Custom View Settings

Question #18 *Topic 5*

HOTSPOT -

You are working for Contoso, Ltd.

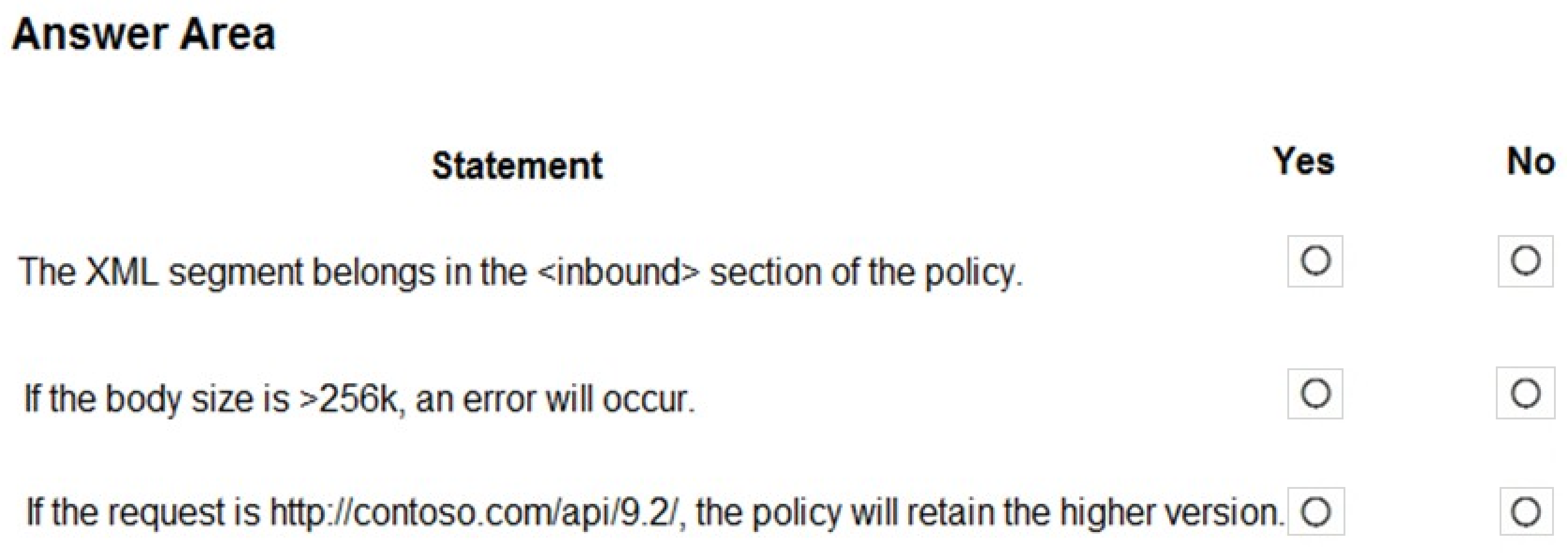
You define an API Policy object by using the following XML markup:

```
<set-variable name= "bodySize" value="@(context.Request.Headers["Content-Length"] [0])"/>
<choose>
 <when condition= "@(int.Parse(context.Variables.GetValueOrDefault<string> ("bodySize"))<512000)">
</when>
<otherwise>
     <rewrite-uri template= "/put"/>
     <set-backend-service base-url= "http://contoso.com/api/9.1/"/>
</otherwise>
</choose>
```

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

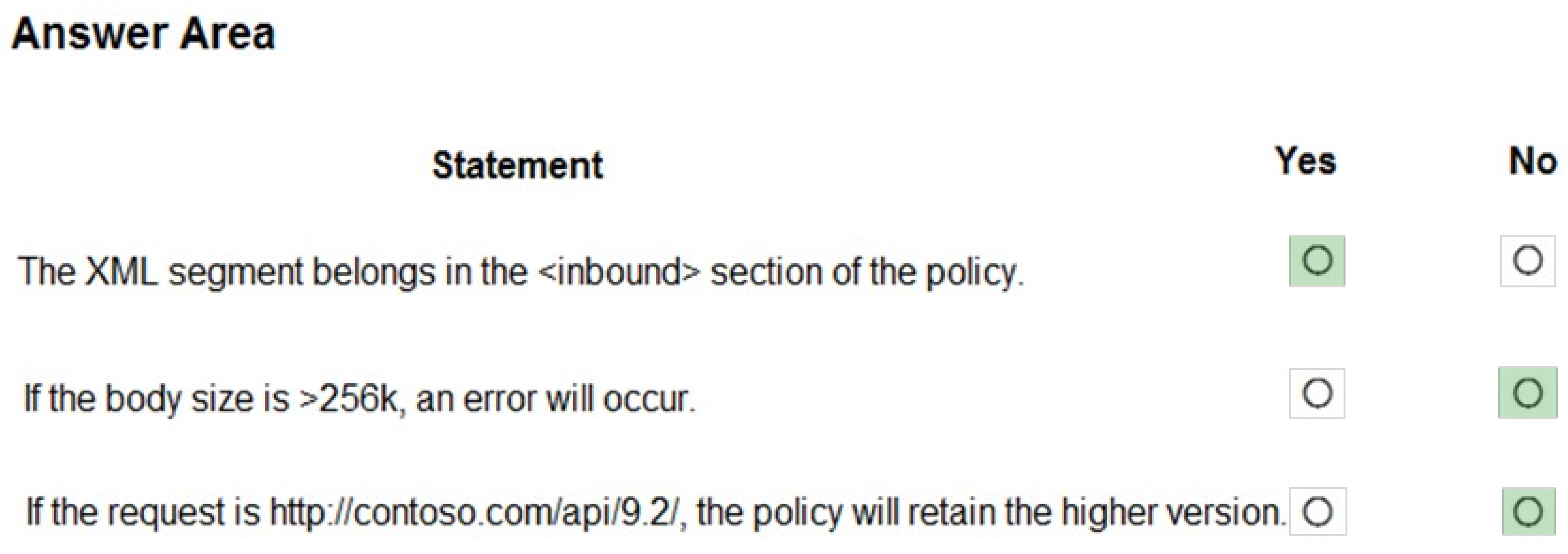
Hot Area:



Answer Area

Statement	Yes	No
The XML segment belongs in the <inbound> section of the policy.	○	○
If the body size is >256k, an error will occur.	○	○
If the request is http://contoso.com/api/9.2/, the policy will retain the higher version.	○	○

Correct Answer:



Answer Area

Statement	Yes	No
The XML segment belongs in the <inbound> section of the policy.	[✓ ○]	○
If the body size is >256k, an error will occur.	○	[✓ ○]
If the request is http://contoso.com/api/9.2/, the policy will retain the higher version.	○	[✓ ○]

Box 1: Yes -

Use the set-backend-service policy to redirect an incoming request to a different backend than the one specified in the API settings for that operation. Syntax:

<set-backend-service base-url="base URL of the backend service" />

Box 2: No -

The condition is on 512k, not on 256k.

Box 3: No -

The set-backend-service policy changes the backend service base URL of the incoming request to the one specified in the policy.

Reference:

https://docs.microsoft.com/en-us/azure/api-management/api-management-transformation-policies

Question #19 *Topic 5*

You are developing a solution that will use Azure messaging services.

You need to ensure that the solution uses a publish-subscribe model and eliminates the need for constant polling.

What are two possible ways to achieve the goal? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

A. Service Bus

B. Event Hub

C. Event Grid

D. Queue

Correct Answer: *AC*

It is strongly recommended to use available messaging products and services that support a publish-subscribe model, rather than building your own. In Azure, consider using Service Bus or Event Grid. Other technologies that can be used for pub/sub messaging include Redis, RabbitMQ, and Apache Kafka.

Reference:

https://docs.microsoft.com/en-us/azure/architecture/patterns/publisher-subscriber

Question #20 *Topic 5*

A company is implementing a publish-subscribe (Pub/Sub) messaging component by using Azure Service Bus. You are developing the first subscription application.

In the Azure portal you see that messages are being sent to the subscription for each topic. You create and initialize a subscription client object by supplying the correct details, but the subscription application is still not consuming the messages.

You need to ensure that the subscription client processes all messages.

Which code segment should you use?

A. await subscriptionClient.AddRuleAsync(new RuleDescription(RuleDescription.DefaultRuleName, new TrueFilter()));

B. subscriptionClient = new SubscriptionClient(ServiceBusConnectionString, TopicName, SubscriptionName);

C. await subscriptionClient.CloseAsync();

D. subscriptionClient.RegisterMessageHandler(ProcessMessagesAsync, messageHandlerOptions);

Correct Answer: *D*

Using topic client, call RegisterMessageHandler which is used to receive messages continuously from the entity. It registers a message handler and begins a new thread to receive messages. This handler is waited on every time a new message is received by the receiver.

subscriptionClient.RegisterMessageHandler(ReceiveMessagesAsync, messageHandlerOptions);

Reference:

https://www.c-sharpcorner.com/article/azure-service-bus-topic-and-subscription-pub-sub/

Question #21 *Topic 5*

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You are developing an Azure Service application that processes queue data when it receives a message from a mobile application. Messages may not be sent to the service consistently.
You have the following requirements:
✑ Queue size must not grow larger than 80 gigabytes (GB).
✑ Use first-in-first-out (FIFO) ordering of messages.
✑ Minimize Azure costs.
You need to implement the messaging solution.
Solution: Use the .Net API to add a message to an Azure Storage Queue from the mobile application. Create an Azure VM that is triggered from Azure Storage
Queue events.
Does the solution meet the goal?

A. Yes

[✓ B. No]

Correct Answer: *B*
Don't use a VM, instead create an Azure Function App that uses an Azure Service Bus Queue trigger.
Reference:
https://docs.microsoft.com/en-us/azure/azure-functions/functions-create-storage-queue-triggered-function

Question #22 *Topic 5*

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You are developing an Azure Service application that processes queue data when it receives a message from a mobile application. Messages may not be sent to the service consistently.
You have the following requirements:
✑ Queue size must not grow larger than 80 gigabytes (GB).
✑ Use first-in-first-out (FIFO) ordering of messages.
✑ Minimize Azure costs.
You need to implement the messaging solution.
Solution: Use the .Net API to add a message to an Azure Service Bus Queue from the mobile application. Create an Azure Windows VM that is triggered from
Azure Service Bus Queue.
Does the solution meet the goal?

A. Yes

[✓ B. No]

Correct Answer: *B*
Don't use a VM, instead create an Azure Function App that uses an Azure Service Bus Queue trigger.
Reference:
https://docs.microsoft.com/en-us/azure/azure-functions/functions-create-storage-queue-triggered-function

← Previous Questions Next Questions →