**Lab Manual- Performing the Blob Operation using VS Code**

**Prepared for**: TechPledge

**Date:** 18th Nov 2018

**Prepared by:** Shruti Sinhaa

Document Name: Lab Manual

**Document Number** DevOpsLab401

**Contributor:**

Bipin Sinhaa

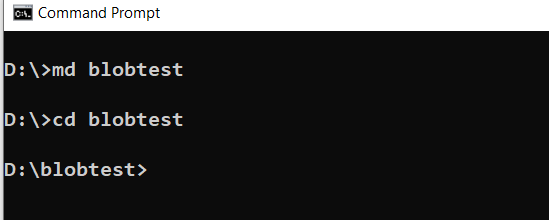
# Lab Scenario

## **Create an Folder and Clone the Code from Github**

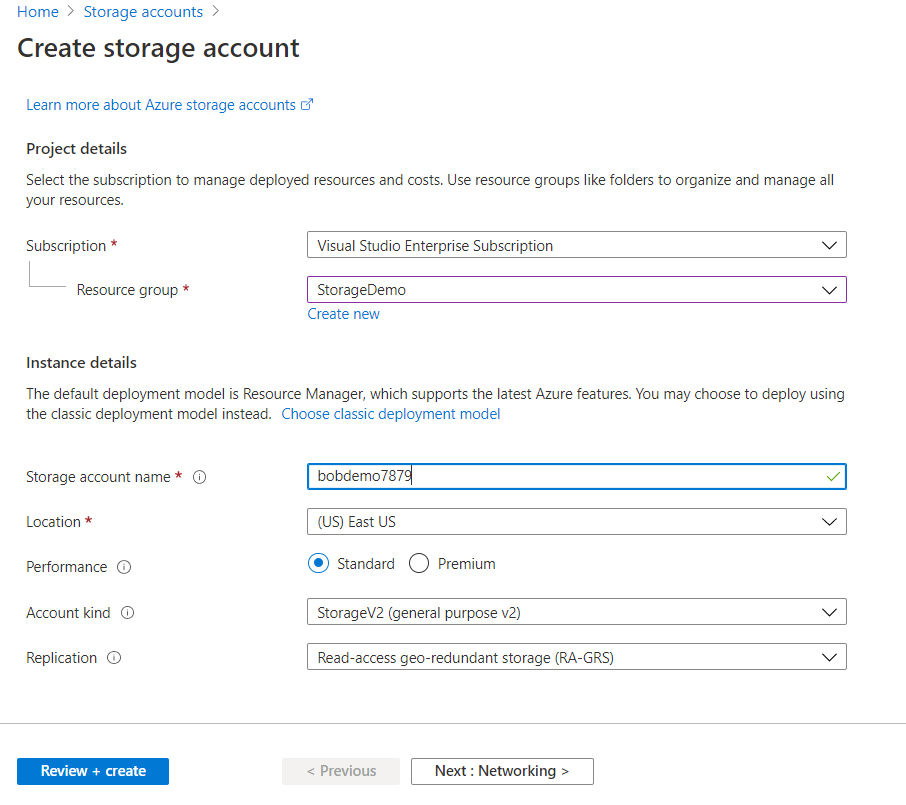
* 1. Open Command Prompt and Create a Folder ad then go inside it.

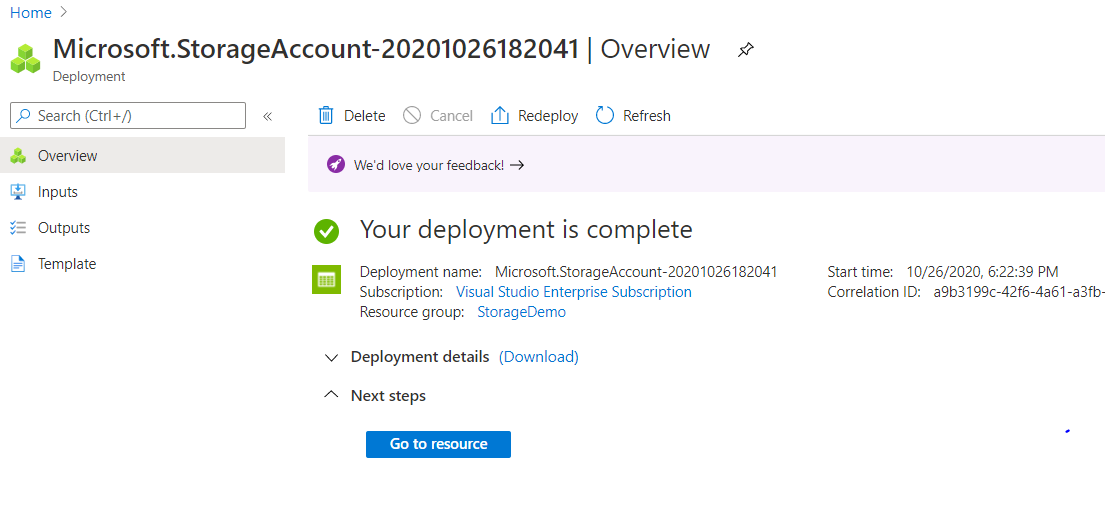
md blobtest

cd blobtest

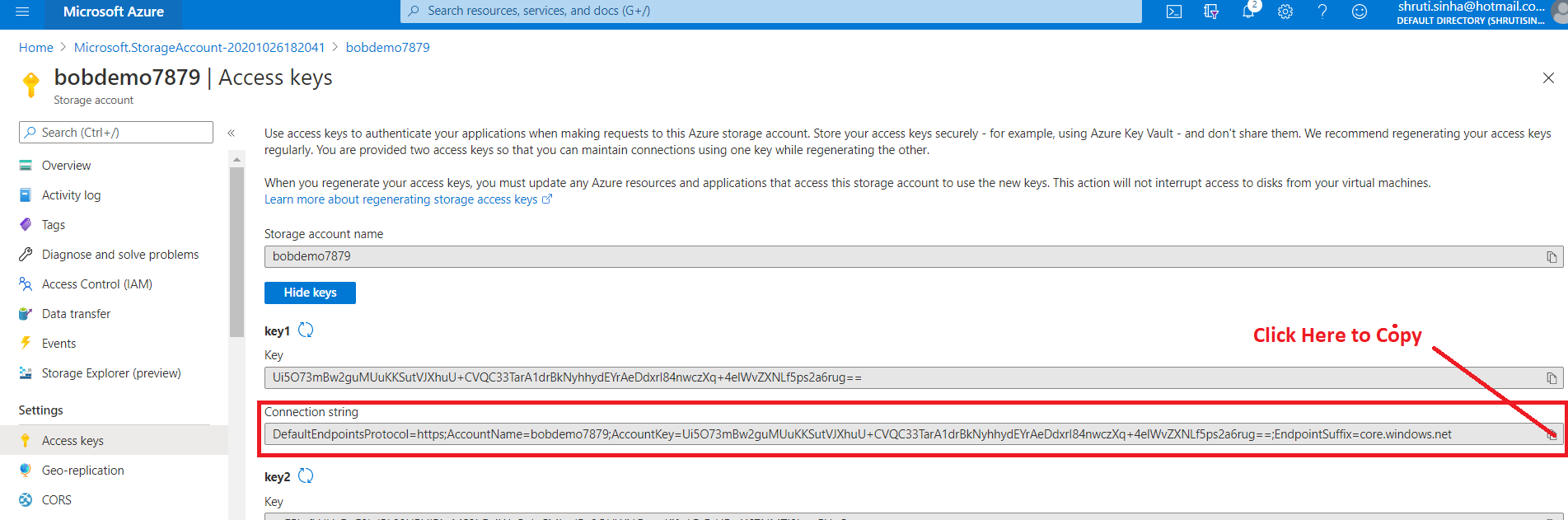


## **Step3 : Create an a Storage Account and Set the Enviornment Variable**





* In the **Settings** section of the storage account overview, select **Access keys**. Here, you can view your account access keys and the complete connection string for each key.



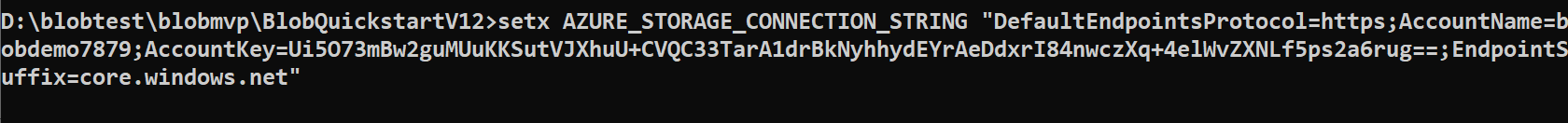
* Find the **Connection string** value under **key1**, and select the **Copy** button to copy the connection string. You will add the connection string value to an environment variable in the next step.

### **Configure your storage connection string**

* To set the environment variable, open a **DOS** console window and Type below command

setx AZURE\_STORAGE\_CONNECTION\_STRING

"DefaultEndpointsProtocol=https;AccountName=funcstorshruti;AccountKey=r2pI9EDNYbffYjeGeU5Ffg0asft9+cqYkJ5hjZ4OWpq5Kn5zjycxn0Alf74be/y3DXHFyizxH984QN7CTZoCoA==;EndpointSuffix=core.windows.net"

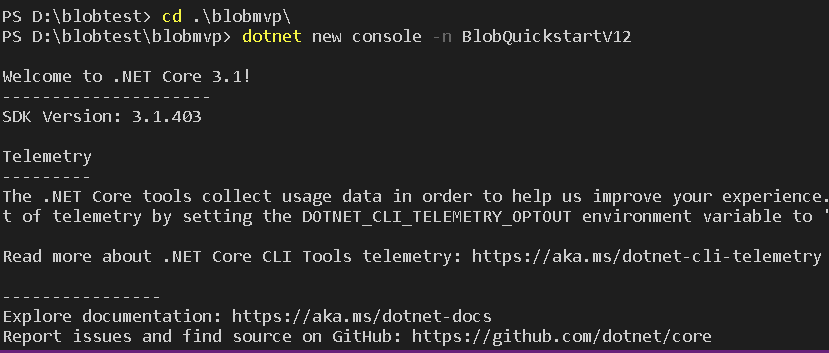


* Now Close the DOS Console and Relaunch it again

## **Create a .NET Core application named BlobQuickstartV12.**

* Now Create New Project use the **dotnet new** command to create a new console app with the name *BlobQuickstartV12*

**dotnet new console -n BlobQuickstartV12**

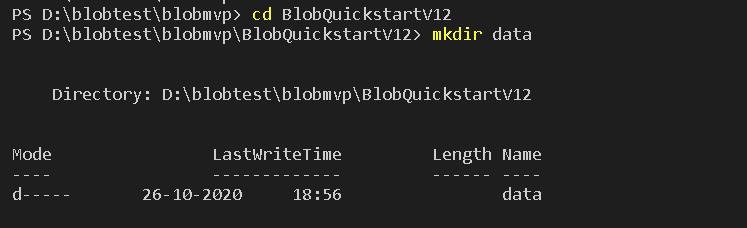


* Switch to the newly created **BlobQuickstartV12**directory.

cd BlobQuickstartV12

* In side the **BlobQuickstartV12** directory, create another directory called **data.** This is where the blob data files will be created and stored.

Mkdir data



### **install the Azure Blob storage client library for .NET package**

While still in the application directory, install the Azure Blob storage client library for .NET package by using the dotnet add package command.

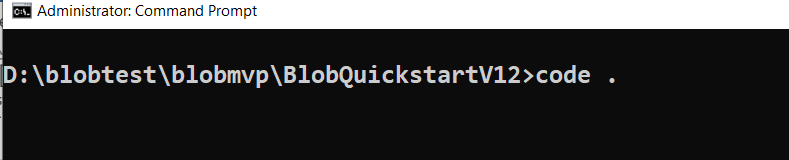
**dotnet add package Azure.Storage.Blobs**



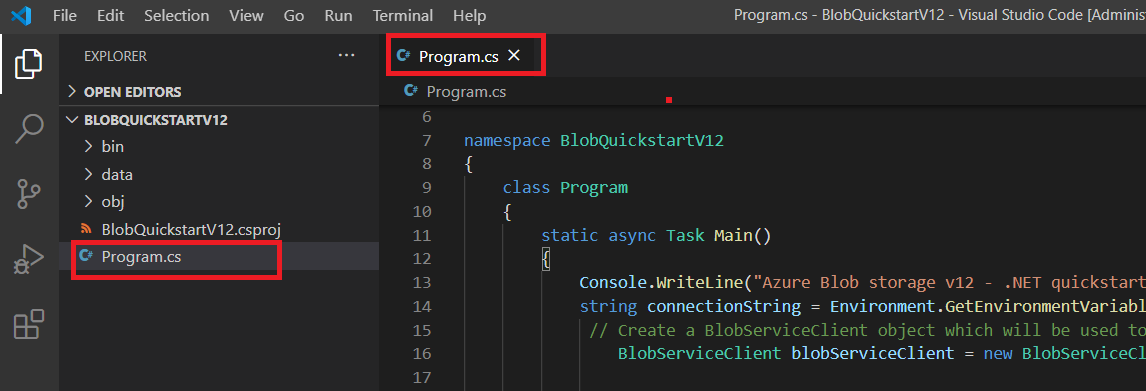
### **Set up the app framework**

1. To Launch the Code Editor type  ***Code .*** file in your editor

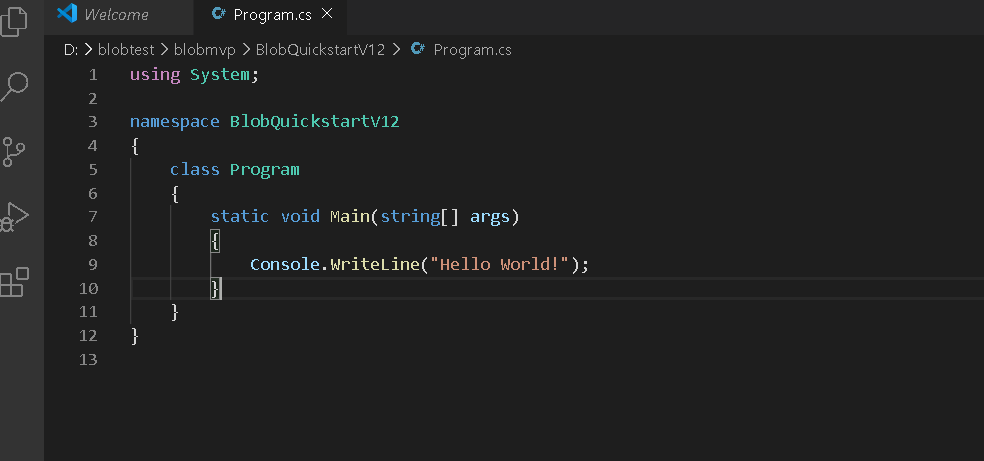
Code .



1. Open ***program.cs*** file for editing



1. You Got the Default Code



# Create the Container in Azure Storage and Upload the Blob

## **Add this code inside the Main method:**

using Azure.Storage.Blobs;

using Azure.Storage.Blobs.Models;

using System;

using System.IO;

using System.Threading.Tasks;

namespace BlobQuickstartV12

{

class Program

{

static async Task Main()

{

Console.WriteLine("Azure Blob storage v12 - .NET quickstart sample\n");

string connectionString = Environment.GetEnvironmentVariable("AZURE\_STORAGE\_CONNECTION\_STRING");

// Create a BlobServiceClient object which will be used to create a container client

BlobServiceClient blobServiceClient = new BlobServiceClient(connectionString);

//Create a unique name for the container

string containerName = "quickstartblobs" + Guid.NewGuid().ToString();

// Create the container and return a container client object

BlobContainerClient containerClient = await blobServiceClient.CreateBlobContainerAsync(containerName);

// Create a local file in the ./data/ directory for uploading and downloading

string localPath = "./data/";

string fileName = "quickstart" + Guid.NewGuid().ToString() + ".txt";

string localFilePath = Path.Combine(localPath, fileName);

// Write text to the file

await File.WriteAllTextAsync(localFilePath, "Hello, World!");

// Get a reference to a blob

BlobClient blobClient = containerClient.GetBlobClient(fileName);

Console.WriteLine("Uploading to Blob storage as blob:\n\t {0}\n", blobClient.Uri);

// Open the file and upload its data

using FileStream uploadFileStream = File.OpenRead(localFilePath);

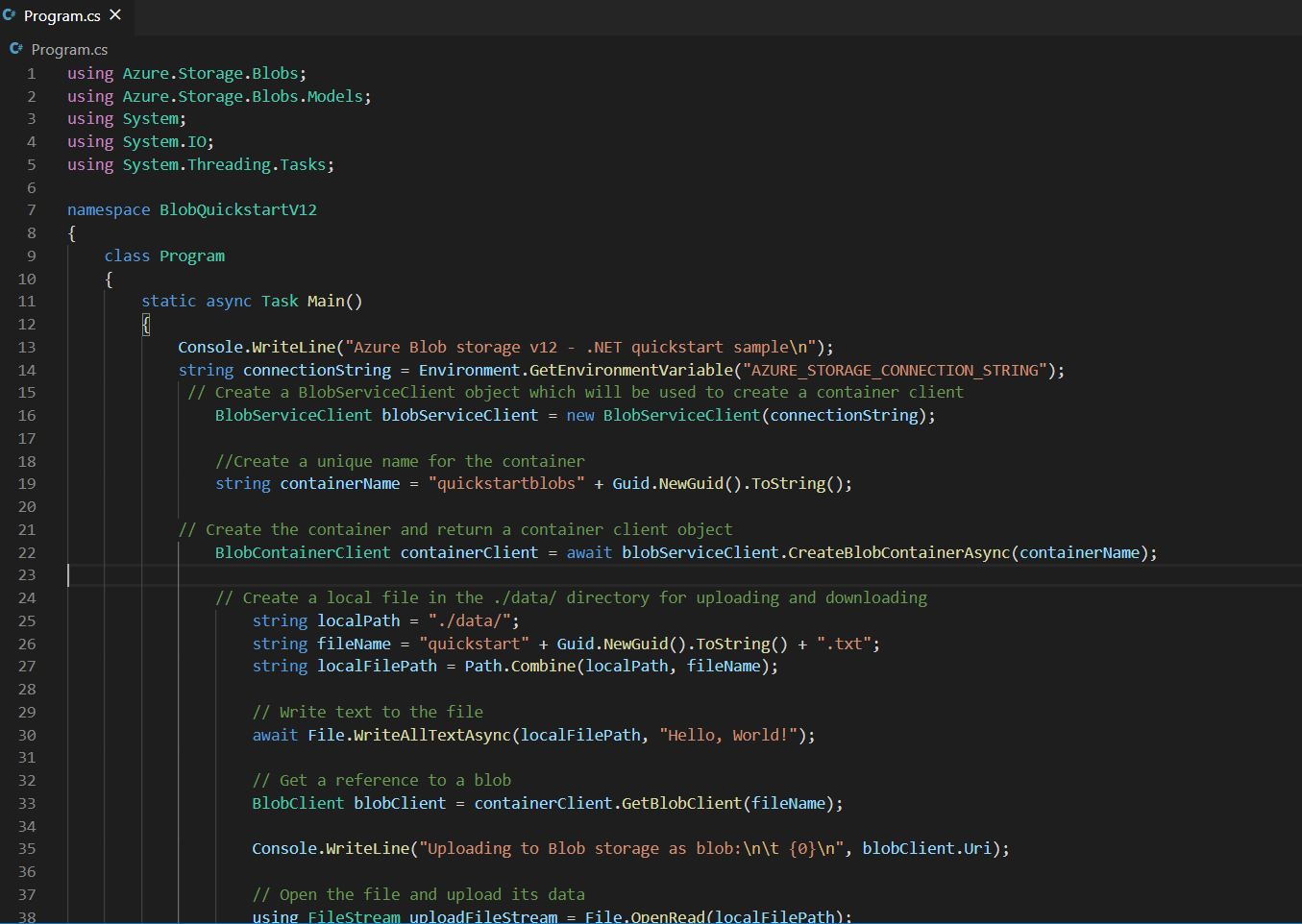
await blobClient.UploadAsync(uploadFileStream, true);

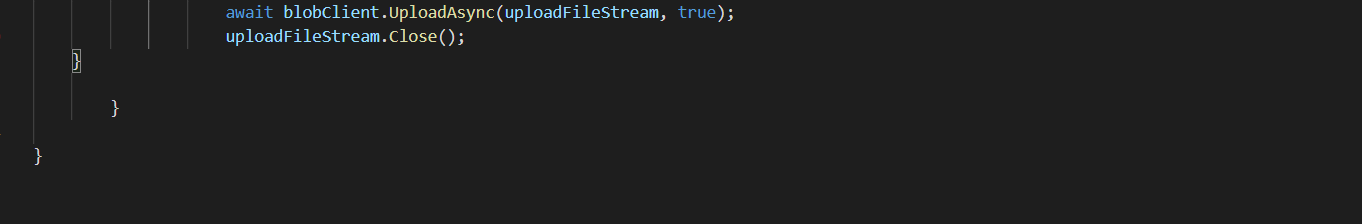
uploadFileStream.Close();

}

}

}



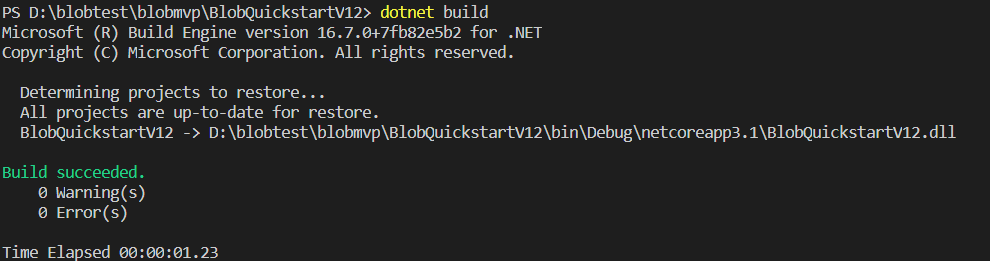


* Now Save the File with File🡪Save

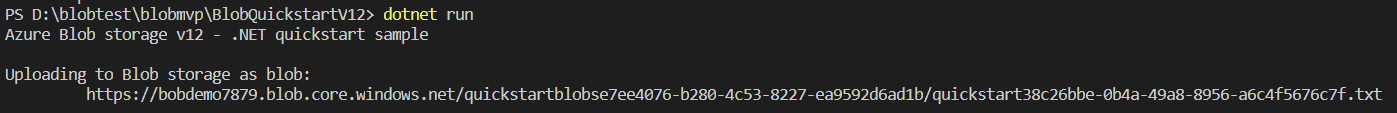
## **Now Build and Run The Code**

* In Terminal Type below command

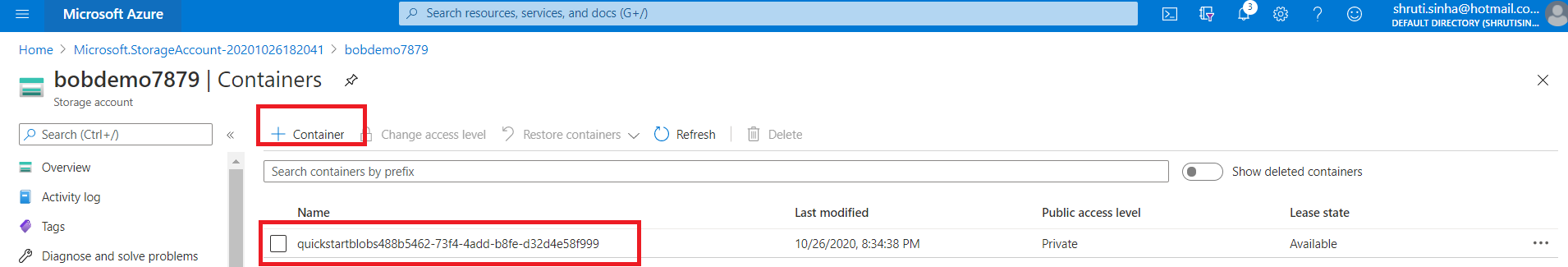
dotnet build



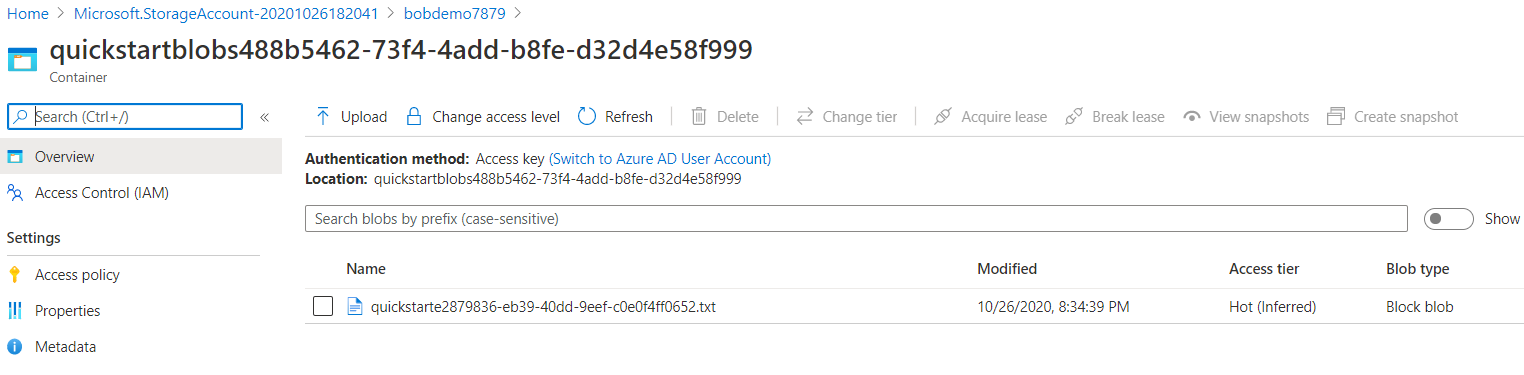
dotnet run



* Now Go to Portal and Check you have Blob Container Created



* Go Inside the Container and Check the Blob



# List the blobs in a container

* Add this code to the end of the Main method:

Console.WriteLine("Listing blobs...");

// List all blobs in the container

await foreach (BlobItem blobItem in containerClient.GetBlobsAsync())

{

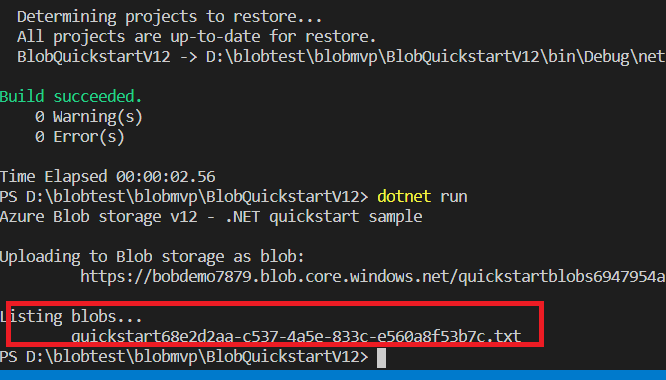
Console.WriteLine("\t" + blobItem.Name);

}

* Now Save the File with File🡪Save
* Now again Build and Run the Code

Dotnet Build

Dotnet Run



# Download blobs

* Add this code to the end of the Main method:

// Download the blob to a local file

// Append the string "DOWNLOADED" before the .txt extension

// so you can compare the files in the data directory

string downloadFilePath = localFilePath.Replace(".txt", "DOWNLOADED.txt");

Console.WriteLine("\nDownloading blob to\n\t{0}\n", downloadFilePath);

// Download the blob's contents and save it to a file

BlobDownloadInfo download = await blobClient.DownloadAsync();

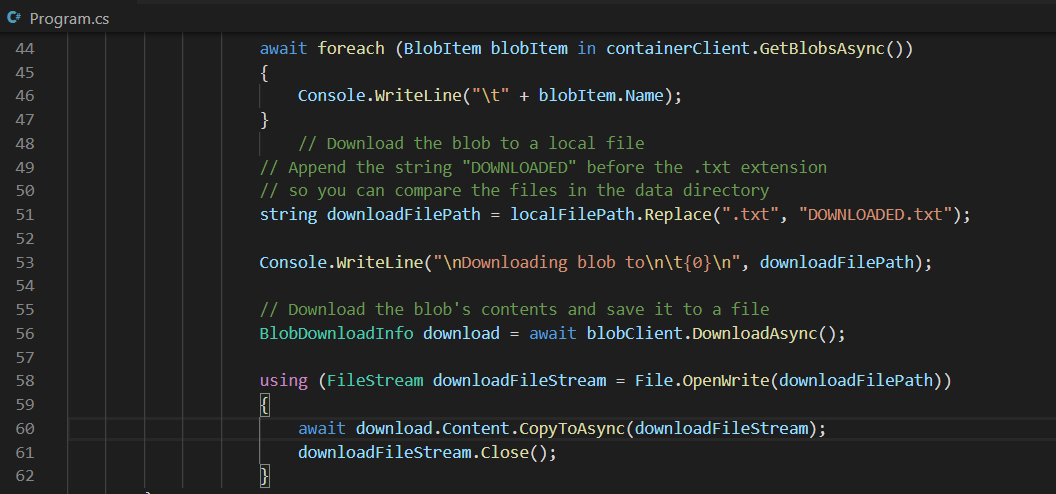
using (FileStream downloadFileStream = File.OpenWrite(downloadFilePath))

{

await download.Content.CopyToAsync(downloadFileStream);

downloadFileStream.Close();

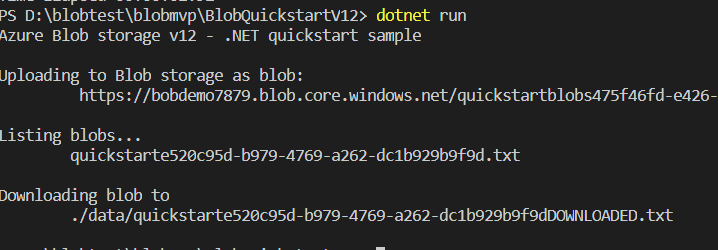
}



* Now Save the File with File🡪Save
* Now again Build and Run the Code

Dotnet Build

Dotnet Run



# Delete the Container

* Add this code to the end of the Main method:

**// Clean up**

Console.Write("Press any key to begin clean up");

Console.ReadLine();

Console.WriteLine("Deleting blob container...");

await containerClient.DeleteAsync();

Console.WriteLine("Deleting the local source and downloaded files...");

File.Delete(localFilePath);

File.Delete(downloadFilePath);

Console.WriteLine("Done");

* Now Save the File with File🡪Save
* Now again Build and Run the Code

Dotnet Build

Dotnet Run