligible staff members.

3. Implements row-level security for the 'financial' table.

4. Apply row-level security permissions to the 'finance' Azure Active Directory Group.

A) ALTER ROLE db_owner ADD MEMBER finance;

B) ALTER TABLE [financial] ADD CONSTRAINT chk_employee CHECK (employee_id IN (SELECT id FROM dbo.Employees WHERE department = 'finance'));

C) EXECUTE AS USER = '<username>'; REVERT;

D) CREATE SECURITY POLICY FinancePolicy ADD FILTER PREDICATE ON finances BY employee_id WITH SCHEMABINDING = employees;

Answer: D

Feedback(if correct):

Option D) CREATE SECURITY POLICY FinancePolicy ADD FILTER PREDICATE ON finances BY employee_id WITH SCHEMABINDING = employees; is indeed the correct choice. This option creates a security policy named FinancePolicy and applies a filter predicate on the 'financial' table by the employee_id column. This ensures that only rows where the employee_id matches those in the 'finance' group are accessible, effectively implementing row-level security based on the specified criteria.

Feedback(if wrong):

Option A) ALTER ROLE db_owner ADD MEMBER finance; is incorrect because it adds the 'finance' group as a member of the db_owner role, which grants full control over the database. However, this action does not enforce row-level security.

Option B) ALTER TABLE [financial] ADD CONSTRAINT chk_employee CHECK (employee_id IN (SELECT id FROM dbo.Employees WHERE department = 'finance')); is incorrect as it adds a check constraint to the 'financial' table to restrict access based on the employee's department. However, it does not provide dynamic row-level security based on Azure Active Directory groups.

Option C) EXECUTE AS USER = '<username>'; REVERT; is incorrect because it temporarily executes statements in the context of a specified user and then reverts back to the original context. This action is not related to implementing row-level security based on Azure Active Directory groups.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Implement Azure security

Competencies: Azure SQL Database, Azure Active Directory (Azure AD), Row-Level Security

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

32. Your organization maintains customer records containing personally identifying information (PII) in an Azure SQL Database instance. Due to regulatory compliance requirements, you need to restrict access to this sensitive data. Specifically, you must block non-managerial personnel from viewing such information. Considering the given scenario, which two actions should you undertake to satisfy the requirement?

A) Modify the Azure SQL Database transparency rules to exclude manager accounts.

B) Apply Azure SQL Database Transparent Data Encryption to safeguard sensitive data.

C) Establish an Azure Active Directory Group called 'managers,' adding eligible individuals.

D) Employ Azure PowerShell to establish data masking rules for affected tables.

Answer: C, D

Feedback(if correct):-

The correct actions to undertake to satisfy the requirement are:

C) Establish an Azure Active Directory Group called 'managers,' adding eligible individuals.

Explanation: By creating an Azure Active Directory (Azure AD) group specifically for managers and adding eligible individuals to it, you can control access to sensitive data by granting permissions only to members of this group.

D) Employ Azure PowerShell to establish data masking rules for affected tables.

Explanation: Azure PowerShell can be used to create data masking rules for sensitive columns in the Azure SQL Database tables. By applying data masking, you can ensure that non-managerial personnel only see masked or obfuscated data, while managerial personnel can view the actual data.

Feedback(if correct):

Options C and D are the correct choices. Establishing an Azure AD group for managers ensures that only authorized personnel have access to sensitive data while employing data masking rules via Azure PowerShell helps protect the confidentiality of the data.

Feedback(if wrong):

Option A: Modifying transparency rules does not directly address restricting access based on managerial roles.

Option B: Transparent Data Encryption (TDE) encrypts data at rest and is not specifically tailored to restrict access based on managerial roles.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Implement Azure security

Competencies: Azure SQL Database, Azure Active Directory (Azure AD), Row-Level Security

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

33. You are tasked with deploying a new ASP.NET Core application named "MyApp" within a Docker container. The application setup requires running a script named "setupScript.sh" during the container build process and executing the application's main executable file "MyApp.exe" upon container startup. You need to create a Dockerfile to meet these requirements. Which of the following commands should you include in the Dockerfile?

A) FROM microsoft/dotnet:latest

B) WORKDIR /app/MyApp

C) COPY . /

D) RUN sh setupScript.sh

E) CMD ["dotnet", "MyApp.exe"]

Select the correct sequence of commands from the options below:

A) E, D, C, B

B) A, C, B, D, E

C) B, D, C, E

D) C, B, D, E

Answer: B

Feedback(if correct):-

B) A, C, B, D, E

This order ensures a well-structured Dockerfile for deploying your ASP.NET Core application within a container:

1. A) FROM microsoft/dotnet:latest: Defines the base image with the .NET runtime environment upon which your application will run.

2. C) WORKDIR /app/MyApp: Sets the working directory within the container. Subsequent commands will be executed relative to this directory, making it easier to manage file paths.

3. B) COPY . /app (or specific directory): Copies your application files from the current directory (`.`) or a specific directory containing your application code into the container at the `/app` directory (as set in the previous step).

4. D) RUN sh setupScript.sh: Executes the setup script named "setupScript.sh" during the container build process. This script can be used for various purposes, such as installing dependencies, configuring the application environment, or performing any pre-startup tasks necessary for your application to function correctly.

5. E) CMD ["dotnet", "MyApp.exe"]: Defines the default command to run when the container starts. It instructs the container to execute the ".NET" runtime with the argument "MyApp.exe," launching your ASP.NET Core application upon container startup.

Feedback(if wrong):-

A) E, D, C, B: This order incorrectly places the "WORKDIR" command after copying files. The working directory needs to be set before copying to ensure files are copied to the correct location within the container.

C) B, D, C, E: Similar to A), it sets the working directory after copying files, leading to potential issues if the script relies on specific file paths.

D) C, B, D, E: This order swaps the positions of "WORKDIR" and "COPY" commands. While it might not cause major errors, it's generally considered better practice to set the working directory before copying files for clarity and maintainability.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Implement Azure security

Competencies: Azure SQL Database, Azure Active Directory (Azure AD), Row-Level Security

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

34. You're utilizing Azure Front Door with Brotli compression activated to optimize website performance. Your application serves a variety of assets, including large JavaScript files. However, you observe that these JavaScript files are not being compressed by Brotli, leading to increased loading times.

To diagnose the root cause of the problem, identify the TWO CORRECT statements from the following options:

Statement	Yes	No
The content type (MIME type) of the JavaScript files is incompatible with Brotli compression.		
Azure Front Door might not be compressing files exceeding a specific size threshold.		
Purging cached assets from edge nodes could potentially resolve the Brotli compression issue for JavaScript files.		

Select the correct statement responses

A. Yes, No, No
B. Yes, Yes, Yes
C. Yes, No, No
D. No, Yes, No

Answer: C

Feedback(if correct):

- Yes: The content type (MIME type) of the JavaScript files should be compatible with Brotli compression.

- No: Large files, including JavaScript files, may not be compressed by Azure Front Door if they exceed a specific size threshold.

- No: Purging cached assets from edge nodes won't likely fix the Brotli compression issue for JavaScript files.

Feedback(if wrong):

Option A is incorrect because it incorrectly states that purging cached assets from edge nodes could potentially resolve the Brotli compression issue for JavaScript files, which is not accurate. Option B is incorrect because it wrongly suggests that Azure Front Door might not be compressing files exceeding a specific size threshold, which is not mentioned in the context. Option D is incorrect as it incorrectly claims that the content type (MIME type) of JavaScript files might be incompatible with Brotli compression, which is not stated in the scenario.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Monitor, optimize, and troubleshoot Azure solutions

Competencies: Optimizing Azure Solutions, Troubleshooting Azure Solutions

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

35. You're leveraging Azure Front Door with Brotli compression configured. You've confirmed that Brotli is enabled and the file size thresholds are appropriate. However, you discover that a specific set of large JSON files (around 15 MB each) is not being compressed. To troubleshoot the cause of the Brotli compression failure for these JSON files, which of the following actions is the MOST suitable approach? (Choose one)

A) Analyze the response headers delivered by the server for the JSON files to determine if Brotli encoding is present.

B) Modify the Brotli configuration within your application code to explicitly include JSON files for compression.

C) Increase the global Brotli compression threshold in Azure Front Door to encompass the size of the JSON files.

D) Temporarily disable Brotli compression and observe the impact on website performance.

Answer: A

Feedback(if correct):

The correct answer is A) Analyze the response headers delivered by the server for the JSON files to determine if Brotli encoding is present. This option allows you to directly inspect the response headers to verify whether Brotli compression is being applied to the JSON files, helping diagnose any issues with the compression setup.

Feedback(if wrong):

B) Modify the Brotli configuration within your application code to explicitly include JSON files for compression. This option suggests modifying the application code to include JSON files for compression, which might not address the root cause of the compression failure and could introduce unnecessary complexity.

C) Increase the global Brotli compression threshold in Azure Front Door to encompass the size of the JSON files. This option proposes adjusting the compression threshold, which may not resolve the issue if the problem lies elsewhere, such as with the file format or server configuration.

D) Temporarily disable Brotli compression and observe the impact on website performance. This option suggests disabling Brotli compression entirely, which is not recommended as it does not directly address the problem and may degrade website performance unnecessarily.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Monitor, optimize, and troubleshoot Azure solutions

Competencies: Optimizing Azure Solutions, Troubleshooting Azure Solutions

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

36. You are responsible for managing the cost of an Azure web app that utilizes Application Insights for performance monitoring. Your goal is to ensure that the cost of Application Insights stays within a predetermined budget. Which action should you take to achieve this objective?

A) Implement adaptive sampling using the Azure AD.

B) Enforce a daily cap for the Application performance monitoring.

C) Implement adaptive sampling using the Application Insights SDK in the ASP.NET SDK in the .config file; or in the Java SDK in the ApplicationInsights.xml file

D) Configure ingestion sampling using the Azure portal.

Answer: C

Feedback(if correct):

Implementing adaptive sampling directly within the Application Insights SDK configuration files allows for fine-tuning the telemetry sampling process, ensuring that only necessary data is collected, thereby helping to manage costs effectively.

Feedback(if wrong):

Azure AD (Option A) is not directly related, enforcing a daily cap (Option B) may limit costs but doesn't optimize telemetry sampling, and configuring ingestion sampling (Option D) through the Azure portal doesn't offer the same level of control over sampling as directly adjusting the sampling rates in the SDK configuration files.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Monitor, optimize, and troubleshoot Azure solutions

Competencies: Optimizing Azure Solutions, Troubleshooting Azure Solutions

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

37. A taxi dispatch center sends trip information updates via an Azure Event Hub. The updates include pick-up locations, driver IDs, passenger counts, fare estimates, vehicle types, and timestamp. There are four categories of drivers: Regular, Silver, Gold, and Diamond.

Requirements:

Category A clients (Regular and Silver drivers) should receive all updates except for emergencies flagged with "Diamond Status."

Category B clients (Diamond drivers) should receive all updates, including emergencies flagged with "Diamond Status."

Category C clients (Gold drivers) should receive all updates plus maintenance schedules for their vehicles.

For Category A clients, which include Regular and Silver drivers, the requirement is to receive all updates except for emergencies flagged with "Diamond Status." To achieve this, what filter type should be implemented?

A) SQLFilter

B) CorrelationFilter

C) No Filter

D) Boolean Filter

Answer: A

Feedback(if correct):-

For Category A clients (Regular and Silver drivers), the requirement is to receive all updates except for emergencies flagged with "Diamond Status." To achieve this, you should implement an SQLFilter when configuring the Event Hub subscription for Category A clients. By using an SQLFilter, you can filter messages based on a SQL-like conditional expression, which allows you to exclude messages with "Diamond Status."

Example Expression: `NOT (vehicleStatus="Diamond Status")`

This filter will pass all messages except those with "Diamond Status."

Feedback(if wrong):-

CorrelationFilter: Correlation filters are used to select messages based on specific properties. However, in this scenario, there is no mention of any specific property related to emergencies that need to be filtered using correlation. Therefore, using a CorrelationFilter is unnecessary for Category A clients.

No Filter: Using No Filter means that all messages will be passed to the subscription, including emergencies flagged with "Diamond Status." This contradicts the requirement for Category A clients (Regular and Silver drivers) to receive all updates except for emergencies.

Boolean Filter: Boolean filters are not applicable in this context as they are not supported for message filtering in Azure Event Hubs. Therefore, choosing a Boolean Filter is incorrect for this scenario.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Monitor, optimize, and troubleshoot Azure solutions

Competencies: Optimizing Azure Solutions, Troubleshooting Azure Solutions

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

38. A taxi dispatch center sends trip information updates via an Azure Event Hub. The updates include pick-up locations, driver IDs, passenger counts, fare estimates, vehicle types, and timestamp. There are four categories of drivers: Regular, Silver, Gold, and Diamond.

Requirements:

Category A clients (Regular and Silver drivers) should receive all updates except for emergencies flagged with "Diamond Status."

Category B clients (Diamond drivers) should receive all updates, including emergencies flagged with "Diamond Status."

Category C clients (Gold drivers) should receive all updates plus maintenance schedules for their vehicles.

To ensure that Category B clients, specifically Diamond drivers, receive all updates, including emergencies flagged with "Diamond Status," what approach should be taken for filtering?

A) SQLFilter

B) CorrelationFilter

C) No Filter

D) Boolean Filter

Answer: C

Feedback(if correct):-

Since Category B clients (Platinum users) should receive all updates, including special announcements labeled with "Platinum," you do not need to apply any filter. Simply attaching the receiver to the Event Hub will suffice. This approach ensures that all messages published to the Event Hub are consumed by the receivers associated with Category B clients.

Feedback(if wrong):-
SQLFilter is incorrect because it involves filtering messages based on specific SQL expressions, which is not necessary for Category B clients who need to receive all updates.

CorrelationFilter is incorrect because it filters messages based on correlation properties, which is not needed for Category B clients who require all updates without any filtering.

Boolean filter is incorrect because it applies a true or false condition to filter messages, which is not suitable for ensuring Category B clients receive all updates.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Monitor, optimize, and troubleshoot Azure solutions

Competencies: Optimizing Azure Solutions, Troubleshooting Azure Solutions

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

39. A taxi dispatch center sends trip information updates via an Azure Event Hub. The updates include pick-up locations, driver IDs, passenger counts, fare estimates, vehicle types, and timestamp. There are four categories of drivers: Regular, Silver, Gold, and Diamond.

Requirements:

Category A clients (Regular and Silver drivers) should receive all updates except for emergencies flagged with "Diamond Status."

Category B clients (Diamond drivers) should receive all updates, including emergencies flagged with "Diamond Status."

Category C clients (Gold drivers) should receive all updates plus maintenance schedules for their vehicles.

To provide maintenance schedules for their vehicles to Category C clients (Gold drivers), which filter type should you implement for the Azure Event Hub subscription?

A) SQLFilter

B) CorrelationFilter

C) No Filter

D) Boolean filter

Answer: B

Feedback(if correct): CorrelationFilter is the correct choice because it allows you to filter messages based on specific properties. In this scenario, you can use CorrelationFilter to ensure that only maintenance schedules for vehicles are received by Category C clients (Gold drivers), meeting the requirement to provide maintenance schedules for their vehicles.

Feedback(if wrong):

A) SQLFilter: This is incorrect because SQLFilter is used to filter messages based on SQL-like conditional expressions. It's not suitable for filtering based on specific properties like maintenance schedules for vehicles.

C) No Filter: This is incorrect because it doesn't allow for targeted filtering based on specific properties. Without any filter, Category C clients would receive all updates, including those unrelated to maintenance schedules for their vehicles.

D) Boolean filter: There's no such thing as a "Boolean filter" in the context of Azure Event Hubs. Therefore, this option is incorrect.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Monitor, optimize, and troubleshoot Azure solutions

Competencies: Optimizing Azure Solutions, Troubleshooting Azure Solutions

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

40. To resolve the CORS-related testing error in your Azure App Service Web App APIs, requiring the configuration of specific origins to permit cross-origin requests, which command sequence should you utilize in Azure CLI?

A) Command Sequence: az webapp cors add allowed-origins

B) Command Sequence: az webapp add cors allowed-origins

C) Command Sequence: az webapp cors allowed-origins add

D) Command Sequence: az webapp cors add http://test.besttacotables.com

Answer: A

Feedback(if correct):-

az webapp cors add allowed-origins
This command adds the specified origins to the allowed list for CORS (Cross-Origin Resource Sharing), which is the appropriate action for addressing the CORS-related issue encountered during testing.

Feedback(if wrong):-

B) az webapp add cors --allowed-origins

This option is incorrect because it misplaces the "cors" parameter before "add," which is not the correct syntax for configuring CORS using the Azure CLI.

C) az webapp cors --allowed-origins add

This option is incorrect because it incorrectly orders the parameters, putting "allowed-origins" before "cors," which does not align with the proper syntax for the Azure CLI command.

D) az webapp cors add --allowed-origins "<https://test.besttacotables.com>"

This option is incorrect because it directly adds a specific origin ("<https://test.besttacotables.com>") without utilizing the "--allowed-origins" parameter, which is necessary for specifying the allowed origins for CORS configuration.

Skill mapping:

Skills: Azure Developer Certification (AZ 204)

Subskill: Connect to and consume Azure services and third-party services

Competencies: Azure Services Integration, Azure Logic Apps, Azure Service Bus. Azure Event Grid. Integration with third-party services

Difficulty Level: Intermediate or Expert

Bloom's Taxonomy Level: Application or Analysis

41. You are developing an Azure Function App using Visual Studio. The app processes orders input by an Azure Web App, which places order information into Azure Queue Storage. You need to evaluate the Azure Function App code provided below:

```csharp
public static class OrderProcessor
{
  public static void ProcessOrders([QueueTrigger("incoming-orders")] Order order, ILogger log)
  {
    log.Info($"Processing Order: {order.Id}");
    log.Info($"Queue Insertion Time: {order.InsertionTime}");
    log.Info($"Queue Expiration Time: {order.ExpirationTime}");
    // Additional code to process the order
  }

  public static void ProcessFailedOrders([QueueTrigger("failed-orders")] string failedOrder, ILogger log)
  {
    log.LogError($"Failed to process order: {failedOrder}");
  }
}
```

Which statements are correct regarding the provided code? (Select all that apply)

A) The code will log the time when the order was processed from the queue.

B) In case of failure in the ProcessOrders function, it will be retried up to five times for a given order, including the initial attempt.

C) When there are multiple orders queued, the ProcessOrders function will retrieve and process them concurrently in batches.

D) The ProcessOrders function will save the order information to an Azure Table Storage table named "Orders."

Answer: B

Feedback(if correct):-

B) In case of failure in the ProcessOrders function, it will be retried up to five times for a given order, including the initial attempt.

. Azure Functions' default behavior for a queue trigger involves retrying the execution of a triggered function up to five times upon failure before considering the message as poisonous and moving it to a poison queue. This retry mechanism helps in handling transient errors or issues that might occur during function execution.

 Feedback(if wrong):

A) The code will log the time when the order was processed from the queue.

- This statement is incorrect because the code logs properties of the `Order` object that are assumed to be `InsertionTime` and `ExpirationTime`, which do not reflect the time the function processes the order. There's no code explicitly logging the current time at which the order processing occurs.

C) When there are multiple orders queued, the ProcessOrders function will retrieve and process them concurrently in batches.

- This statement is misleading or incorrect as the provided code snippet does not specify the concurrency level or batch processing behavior. Azure Functions indeed supports concurrent execution, but the extent of concurrency and batch processing depends on host configuration settings and is not dictated by the provided code.

D) The ProcessOrders function will save the order information to an Azure Table Storage table named "Orders."

- This is incorrect because the provided code snippet does not include any operations related to Azure Table Storage or any explicit saving mechanism. The code only demonstrates logging operations and does not interact with any storage solution or specifically mention a table named "Orders".

skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competencies: Azure Services Integration

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

42. In a global event platform hosted on Azure, how can you ensure fast content delivery for static assets to users worldwide while minimizing latency?

A) Leverage Azure Traffic Manager for optimal request routing.

B) Implement horizontal pod autoscaling using HPA in AKS.

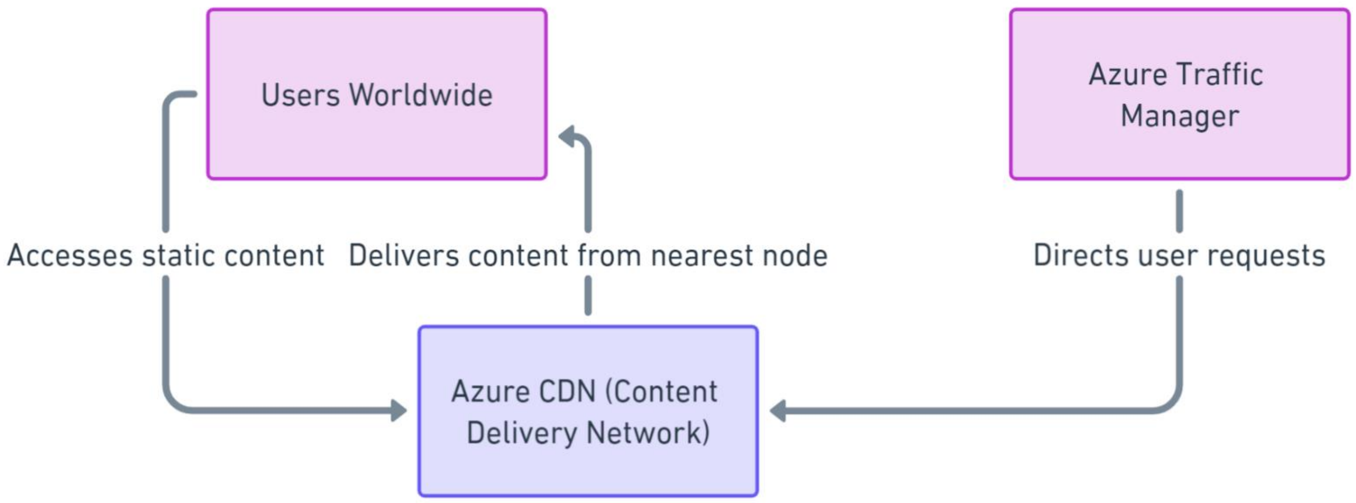
C) Utilize Azure CDN to cache static assets in geographically distributed locations.

D) Configure Azure Application Gateway for layer 7 load balancing.

Answer: C

Feedback(if correct):-

Correct Answer (C): Utilize Azure CDN to cache static assets in geographically distributed locations. This option ensures fast content delivery by caching static assets closer to users, minimizing latency through content delivery networks.



Feedback(if wrong):-

A) Azure Traffic Manager is primarily used for DNS-based traffic routing to endpoints, not for content delivery or minimizing latency.

B) Horizontal pod autoscaling using HPA in AKS is related to managing the number of running instances of a Kubernetes pod based on metrics like CPU utilization, which is not directly relevant to content delivery or minimizing latency.

D) Azure Application Gateway is a layer 7 load balancer that directs traffic to backend servers based on complex routing rules, but it's not specifically designed for caching or optimizing content delivery for static assets.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competencies: Azure Services Integration, Azure Traffic Manager, Azure CDN, Azure Application Gateway

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

43. You are an Azure developer working on a resilient microservice architecture deployed on Azure Kubernetes Service (AKS). As part of your deployment strategy, you need to ensure that the health status of running containers is continuously monitored, and any unhealthy containers are automatically restarted to maintain the availability of your microservices. Within your resilient microservice architecture on AKS, what Kubernetes component monitors the health status of running containers and triggers restarts if necessary?

A) Ingress Controller

B) ConfigMap

C) HPA (Horizontal Pod Autoscaler)

D) Liveness Probe

Answer: D

Feedback(if correct):-

In the given scenario, where the goal is to continuously monitor the health status of running containers and automatically restart any unhealthy containers to maintain availability, Liveness Probes are the appropriate Kubernetes component. Liveness Probes are configured within Kubernetes pod definitions to periodically check the health of containers. If a container fails the liveness check, Kubernetes automatically restarts it. This ensures that any issues affecting container health are promptly addressed, contributing to the resilience of the microservice architecture.

Feedback(if wrong):-

A) Ingress Controller - An Ingress Controller manages external access to services in a Kubernetes cluster, typically by providing HTTP and HTTPS routing to services. While important for managing traffic flow, it does not directly monitor container health or trigger restarts.

B) ConfigMap - ConfigMaps are used for storing non-sensitive configuration data in key-value pairs. They are not involved in monitoring container health or managing container restarts.

C) HPA (Horizontal Pod Autoscaler) - HPA automatically scales the number of pod replicas based on observed CPU utilization. While useful for scaling applications dynamically, it does not monitor container health or trigger restarts based on health status.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competencies: Azure Services Integration, Azure Kubernetes Service (AKS), Ingress Controllers

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application

44. Your company is using Azure Search to index documents stored in Azure Blob Storage. Customers report that they cannot find products when searching by product categories. What actions would you take to address this issue?

A) Ensure that the product categories are properly mapped to searchable fields in the Azure Search index settings.

B) Implement faceted navigation by configuring facets based on product categories to enable users to filter search results.

C) Add a custom skill to the Azure Cognitive Search pipeline to extract product categories from the documents and enrich the index with this information.

D) Modify the search query to include category-specific filters to refine search results based on user-selected product categories.

Answer: A

Feedback (if correct):

- Option A is correct because ensuring proper mapping of product categories to searchable fields in the Azure Search index settings is essential for accurate search functionality. This step ensures that search queries align with the indexed data, improving search accuracy and relevance.

Feedback (if wrong):

- Option B is incorrect because while implementing faceted navigation can enhance the user experience by enabling users to filter search results, it does not directly address the issue of products not being found when searching by categories.

- Option C is incorrect because adding a custom skill to the Azure Cognitive Search pipeline to extract product categories and enrich the index with this information might enhance search capabilities, but it does not directly address the reported issue of products not being found when searching by categories.

- Option D is incorrect because modifying the search query to include category-specific filters may help refine search results based on user-selected categories, but it does not address the underlying issue of products not being found when searching by categories.

skill mapping :

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competencies: Azure Services Integration

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application, Analysis

45. You are developing an application that manages user data for a video game. The application will store information such as region, email address, and optionally, phone number for each player. The region data will be used for load-balancing purposes. All data will be stored in Azure Table Storage. You need to implement code to retrieve data for an individual player.

Select the appropriate options to complete the code snippet below:

```csharp
public class PlayerEntity : TableEntity
{
    public PlayerEntity()
    {
    }

    public PlayerEntity(string region, string email)
    {
        PartitionKey = Slot1;
        RowKey = Slot2;
    }

    public string Phone { get; set; }
}

public class Player
{
    protected PlayerEntity player;

    async void GetPlayer(string cs, CloudTable table, string pk, string rk)
    {
        TableOperation query = TableOperation.Retrieve<PlayerEntity>(Slot3, Slot4);
        TableResult data = await table.ExecuteAsync(query);
        player = data.Result as PlayerEntity;
    }
}
```

What should be placed in Slot1 to properly complete the code snippet?

A) The region of the player

B) The email address of the player

C) The Partition Key for the Azure Table entity

D) The base class for Azure Table entities

Answer: A

Feedback(if correct):-

Slot1 corresponds to the Partition Key in the Azure Table entity. The Partition Key is used to partition data across multiple storage nodes for scalability and performance. In this scenario, the region of the player is used as the Partition Key, enabling efficient load-balancing of data. Therefore, placing the region of the player in Slot1 completes the code snippet for creating a PlayerEntity with the appropriate Partition Key.

Feedback(if wrong):-

B) The email address of the player is incorrect. Slot1 represents the Partition Key, not the player's email address.

C) The Partition Key for the Azure Table entity is incorrect. Slot1 requires the actual value of the Partition Key, which is the region of the player.

D) The base class for Azure Table entities is incorrect. Slot1 represents the Partition Key, not the base class for Azure Table entities.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competencies: Azure Services Integration

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application, Analysis

46. You are developing an application that manages user data for a video game. The application will store information such as region, email address, and optionally, phone number for each player. The region data will be used for load-balancing purposes. All data will be stored in Azure Table Storage. You need to implement code to retrieve data for an individual player.

Select the appropriate options to complete the code snippet below:

```csharp
public class PlayerEntity : TableEntity
{
    public PlayerEntity()
    {
    }

    public PlayerEntity(string region, string email)
    {
        PartitionKey = Slot1;
        RowKey = Slot2;
    }
    public string Phone { get; set; }
}

public class Player
{
    protected PlayerEntity player;

    async void GetPlayer(string cs, CloudTable table, string pk, string rk)
    {
        TableOperation query = TableOperation.Retrieve<PlayerEntity>(Slot3, Slot4);
        TableResult data = await table.ExecuteAsync(query);
        player = data.Result as PlayerEntity;
    }
}
```

What should be placed in Slot2 to properly complete the code snippet?

A) The region of the player

B) The email address of the player

C) The Row Key for the Azure Table entity

D) The base class for Azure Table entities

Answer: B

Feedback(if correct):-

Slot2 corresponds to the Row Key in the Azure Table entity. The Row Key uniquely identifies each record within a partition. In this scenario, the email address of the player is used as the Row Key, ensuring each player's data can be efficiently retrieved. Therefore, placing the email address of the player in Slot2 completes the code snippet for creating a PlayerEntity with the appropriate Row Key.

Feedback(if wrong):-

A) The region of the player - Incorrect. Slot2 represents the Row Key, not the player's region.

C) The Row Key for the Azure Table entity is incorrect. Slot2 requires the actual value of the Row Key, which is the email address of the player.

D) The base class for Azure Table entities is incorrect. Slot2 represents the Row Key, not the base class for Azure Table entities.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competencies: Azure Services Integration

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application, Analysis

47. You are developing an application that manages user data for a video game. The application will store information such as region, email address, and optionally, phone number for each player. The region data will be used for load-balancing purposes. All data will be stored in Azure Table Storage. You need to implement code to retrieve data for an individual player. Select the appropriate options to complete the code snippet below:

```csharp
public class PlayerEntity : TableEntity
{
  public PlayerEntity()
  {
  }

  public PlayerEntity(string region, string email)
  {
    PartitionKey = Slot1;
    RowKey = Slot2;
  }

  public string Phone { get; set; }
```

}

public class Player

{

  protected PlayerEntity player;

  async void GetPlayer(string cs, CloudTable table, string pk, string rk)

  {

    TableOperation query = TableOperation.Retrieve<PlayerEntity>(Slot3, Slot4);

    TableResult data = await table.ExecuteAsync(query);

    player = data.Result as PlayerEntity;

  }

}

```

What should be placed in Slot3 to properly complete the code snippet?

A) The Partition Key for the Azure Table entity

B) The Row Key for the Azure Table entity

C) the region of the player

D) the email address of the player

Answer: C

Feedback(if correct):-

Slot3 corresponds to the Partition Key in the Azure Table entity. The Partition Key is used to distribute table data across storage nodes for scalability and performance. In this scenario, the region of the player is used for load-balancing purposes and is therefore set as the Partition Key. Placing the region of the player in Slot3 completes the code snippet by specifying the Partition Key for the PlayerEntity.

Feedback(if wrong):-  A) The Partition Key for the Azure Table entity is incorrect. Slot3 represents the actual value of the Partition Key, which in this case is the region of the player.

B) The Row Key for the Azure Table entity is incorrect. Slot3 corresponds to the Partition Key, not the Row Key.

D) the email address of the player is incorrect. Slot3 requires the value of the Partition Key, which is the region of the player, not the email address.
```

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competencies: Azure Services Integration

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application, Analysis

48. You are developing an application that manages user data for a video game. The application will store information such as region, email address, and optionally, phone number for each player. The region data will be used for load-balancing purposes. All data will be stored in Azure Table Storage. You need to implement code to retrieve data for an individual player.

Select the appropriate options to complete the code snippet below:

```csharp
public class PlayerEntity : TableEntity
{
  public PlayerEntity()
  {
  }

  public PlayerEntity(string region, string email)
  {
    PartitionKey = Slot1;
    RowKey = Slot2;
  }

  public string Phone { get; set; }
}

public class Player
{
    protected PlayerEntity player;


    async void GetPlayer(string cs, CloudTable table, string pk, string rk)
    {
        TableOperation query = TableOperation.Retrieve<PlayerEntity>(Slot3, Slot4);
        TableResult data = await table.ExecuteAsync(query);
        player = data.Result as PlayerEntity;
    }
}
```

What should be placed in Slot4 to properly complete the code snippet?

A) The Partition Key for the Azure Table entity

B) The Row Key for the Azure Table entity

C) the region of the player

D) the email address of the player

Answer: D

Feedback(if correct):-

In Azure Table Storage, entities are uniquely identified by a combination of Partition Key and Row Key. The Row Key uniquely identifies each entity within its partition. In the provided code snippet, Slot4 is used as the placeholder for the Row Key, which means it should contain the value that uniquely identifies the player entity within the specified partition.

Feedback(if wrong):-

A) The Partition Key for the Azure Table entity is incorrect. The Partition Key is already set to Slot1 in the code snippet, so Slot4 should contain the Row Key, not the Partition Key, which uniquely identifies the partition.

C) The region of the player - Incorrect. Slot4 should contain the Row Key value, not the player's region. The Row Key is used for unique identification within the partition, while the player's region is a separate attribute.

D) The email address of the player is incorrect. Slot4 should contain the Row Key value, not the player's email address. The Row Key is used for unique identification within the partition, while the email address is a separate attribute.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competencies: Azure Services Integration

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application, Analysis

49. You're developing a suite of microservices on Azure Kubernetes Service (AKS). These microservices need to communicate with each other securely and efficiently. You plan to use a messaging system for inter-service communication. Which solution would best meet the specified requirements?

A) Configure Azure API Management to manage APIs and enforce policies for communication between microservices.

B) Utilize Azure Event Grid to handle event-driven communication between microservices.

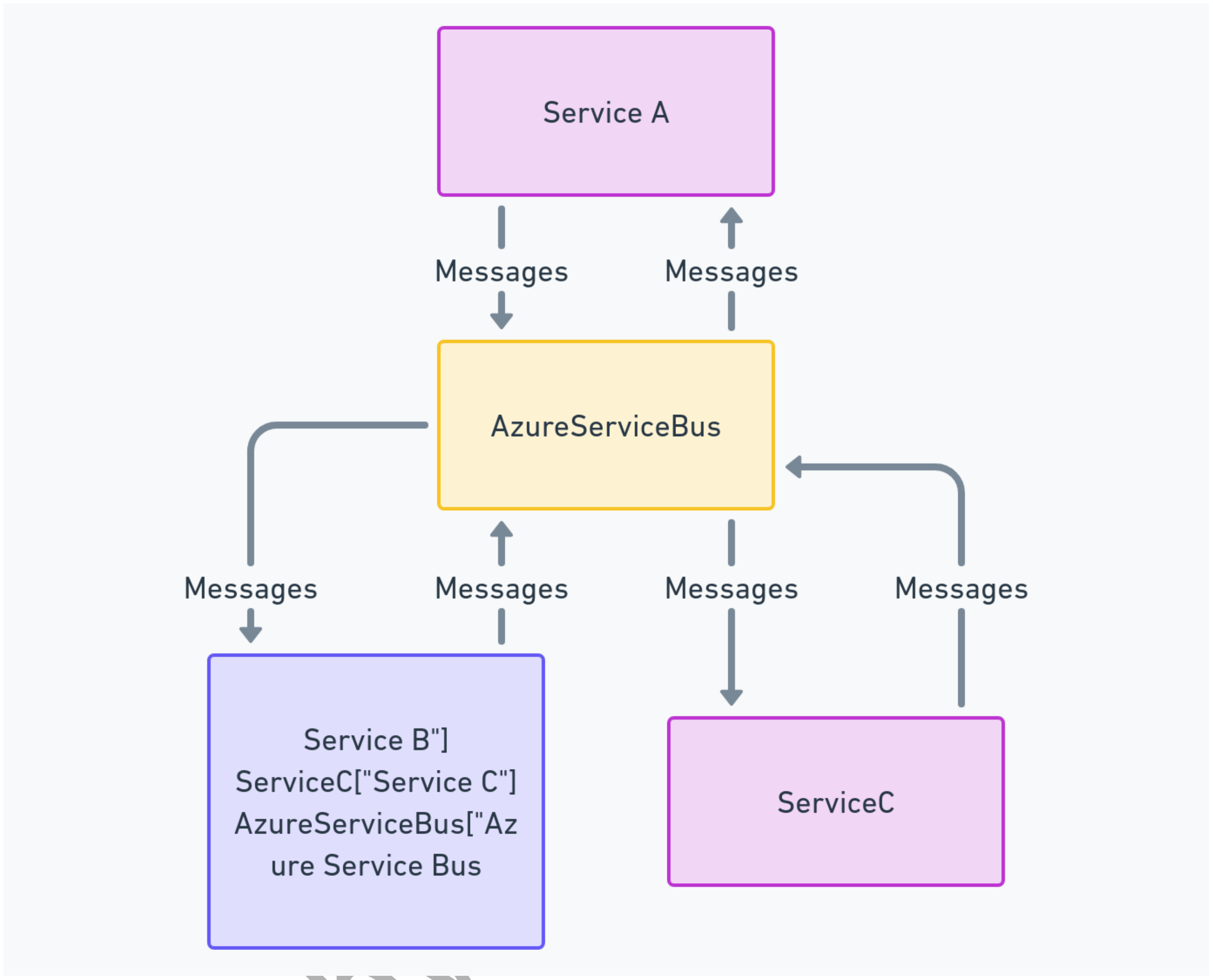
C) Implement Azure Logic Apps to orchestrate workflows and integrate services seamlessly.

D) Deploy Azure Service Bus to enable reliable messaging between microservices.

Answer: D

Feedback(if correct):-

Azure Service Bus is specifically designed for reliable messaging between applications and services. It provides features such as queues, topics, and subscriptions, which are essential for asynchronous and decoupled communication in a microservices architecture. Service Bus ensures message delivery and supports advanced messaging patterns, making it suitable for inter-service communication in distributed systems like microservices.



Feedback(if wrong):-

Option B (Azure Event Grid) is more suitable for handling event-driven communication and routing events to specific endpoints or handlers, rather than direct communication between microservices.

Option C (Azure Logic Apps) is designed for workflow automation and integrating various services and systems, but it is not optimized for microservices communication.

Option A (Azure API Management) focuses on managing APIs, enforcing policies, and controlling access to APIs, but it is not intended for direct communication between microservices within an application.

Skill mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competency: Azure Service Bus, Azure Event Grid, Azure Logic Apps, Azure API Management

Difficulty Level: Intermediate

50. You are designing a real-time data processing solution on Azure. Your solution requires a mechanism to handle events asynchronously in a publish-subscribe fashion without the need for constant polling. Which two Azure services can you utilize to meet this requirement? Select the best options from the choices below:

A) Azure Blob Storage

B) Azure Event Hubs

C) Azure Logic Apps

D) Azure Functions

Answer: B, D

Feedback(if correct):

Option B) Azure Event Hubs is the correct choice. Azure Event Hubs is a scalable and distributed event streaming platform that can handle millions of events per second, making it ideal for scenarios requiring asynchronous event processing in a publish-subscribe model.

Option D) Azure Functions is also a suitable choice. Azure Functions can be triggered by events from Azure Event Hubs, allowing for asynchronous event processing without the need for constant polling. This combination enables efficient and scalable event-driven architectures.

Feedback(if wrong):

Option A) Azure Blob Storage is not designed for real-time event processing or asynchronous communication in a publish-subscribe model. It is primarily used for storing unstructured data, such as images, documents, and log files, rather than handling real-time events.

Option C) Azure Logic Apps is a workflow automation service that allows you to automate business processes by orchestrating workflows and integrating with various Azure and third-party services. While it supports event-driven workflows, it is not specifically designed for handling real-time events in a publish-subscribe model like Azure Event Hubs.

Skill Mapping:

Skills: Azure Developer Certification (AZ-204)

Subskills: Connect to and consume Azure services and third-party services

Competencies: Event-driven architecture, Asynchronous messaging

Difficulty Level: Intermediate

Bloom's Taxonomy Level: Application