AZ-201$^{Q\&As}$

Microsoft Azure Developer Advanced Solutions (beta)

Pass Microsoft AZ-201 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.pass4itsure.com/az-201.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Microsoft Official Exam Center

QUESTION 1

DRAG DROP

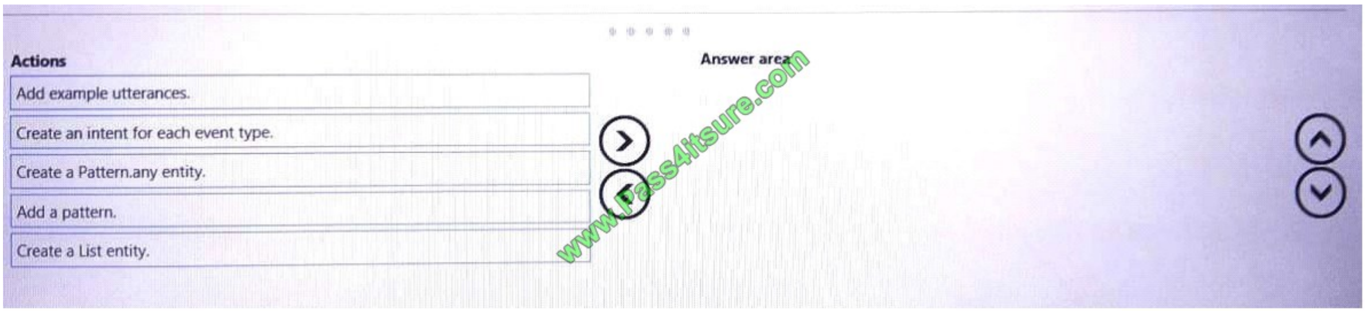
You develop a bot by using Language Understanding Intelligence Service (LUIS) and the MI Bot framework. You use LUIS m the Azure portal to optimize the bot

You\\'re view the utterancesand determine that users are requesting time andvenue informationfor events.

You need to improve the prediction efficiency of the bot.

Which three actions should you perform in sequence? To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.

Select and Place:



Actions

Add example utterances.

Create an intent for each event type.

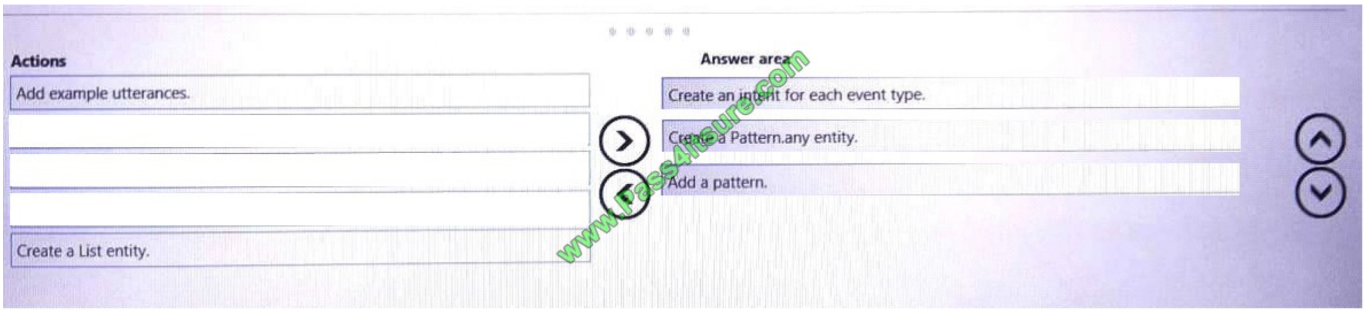
Create a Pattern.any entity.

Add a pattern.

Create a List entity.

Answer area

Correct Answer:



Actions

Add example utterances.

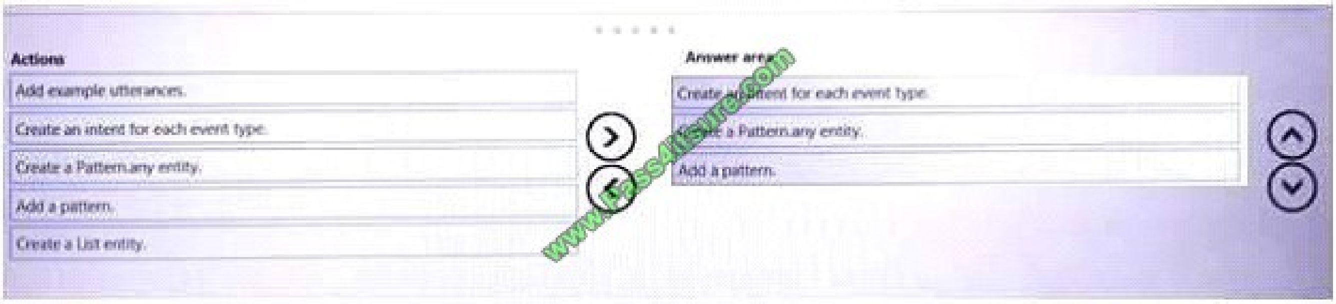
Create a List entity.

Answer area

Create an intent for each event type.

Create a Pattern.any entity.

Add a pattern.



Actions

Add example utterances.

Create an intent for each event type.

Create a Pattern.any entity.

Add a pattern.

Create a List entity.

Answer area

Create an intent for each event type.

Create a Pattern.any entity.

Add a pattern.

QUESTION 2

Note:this question is partof a series of questionsthat presentthe samescenario. Each questionin 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 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 use ASP.NET Core MVC with ADO.NET to develop an application. You implement database sharding for the application by using Azure SQL Database. You establish communication to implement a strategy that allows a group of

operations that are performed on multiple Azure

databases to be rolled back on all database if any of the operations fail.

Solution

Create stored procedures in each Azure SQL database instance to perform operations for each respective database.

Invoke anamedtransaction and usethe same namefor thetransactionin each stored procedure.

Establish a new transaction scope in a using block. Within the block, establish connections to each Azure SQL Database instance and run the stored procedure.

If no exception occurs, commit the scoped transaction.

Does the solution meet the goal?

A. Yes

B. No

Correct Answer: B

QUESTION 3

You have implemented code that uses elastic transactions spanning across three different Azure SQL Database logical servers. Database administrators report that some transactions take longer to complete than expected. You need to use the correct tool to monitor all the transactions originating from the elastic transaction implementation. Which tool should you use?

A. Run the sys.dm_tran_active_transactions dynamic management view.

B. Run the sys.dm_tran_current_snapshot dynamic management view.

C. Run the sys.dm_tran_active_snapshot_database_transactions dynamicmanagement

D. Use the dependencies section of Azure Applications Insights.

Correct Answer: A

QUESTION 4

DRAG DROP

Contoso. Ltd. provides an API to customers by using Azure API Management (ARM). The API authorizes users with a JWT token

You must implement response caching for the APIM gateway. The caching mechanism must detect the user IDof the client that accessesdata for a given location and cache the response for that user ID.

You need to add the following policies to the policies file:

a set-variable policy to store the detected user identity

a cache-lookup-value policy

a cache-store-value policy

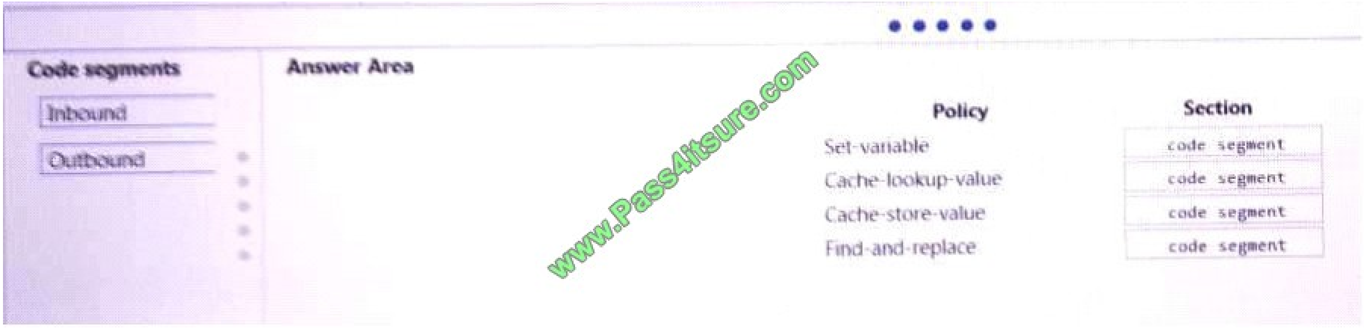
a find-and-replace policy to update the response body with the user profile information

To which policy section should you add the policies? To answer, drag the appropriate sections to the correct policies. Each section may be used once, more than once, or not at all.

You may need to drag the split bar between panes or scrol\lt oview content.

NOTE: Each correct selection is worth one point.

Select and Place:



Code segments

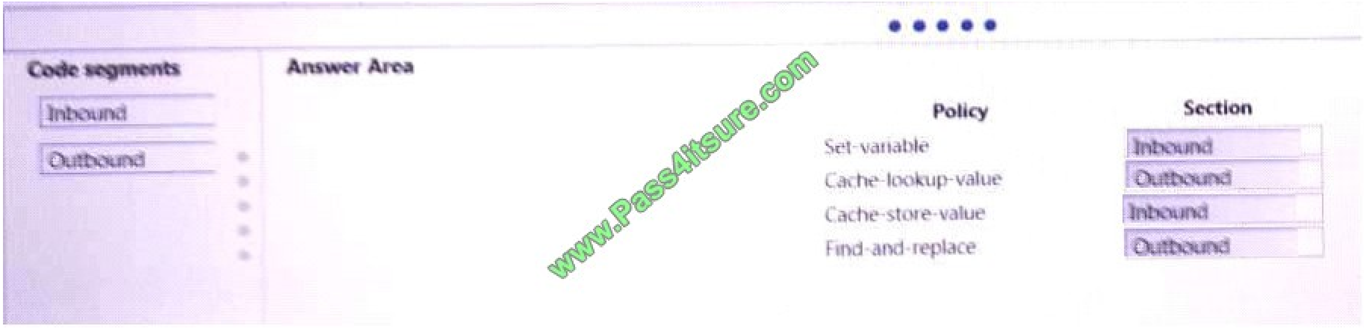
Inbound

Outbound

Answer Area

Policy	Section
Set-variable	code segment
Cache-lookup-value	code segment
Cache-store-value	code segment
Find-and-replace	code segment

Correct Answer:



Code segments

Inbound

Outbound

Answer Area

Policy	Section
Set-variable	Inbound
Cache-lookup-value	Outbound
Cache-store-value	Inbound
Find-and-replace	Outbound

QUESTION 5

HOTSPOT

You are developing an SMS based testing solution. The solution sends users a question by using SMS.

Early responders may qualify for prizes.

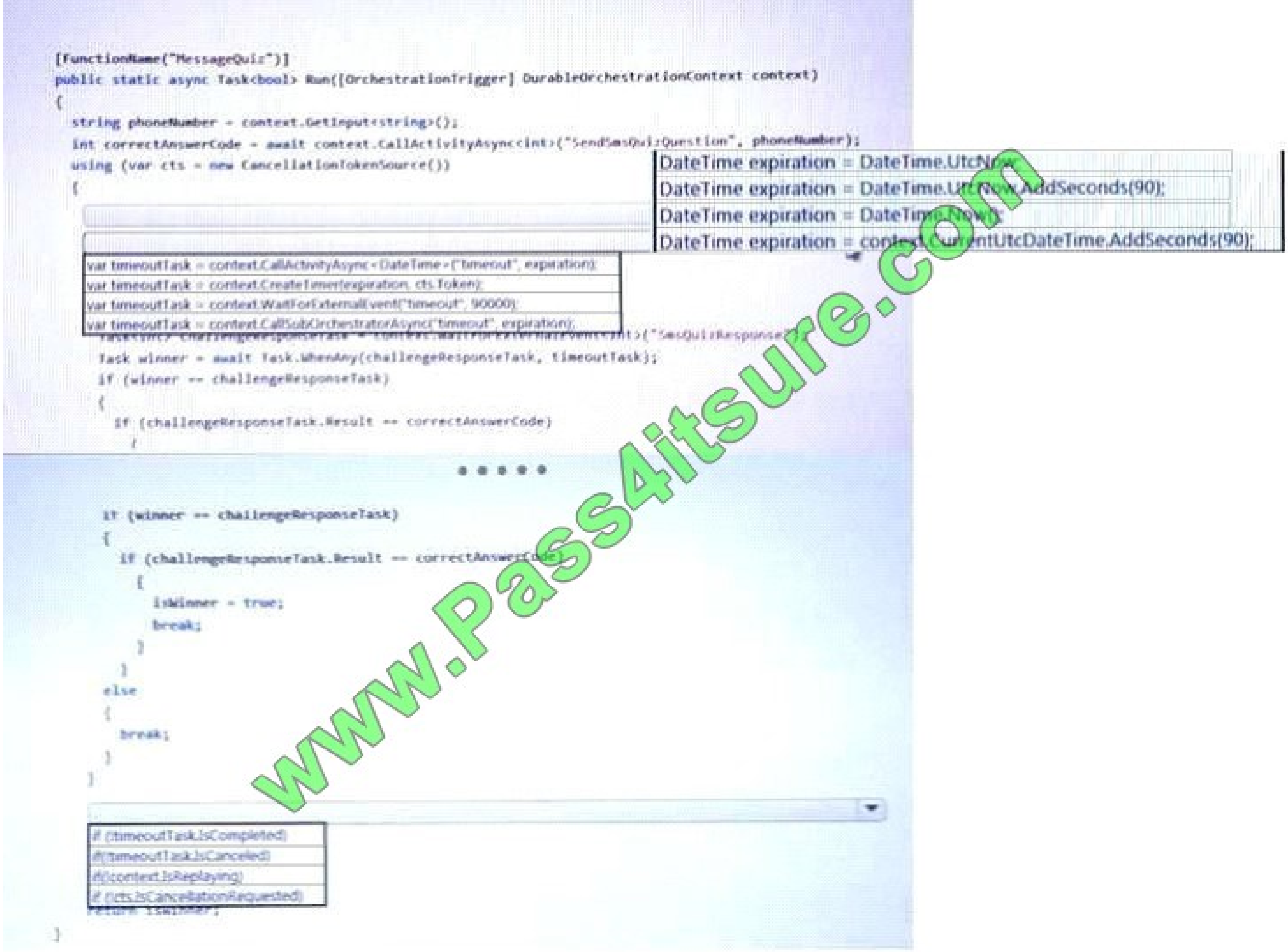
Users must respond with an answer choice with in 90 seconds.You must be able to track how long it takes each user to respond. You create a durable Azure Function named Send Sms Quiz Question that uses Twilio to send messages.

You need to write the code for MessageQuiz.

How should you complete the code? To answer, select the appropriate options in the answer area;

NOTE: Each correct selection is worth one point.

Hot Area:

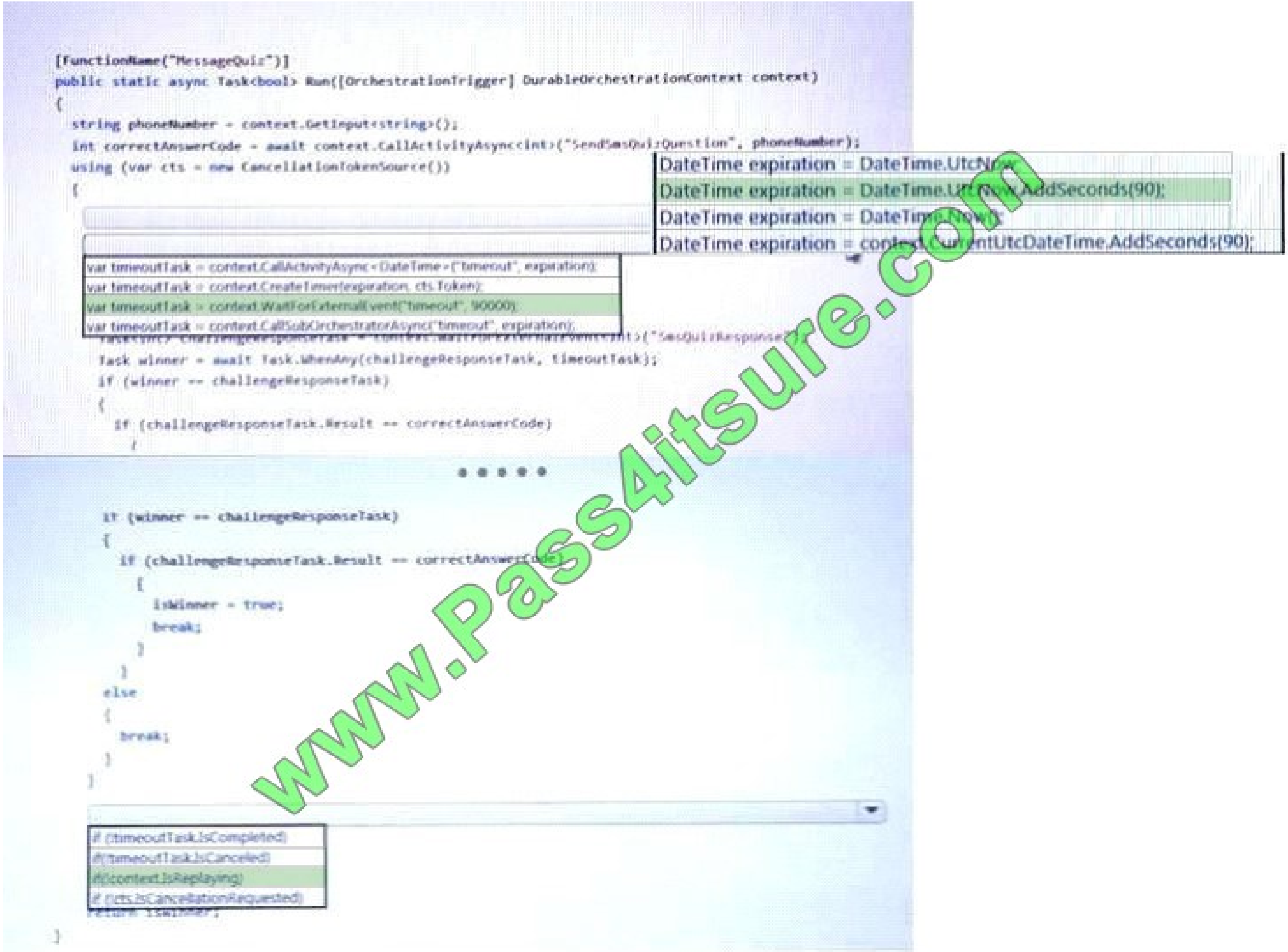


```
[FunctionName("MessageQuiz")]
public static async Task<bool> Run([OrchestrationTrigger] DurableOrchestrationContext context)
{
  string phoneNumber = context.GetInput<string>();
  int correctAnswerCode = await context.CallActivityAsync<int>("SendSmsQuizQuestion", phoneNumber);
  using (var cts = new CancellationTokenSource())
  {
                                                 DateTime expiration = DateTime.UtcNow;
                                                 DateTime expiration = DateTime.UtcNow.AddSeconds(90);
                                                 DateTime expiration = DateTime.Now;
                                                 DateTime expiration = context.CurrentUtcDateTime.AddSeconds(90);
    var timeoutTask = context.CallActivityAsync<DateTime>("timeout", expiration);
    var timeoutTask = context.CreateTimer(expiration, cts.Token);
    var timeoutTask = context.WaitForExternalEvent("timeout", 90000);
    var timeoutTask = context.CallSubOrchestratorAsync("timeout", expiration);
    Task<int> challengeResponseTask = context.WaitForExternalEvent<int>("SmsQuizResponse");
    Task winner = await Task.WhenAny(challengeResponseTask, timeoutTask);
    if (winner == challengeResponseTask)
    {
      if (challengeResponseTask.Result == correctAnswerCode)
      {
```

• • • • •

```
    if (winner == challengeResponseTask)
    {
      if (challengeResponseTask.Result == correctAnswerCode)
      {
        isWinner = true;
        break;
      }
    }
    else
    {
      break;
    }
  }
    if (!timeoutTask.IsCompleted)
    if(!timeoutTask.IsCanceled)
    if(!context.IsReplaying)
    if (!cts.IsCancellationRequested)
    return isWinner;
}
```

Correct Answer:



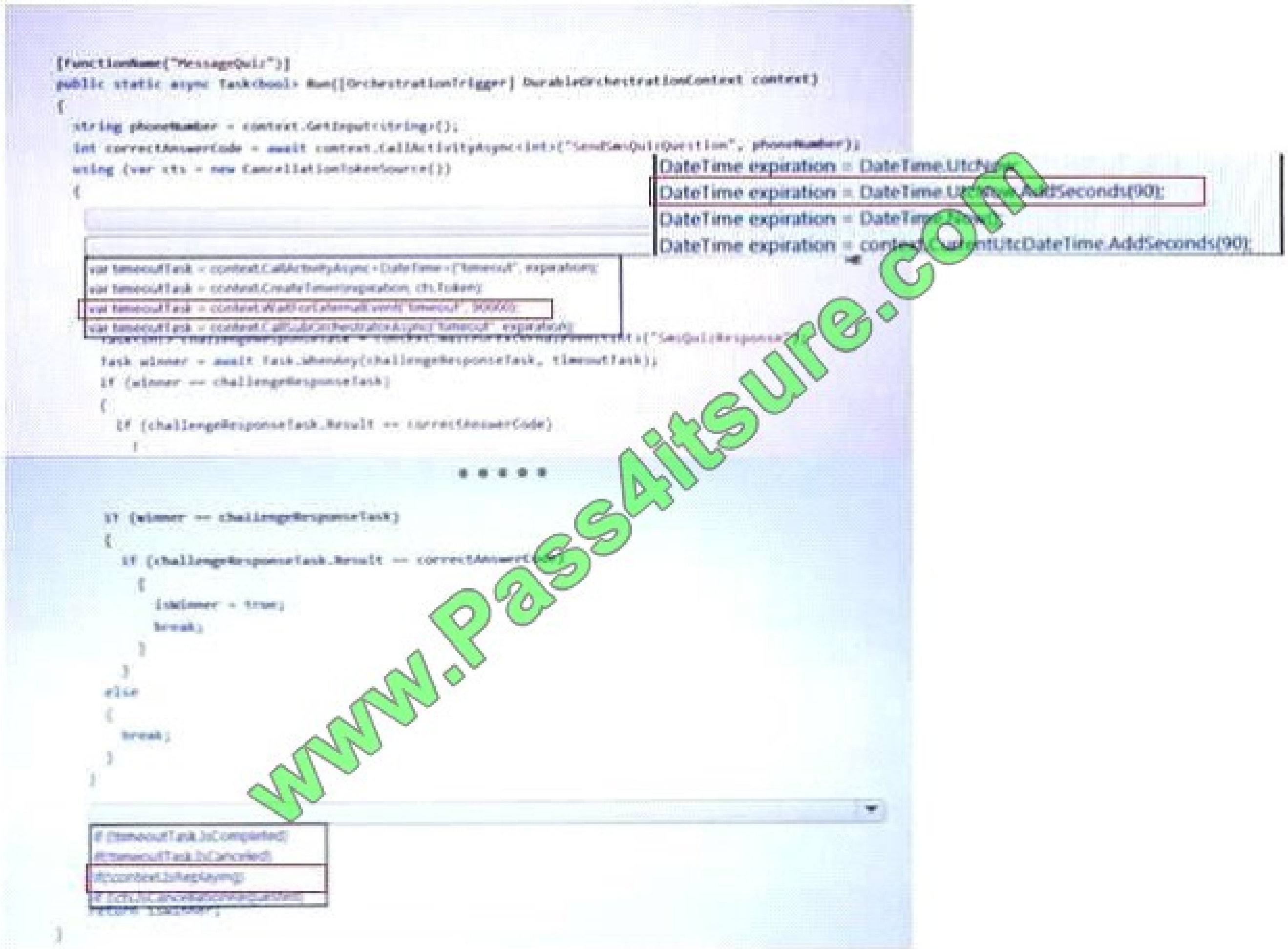
```
[FunctionName("MessageQuiz")]
public static async Task<bool> Run([OrchestrationTrigger] DurableOrchestrationContext context)
{
  string phoneNumber = context.GetInput<string>();
  int correctAnswerCode = await context.CallActivityAsync<int>("SendSmsQuizQuestion", phoneNumber);
  using (var cts = new CancellationTokenSource())
  {
                                                      DateTime expiration = DateTime.UtcNow;
                                                      DateTime expiration = DateTime.UtcNow.AddSeconds(90);
                                                      DateTime expiration = DateTime.Now;
                                                      DateTime expiration = context.CurrentUtcDateTime.AddSeconds(90);

    var timeoutTask = context.CallActivityAsync<DateTime>("timeout", expiration);
    var timeoutTask = context.CreateTimer(expiration, cts.Token);
    var timeoutTask = context.WaitForExternalEvent("timeout", 90000);
    var timeoutTask = context.CallSubOrchestratorAsync("timeout", expiration);
    Task<int> challengeResponseTask = context.WaitForExternalEvent<int>("SmsQuizResponse
    Task winner = await Task.WhenAny(challengeResponseTask, timeoutTask);
    if (winner == challengeResponseTask)
    {
      if (challengeResponseTask.Result == correctAnswerCode)
      {

                                  • • • • •

    if (winner == challengeResponseTask)
    {
      if (challengeResponseTask.Result == correctAnswerCode)
      {
        isWinner = true;
        break;
      }
    }
    else
    {
      break;
    }
  }

  if (!timeoutTask.IsCompleted)
  if(!timeoutTask.IsCanceled)
  if(!context.IsReplaying)
  if (!cts.IsCancellationRequested)
  return isWinner;
}
```



```
[FunctionName("MessageQuiz")]
public static async Task<bool> Run([OrchestrationTrigger] DurableOrchestrationContext context)
{
  string phoneNumber = context.GetInput<string>();
  int correctAnswerCode = await context.CallActivityAsync<int>("SendSmsQuizQuestion", phoneNumber);
  using (var cts = new CancellationTokenSource())
  {
                                                      DateTime expiration = DateTime.UtcNow;
                                                      DateTime expiration = DateTime.UtcNow.AddSeconds(90);
                                                      DateTime expiration = DateTime.Now;
                                                      DateTime expiration = context.CurrentUtcDateTime.AddSeconds(90);
    var timeoutTask = context.CallActivityAsync<DateTime>("timeout", expiration);
    var timeoutTask = context.CreateTimer(expiration, cts.Token);
    var timeoutTask = context.WaitForExternalEvent("timeout", 90000);
    var timeoutTask = context.CallSubOrchestratorAsync("timeout", expiration);
    Task<int> challengeResponseTask = context.WaitForExternalEvent<int>("SmsQuizResponse");
    Task winner = await Task.WhenAny(challengeResponseTask, timeoutTask);
    if (winner == challengeResponseTask)
    {
      if (challengeResponseTask.Result == correctAnswerCode)
      {

                                    • • • • •

    if (winner == challengeResponseTask)
    {
      if (challengeResponseTask.Result == correctAnswerCode)
      {
        isWinner = true;
        break;
      }
    }
    else
    {
      break;
    }
  }

  if (!timeoutTask.IsCompleted)
  if(!timeoutTask.IsCanceled)
  if(!context.IsReplaying)
  if (!cts.IsCancellationRequested)
  return isWinner;
}
```

[AZ-201 Study Guide](https://www.pass4itsure.com/az-201.html) [AZ-201 Exam Questions](https://www.pass4itsure.com/az-201.html) [AZ-201 Braindumps](https://www.pass4itsure.com/az-201.html)