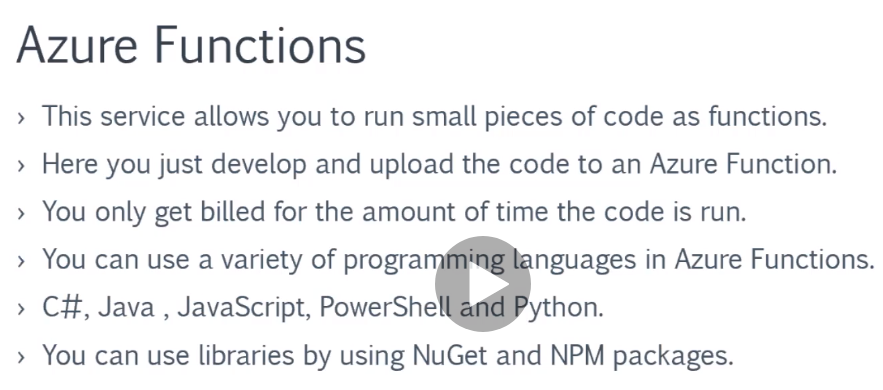
**Lesson14 What is Azure Functions**

**Notes: -**

**1-the high advantage point for the Azure function is in nature, so you need to manage the underlying infrastructure that is used to host your code**

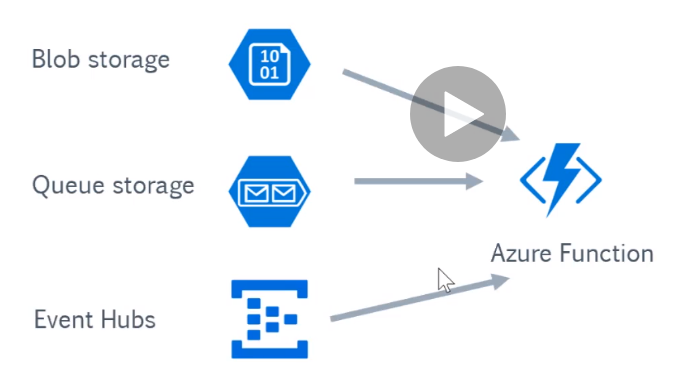


**Invoking your Function Ways**

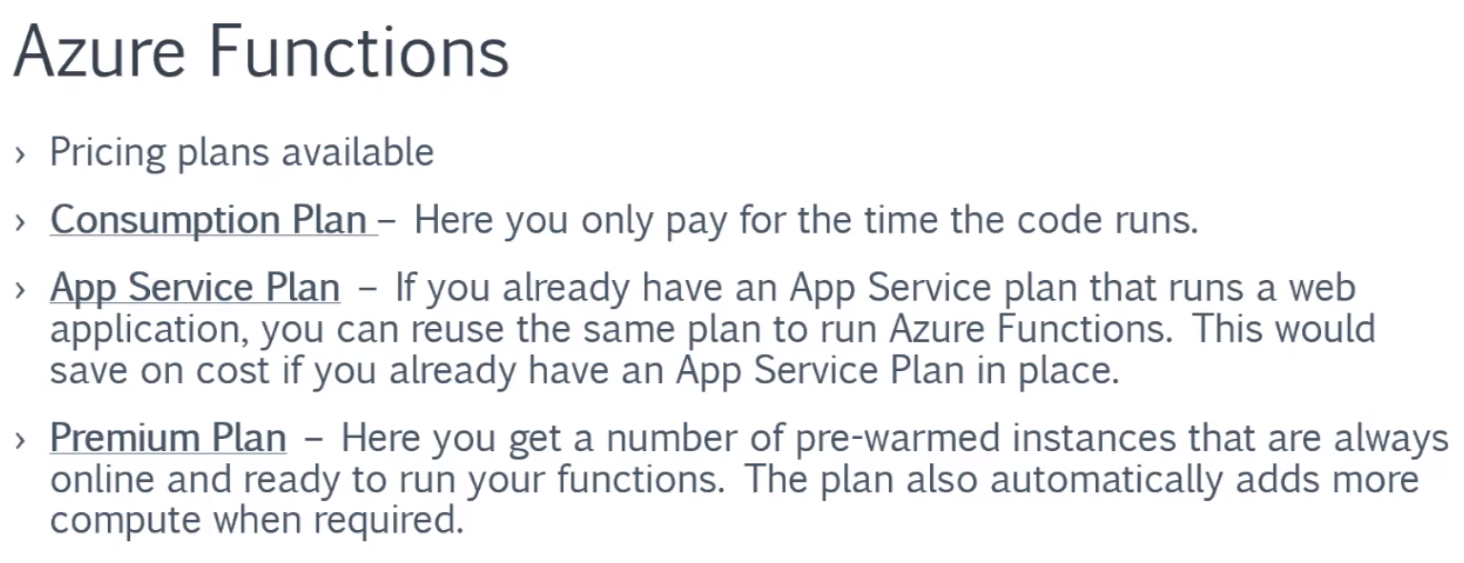
**1-by using HTTP triggers**

**2-by using Time triggers if you want to run your JavaScript code in timer trigger**

**3-by using Blob Storage, Queue storage, Event Hubs**



**Azure Functions pricing**



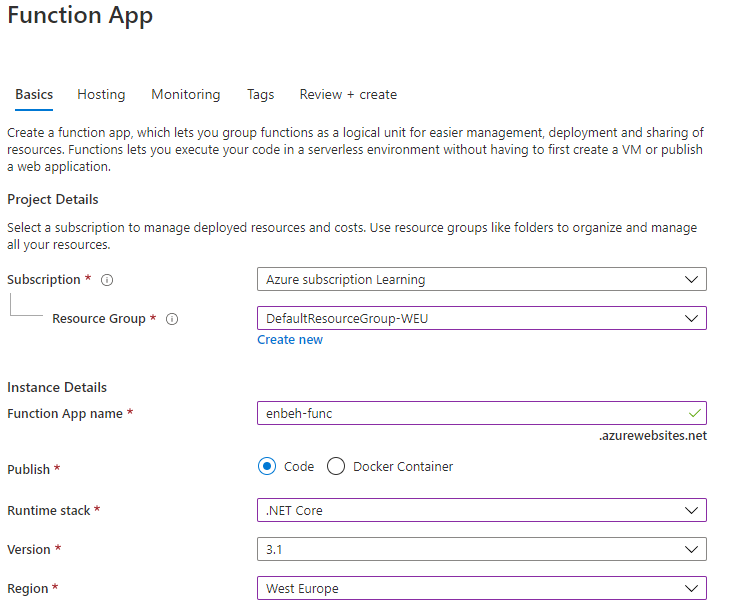
**Implements of Azure Functions**

**Notes: -**

**1-Azure function is container to group of functions**

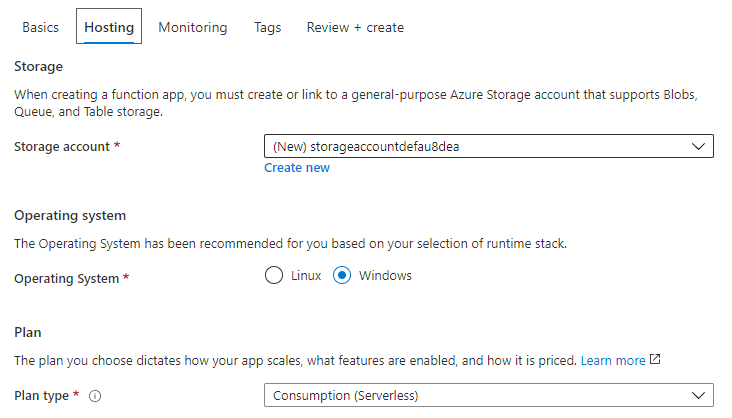
**2-you have to create storage account to the Azure Functions**

**3-on Azure Portal > create function app and set the basic information**

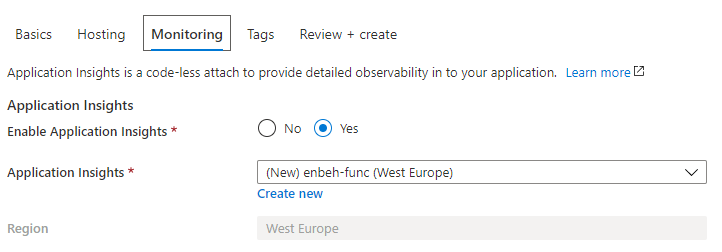


**4-on Hosting tab we see that we select the storage account that used to store function we declare in this resource group**

**(We see also that we have 3 types of service plan, we choose consumption service plan which is cold service take some time (pay as you go))**



**5-on Management tab we select the application insight that used to store the logs that we set inside the function to detect info / issue when the function running**



**6-we see that the resources created is**

**Function App**

**App Service Plan**

**Storage Account**

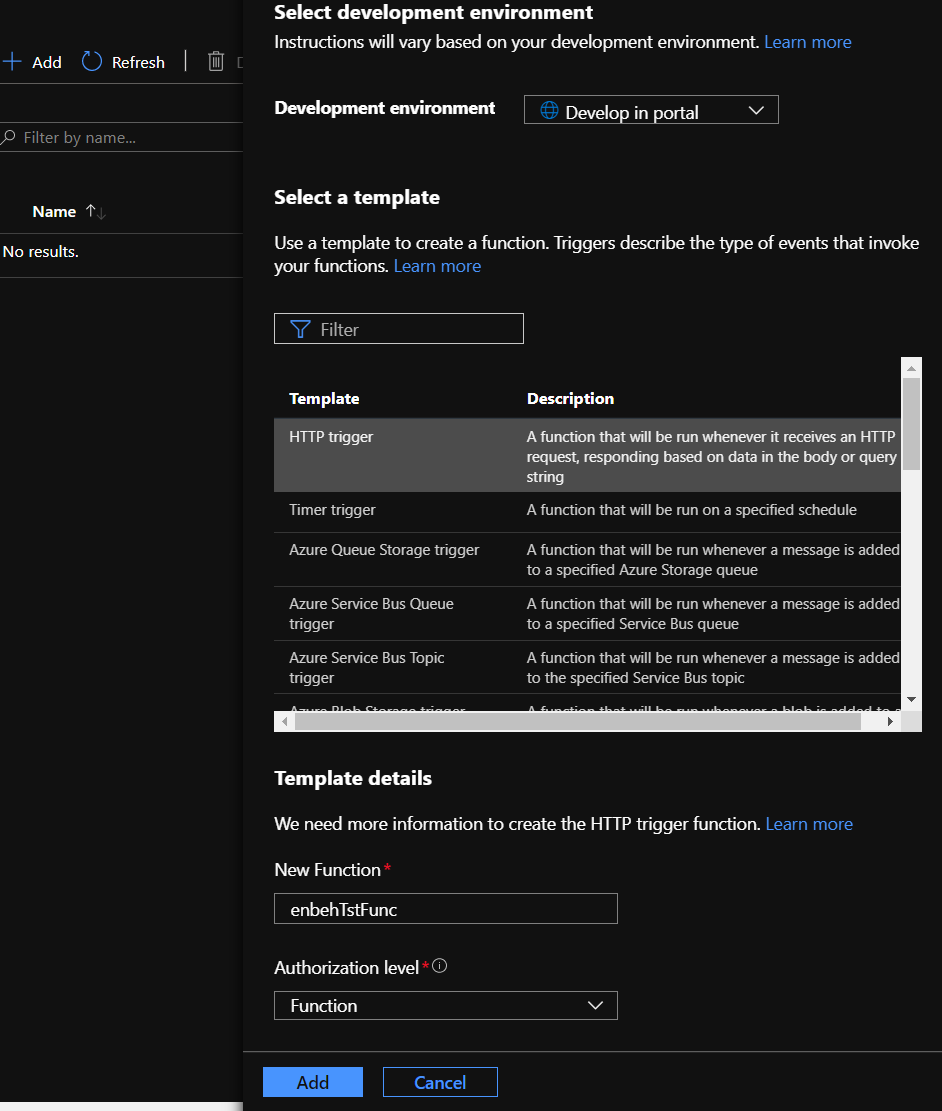
**Application Insights**

**Adding Functions in Azure Functions**

**Steps: -**

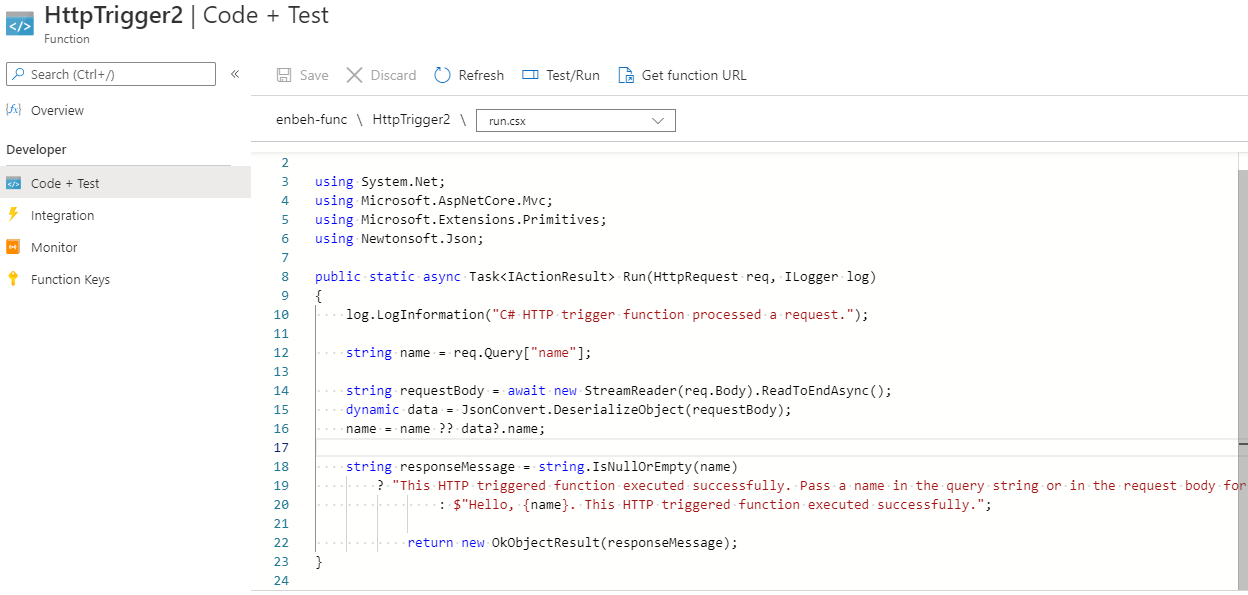
**1-we see that Azure App function created > Function > Add > HTTP Trigger**

**(You will see different types of Azure functions such as timer function / Azure service bus queue, RabbitMQ trigger, etc…)**

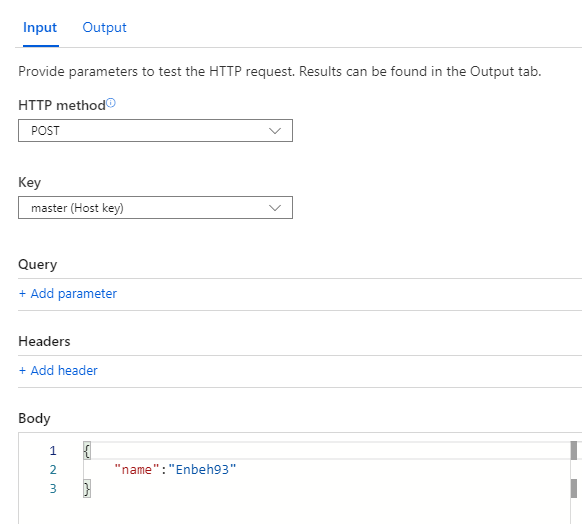


**2-on Azure function > function > Http Trigger > Code and test**

**3-we see sample code that accept request Post with optional parameter name and return response**



**It will store the log in the application insight**



**You can test it under click on Test button or click on the get function URL to get the URL and test it through using postman**

**By using PostMan , you can copy the Get Function URL from the azure function and hit the request**

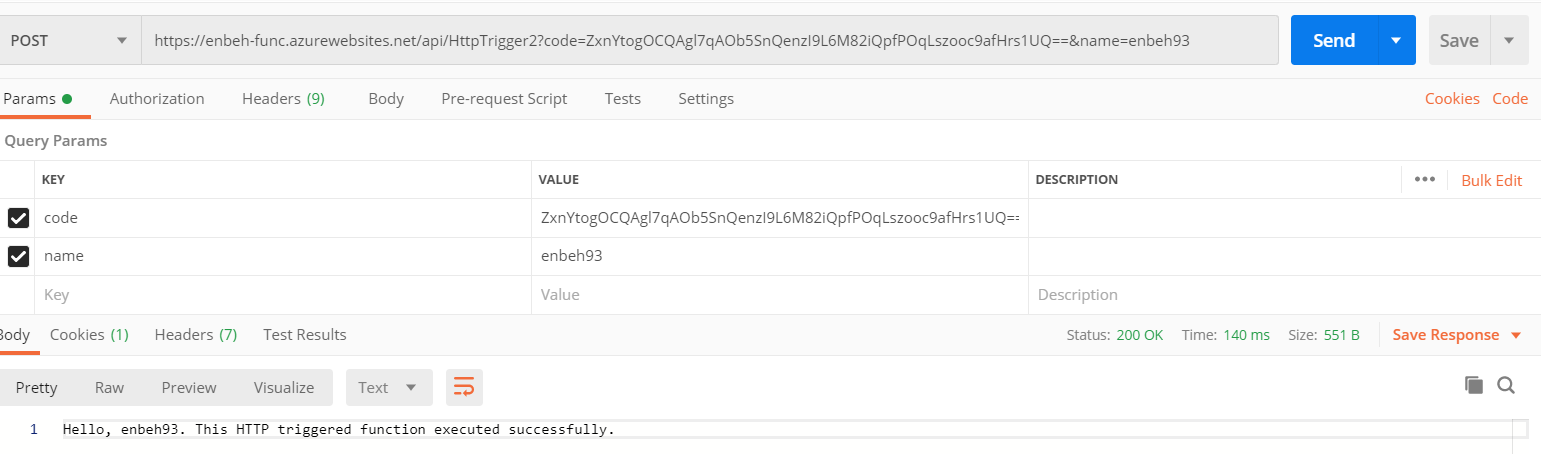
**HTTPPOST**

[**https://enbehfunc.azurewebsites.net/api/enbehTstFunc?code=PMyu9nJ3cMZW6NxtQWFvVu0RtORQIZURrQeucWHnqUiON7Y12w4a4w==**](https://enbehfunc.azurewebsites.net/api/enbehTstFunc?code=PMyu9nJ3cMZW6NxtQWFvVu0RtORQIZURrQeucWHnqUiON7Y12w4a4w==)

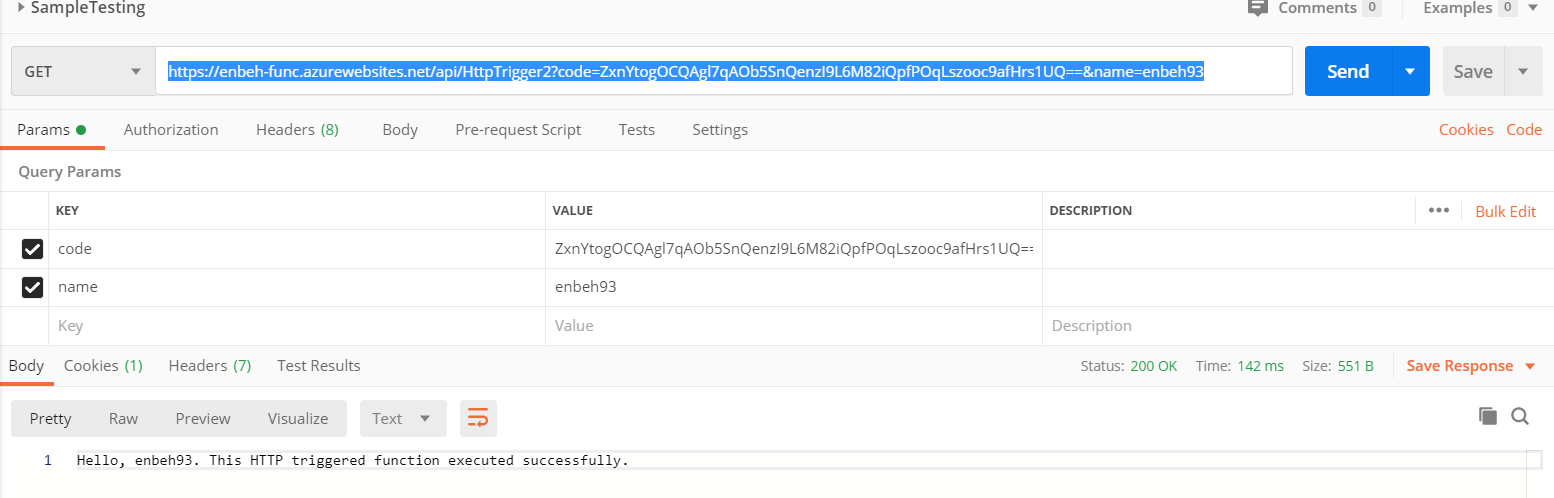
**{**

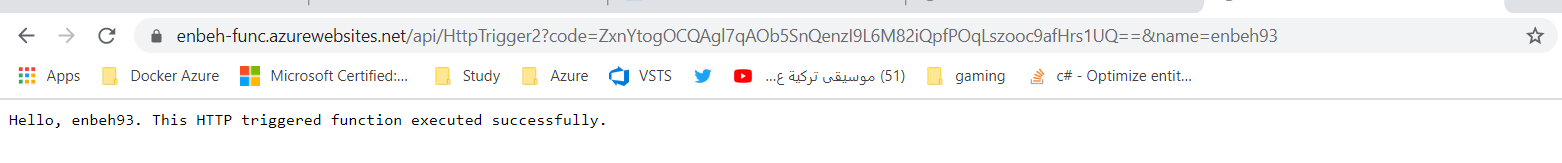
**"name": "Azure123"**

**}**



**Or by using Get with pass parameter**





**Azure Function Lab Part 02**

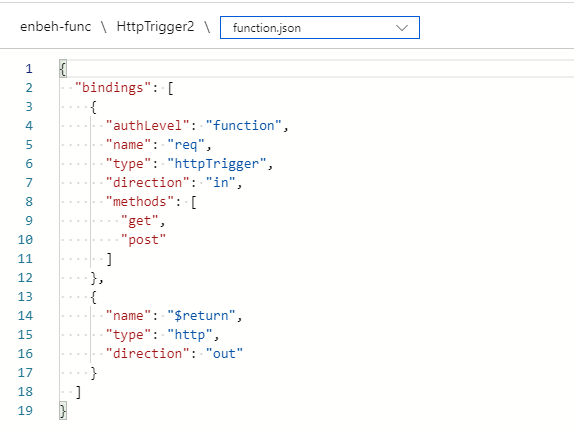
**Notes: -**

**1-we see that the azure function run under CSX file (c sharp script file) that run your c sharp code in azure function**

**(You see this expression #r which used to reference library such as newtonsoft. json)**



**2-we see that the azure function has function.json which is configuration of the azure function that created.**



**3-on azure portal > function > app files > you see that there is configuration of the azure function app as container of all functions that declare**

**#r "Newtonsoft.Json"**

**using System.Net;**

**using Microsoft.AspNetCore.Mvc;**

**using Microsoft.Extensions.Primitives;**

**using Newtonsoft.Json;**

**//we take two query parameter firstName , lastName and using Get request**

**public static async Task<IActionResult> Run(HttpRequest req, ILogger log){**

**log.LogInformation("C# HTTP trigger function processed a request.");**

**string fname = req.Query["firstName"].ToString();**

**string lname = req.Query["lastName"].ToString();**

**string response = $"hello {fname} {lname}";**

**return new OkObjectResult(response);}**

**4-for POST function we must create another function inside the azure function**

**#r "Newtonsoft.Json"**

**using System.Net;**

**using Microsoft.AspNetCore.Mvc;**

**using Microsoft.Extensions.Primitives;**

**using Newtonsoft.Json;**

**public static async Task<IActionResult> Run(HttpRequest req, ILogger log){**

**log.LogInformation("C# HTTP trigger function processed a request.");**

**string requestBody = await new StreamReader(req.Body).ReadToEndAsync();**

**dynamic data = JsonConvert.DeserializeObject(requestBody);**

**var fname = data?.firstName;**

**var lname = data?.lastName;**

**string responseMessage = $"{fname} {lname}";**

**return new OkObjectResult(responseMessage);}**

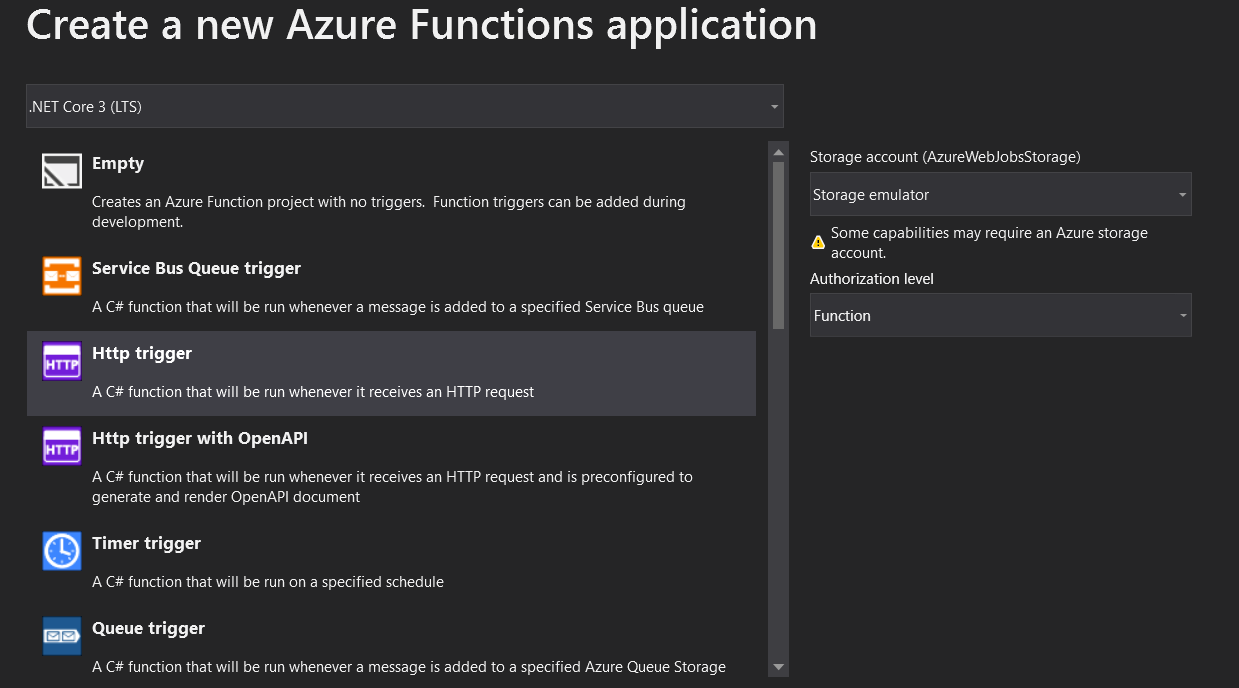
**Apply from VS 2019**

**Notes: -**

**1-you have to remove WEBSITE\_RUN\_FROM\_PACKAGE from Azure function from**

**Azure function > configuration**

**2-we can apply it through VS 2019 > new Azure Function > choose HTTP Trigger**



**3-we set the following code as below**

**using System;**

**using System.IO;**

**using System.Threading.Tasks;**

**using Microsoft.AspNetCore.Mvc;**

**using Microsoft.Azure.WebJobs;**

**using Microsoft.Azure.WebJobs.Extensions.Http;**

**using Microsoft.AspNetCore.Http;**

**using Microsoft.Extensions.Logging;**

**using Newtonsoft.Json;**

**namespace SampleFunc{**

**public static class Function1{**

**//this function accept get or post request**

**[FunctionName("CallData")]**

**public static async Task<IActionResult> Run(**

**[HttpTrigger(AuthorizationLevel.Function, "get", "post", Route = null)] HttpRequest req,**

**ILogger log){**

**log.LogInformation("C# HTTP trigger function processed a request.");**

**string fname = req.Query["fname"];**

**string lname = req.Query["lname"];**

**string requestBody = await new StreamReader(req.Body).ReadToEndAsync();**

**dynamic data = JsonConvert.DeserializeObject(requestBody);**

**fname = fname ?? data?.fname;**

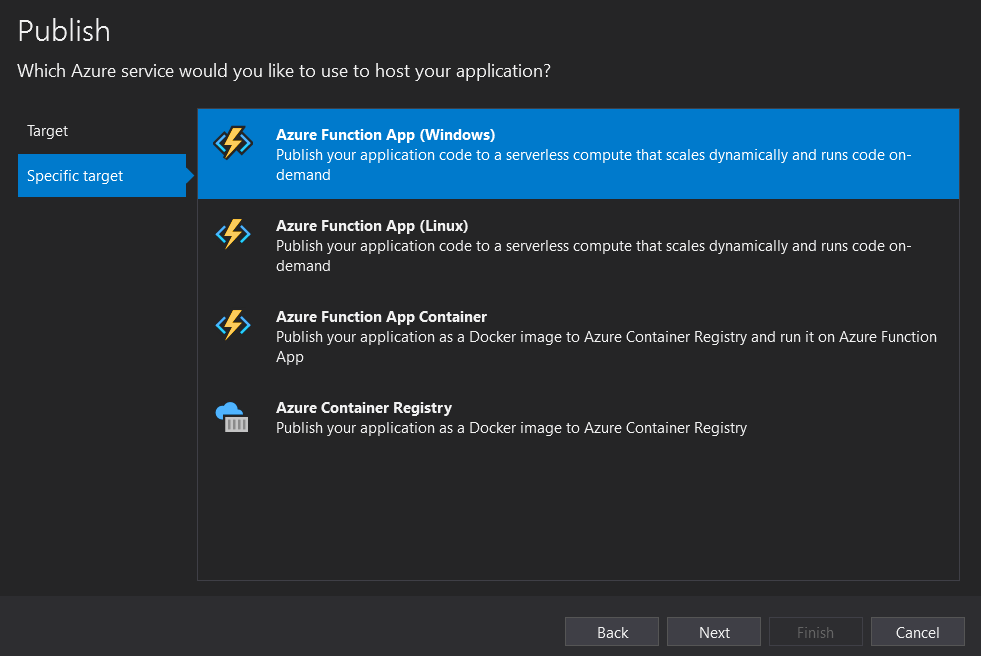
**lname = lname ?? data?.lname;**

**string responseMessage = string.IsNullOrEmpty(fname) && string.IsNullOrEmpty(lname)**

**? $"This HTTP Request Accept GET / POST , Hello {fname} {lname}" : $"Hello {fname} {lname}";**

**return new OkObjectResult(responseMessage);}}}**

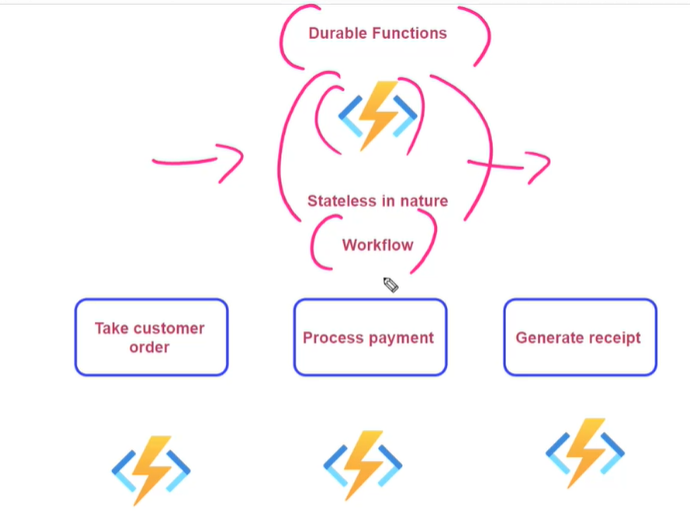
**4-on project > publish > select Azure function app (windows)**



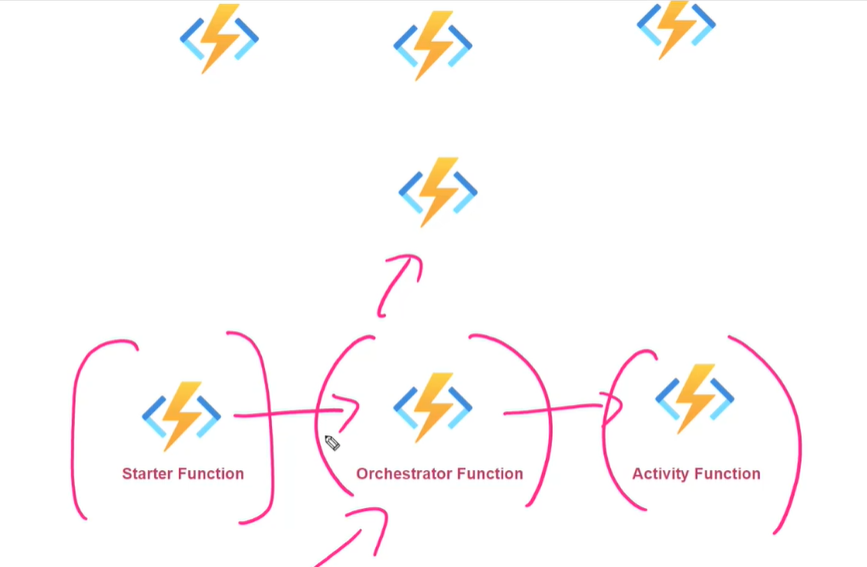
**Azure Durable function**

**Notes: -**

**1-in order to apply sequential workflow flow such as when azure function is executed then execute the next azure function and so on.**



**Or execute multiple azure functions**



**2-there are three types of functions**

**A-Stater function: which used to create orchestrator function**

**B-Orchestrator function: which used to call many activity function and control which activity function will be executed after the current one execute**

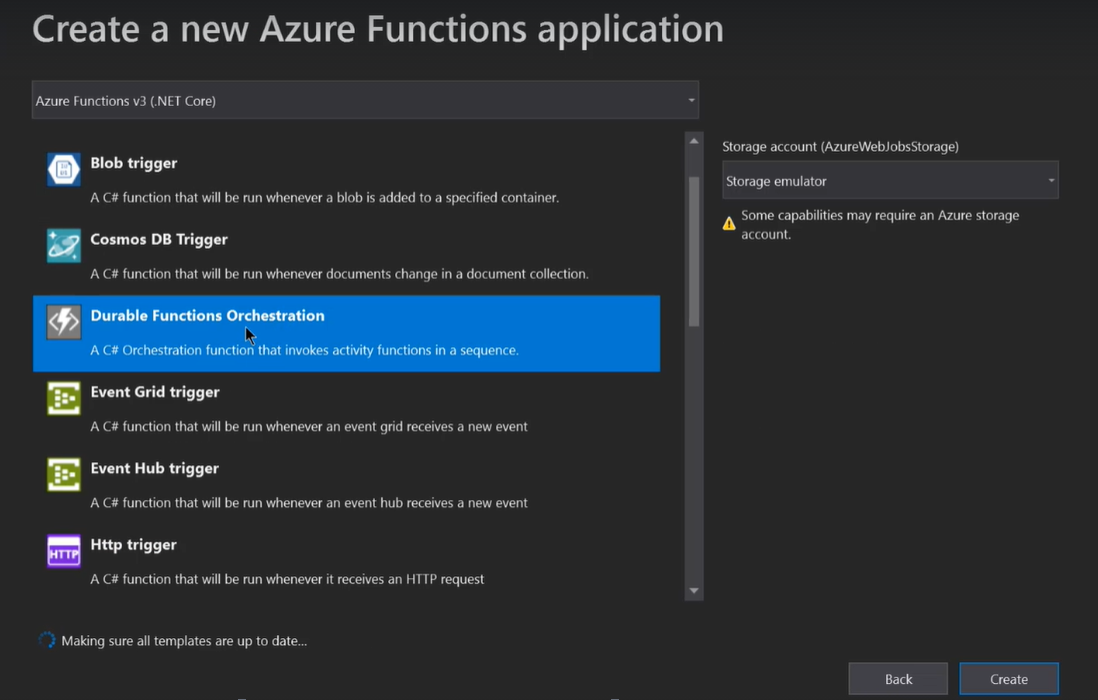
**C-Activity function: which represent the actual code executed**

**Steps: -**

**1-on Azure Portal > create durable azure function**

**(Make sure to remove WEBSITE\_RUN\_FROM\_PACKAGE from configuration)**

**2-on VS2019 > create new Azure Durable function orchestration**



**3-you will see that in you Durable functions orchestration you will see the starter function and orchestrator function and two activity function and we see that orchectrator functions call one of the activity function twice**

**using Microsoft.Azure.WebJobs;**

**using Microsoft.Azure.WebJobs.Extensions.DurableTask;**

**using Microsoft.Azure.WebJobs.Extensions.Http;**

**using Microsoft.Extensions.Logging;**

**using System.Collections.Generic;**

**using System.Net.Http;**

**using System.Threading.Tasks;**

**namespace AzureDurableFunc{**

**public static class Function1{**

**//this orchestrator function call the activity function as below**

**[FunctionName("Function1")]**

**public static async Task<List<string>> RunOrchestrator(**

**[OrchestrationTrigger] IDurableOrchestrationContext context){**

**var outputs = new List<string>();**

**// Replace "hello" with the name of your Durable Activity Function.**

**outputs.Add(await context.CallActivityAsync<string>("SayHello", "Tokyo"));**

**outputs.Add(await context.CallActivityAsync<string>("SayHello", "Seattle"));**

**outputs.Add(await context.CallActivityAsync<string>("ExecFunc", ("Mohammed Enbeh",23)));**

**// returns ["Hello Tokyo!", "Hello Seattle!", "Hello London!"]**

**return outputs;}**

**//this activity function contians the our code we need to execute**

**[FunctionName("SayHello")]**

**public static string SayHello([ActivityTrigger] string name, ILogger log){**

**log.LogInformation($"Saying hello to {name}.");**

**return $"Hello {name}!";}**

**//this activity function contians the our code we need to execute**

**[FunctionName("ExecFunc")]**

**public static string ExecFunc([ActivityTrigger] (string name,int age) person, ILogger log){**

**log.LogInformation($"Saying hello to {person.name} , your age is {person.age}.");**

**return $"Hello {person.name} , {person.age}!";}**

**//this is the starter function that call the orchestrator function**

**[FunctionName("Function1\_HttpStart")]**

**public static async Task<HttpResponseMessage> HttpStart(**

**[HttpTrigger(AuthorizationLevel.Anonymous, "get", "post")] HttpRequestMessage req,**

**[DurableClient] IDurableOrchestrationClient starter,**

**ILogger log){**

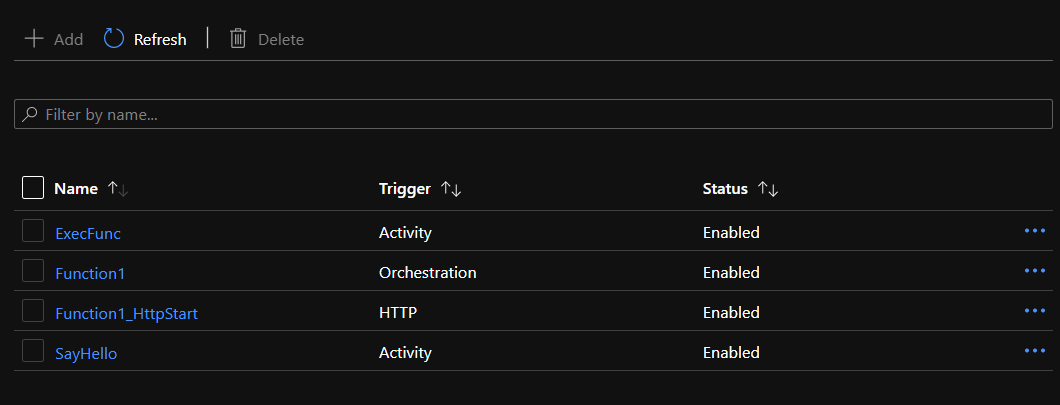
**// Function input comes from the request content.**

**string instanceId = await starter.StartNewAsync("Function1", null);**

**log.LogInformation($"Started orchestration with ID = '{instanceId}'.");**

**return starter.CreateCheckStatusResponse(req, instanceId);}}}**

**(You will see that it will create two activity function and starter function and orchestration function as below)**



**4-copy function url of the starter function and execute it will give response as below**

**{**

**"id": "a06e52891b2d47baaec2530218ae5633",**

**"statusQueryGetUri": "https://enbehfunc.azurewebsites.net/runtime/webhooks/durabletask/instances/a06e52891b2d47baaec2530218ae5633?taskHub=EnbehFunc&connection=Storage&code=W1UsiYXmY/Yo5qfRZjrK0rcTpv55Ws2qzXl7N2kBCQDrYekADOTnzg==",**

**"sendEventPostUri": "https://enbehfunc.azurewebsites.net/runtime/webhooks/durabletask/instances/a06e52891b2d47baaec2530218ae5633/raiseEvent/{eventName}?taskHub=EnbehFunc&connection=Storage&code=W1UsiYXmY/Yo5qfRZjrK0rcTpv55Ws2qzXl7N2kBCQDrYekADOTnzg==",**

**"terminatePostUri": "https://enbehfunc.azurewebsites.net/runtime/webhooks/durabletask/instances/a06e52891b2d47baaec2530218ae5633/terminate?reason={text}&taskHub=EnbehFunc&connection=Storage&code=W1UsiYXmY/Yo5qfRZjrK0rcTpv55Ws2qzXl7N2kBCQDrYekADOTnzg==",**

**"purgeHistoryDeleteUri": "https://enbehfunc.azurewebsites.net/runtime/webhooks/durabletask/instances/a06e52891b2d47baaec2530218ae5633?taskHub=EnbehFunc&connection=Storage&code=W1UsiYXmY/Yo5qfRZjrK0rcTpv55Ws2qzXl7N2kBCQDrYekADOTnzg=="**

**}**

**(Copy the stausQueryGetUri and you will see that it will execute all the three activity functions as below)**

**{"name": "Function1",**

**"instanceId": "a06e52891b2d47baaec2530218ae5633",**

**"runtimeStatus": "Completed",**

**"input": null,**

**"customStatus": null,**

**//you will see that it will execute the three activity functions as below**

**"output": [**

**"Hello Tokyo!",**

**"Hello Seattle!",**

**"Hello Mohammed Enbeh , 23!"],**

**"createdTime": "2021-06-16T20:15:19Z",**

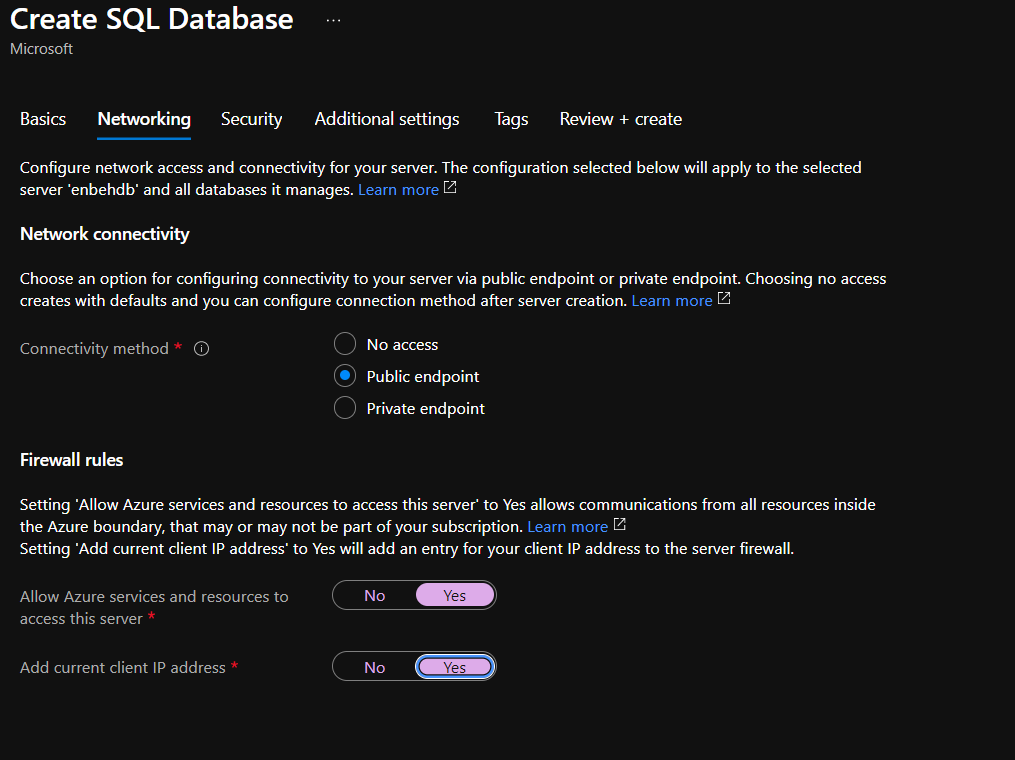
**"lastUpdatedTime": "2021-06-16T20:15:20Z"}**

**Azure Function SQL database**

**Notes: -**

**1-we create Azure SQL database with choose Basic service plan**

**(With set public endpoint and allow firewall rules as below)**



**2-we copy the connection string from the Azure SQL database**

**4-on Azure function we select the subscription and resource group and then create storage account**

**3-on .net core project we create Azure Function > Http Trigger > set the following code**

**using AzureSolPro.Models;**

**using Microsoft.AspNetCore.Http;**

**using Microsoft.AspNetCore.Mvc;**

**using Microsoft.Azure.WebJobs;**

**using Microsoft.Azure.WebJobs.Extensions.Http;**

**using Microsoft.Extensions.Logging;**

**using Newtonsoft.Json;**

**using System.Collections.Generic;**

**using System.Data.SqlClient;**

**using System.Threading.Tasks;**

**namespace AzureSolPro{**

**public static class HttpGetFunc{**

**[FunctionName("HttpGetFunc")]**

**public static async Task<IActionResult> Run(**

**[HttpTrigger(AuthorizationLevel.Function, "get", Route = null)] HttpRequest req,**

**ILogger log){**

**var lst = new List<Course>();**

**var \_connection\_string = "Server=tcp:enbehdb.database.windows.net,1433;Initial Catalog=enbehdb;Persist Security Info=False;User ID=m.enbeh;Password=Mohammed1993$;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;";**

**var \_statement = "select CourseID,CourseName,Rating from Course";**

**var \_conn = new SqlConnection(\_connection\_string);**

**\_conn.Open();**

**var \_sqlCommand = new SqlCommand(\_statement, \_conn);**

**using (SqlDataReader \_reader = \_sqlCommand.ExecuteReader()){**

**while (\_reader.Read()){**

**var \_course = new Course(){**

**CourseID = \_reader.GetInt32(0),**

**CourseName = \_reader.GetString(1),**

**Rating = \_reader.GetDecimal(2)};**

**lst.Add(\_course);}}**

**\_conn.Close();**

**return new OkObjectResult(lst);}}}**

**POST Operation**

**1-on VS 2019 > create Azure Function > HTTP Trigger**

**using System;**

**using System.IO;**

**using System.Threading.Tasks;**

**using Microsoft.AspNetCore.Mvc;**

**using Microsoft.Azure.WebJobs;**

**using Microsoft.Azure.WebJobs.Extensions.Http;**

**using Microsoft.AspNetCore.Http;**

**using Microsoft.Extensions.Logging;**

**using Newtonsoft.Json;**

**using AzureSolPro.Models;**

**using System.Data.SqlClient;**

**using System.Data;**

**namespace AzureSolPro{**

**public static class HttpPostFunc{**

**[FunctionName("HttpPostFunc")]**

**public static async Task<IActionResult> Run(**

**[HttpTrigger(AuthorizationLevel.Function, "get", "post", Route = null)] HttpRequest req,**

**ILogger log){**

**try{**

**string requestBody = await new StreamReader(req.Body).ReadToEndAsync();**

**var data = JsonConvert.DeserializeObject<Course>(requestBody);**

**var \_connection\_string = "Server=tcp:enbehdb.database.windows.net,1433;Initial Catalog=enbehdb;Persist Security Info=False;User ID=m.enbeh;Password=Mohammed1993$;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;";**

**var \_statement = "insert into Course(CourseId,CourseName,Rating) values(@id,@name,@rating)";**

**var \_conn = new SqlConnection(\_connection\_string);**

**\_conn.Open();**

**using (var \_command = new SqlCommand(\_statement, \_conn)){**

**\_command.Parameters.Add("@id", SqlDbType.Int).Value = data.CourseID;**

**\_command.Parameters.Add("@name", SqlDbType.VarChar, 1000).Value = data.CourseName;**

**\_command.Parameters.Add("@rating", SqlDbType.Decimal).Value = data.Rating;**

**\_command.CommandType = CommandType.Text;**

**\_command.ExecuteNonQuery();}**

**\_conn.Close();}**

**catch{return new BadRequestResult();}**

**return new OkObjectResult("Course Added");}}}**

**2-we make publish to Azure function app**

**Using Connection string**

**Notes: -**

**1-we will be using environment variable in order to access to connection string that deployed on the Azure App function**

**//On GET function**

**using AzureSolPro.Models;**

**using Microsoft.AspNetCore.Http;**

**using Microsoft.AspNetCore.Mvc;**

**using Microsoft.Azure.WebJobs;**

**using Microsoft.Azure.WebJobs.Extensions.Http;**

**using Microsoft.Extensions.Logging;**

**using Newtonsoft.Json;**

**using System;**

**using System.Collections.Generic;**

**using System.Data.SqlClient;**

**using System.Threading.Tasks;**

**namespace AzureSolPro{**

**public static class HttpGetFunc{**

**[FunctionName("HttpGetFunc")]**

**public static async Task<IActionResult> Run(**

**[HttpTrigger(AuthorizationLevel.Function, "get", Route = null)] HttpRequest req,**

**ILogger log){**

**var lst = new List<Course>();**

**//we will use the syntax of the as following cutomName\_ConnectionString**

**var \_connection\_string = Environment.GetEnvironmentVariable("EnbehDb\_ConnectionString");**

**var \_statement = "select CourseID,CourseName,Rating from Course";**

**var \_conn = new SqlConnection(\_connection\_string);**

**\_conn.Open();**

**var \_sqlCommand = new SqlCommand(\_statement, \_conn);**

**using (SqlDataReader \_reader = \_sqlCommand.ExecuteReader()){**

**while (\_reader.Read()){**

**var \_course = new Course(){**

**CourseID = \_reader.GetInt32(0),**

**CourseName = \_reader.GetString(1),**

**Rating = \_reader.GetDecimal(2)};**

**lst.Add(\_course);}}**

**\_conn.Close();**

**return new OkObjectResult(lst);}}}**

**//On POST function**

**using System;**

**using System.IO;**

**using System.Threading.Tasks;**

**using Microsoft.AspNetCore.Mvc;**

**using Microsoft.Azure.WebJobs;**

**using Microsoft.Azure.WebJobs.Extensions.Http;**

**using Microsoft.AspNetCore.Http;**

**using Microsoft.Extensions.Logging;**

**using Newtonsoft.Json;**

**using AzureSolPro.Models;**

**using System.Data.SqlClient;**

**using System.Data;**

**namespace AzureSolPro{**

**public static class HttpPostFunc{**

**[FunctionName("HttpPostFunc")]**

**public static async Task<IActionResult> Run(**

**[HttpTrigger(AuthorizationLevel.Function, "get", "post", Route = null)] HttpRequest req,**

**ILogger log){**

**try{**

**string requestBody = await new StreamReader(req.Body).ReadToEndAsync();**

**var data = JsonConvert.DeserializeObject<Course>(requestBody);**

**//we will use the syntax of the as following cutomName\_ConnectionString**

**var \_connection\_string = Environment.GetEnvironmentVariable("EnbehDb\_ConnectionString");**

**var \_statement = "insert into Course(CourseId,CourseName,Rating) values(@id,@name,@rating)";**

**var \_conn = new SqlConnection(\_connection\_string);**

**\_conn.Open();**

**using (var \_command = new SqlCommand(\_statement, \_conn)){**

**\_command.Parameters.Add("@id", SqlDbType.Int).Value = data.CourseID;**

**\_command.Parameters.Add("@name", SqlDbType.VarChar, 1000).Value = data.CourseName;**

**\_command.Parameters.Add("@rating", SqlDbType.Decimal).Value = data.Rating;**

**\_command.CommandType = CommandType.Text;**

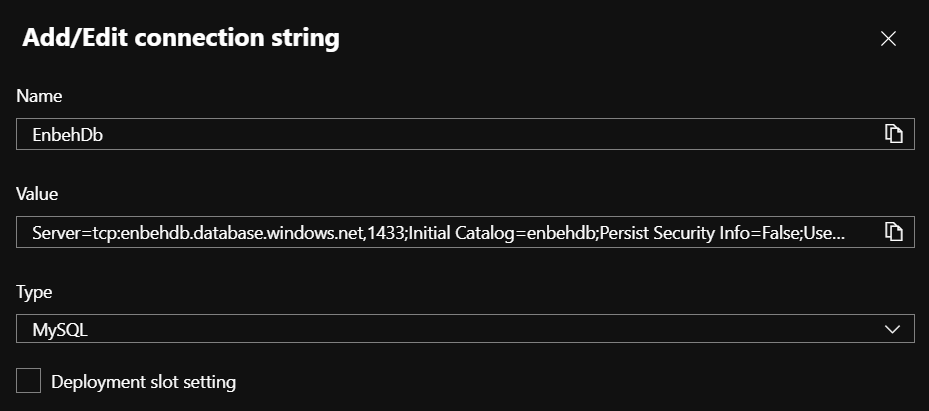
**\_command.ExecuteNonQuery();}**

**\_conn.Close();}**

**catch{return new BadRequestResult();}**

**return new OkObjectResult("Course Added");}}}**

**2-on Azure function > configuration we add connection string as below**



**Using Azure App Service with Azure Function**

**Notes: -**

**\*we will use Azure App Service to communicate with Azure function that call Azure SQL database**

**1-on SqlApp we update the following code as below**

**using Microsoft.AspNetCore.Mvc;**

**using Sqlapp.Models;**

**using Sqlapp.Services;**

**using System.Collections.Generic;**

**using System.Threading.Tasks;**

**namespace Sqlapp.Controllers{**

**public class CourseController : Controller{**

**private readonly CourseService \_course\_service;**

**public CourseController(CourseService \_svc){\_course\_service = \_svc;}**

**// The Index method is used to get a list of courses and return it to the view**

**public async Task<IActionResult> Index(){**

**IEnumerable<Course> \_course\_list = await \_course\_service.GetCourses();**

**return View(\_course\_list);}}}**

**using Microsoft.Extensions.Configuration;**

**using Newtonsoft.Json;**

**using Sqlapp.Models;**

**using System.Collections.Generic;**

**using System.Data.SqlClient;**

**using System.Net.Http;**

**using System.Text.Json;**

**using System.Threading.Tasks;**

**namespace Sqlapp.Services{**

**public class CourseService{**

**// Ensure to change the below variables to reflect the connection details for your database**

**private IConfiguration \_configuration;**

**public CourseService(IConfiguration \_config){\_configuration = \_config;}**

**public async Task<IEnumerable<Course>> GetCourses(){**

**var functionUrl = "https://enbehfunc.azurewebsites.net/api/HttpGetFunc?code=gp6AIilx3TjOFkT848LH4CJb0JNrZzu0SV4J0CERCB0cv2T8bbmQnw==";**

**using (HttpClient \_client = new HttpClient()){**

**HttpResponseMessage \_response = await \_client.GetAsync(functionUrl);**

**string \_content = await \_response.Content.ReadAsStringAsync();**

**return JsonConvert.DeserializeObject<IEnumerable<Course>>(\_content);}}}}**

**(make sure that you publish your azure function app and Azure SQL database and create table courses in order to communicate between them)**

**Azure function more Details**

**Notes: -**

**1-Azure function is serverless that you don’t need to manage the infrastructure**

**2-if you have app with multiple modules if we have to apply email order module to use on multiple module, you have to inject the modules that needs to call it**

**(So instead it we can include it in azure function and call it via http request)**

**2-you can write azure function in different languages such as Java, azure PowerShell, .net core**

**3-you can trigger azure function via another azure resources such as azure queues, azure blobs.**

