Lab: Cosmos DB and Azure Function

Overview

This lab introduces you how to create Azure Cosmos DB account, set up for work and adding collection. Next you will learn how create function instance triggered by HTTP request and how to add output of Cosmos DB source

Task 1: Create Azure Cosmos DB

> In this section you will learn how to create Azure Cosmos DB account from Azure Portal.
>
> If you already have a Azure Storage Cosmos DB account resource, then skip this task

1. Sign in the Azure portal at https://portal.azure.com
2. On the left Hub menu click Azure Cosmos DB.
3. On the Azure Cosmos DB blade click the Add button marked with a plus sign.
4. On the New account blade provide the following configuration:
    - ID: **cosmosdb-student0X**
    - API: **SQL**
    - Subscription: **ChmurowiskoLab**
    - Resource Group: **YOUR_RESOURCE_GROUP**
    - Location: **West Europe**
    - Apply Free Tier Discount: **Do not apply**
    - Select **Enable geo-redundancy**
    - Keep default values for the rest of the settings
    5. Wait for the Azure Cosmos DB account to be deployed.

Task 2: Add database and collection

> In this section you will learn how to add database and first collection from Azure Portal.

1. Open the newly created Azure Cosmos DB.
2. Familiarize yourself with the main dashboard of Azure Document DB.
3. On the main Overview page click on the Add Container button marked with a plus sign.
4. On the Add Container blade provide the following configurations:
    - Database id:
        - Create new: **student-database**
    - Throughput: **400 RU**
    - Container Id: **student-collection**
    - Partition Key: **/id**
    - Keep default values for the rest of the settings
5. The database and first container have been created.

Task 3: Create Azure Functions

> In this section you will learn how to create Azure Function from Azure Portal.
>
> You may also use Function App created in previous labs.

1. On the Azure Portal select Create new Resource -> Compute -> Functions App
2. Provide following configurations:
    - App name: **student[yourname]fa**
    - Subscription: **ChmurowiskoLab**
    - Resource Group: **YOUR_RESOURCE_GROUP**
    - OS: **Windows**
    - Hosting Plan: **Consumption Plan**
    - Location: **West Europe**
    - Runtime Stack: **.NET**
    - Storage: **Create New**
    - Application Insights: **Leave default**
    - Keep default values for the rest of the settings
3. Wait for the Function App to be deployed and open it.

Task 4: Create an instance of function

> In this section you will learn how create an instance of function triggered by HTTP request and how to add our definition of code.
>
> You may also use HttpTrigger function created in previous labs.

1. On the Functions page click on the New function button marked with a plus sign.
2. Select HTTP trigger template
3. On the HTTP Trigger New Function blade provide the following configurations:
    - Name: **HttpTriggerCSharp**
    - Authorization level: **Anoymous**

4. On the HttpTriggerCSharp page function change code in run.csx file:

```
 #r "Newtonsoft.Json"

using System.Net;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Primitives;
using Newtonsoft.Json;

public static IActionResult Run(HttpRequest req, ILogger log, out object outputDocument)
{
    log.LogInformation("C# HTTP trigger function processed a request.");

    string name = req.Query["name"];

    string requestBody = new StreamReader(req.Body).ReadToEnd();
    dynamic data = JsonConvert.DeserializeObject(requestBody);
    outputDocument = data;
    name = name ?? data?.name;

    return name != null
        ? (ActionResult)new OkObjectResult($"Hello, {name}")
        : new BadRequestObjectResult("Please pass a name on the query string or in the request body");
}
```

Task 5: Add Azure Cosmos DB output

> In this section you will learn how to add Azure Cosmos DB account connection as output in Azure Functions.

1. On the HttpTriggerCSharp page function click on **Integrate**.
2. On the Integrate page click on **Add Output** and select **Azure Cosmos DB**.
3. Next on Azure Comos DB Output section provide the following configurations:
    - Document parameter name: **outputDocument**
    - Database name: **student-database**
    - Unselect Use Function return value
    - Collection Name: **student-collection**
    - Partition key: **Leave empty**
    - Azure Cosmos DB account connection:
        - New: Select created Azure Cosmos DB from previous lab **cosmosdb-student0X**
4. On the HttpTriggerCSharp page function click on **Run/Test**, you also can add your JSON definition of message on the Input blade in section Request body.
5. Next go on your resource of Azure Cosmos DB - cosmosdb-student0X, click on Data Explorer and check in student-collection new added document by Azure Functions.