Ourexam

Higher Quality Better Service!

We offer free update service for one year

Http://www.ourexam.com

Exam : **AZ-302**

Title : Microsoft Azure Solutions Architect Certification Transition

Version : DEMO

1. This is a case study. Case studies are not timed separately. You can use as much exam time as you would like to complete each case. However, there may be additional case studies and sections on this exam. You must manage your time to ensure that you are able to complete all questions included on this exam in the time provided.
To answer the questions included in a case study, you will need to reference information that is provided in the case study, Case studies might contain exhibits and other resources that provide more information about the scenario that is described in the case study. Each question is independent of the other questions in this case study.
At the end of this case study, a review screen will appear. This screen allows you to review your answers and to make changes before you move to the next section of the exam. After you begin a new section, you cannot return to this section.

To start the case study
To display the first question in this case study, click the Next button. Use the buttons in the left pane to explore the content of the case study before you answer the questions. Clicking these buttons displays information such as business requirements, existing environment, and problem statements.
If the case study has an All Information tab, note that the information displayed is identical to the information displayed on the subsequent tabs.
When you are ready to answer a question, click the Question button to return to the question.

You are a developer for Proseware, Inc. You are developing an application that applies a set of governance policies for Proseware's internal services, external services, and applications. The application will also provide a shared library for common functionality.

You develop and deploy a stateful ASP. NET Core 2.1 web application named Policy service to an Azure App Service Web App. The application reacts to events from Azure Event Grid and performs policy actions based on those events.
The application must include the Event Grid Event ID field in all Application Insights telemetry.
Policy service must use Application Insights to automatically scale with the number of policy actions that it is performing.

Log policy
All Azure App Service Web Apps must write logs to Azure Blob storage. All log files should be saved to a container named logdrop Logs must remain in the container for 15 days.

Authentication events
Authentication events are used to monitor users signing in and signing out. All authentication events must be processed by Policy service. Sign outs must be processed as quickly as possible

PolicyLib
You have a shared library named PolicyLib that contains functionality common to all ASP. NET Core web services and applications.
The PolicyLib library must:
- Exclude non-user actions from Application Insights telemetry

- Provide methods that allow a web service to scale itself.
- Ensure that scaling actions do not disrupt application usage

Anomaly detection service
You have an anomaly detection service that analyzes log information for anomalies. It is implemented as an Azure Machine Learning model. The model is deployed as a web service.
If an anomaly is detected, an Azure Function that emails administrators is called by using an HTTP Web Hook.

Health monitoring
All web applications and services have health monitoring at the/health service endpoint.

Policy loss
When you deploy Policy service, policies may not be applied if they were in the process of being applied during the deployment.

Performance issue
When under heavy load, the anomaly detection service undergoes slowdowns and rejects connections

Notification latency
Users report that anomaly detection emails can sometimes arrive several minutes after an anomaly is detected.

Relevant portions of the app files are shown below. Line numbers are included for reference only and include a two-character prefix that denotes the specific tile to which they belong.

```
EventGridController.cs
EG01 public class EventGridController : Controller
EG02 {
EG03   public static AsyncLocal<string> EventId = new AsyncLocal<string>();
EG04   public IActionResult Process([FromBody] string eventsJson)
EG05   {
EG06     var events = JArray.Parse(eventsJson);
EG07
EG08     foreach (var @event in events)
EG09     {
EG10       EventId.Value = @event["id"].ToString();
EG11       if (@event["topic"].ToString().Contains("providers/Microsoft.Storage"))
EG12       {
EG13         SendToAnomalyDetectionService(@event["data"]["url"].ToString());
EG14       }
EG15
EG16       {
EG17         EnsureLogging(@event["subject"].ToString());
EG18       }
EG19     }
EG20     return null;
EG21   }
EG22   private void EnsureLogging(string resource)
EG23   {
EG24     . . .
EG25   }
EG26   private async Task SendToAnomalyDetectionService(string uri)
EG27   {
EG28     var content = GetLogData(uri);
EG29     var scoreRequest = new
EG30     {
EG31       Inputs = new Dictionary<string, List<Dictionary<string, string>>>()
EG32       {
EG33         {
EG34           "input1",
EG35           new List<Dictionary<string, string>>()
```

```
EG17          EnsureLogging(@event["subject"].ToString());
EG18        }
EG19      }
EG20      return null;
EG21    }
EG22    private void EnsureLogging(string resource)
EG23    {
EG24      . . .
EG25    }
EG26    private async Task SendToAnomalyDetectionService(string uri)
EG27    {
EG28      var content = GetLogData(uri);
EG29      var scoreRequest = new
EG30      {
EG31        Inputs = new Dictionary<string, List<Dictionary<string, string>>>()
EG32        {
EG33          {
EG34            "input1",
EG35            new List<Dictionary<string, string>>()
EG36            {
EG37              new Dictionary<string, string>()
EG38              {
EG39                {
EG40                  "logcontent", content
EG41                }
EG42              }
EG43            }
EG44          },
EG45        },
EG46        GlobalParameters = new Dictionary<string, string>() { }
EG47      };
EG48      var result = await (new HttpClient()).PostAsJsonAsync("...", scoreRequest);
EG49      var rawModelResult = await result.Content.ReadAsStringAsync();
EG50      var modelResult = JObject.Parse(rawModelResult);
EG51      if (modelResult["notify"].HasValues)
EG52      {
EG53        . . .
EG54      }
EG55    }
EG56    private (string name, string resourceGroup) ParseResourceId(string resourceId)
```

```
EG29      var scoreRequest = new
EG30      {
EG31        Inputs = new Dictionary<string, List<Dictionary<string, string>>>()
EG32        {
EG33          {
EG34            "input1",
EG35            new List<Dictionary<string, string>>()
EG36            {
EG37              new Dictionary<string, string>()
EG38              {
EG39                {
EG40                  "logcontent", content
EG41                }
EG42              }
EG43            }
EG44          },
EG45        },
EG46        GlobalParameters = new Dictionary<string, string>() { }
EG47      };
EG48      var result = await (new HttpClient()).PostAsJsonAsync("...", scoreRequest);
EG49      var rawModelResult = await result.Content.ReadAsStringAsync();
EG50      var modelResult = JObject.Parse(rawModelResult);
EG51      if (modelResult["notify"].HasValues)
EG52      {
EG53        . . .
EG54      }
EG55    }
EG56    private (string name, string resourceGroup) ParseResourceId(string resourceId)
EG57    {
EG58      . . .
EG59    }
EG60    private string GetLogData(string uri)
EG61    {
EG62      . . .
EG63    }
EG64    static string BlobStoreAccountSAS(string containerName)
EG65    {
EG66      . . .
EG67    }
EG68  }
```

Relevant portions of the app files are shown below. Line numbers are included for reference only and include a two-character prefix that denotes the specific file to which they belong.

```
LoginEvent.cs
LE01 public class LoginEvent
LE02 {
LE03
LE04   public string subject { get; set; }
LE05   public DateTime eventTime { get; set; }
LE06   public Dictionary<string, string> data { get; set; }
LE07   public string Serialize()
LE08   {
LE09     return JsonConvert.SerializeObject(this);
LE10   }
LE11 }
```

You need to meet the scaling requirements for Policy Service.
What should you store in Azure Redis Cache?
A. Session state
B. View State
C. Temp Data
D. Http Context.Items
Answer:A

2. You need to ensure that the Policy service can implement the policy actions.
Which code segment should you insert at line EG07 in Event Grid Controllercs?

A.
```
if (HttpContext.Request.Headers["aeg-event-type"].FirstOrDefault() == "SubscriptionValidation")
{
  return new JsonResult(new
  {
    validationResponse = events[0]["validationCode"]
  });
}
```

B.
```
if (events[0]["eventType"].ToString() == "SubscriptionValidation")
{
  return new JsonResult(new
  {
    validationResponse = events[0]["validationCode"]
  });
}
```

C.
```
if (HttpContext.Request.Headers["aeg-event-type"].FirstOrDefault() == "SubscriptionValidation")
{
  return new JsonResult(new
  {
    validationResponse = events[0]["data"]["validationCode"]
  });
}
```

D.
```
if (events[0]["subject"].ToString() == "SubscriptionValidation")
{
  return new JsonResult(new
  {
    validationResponse = events[0]["data"]["validationCode"]
  });
}
```

A. Option A
B. Option B
C. Option C
D. Option D
Answer:C

3. You need to ensure that the solution can meet the scaling requirements for Policy Service.
Which Azure Application Insights data model should you use?
A. an Application Insights trace
B. an Application Insights metric
C. an Application Insights dependency
D. an Application Insights event
Answer:B

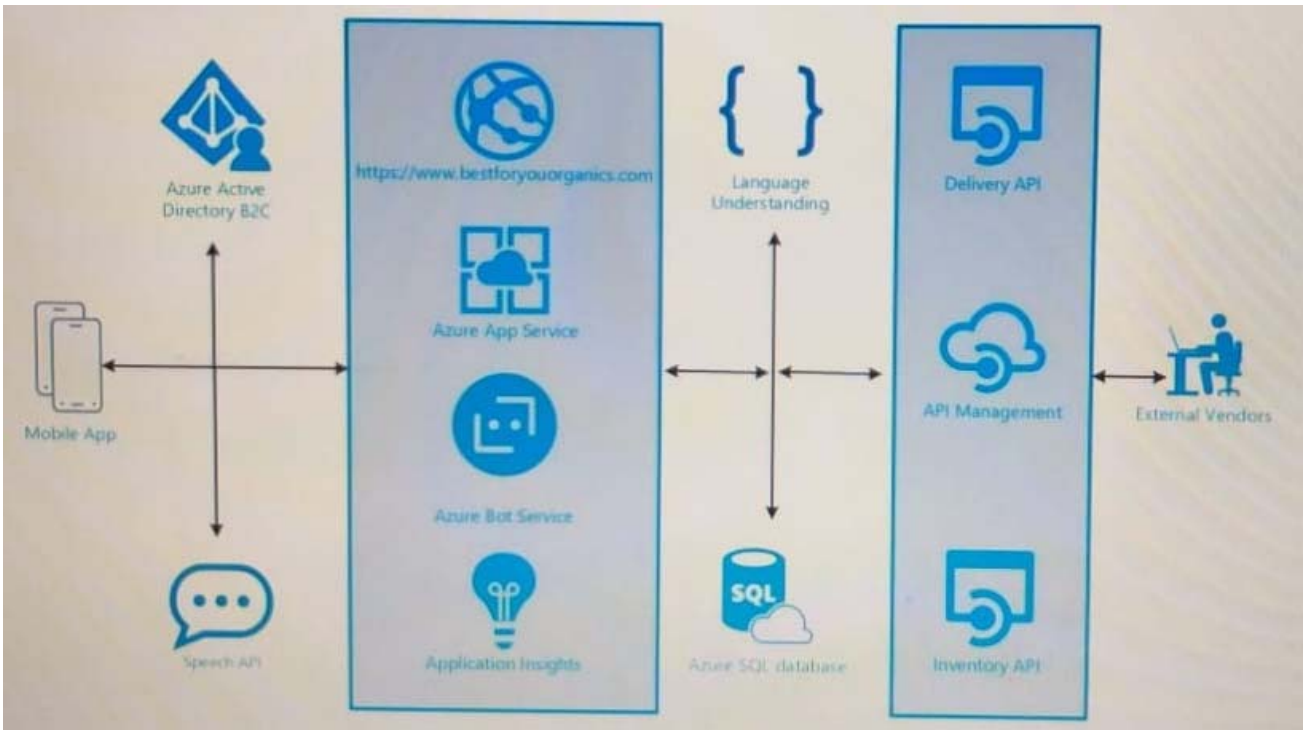
4. Best for You Organics Company is a global restaurant franchise that has multiple locations The company wants to enhance user experiences and vendor integrations. The company plans to implement automated mobile ordering and delivery services.
Best For You Organics hosts an Azure web app at the https://www.bestforyouorganicscom. Users can use

the web app to browse restaurant locations, menu items, nutritional. information, and company information. The company developed and deployed a cross-platform mobile app.

You must develop a chatbot by using the Bot Builder SDK Intelligence Service ('LUIS). The chatbot must allow users to order food for Pickup or den-very.
The chatbot must meet the following requirements
- Ensure that chatbot endpoint can be accessed only by the Bot Framework connector
- Use natural language processing and speech recognition So that users can with the chatbot by using text and voice. Processing mast be server-based.
- Alert users about promotions at local restaurants.
- Enable users to an Older for delivery Pickup by using their voice.
- Greet the user upon sign-in by displaying a graphical Interface that contains action buttons.
- The chatbot greeting Inletface match the 01 'he following example:

Vendors receive and provide updates for the restaurant inventory and delivery services by using Azure API Management hosted APE.
Each vendor uses their own subscription to access each of the APIs.
APIs must meet the following conditions:
- API usage must not exceed 5,000 calls and 50,000 kilobytes of bandwidth per hour per vendor.
- It a vendor 's nearing the number of calls or bandwidth limit the API must trigger email nonfictions to the vendor.
- APIs must prevent API usage spikes On a per-subscription basis by limiting the call rate to 100 calls per
- The Inventory API must be written by using ASP.NET Core and Node.js.
- The API must be updated to provide an interface to Azure SQL Database. Database objects must be managed by using code.
- The Delivery API must be protected by using the OAuth 2.0 protocol with Azure AL-five Directory (Azure AD) when called from the Azure web app. You the Delivery API and web app in AD. You enable OAuth 2.0 in the web app.
- The deliver/ API must update the Products table, the Vendor transactions table, and the Billing table in single transaction.

The Best For You Organics Company architecture team has created the following diagram depicting the expected deployments into Azure:



Delivery API
The Delivery API intermittently throws the following exception:

“System.Data.Entity.Core.EntityCommandExecutionException: An error occurred while executing the command definition. See the inner exception for details. ---> System.Data.SqlClient.SqlExeption: A transport-level error has occurred when receiving results from the server. (provider: Session Provider, error: 19 – Physical connection is not usable)”

Chatbot greeting
The chatbot's greeting does not show the user's name. You need to debug the chatbot locally.

Language processing
Users report that the bot fails to understand when a customer attempts to order dishes that use Italian names,

Relevant portions of the app files are shown below. Line numbers are included for reference only and include a two-character prefix that denotes the specific file to which they belong.

Startup.cs

```
SU01 namespace DeliveryApi
SU02 {
SU03   public class Startup
SU04   {
SU05     public Startup(IConfiguration configuration)
SU06     {
SU07       Configuration = configuration;
SU08     }
SU09     public IConfiguration Configuration { get; }
SU10     public void ConfigureServices(IServiceCollection services)
SU11     {
SU12       services.AddDbContext<RestaurantsContext>(opt =>
SU13         opt.UseSqlServer(Configuration.GetSection("ConnectionStrings")["RestaurantsDatabase"],
SU14          sqlServerOptionsAction: sqlOptions =>
SU15          {
SU16            . . .
SU17          }));
SU18       services.AddMvc()
SU19         .SetCompatibilityVersion(CompatibilityVersion.Version_2_1);
SU20     }
SU21     public void Configure(IApplicationBuilder app)
SU22     {
SU23       app.UseMvc();
SU24     }
SU25   }
SU26 }
```

In this section, you will see one or more sets of questions with the same scenario and problem. Each question presents a unique solution to the problem, and you must determine whether the solution meets the stated goals. More than one solution might solve the problem. it is also possible that none of the solutions solve the problem.
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.

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution. Determine whether the solution meets the stated goals.
You need to meet the vendor notification requirement.
Solution: Update the Delivery API to send emails by using a cloud-based email service.
Does the solution meet the goal?
A. Yes
B. No
Answer:B

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. Determine whether the solution meets the stated goals.
You need to meet the vendor notification requirement.
Solution: Create and apply a custom outbound Azure API Management policy.
Does the solution meet the goal?

A. Yes
B. No
Answer:B