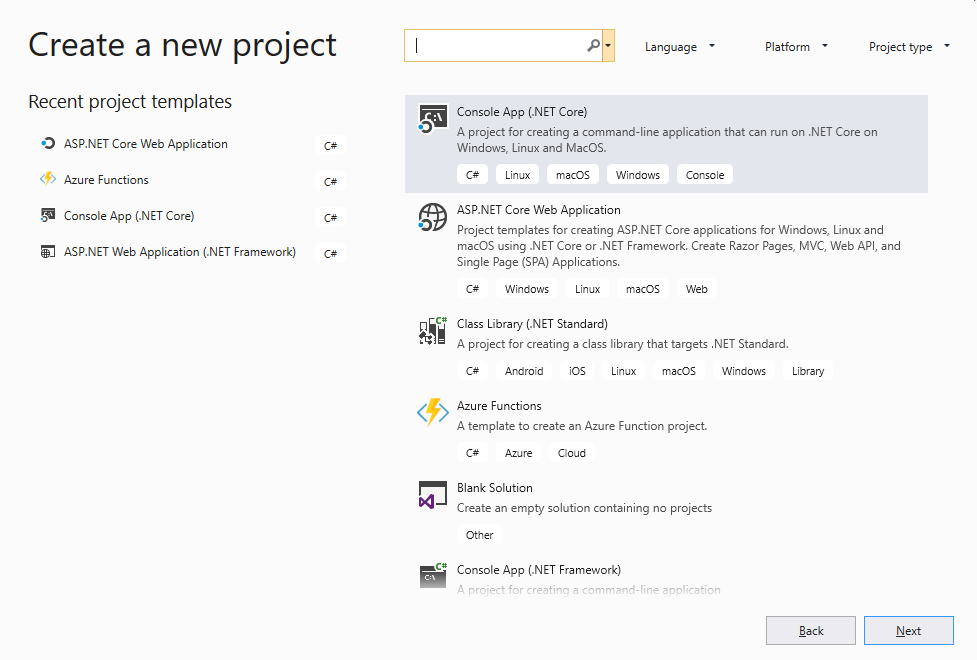
**In this article we will do following things :**

1. Create HTTP trigger
2. Call HTTP trigger function from post man
3. Create queue.
4. Add data in queue.
5. Consume queue data using queue listener
6. Add queue data in file and upload file into azure blog.
7. Create CRON job using timer trigger which sent email in specific given interval of time.

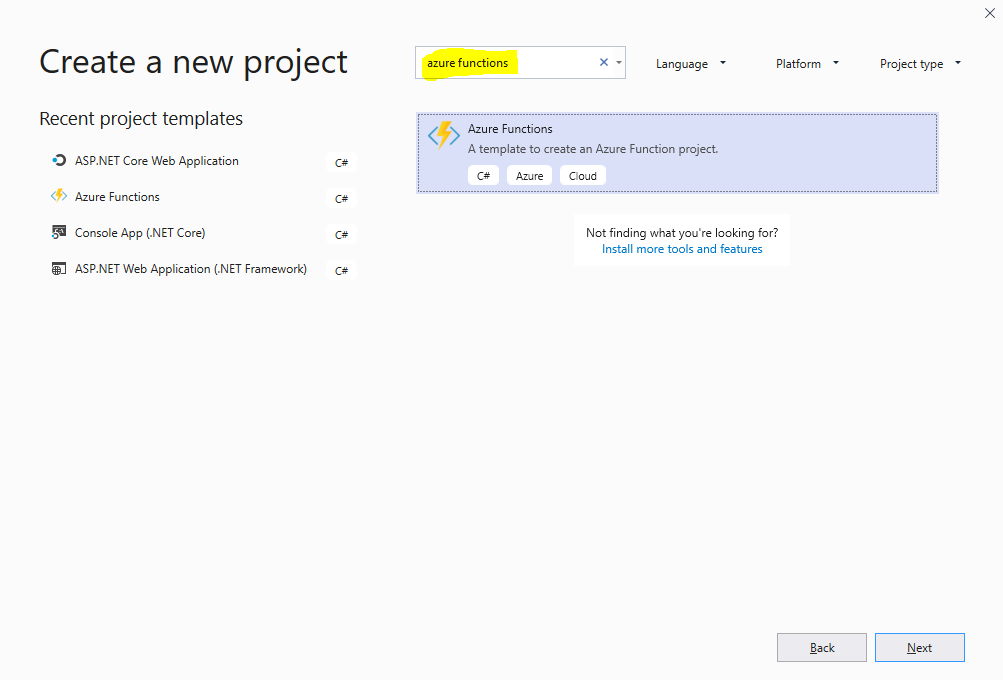
Step 1

Open VS 2019 and click on create new project.



Step 2

In search bar search for azure functions.

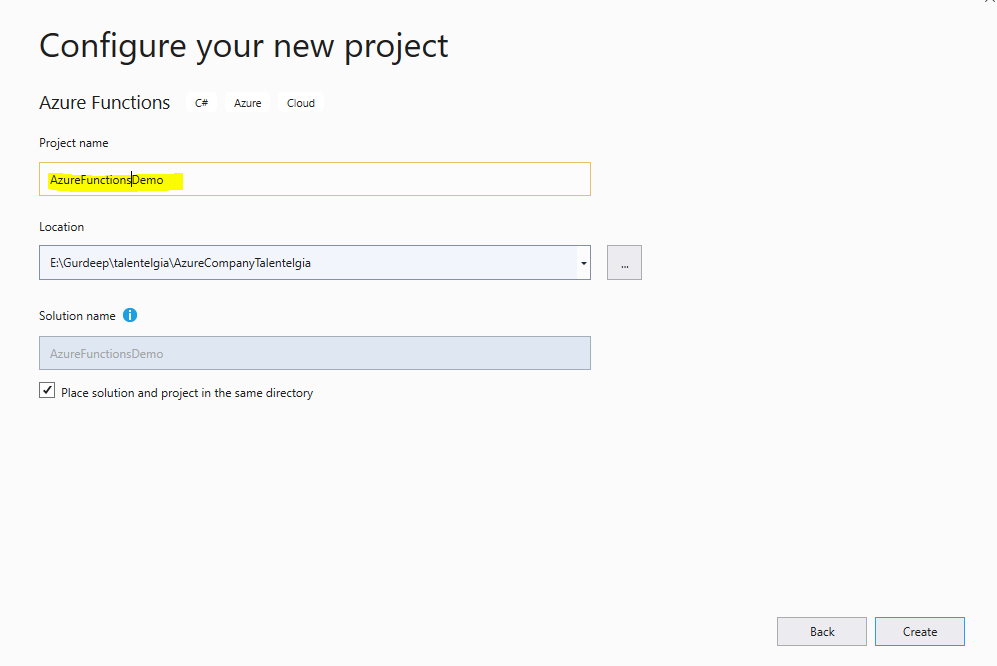


Step 3

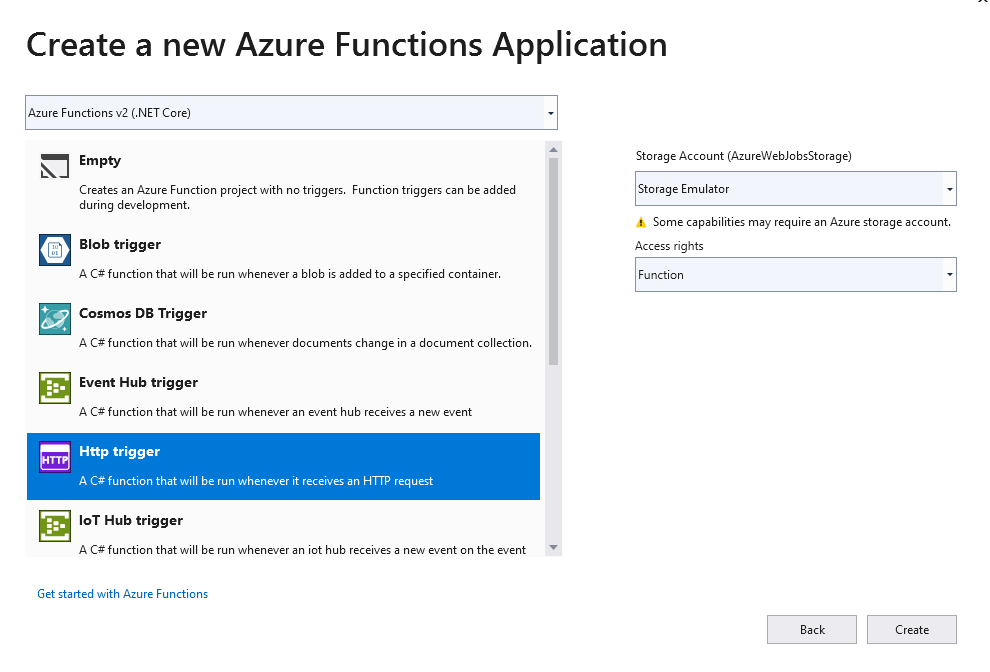
Select Azure Function template and click on Next button

Step 4

Give Project name and click on create button.

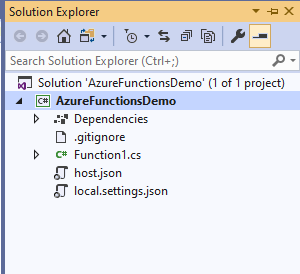


Step 5

select HTTP trigger and click on create button.

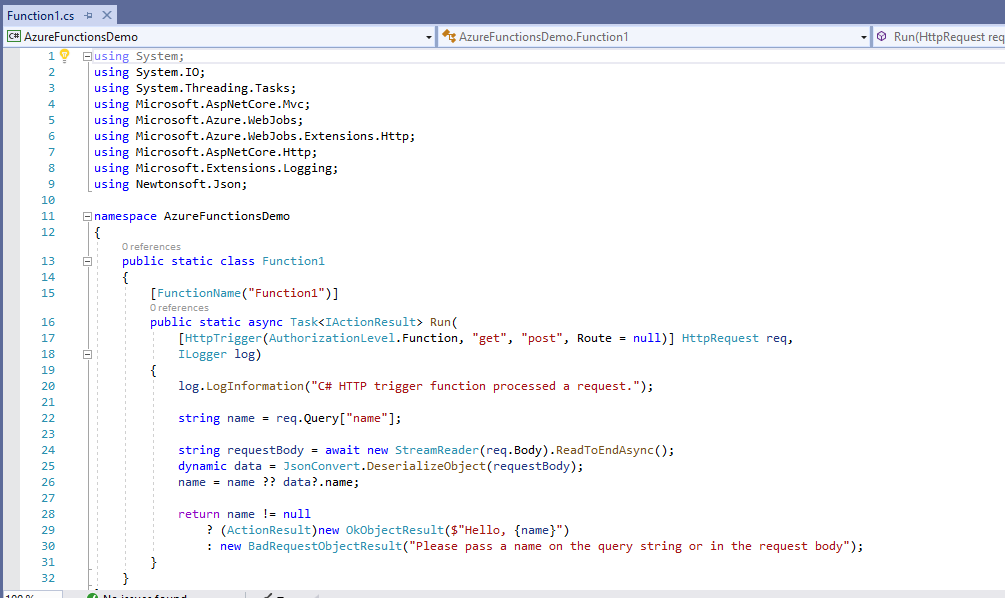
Step 6

We got this kind of folder structure.



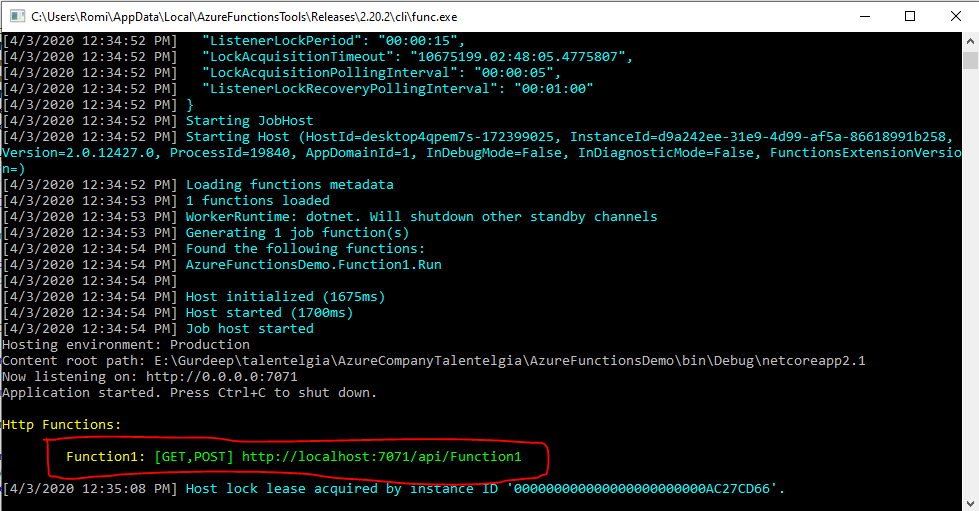
Step 7

Open Function1.cs file which contain http trigger which has type GET and POST.



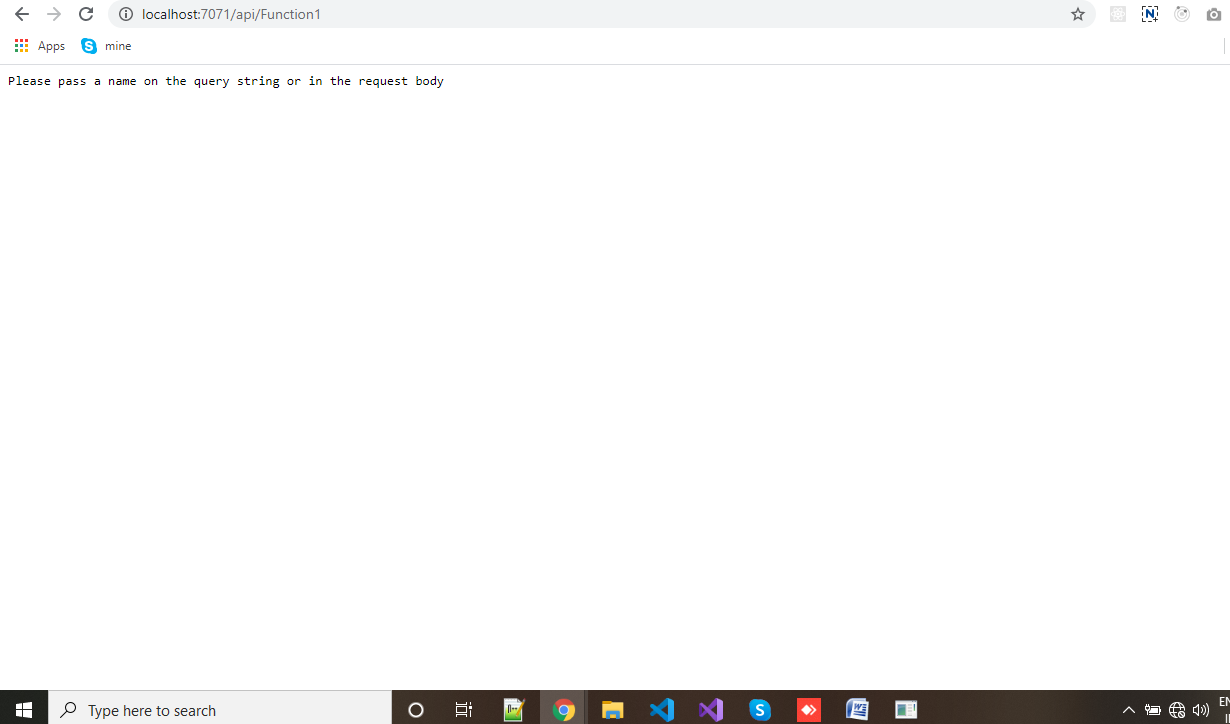
Step 8

Run your application. A console popup window appears which contains HTTP trigger function route.



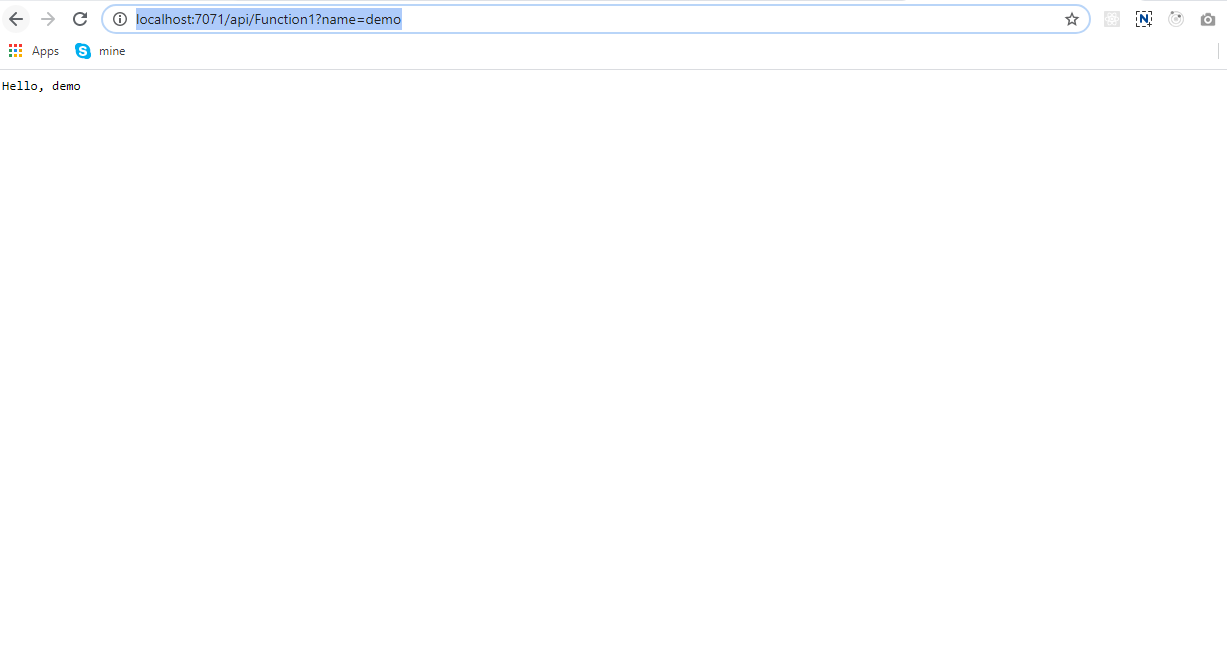
Step 9

Copy route and open it in chrome. You got a message "Please pass a name on the query string or in the request body". It is expecting query string or request body parameter name property.



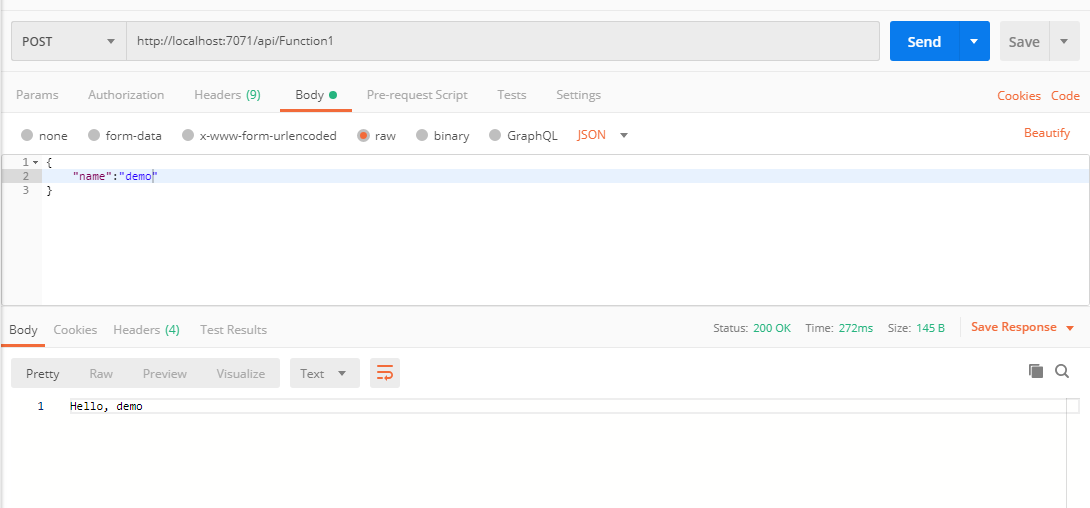
Step 10

Now we’ll pass name in query string " http://localhost:7071/api/Function1?name=demo". We got Hello, demo in output.



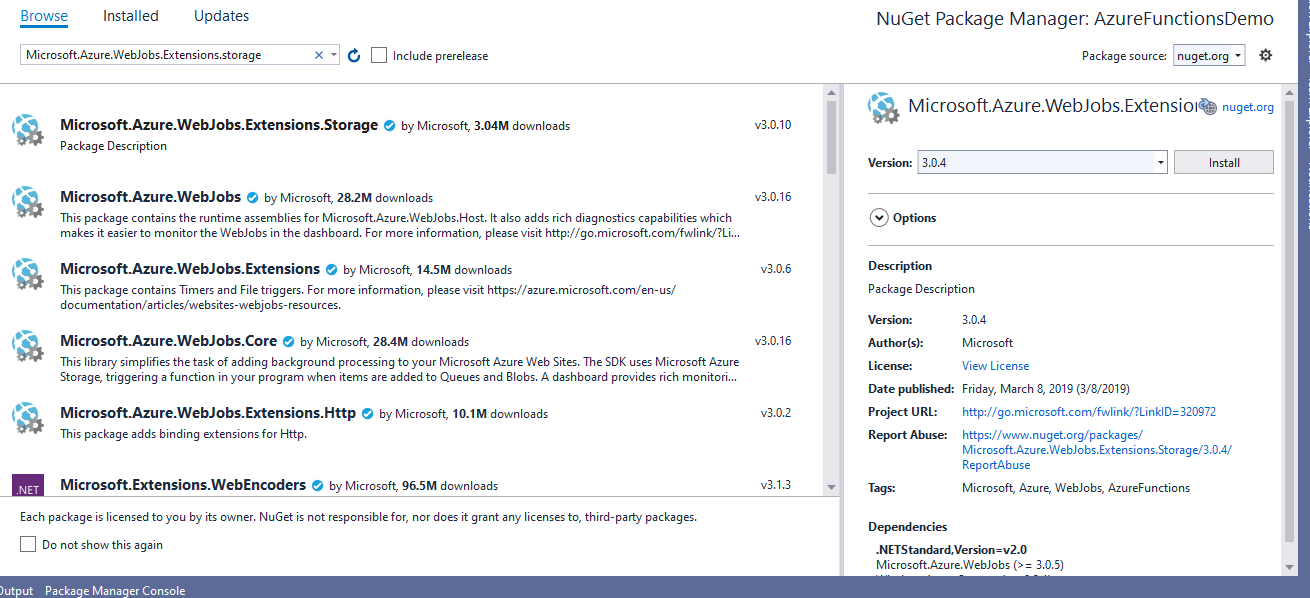
Step 11

Now we’ll make POST request using postman.



Step 12

Now we’ll modify HTTP trigger function for create queue and add data in queue. Before that we have to install nugget package **“Microsoft.Azure.WebJobs.Extensions.storage”**



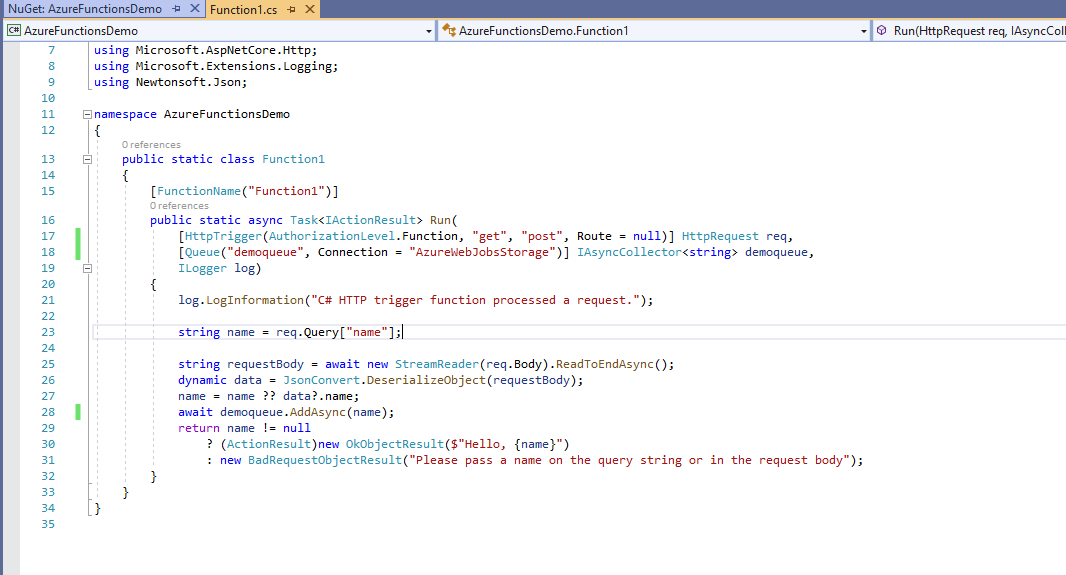
Add below line of code as second parameter in run function.

[Queue("demoqueue", Connection = "AzureWebJobsStorage")] IAsyncCollector<string> demoqueue

Then in Code block add below line of code for push data in queue.

await demoqueue.AddAsync(name);

Now we finish the part of adding data in queue. Please have a look on code of function1.cs file.

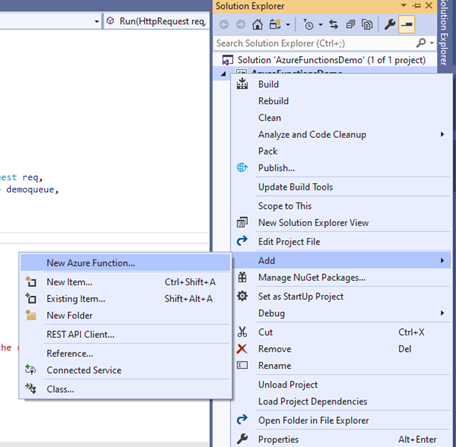


Step 13

Now we have to consume this queue data so we’ll create another queue trigger function which will read data from queue.

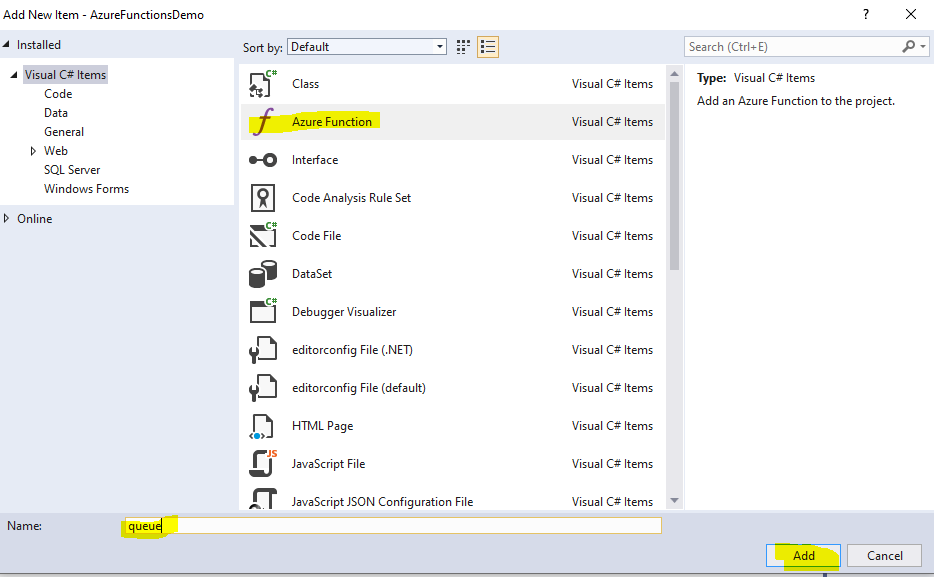
Step 14

Right click on project and click on add then add new Azure function.



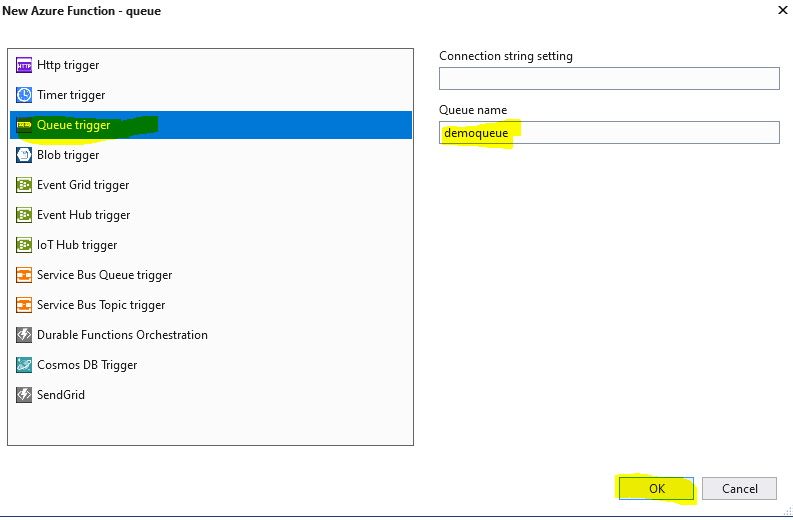
Step 15

A popup window appears. Select Azure function and give name to function as “queue” and click on Add.

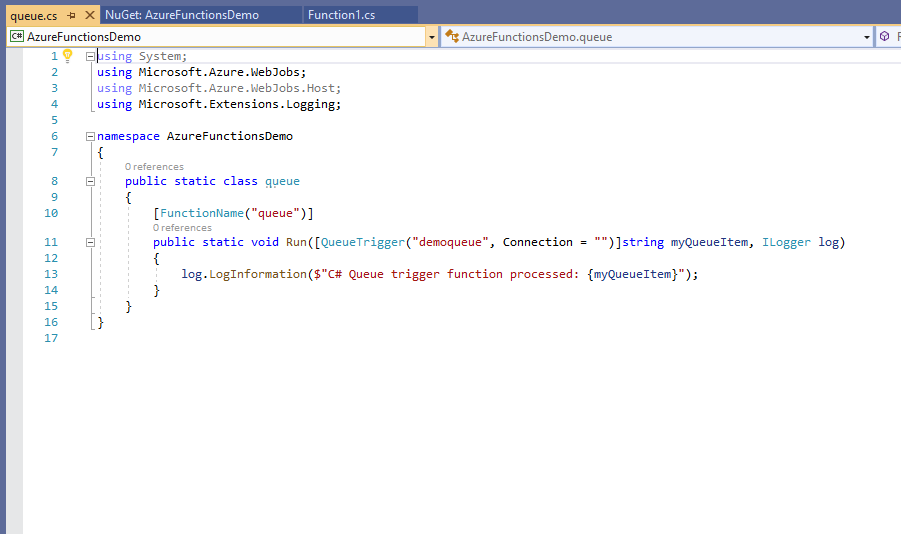


Step 16

Select queue trigger and give queue name which we used in HTTP trigger then click ok.

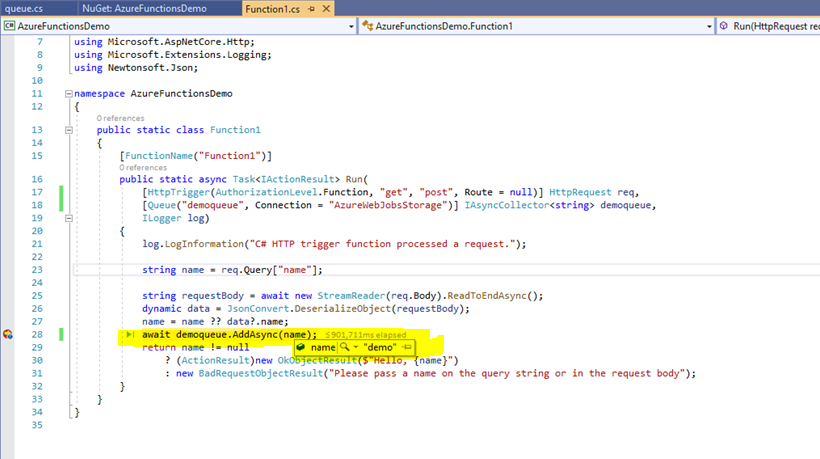


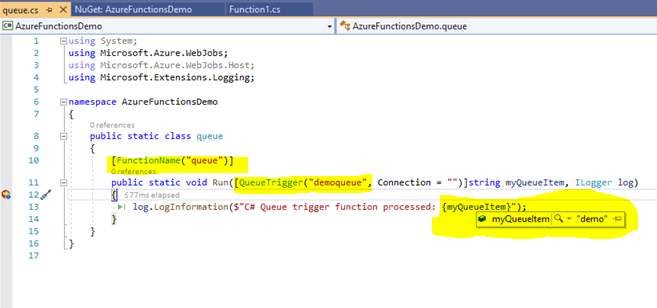
Queue trigger function file is generated.



Step 17

Run your application. When code runs data is pushed in queue and read by queue trigger. Please have a look on below image for your reference.

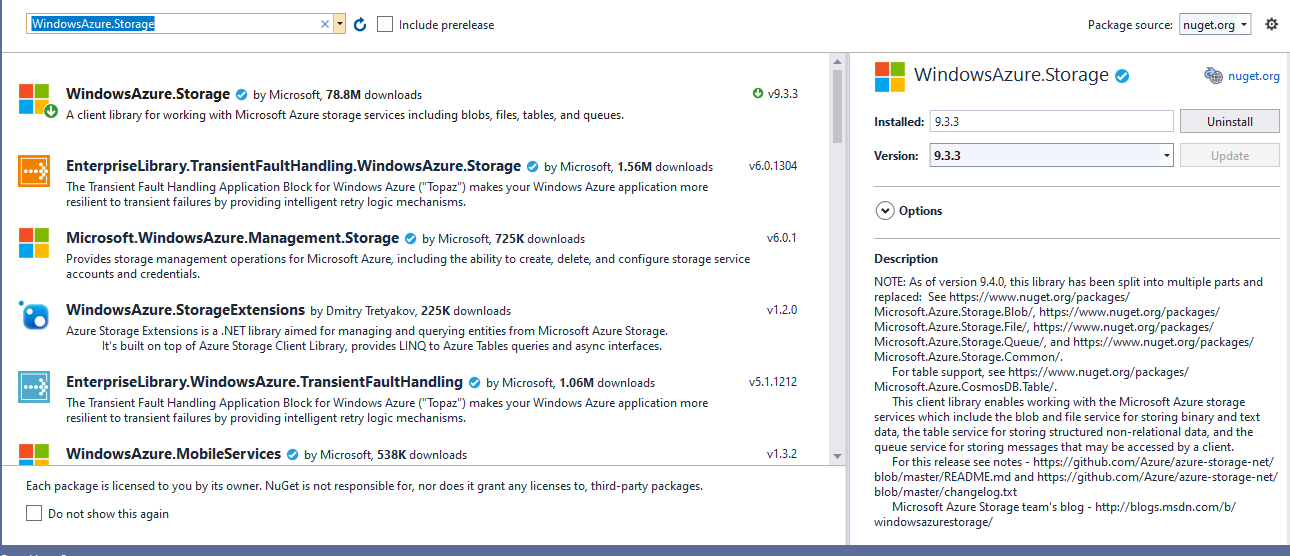
****



Step 18

Now we’ll upload myqueueItem data in azure blob.

Add Nuget “WindowsAzure.Storage”



Add below line of code as second parameter in Run function in queue.cs file.

[Blob("demoBlob ", Connection = "AzureWebJobsStorage")] CloudBlobContainer container

Add below code in code block.

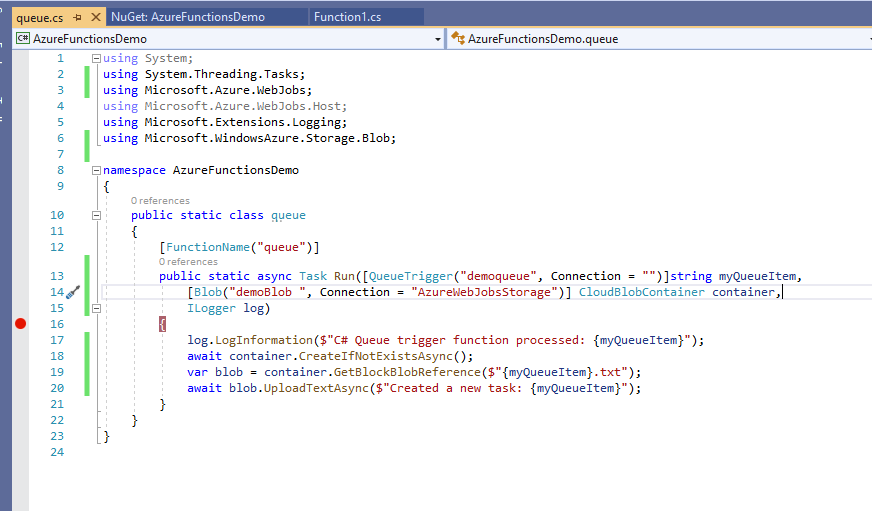
log.LogInformation($"C# Queue trigger function processed: {myQueueItem}");

await container.CreateIfNotExistsAsync();

var blob = container.GetBlockBlobReference($"{myQueueItem}.txt");

await blob.UploadTextAsync($"Created a new task: {myQueueItem}");

Your code looks like this.



Run your application and your file is uploaded in blob. You can check in server explorer.