Introduction

In this document, we will discuss about Azure Storage and different types of storage services provided by Azure. We will create a storage account from Azure portal and we will then create Azure function to store the files into Azure blob storage, and finally, we will deploy the same function to the Azure function app from Visual Studio.

If you are new to Azure Functions/Storage then ask **ChatGPT** or check out the below links,

Azure Functions documentation | Microsoft Learn

Azure Storage documentation | Microsoft Learn

What is Azure Storage Service

Whenever we create an application we have to first decide what storage we can use to store the data. Azure Storage provides solution to different storage problems. It provides scalable, available, durable storage for data in azure. Azure storage is a set of different services such as Blobs, Files, Queues, etc. We can interact with azure storage using different programming languages such .NET, JAVA, Golang, Javascript, etc. using the SDKs.

Different Azure Storage Service

1. **Azure Blob** - also known as blobs which are mainly used to store the binary/text data such as photos, videos, documents, etc.
2. **Azure Tables** - which is nothing but a NoSQL storage for schemaless structured data.
3. **Azure Queue** - it is used to store the messages which can be used to communicate with applications
4. **Azure Files** - it is managed file share service for cloud or on-premise using SMB/NFS protocol.
5. **Azure Disks** - It is a virtual hard disk (VHD) that is of two types: managed and unmanaged.

What is Azure Blob storage?

Azure Blob storage is a storage service used to store schema-less structured data in the cloud. We can store data such as documents, pictures, videos, log files, backup files which don’t have a fixed structure. Blob storage is similar to directory structure where we can create different containers to store the blobs.

There are 3 types of Blobs:

1. Block Blobs - which are used to store the text/binary data.
2. Append Blobs - nothing but block blobs that are optimized for append operation.
3. Page Blobs - which store random access files up to 8 TiB in size.

Create simple azure function using C# to upload files in Azure Blog Storage

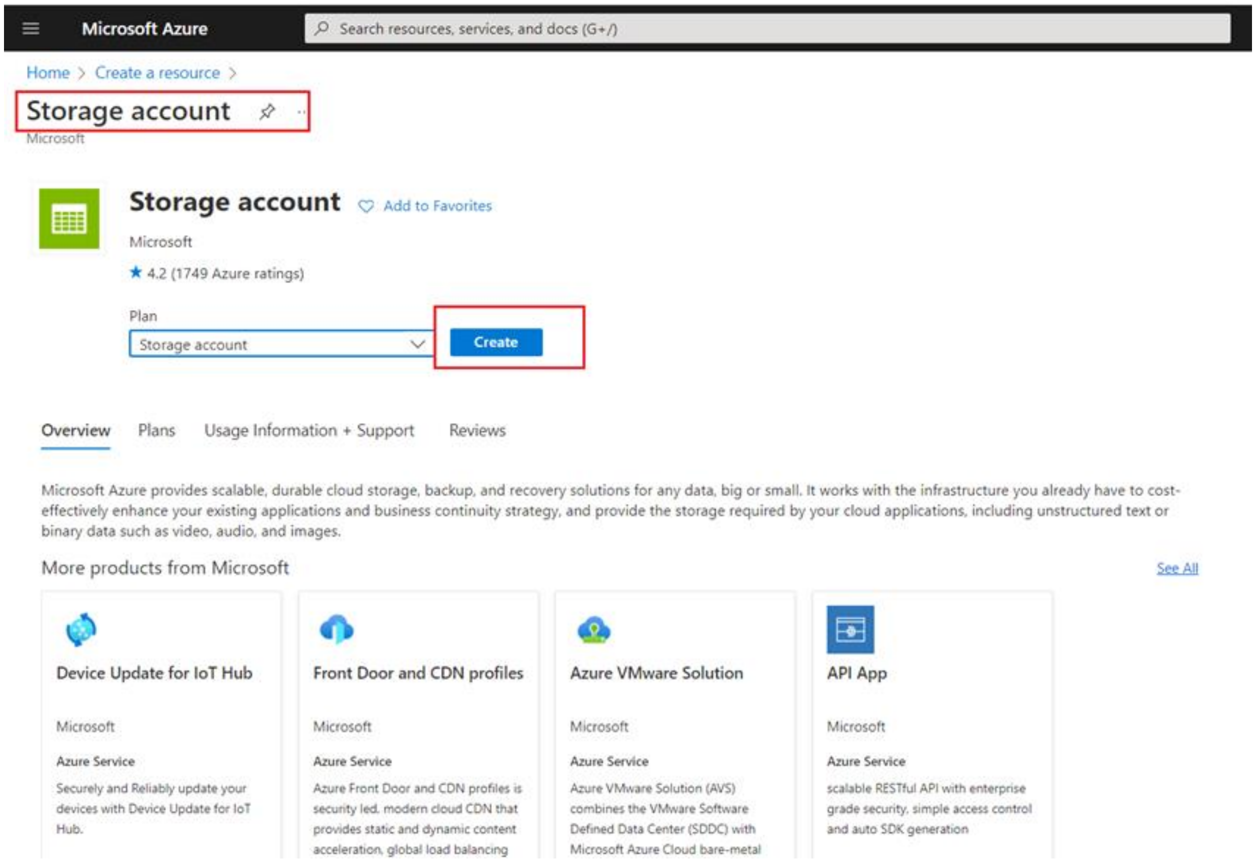
Prerequisites

1. Azure account (If you don't have an Azure subscription, create a free trial account)
2. Visual Studio 22 with Azure Development workload
3. Basic knowledge of C#

Create a Azure Storage resource using Azure Portal

Login to Azure and click on Create a resource button.

In search box type “storage account” and select it.


Microsoft Azure

Search resources, services, and docs (G+/)

Home > Create a resource >

Storage account

Microsoft

Storage account

Add to Favorites

Microsoft

★ 4.2 (1749 Azure ratings)

Plan

Storage account

Create

Overview | Plans | Usage Information + Support | Reviews

Microsoft Azure provides scalable, durable cloud storage, backup, and recovery solutions for any data, big or small. It works with the infrastructure you already have to cost-effectively enhance your existing applications and business continuity strategy, and provide the storage required by your cloud applications, including unstructured text or binary data such as video, audio, and images.

More products from Microsoft

See All

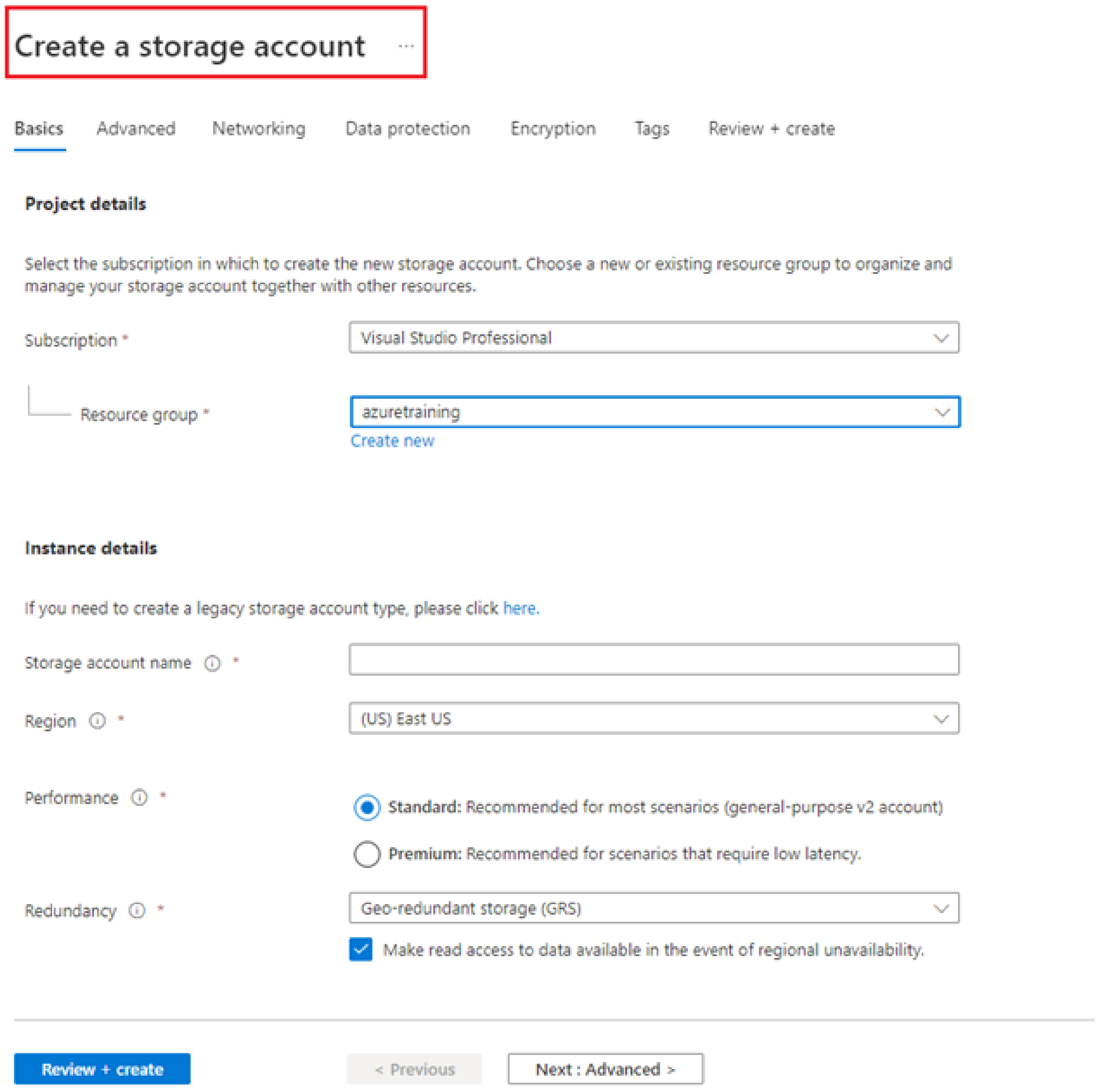
Device Update for IoT Hub
Microsoft
Azure Service
Securely and Reliably update your devices with Device Update for IoT Hub.

Front Door and CDN profiles
Microsoft
Azure Service
Azure Front Door and CDN profiles is security led, modern cloud CDN that provides static and dynamic content acceleration, global load balancing

Azure VMware Solution
Microsoft
Azure Service
Azure VMware Solution (AVS) combines the VMware Software Defined Data Center (SDDC) with Microsoft Azure Cloud bare-metal

API App
Microsoft
Azure Service
scalable RESTful API with enterprise grade security, simple access control and auto SDK generation

Click on Create button and you can see the Create storage account page.



Project details

Select the subscription in which to create the new storage account. Choose a new or existing resource group to organize and manage your storage account together with other resources.

Subscription *	Visual Studio Professional
Resource group *	azuretraining

Create new

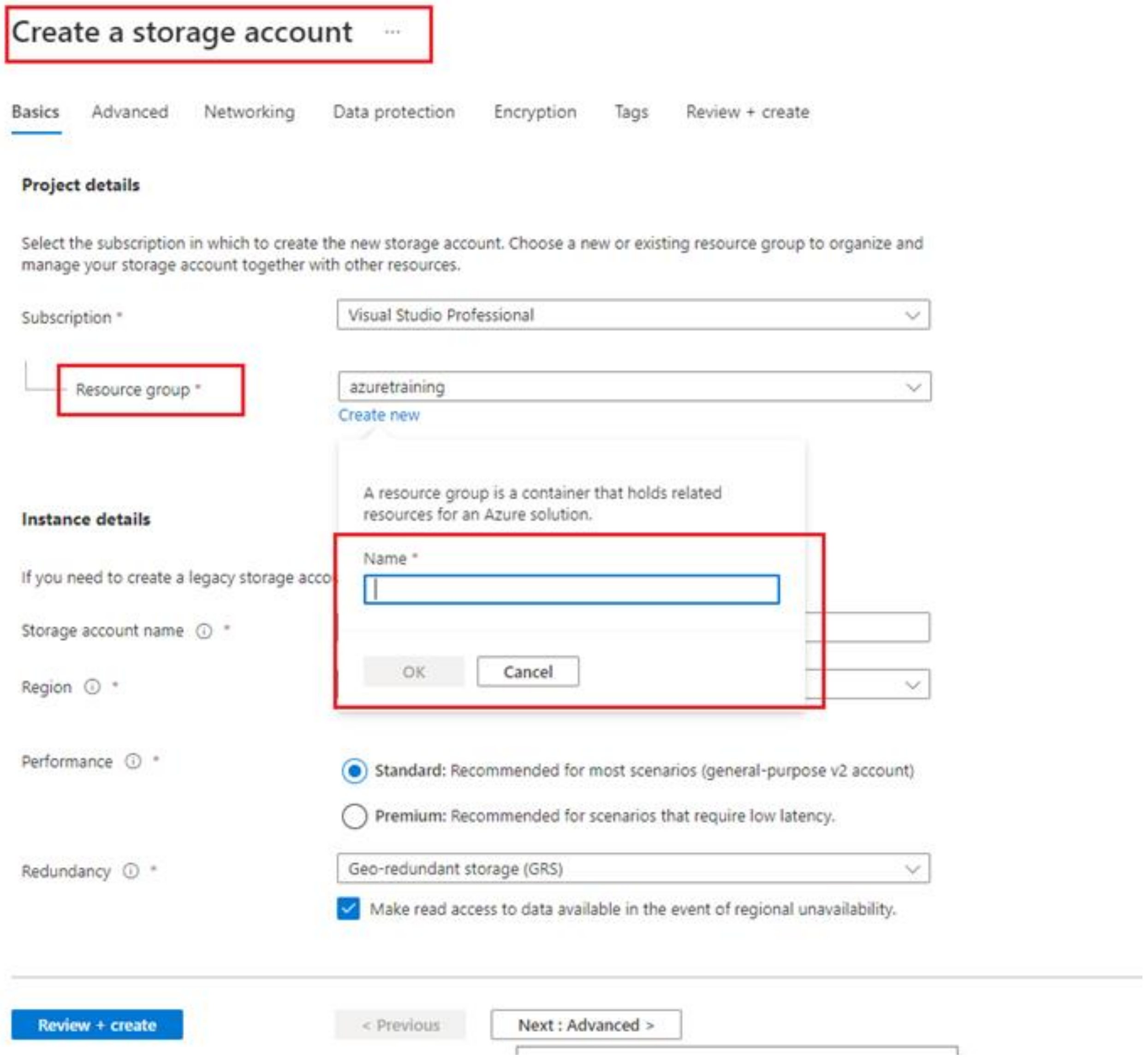
Instance details

If you need to create a legacy storage account type, please click here.

Storage account name *	
Region *	(US) East US
Performance *	[✓ **Standard:** Recommended for most scenarios (general-purpose v2 account)] **Premium:** Recommended for scenarios that require low latency.
Redundancy *	Geo-redundant storage (GRS)

☑ Make read access to data available in the event of regional unavailability.

Azure has Resource Groups (RG) act as a container for your resources. So now we are going to create a Storage account resource. First we need to create Resource Group. If you have already created RG then you can use the same here. Under Resource group click on Create New button and give a unique RG name.



Create a storage account ···

Basics | Advanced | Networking | Data protection | Encryption | Tags | Review + create

Project details

Select the subscription in which to create the new storage account. Choose a new or existing resource group to organize and manage your storage account together with other resources.

Subscription * : Visual Studio Professional

Resource group * : azuretraining

Create new

A resource group is a container that holds related resources for an Azure solution.

Name *

OK | Cancel

Instance details

If you need to create a legacy storage acco

Storage account name *

Region *

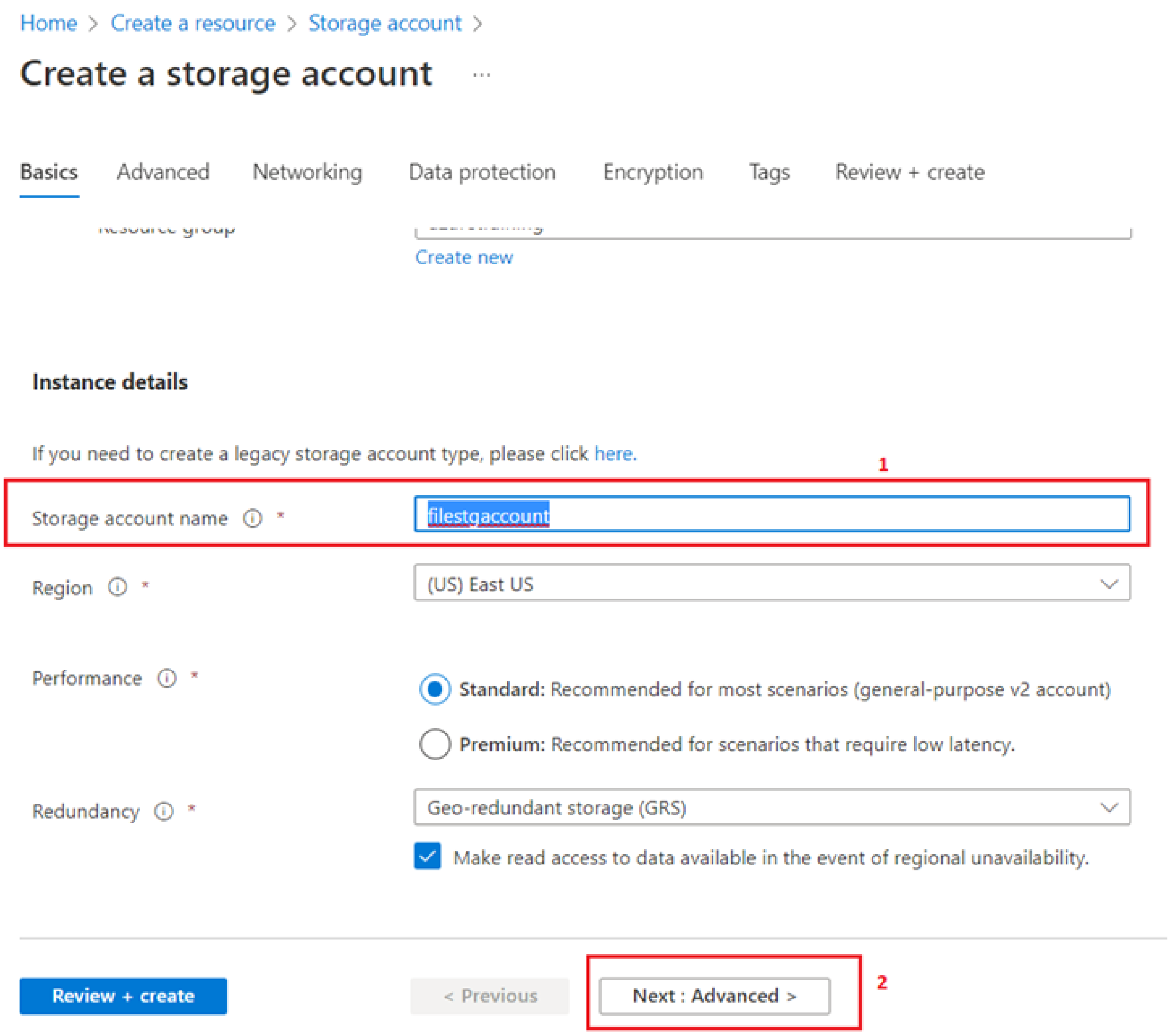
Performance * : [✓ Standard: Recommended for most scenarios (general-purpose v2 account)] Premium: Recommended for scenarios that require low latency.

Redundancy * : Geo-redundant storage (GRS)

[✓] Make read access to data available in the event of regional unavailability.

Review + create | < Previous | Next : Advanced >

Give the unique name to storage account and select the region. Click on **Next : Advanced** button.


Home > Create a resource > Storage account >

Create a storage account ...

Basics | Advanced | Networking | Data protection | Encryption | Tags | Review + create

Resource group

Create new

Instance details

If you need to create a legacy storage account type, please click here.

1

Storage account name ⓘ * : filestgaccount

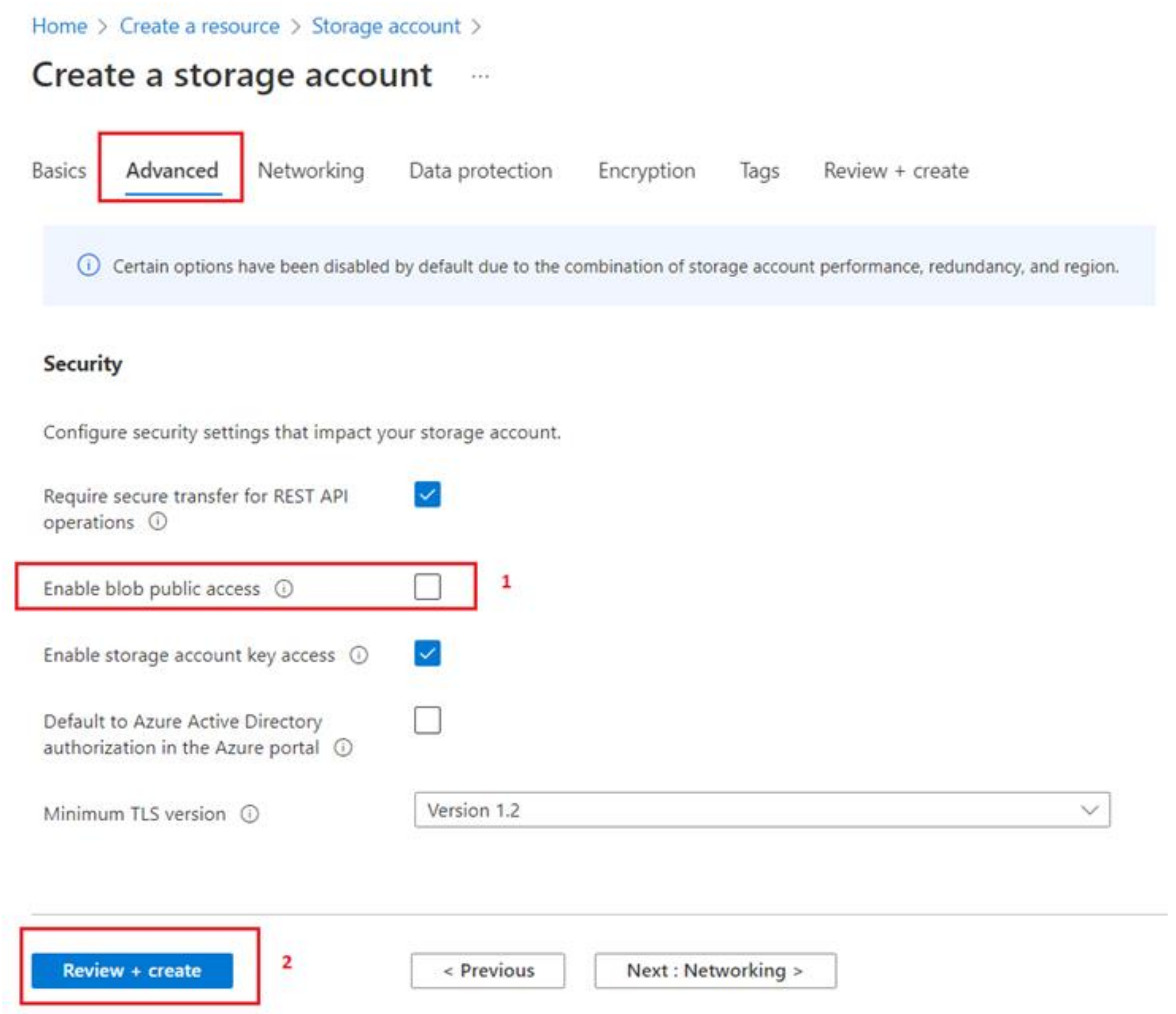
Region ⓘ * : (US) East US

Performance ⓘ * :
- ● **Standard:** Recommended for most scenarios (general-purpose v2 account)
- ○ **Premium:** Recommended for scenarios that require low latency.

Redundancy ⓘ * : Geo-redundant storage (GRS)

✓ Make read access to data available in the event of regional unavailability.

By default the “Enable blob public access” is checked which means that any anonymous user can access blobs within the containers. So uncheck the option and click on Review + Create button


Home > Create a resource > Storage account >

Create a storage account ...

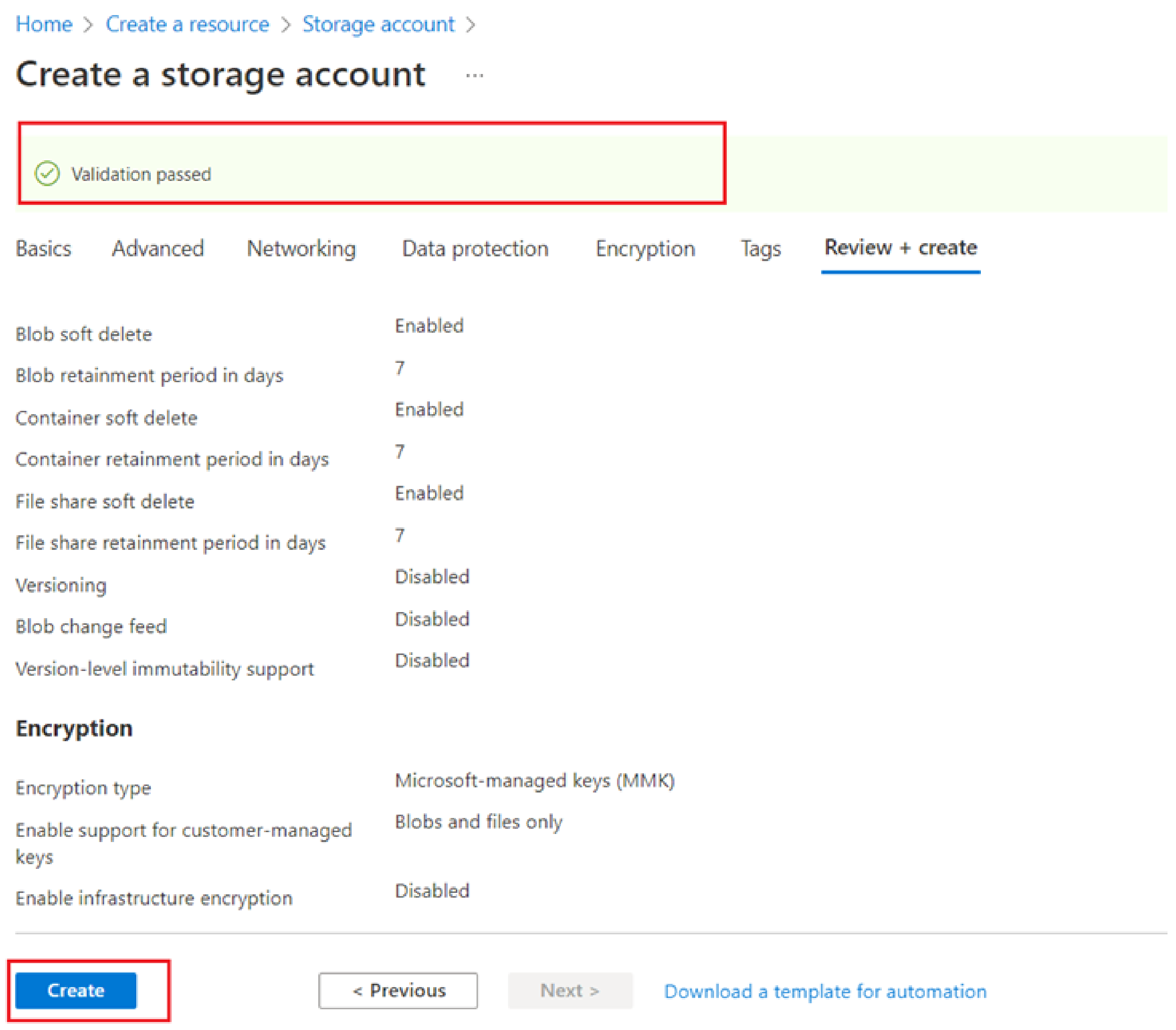
Certain options have been disabled by default due to the combination of storage account performance, redundancy, and region.

Security

Configure security settings that impact your storage account.

Require secure transfer for REST API operations	✓
Enable blob public access	☐
Enable storage account key access	✓
Default to Azure Active Directory authorization in the Azure portal	☐
Minimum TLS version	Version 1.2

Wait for few sec to complete the validation and Review everything is added properly after that click on Create button.

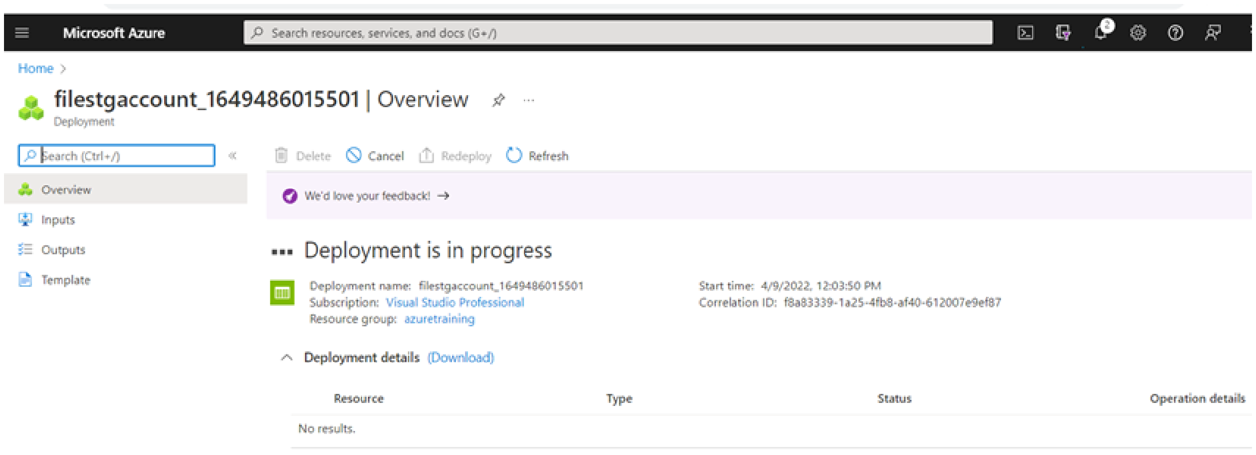


Blob soft delete	Enabled
Blob retainment period in days	7
Container soft delete	Enabled
Container retainment period in days	7
File share soft delete	Enabled
File share retainment period in days	7
Versioning	Disabled
Blob change feed	Disabled
Version-level immutability support	Disabled

Encryption

Encryption type	Microsoft-managed keys (MMK)
Enable support for customer-managed keys	Blobs and files only
Enable infrastructure encryption	Disabled

Creating resources will take time so wait for deployment process to finish.


Microsoft Azure
Search resources, services, and docs (G+/)
Home >
filestgaccount_1649486015501 | Overview
Deployment
Search (Ctrl+/)
Overview
Inputs
Outputs
Template
Delete
Cancel
Redeploy
Refresh
We'd love your feedback! →

Deployment is in progress

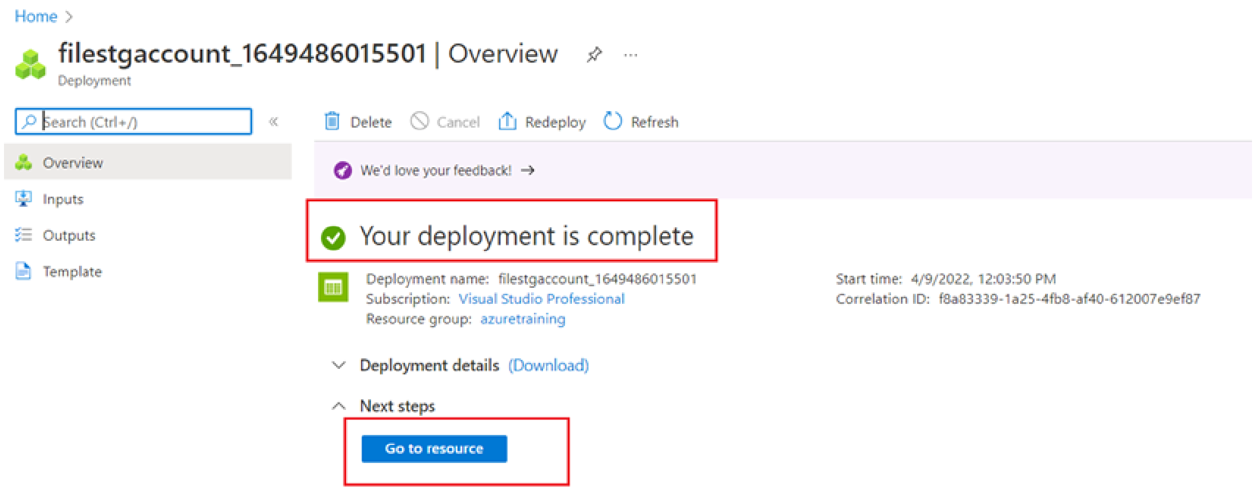
Deployment name: filestgaccount_1649486015501
Subscription: Visual Studio Professional
Resource group: azuretraining

Start time: 4/9/2022, 12:03:50 PM
Correlation ID: f8a83339-1a25-4fb8-af40-612007e9ef87

Deployment details (Download)

Resource	Type	Status	Operation details
No results.			

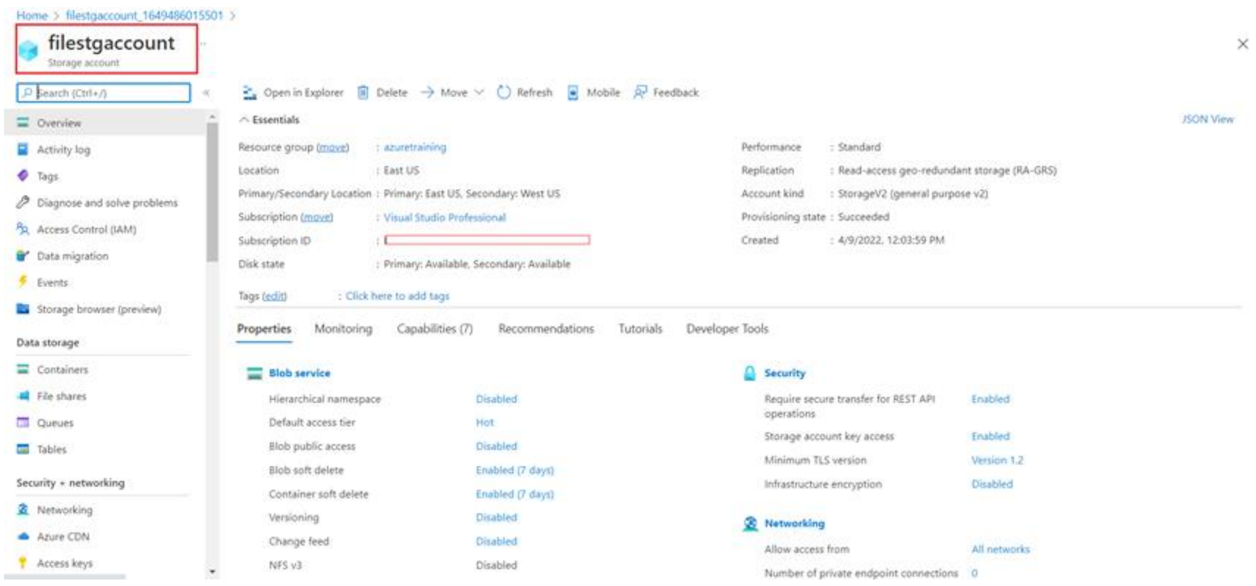
Our storage account is created successfully. Now click on Go to resource button to see the storage account.



Your deployment is complete

Deployment name: filestgaccount_1649486015501
Subscription: Visual Studio Professional
Resource group: azuretraining

Start time: 4/9/2022, 12:03:50 PM
Correlation ID: f8a83339-1a25-4fb8-af40-612007e9ef87


Home > filestgaccount_1649486015501 >

filestgaccount
Storage account

Essentials

Resource group (move)	: azuretraining
Location	: East US
Primary/Secondary Location	: Primary: East US, Secondary: West US
Subscription (move)	: Visual Studio Professional
Subscription ID	: [redacted]
Disk state	: Primary: Available, Secondary: Available
Tags (edit)	: Click here to add tags
Performance	: Standard
Replication	: Read-access geo-redundant storage (RA-GRS)
Account kind	: StorageV2 (general purpose v2)
Provisioning state	: Succeeded
Created	: 4/9/2022, 12:03:59 PM

Blob service

Hierarchical namespace	Disabled
Default access tier	Hot
Blob public access	Disabled
Blob soft delete	Enabled (7 days)
Container soft delete	Enabled (7 days)
Versioning	Disabled
Change feed	Disabled
NFS v3	Disabled

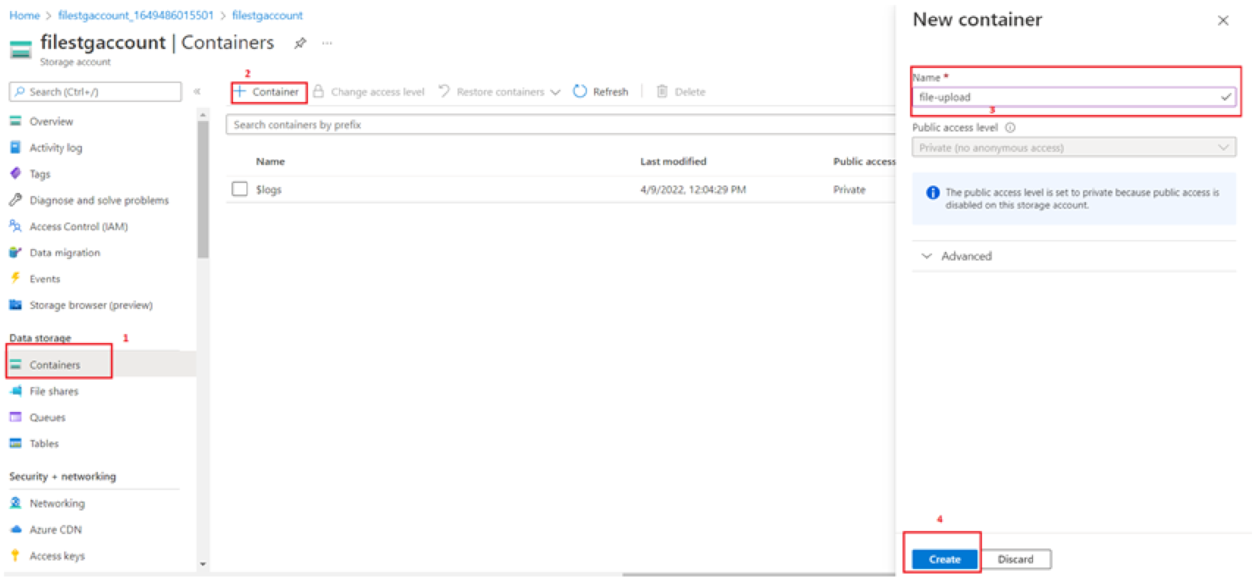
Security

Require secure transfer for REST API operations	Enabled
Storage account key access	Enabled
Minimum TLS version	Version 1.2
Infrastructure encryption	Disabled

Networking

Allow access from	All networks
Number of private endpoint connections	0

Now the next step is to create a container to store our blobs. So in the left section click on Containers and then click on + Container button. Give the proper name to container in which we are going to upload our files and click on Create button.


Home > filestgaccount_1649486015501 > filestgaccount

filestgaccount | Containers
Storage account

Name	Last modified	Public access
$logs	4/9/2022, 12:04:29 PM	Private

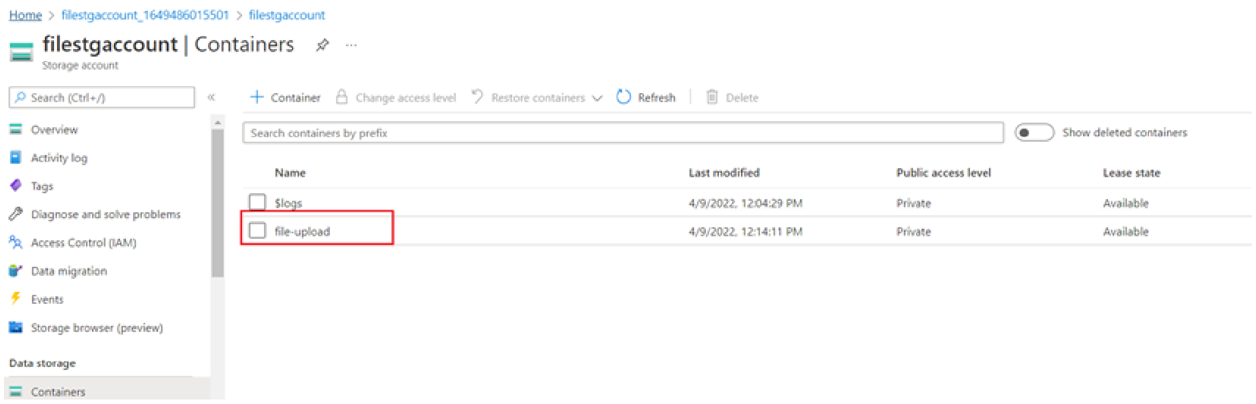
New container

Name *
file-upload

Public access level
Private (no anonymous access)

The public access level is set to private because public access is disabled on this storage account.

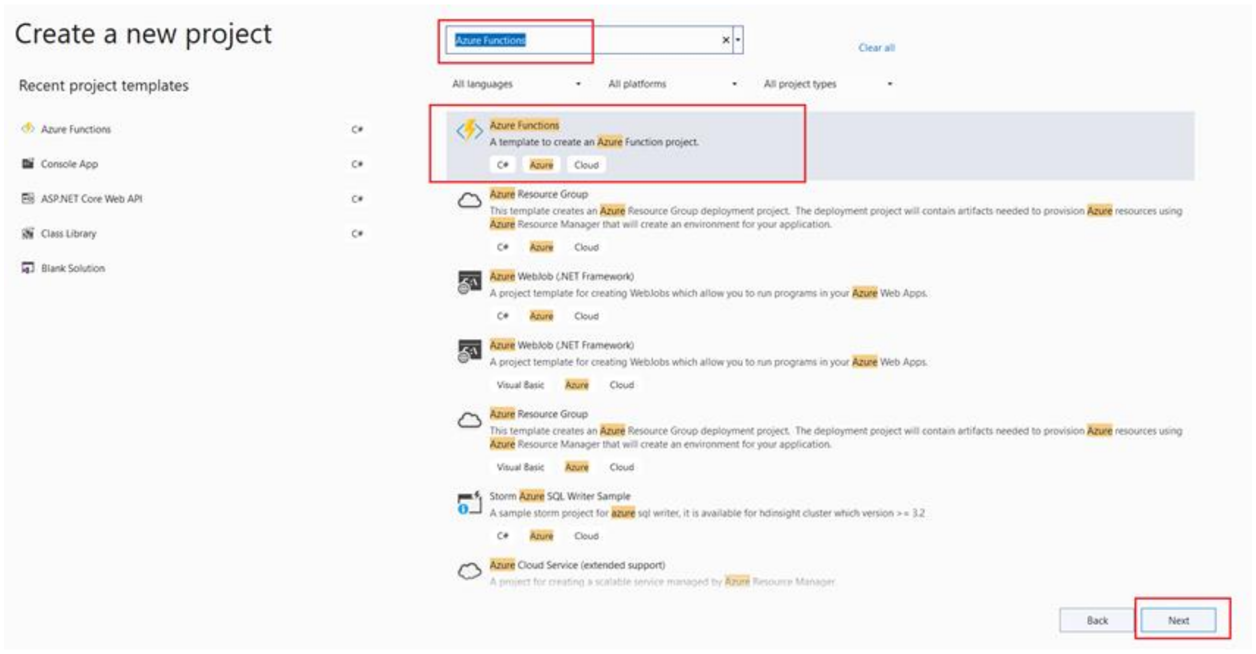
Next step is to create an Azure function that uploads files into the “file-upload” container.



Name	Last modified	Public access level	Lease state
$logs	4/9/2022, 12:04:29 PM	Private	Available
file-upload	4/9/2022, 12:14:11 PM	Private	Available

Create a HTTP trigger azure function using C# to upload files to blob storage

So open Visual Studio and Go to File -> New -> Project. Search "Azure Functions" in the search box and select the Azure function template and click on Next.



Azure Functions
A template to create an Azure Function project.
C# Azure Cloud

Azure Resource Group
This template creates an Azure Resource Group deployment project. The deployment project will contain artifacts needed to provision Azure resources using Azure Resource Manager that will create an environment for your application.
C# Azure Cloud

Azure WebJob (.NET Framework)
A project template for creating WebJobs which allow you to run programs in your Azure Web Apps.
C# Azure Cloud

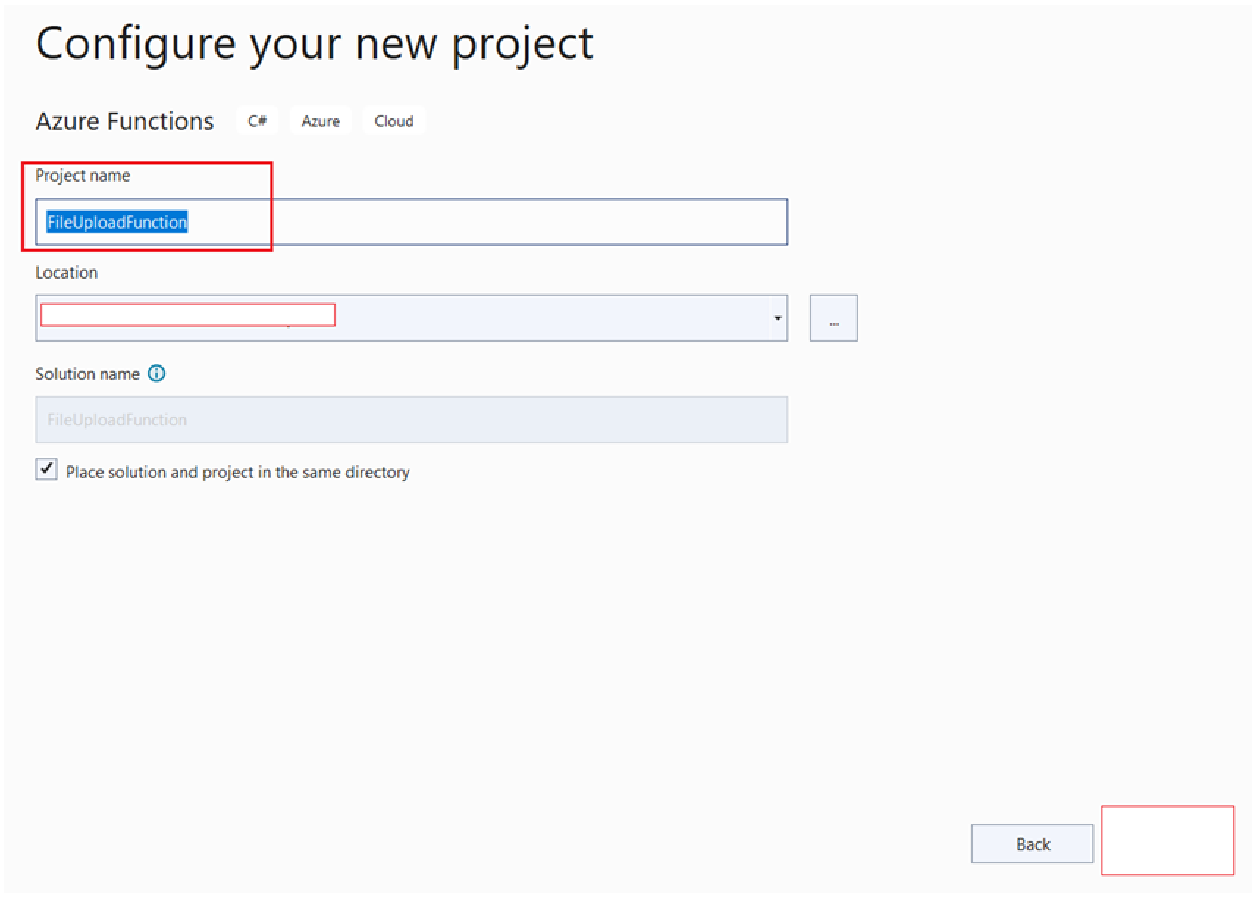
Azure WebJob (.NET Framework)
A project template for creating WebJobs which allow you to run programs in your Azure Web Apps.
Visual Basic Azure Cloud

Azure Resource Group
This template creates an Azure Resource Group deployment project. The deployment project will contain artifacts needed to provision Azure resources using Azure Resource Manager that will create an environment for your application.
Visual Basic Azure Cloud

Storm Azure SQL Writer Sample
A sample storm project for azure sql writer, it is available for hdinsight cluster which version >= 3.2
C# Azure Cloud

Azure Cloud Service (extended support)
A project for creating a scalable service managed by Azure Resource Manager

Give a name to the function project and click on Create.



Configure your new project

Azure Functions

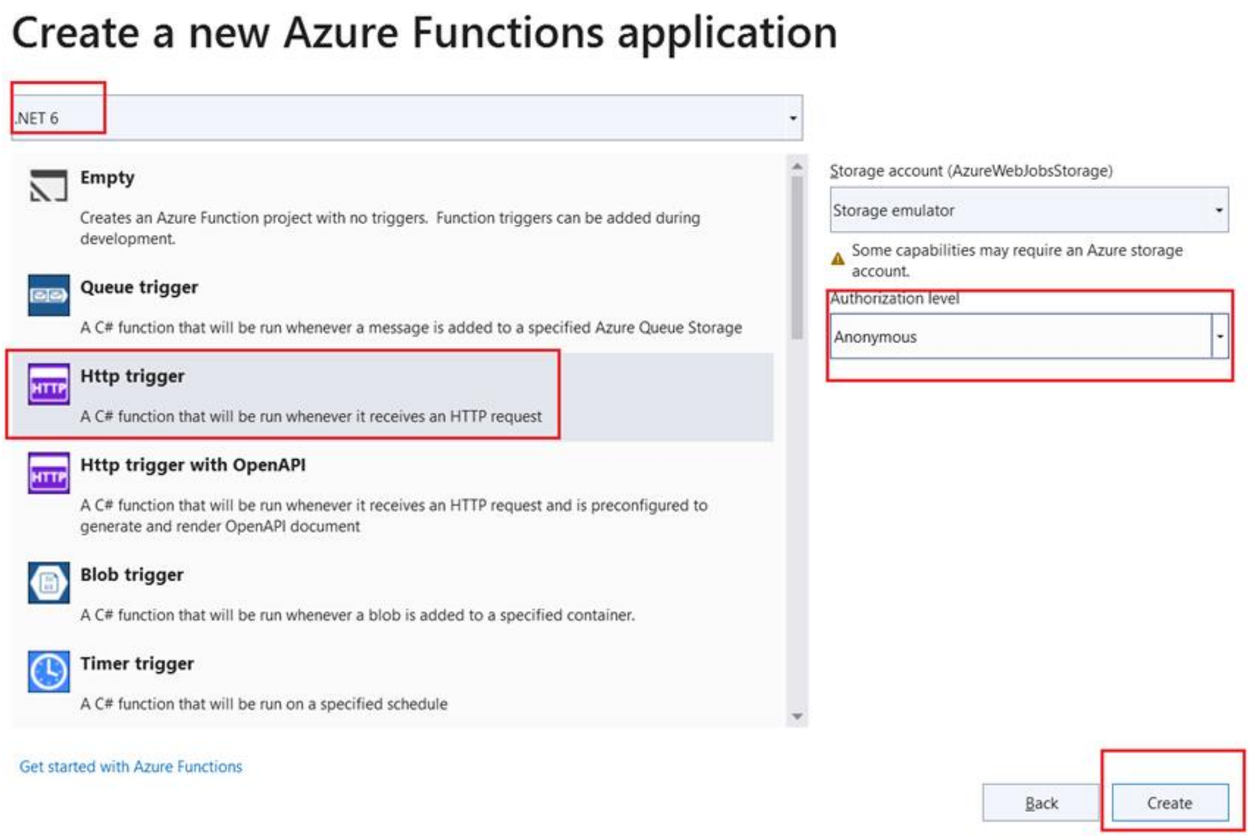
Project name: FileUploadFunction

Location: [redacted]

Solution name: FileUploadFunction

✓ Place solution and project in the same directory

Select the HTTP trigger template and set the Authorization level as Anonymous and click on Create.


Create a new Azure Functions application

.NET 6

Empty
Creates an Azure Function project with no triggers. Function triggers can be added during development.

Queue trigger
A C# function that will be run whenever a message is added to a specified Azure Queue Storage

Http trigger
A C# function that will be run whenever it receives an HTTP request

Http trigger with OpenAPI
A C# function that will be run whenever it receives an HTTP request and is preconfigured to generate and render OpenAPI document

Blob trigger
A C# function that will be run whenever a blob is added to a specified container.

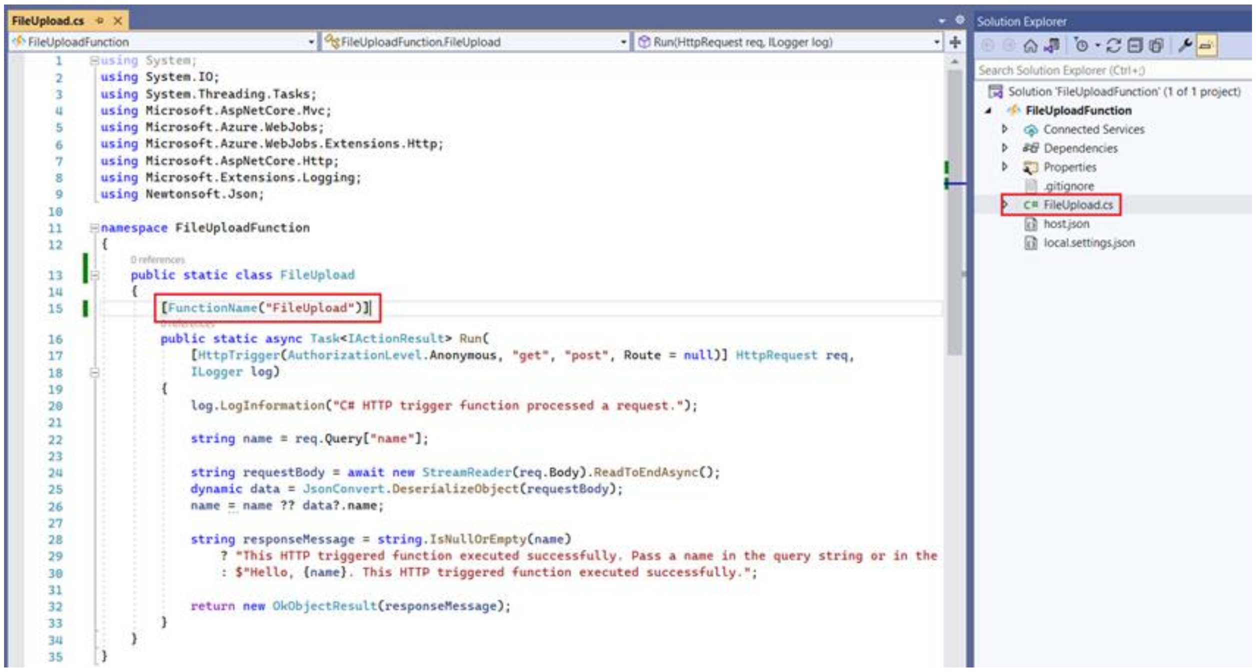
Timer trigger
A C# function that will be run on a specified schedule

Storage account (AzureWebJobsStorage): Storage emulator

Some capabilities may require an Azure storage account.

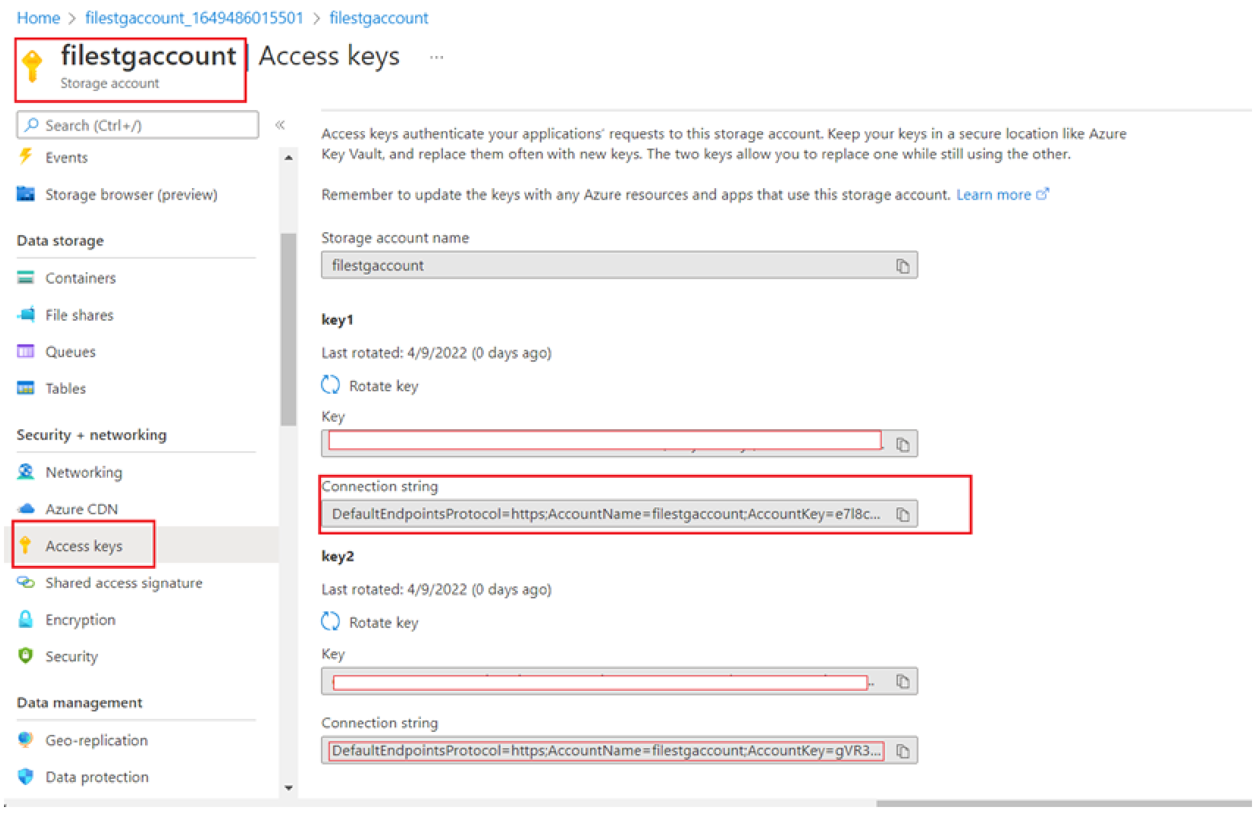
Authorization level: Anonymous

That's it. We have created our first Azure function. By default the name of the function is Function1.cs so now change it to "**FileUpload**".



```
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 FileUploadFunction
{
    public static class FileUpload
    {
        [FunctionName("FileUpload")]
        public static async Task<IActionResult> Run(
            [HttpTrigger(AuthorizationLevel.Anonymous, "get", "post", Route = null)] HttpRequest req,
            ILogger log)
        {
            log.LogInformation("C# HTTP trigger function processed a request.");

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

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

            string responseMessage = string.IsNullOrEmpty(name)
                ? "This HTTP triggered function executed successfully. Pass a name in the query string or in the
                : $"Hello, {name}. This HTTP triggered function executed successfully.";

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

So for our application, we are creating HTTP Trigger function which takes file as input and uploads it into the Azure Blob. To connect with the Blob Storage container we need a connection string. So let's open the Azure Storage resource in portal ->Access Keys -> Click on Show Keys - > Copy the Key 1 Connection String.



Access keys authenticate your applications' requests to this storage account. Keep your keys in a secure location like Azure Key Vault, and replace them often with new keys. The two keys allow you to replace one while still using the other.

Remember to update the keys with any Azure resources and apps that use this storage account. Learn more

Storage account name

filestgaccount

key1

Last rotated: 4/9/2022 (0 days ago)

Rotate key

Key

[redacted]

Connection string

DefaultEndpointsProtocol=https;AccountName=filestgaccount;AccountKey=e7l8c...

key2

Last rotated: 4/9/2022 (0 days ago)

Rotate key

Key

[redacted]

Connection string

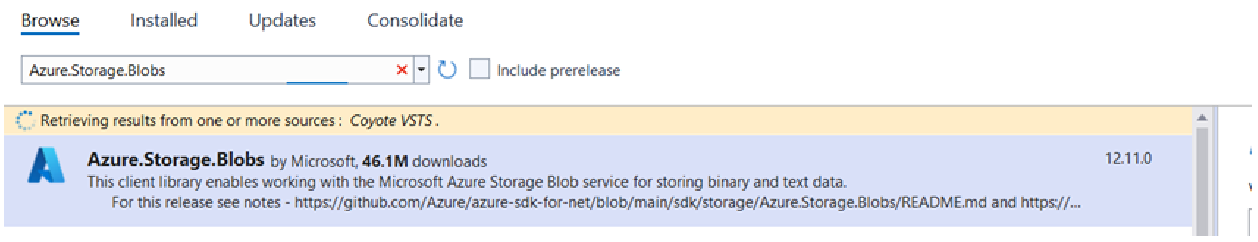
DefaultEndpointsProtocol=https;AccountName=filestgaccount;AccountKey=gVR3...

Open the local.settings.json file in our function app and paste the connection string of our Azure Storage resource as value of “AzureWebJobsStorage” key. Also add another key called “ContainerName” and paste the name of container we have created earlier.

```
{
    "IsEncrypted": false,
    "Values": {
        "AzureWebJobsStorage": "<replace your blob storage connection key here>",
        "ContainerName": "file-upload", // Container name
        "FUNCTIONS_WORKER_RUNTIME": "dotnet"
    }
}
```

JavaScript
Copy

To interact with Azure storage we have to first install the below NuGet package.



Azure.Storage.Blobs

Retrieving results from one or more sources : Coyote VSTS .

Azure.Storage.Blobs by Microsoft, **46.1M** downloads 12.11.0
This client library enables working with the Microsoft Azure Storage Blob service for storing binary and text data.
For this release see notes - https://github.com/Azure/azure-sdk-for-net/blob/main/sdk/storage/Azure.Storage.Blobs/README.md and https://...

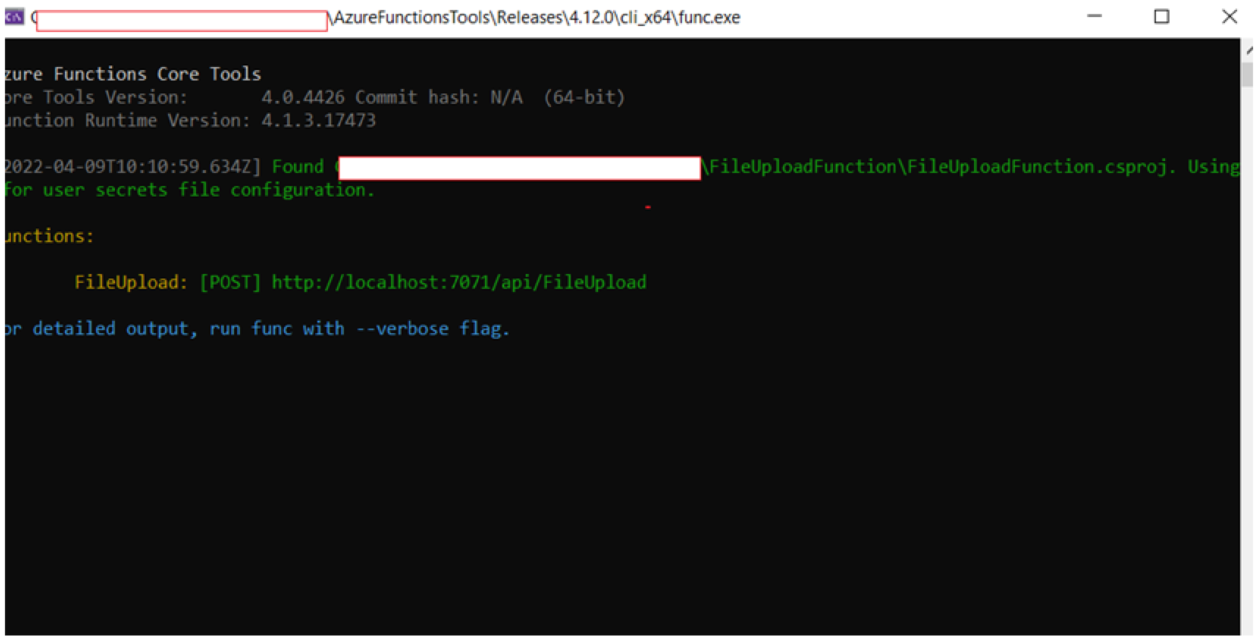
Now add the below code to upload a file into the container in our blob storage.

```
using Azure.Storage.Blobs;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Azure.WebJobs;
using Microsoft.Azure.WebJobs.Extensions.Http;
using Microsoft.Extensions.Logging;
using System;
using System.IO;
using System.Threading.Tasks;
namespace FileUploadFunction {
    public static class FileUpload {
        [FunctionName("FileUpload")]
        public static async Task < IActionResult > Run(
            [HttpTrigger(AuthorizationLevel.Anonymous, "post", Route =
null)] HttpRequest req, ILogger log) {
            string Connection =
Environment.GetEnvironmentVariable("AzureWebJobsStorage");
            string containerName =
Environment.GetEnvironmentVariable("ContainerName");
            Stream myBlob = new MemoryStream();
            var file = req.Form.Files["File"];
            myBlob = file.OpenReadStream();
            var blobClient = new BlobContainerClient(Connection,
containerName);
            var blob = blobClient.GetBlobClient(file.FileName);
            await blob.UploadAsync(myBlob);
            return new OkObjectResult("file uploaded successfylly");
        }
    }
}
```
C#
Copy

First we have stored the Blob Storage connection string and container name from configuration into local variables. After that we have read the file request by using key called “File” into variable and using OpenReadStream() function we have read the file into the steam. To interact with Blob storage we first have to create BlobContainerClient by passing connection string and container name. After creating client, we have to create BlobClient using GetBlobClient method by passing input file name.

Finally to upload file to container we have used UploadAsync method by passing stream. You can read more about these methods here.

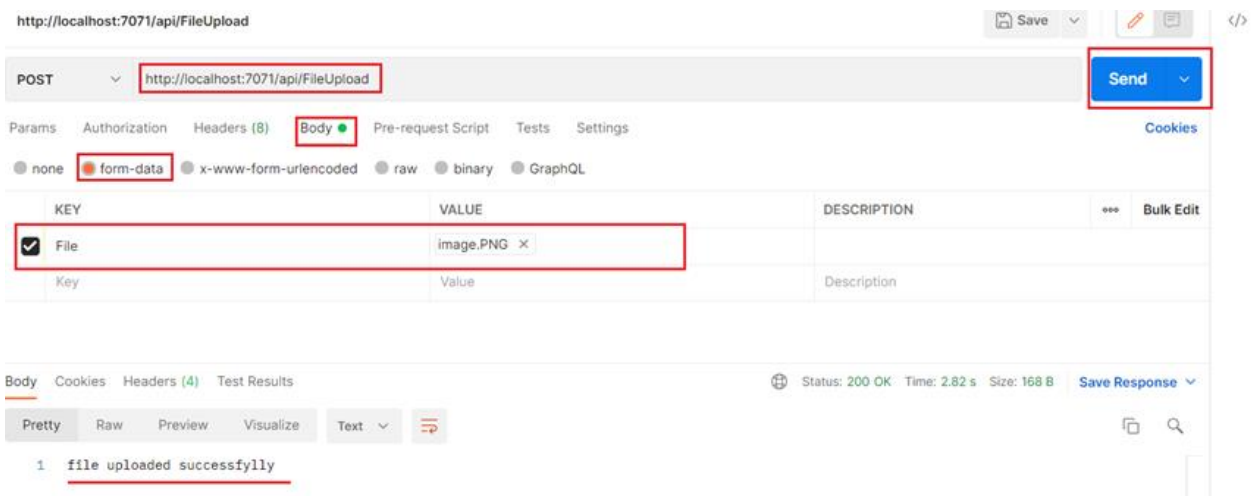
Now run the FileUpload function.



```
C[redacted]\AzureFunctionsTools\Releases\4.12.0\cli_x64\func.exe

zure Functions Core Tools
ore Tools Version:       4.0.4426 Commit hash: N/A  (64-bit)
unction Runtime Version: 4.1.3.17473

2022-04-09T10:10:59.634Z] Found C[redacted]\FileUploadFunction\FileUploadFunction.csproj. Using
for user secrets file configuration.

unctions:

        FileUpload: [POST] http://localhost:7071/api/FileUpload

or detailed output, run func with --verbose flag.
```

For testing the function open postman and paste the about function URL i.e. http://localhost:7071/api/FileUpload

Now in our code we have read the file from form data so click on Body and select form-data then add key as “File” and for value select which file you want to upload and click on Send button.

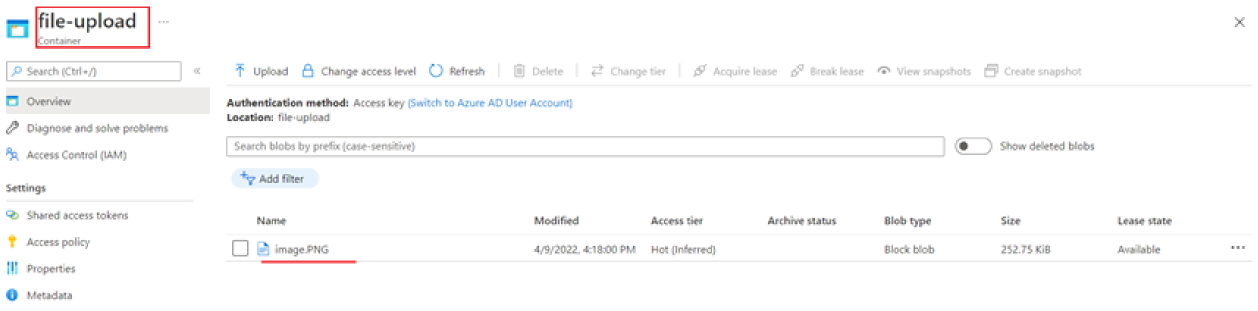


KEY	VALUE	DESCRIPTION
File	image.PNG	
Key	Value	Description

Status: 200 OK Time: 2.82 s Size: 168 B

```
file uploaded successfylly
```

Now if we open our “**file-upload**” container we can see the file which we uploaded.



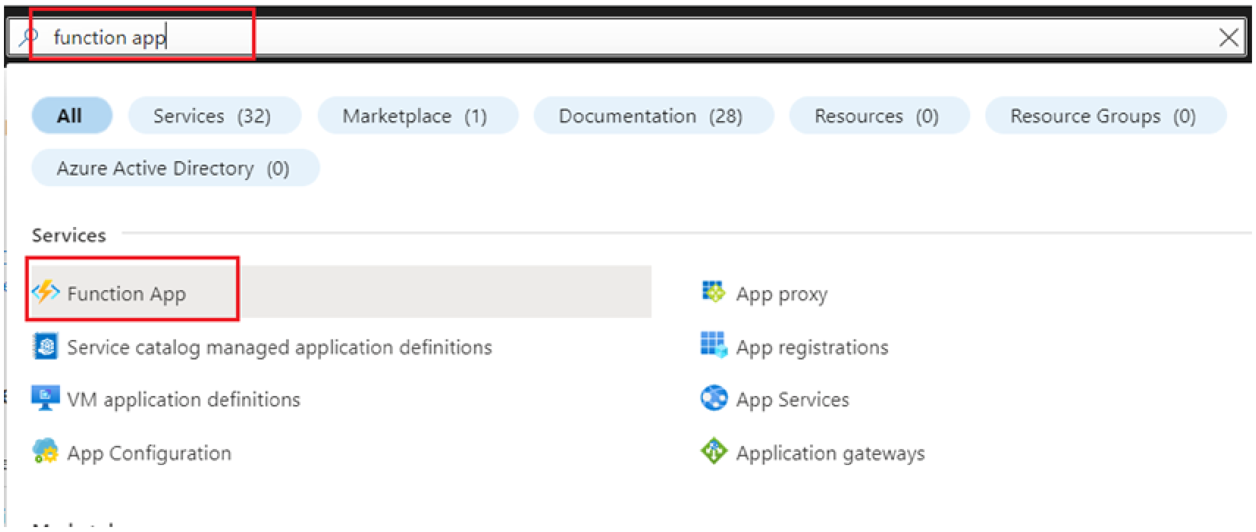
Authentication method: Access key (Switch to Azure AD User Account)
Location: file-upload

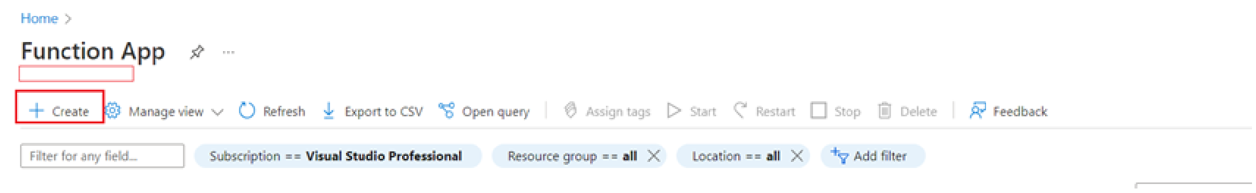
Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
image.PNG	4/9/2022, 4:18:00 PM	Hot (Inferred)		Block blob	252.75 KiB	Available

Publish Azure function to Azure function app using Visual Studio

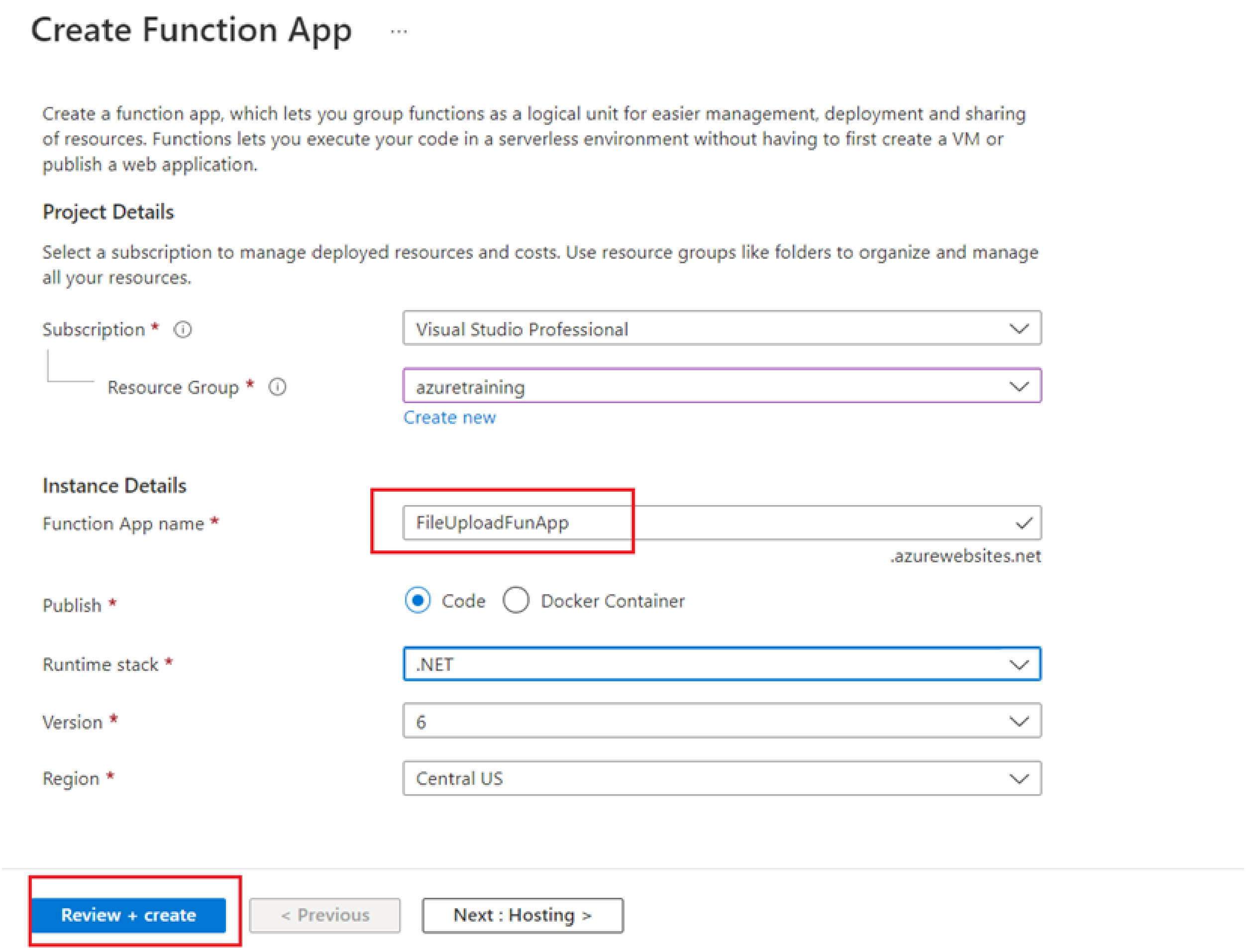
Create a function app using Azure portal

Log in to the Azure portal In the Search bar, search as "function app" and then select the Function app.





After that click on the " + Create" button to add a new function app. Fill out the basic details,



Create Function App

Create a function app, which lets you group functions as a logical unit for easier management, deployment and sharing of resources. Functions lets you execute your code in a serverless environment without having to first create a VM or publish a web application.

Project Details

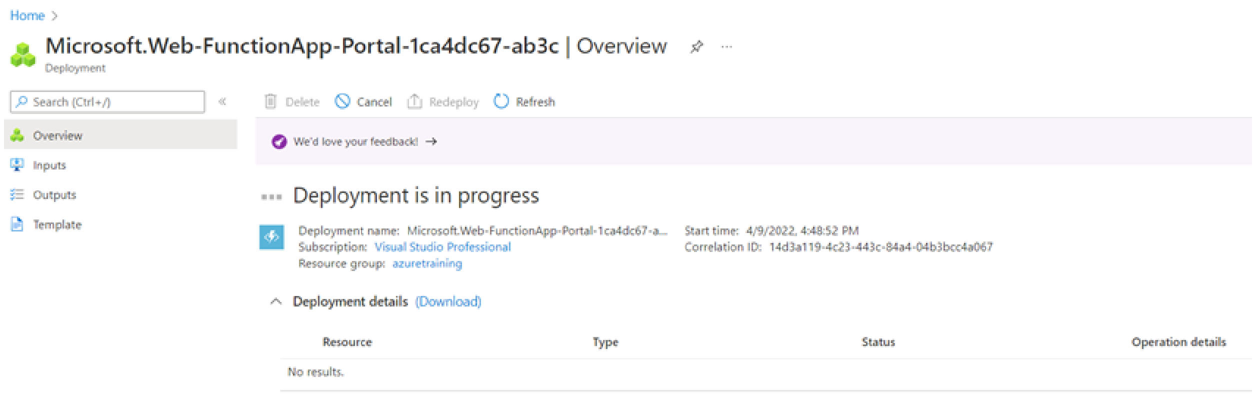
Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	Visual Studio Professional
Resource Group *	azuretraining

Instance Details

Function App name *	FileUploadFunApp .azurewebsites.net
Publish *	[✓ Code] Docker Container
Runtime stack *	.NET
Version *	6
Region *	Central US

Now click on Review: Create button and review all the details and click on the Create button. Wait for a few minutes to create the resources.



Microsoft.Web-FunctionApp-Portal-1ca4dc67-ab3c | Overview

Deployment

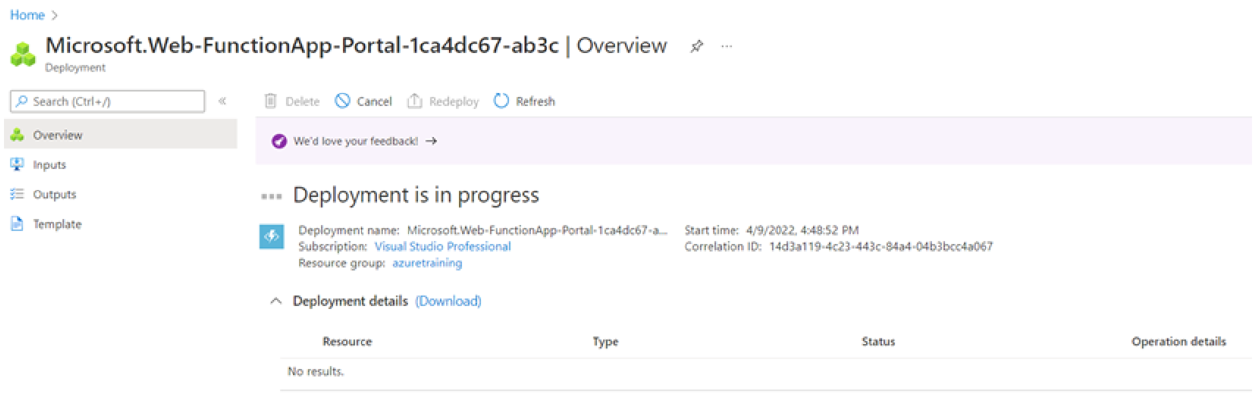
Deployment is in progress

Deployment name: Microsoft.Web-FunctionApp-Portal-1ca4dc67-a...
Subscription: Visual Studio Professional
Resource group: azuretraining
Start time: 4/9/2022, 4:48:52 PM
Correlation ID: 14d3a119-4c23-443c-84a4-04b3bcc4a067

Deployment details (Download)

Resource	Type	Status	Operation details
No results.			

Once deployment is complete click on Go to resource button to see our new function app.


Home >

Microsoft.Web-FunctionApp-Portal-1ca4dc67-ab3c | Overview

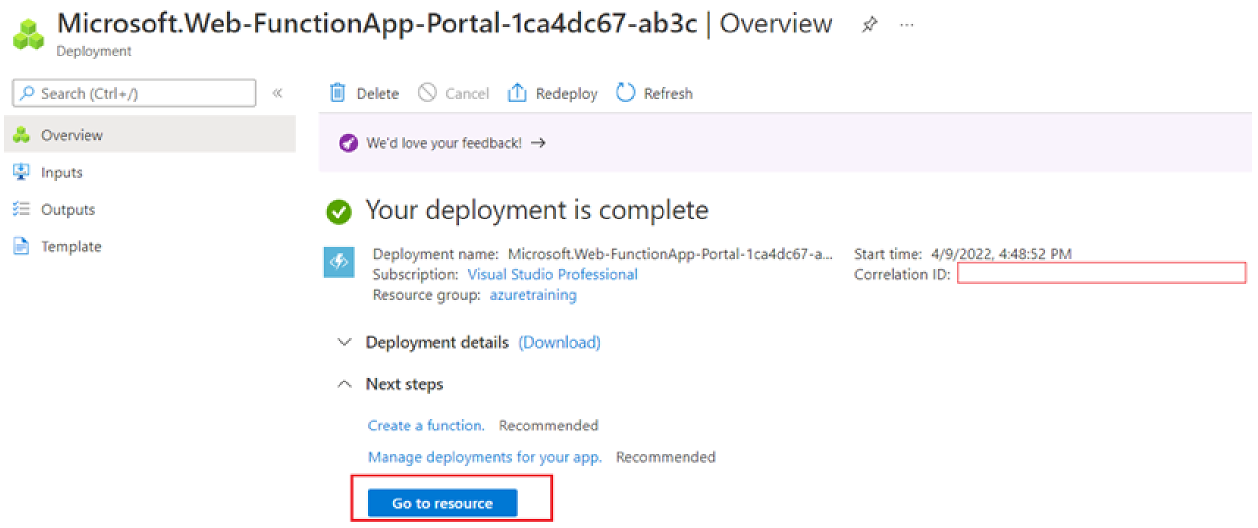
Deployment

Deployment is in progress

Deployment name: Microsoft.Web-FunctionApp-Portal-1ca4dc67-a...
Subscription: Visual Studio Professional
Resource group: azuretraining
Start time: 4/9/2022, 4:48:52 PM
Correlation ID: 14d3a119-4c23-443c-84a4-04b3bcc4a067

Deployment details (Download)

Resource	Type	Status	Operation details
No results.			


Microsoft.Web-FunctionApp-Portal-1ca4dc67-ab3c | Overview

Deployment

Your deployment is complete

Deployment name: Microsoft.Web-FunctionApp-Portal-1ca4dc67-a...
Subscription: Visual Studio Professional
Resource group: azuretraining
Start time: 4/9/2022, 4:48:52 PM
Correlation ID:

Deployment details (Download)

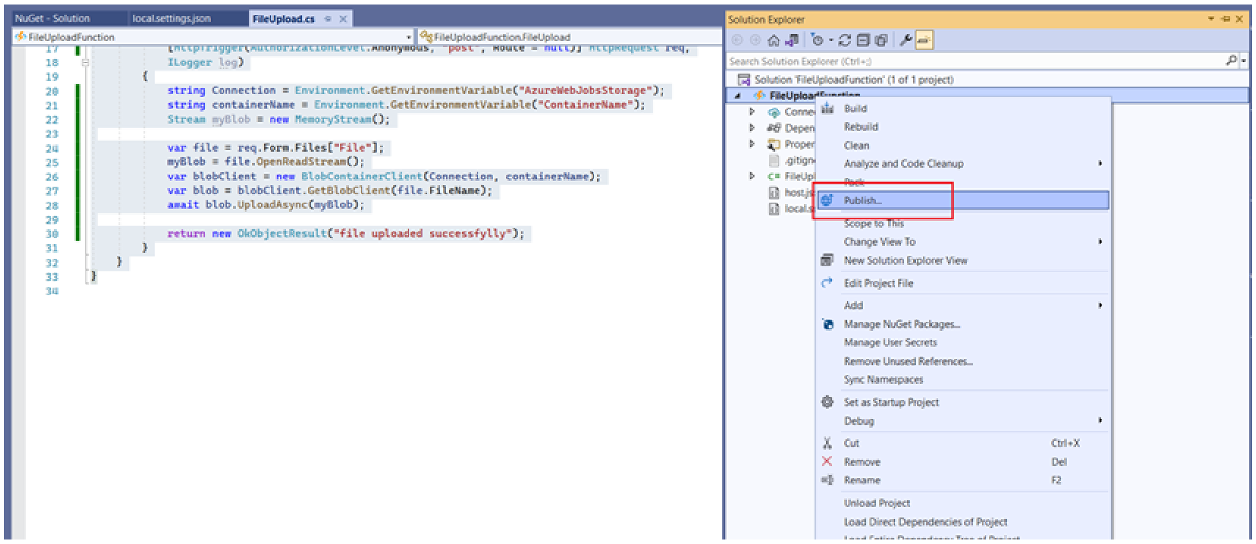
Next steps

Create a function. Recommended

Manage deployments for your app. Recommended

Publish the azure function into the function app

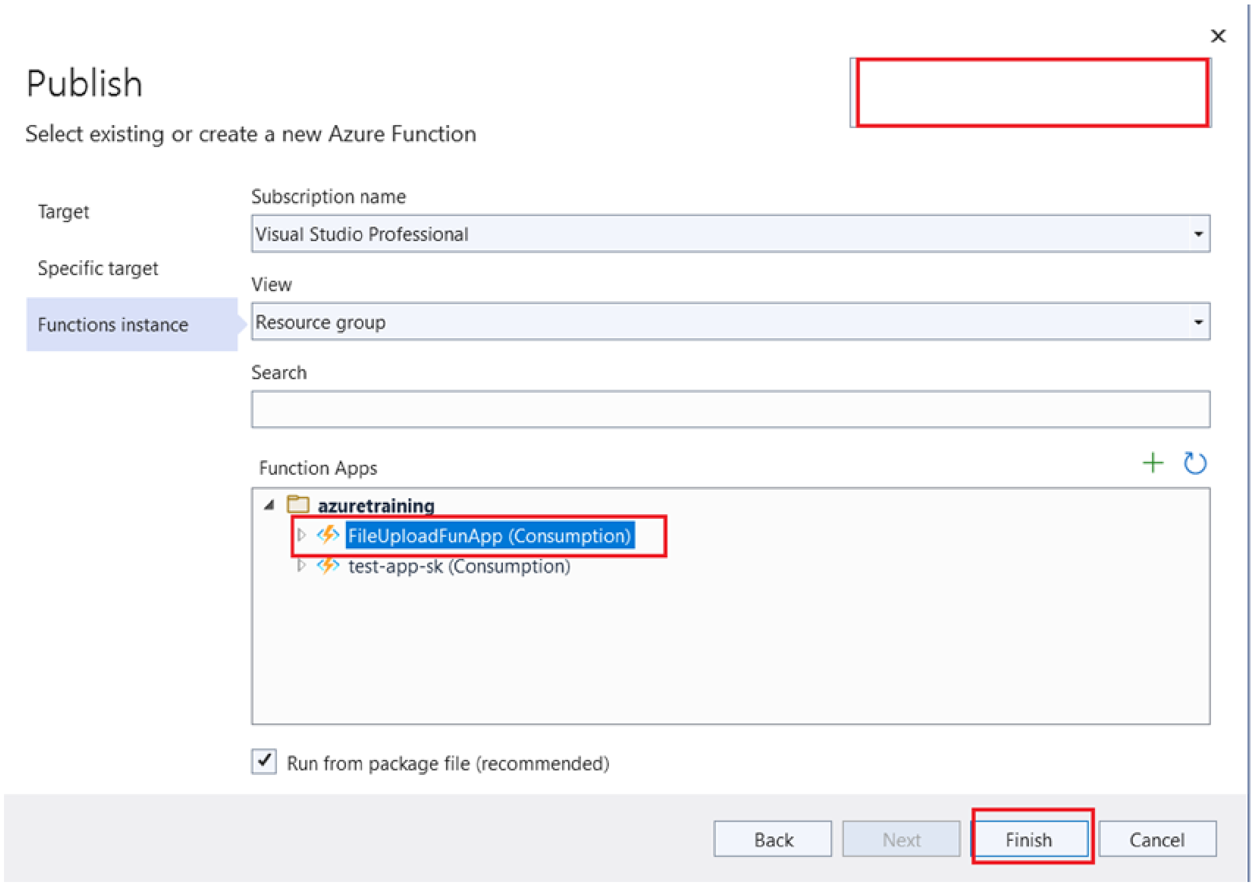
To deploy our function to Azure we need Azure Function App. Right Click on our solution and click on the Publish button.



```
            ILogger log)
        {
            string Connection = Environment.GetEnvironmentVariable("AzureWebJobsStorage");
            string containerName = Environment.GetEnvironmentVariable("ContainerName");
            Stream myBlob = new MemoryStream();

            var file = req.Form.Files["File"];
            myBlob = file.OpenReadStream();
            var blobClient = new BlobContainerClient(Connection, containerName);
            var blob = blobClient.GetBlobClient(file.FileName);
            await blob.UploadAsync(myBlob);

            return new OkObjectResult("file uploaded successfylly");
        }
    }
}
```

As we are publishing into Azure then Select Azure and click on Next. Select target as "Azure Function App(Windows)" and click on Next. We can either select an already created function app or we can create a new Function app.

Click on Finish button.



Publish

Select existing or create a new Azure Function

Subscription name: Visual Studio Professional

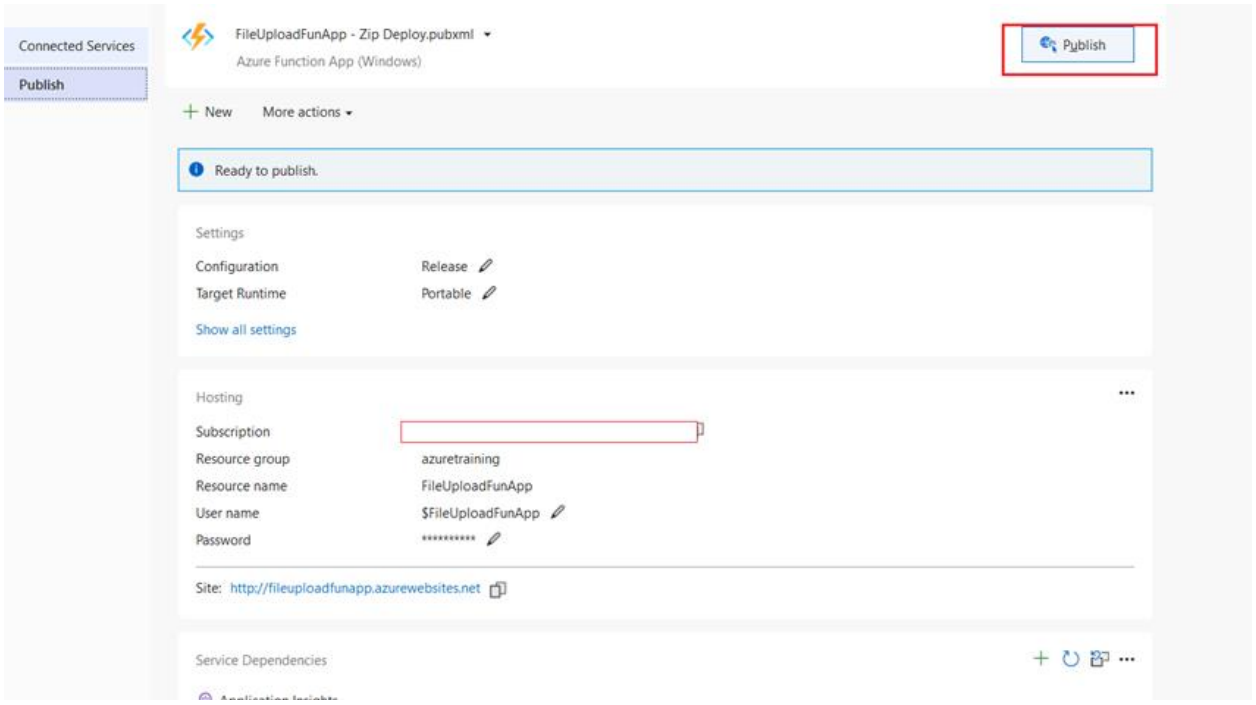
View: Resource group

Search

azuretraining
- FileUploadFunApp (Consumption)
- test-app-sk (Consumption)

✓ Run from package file (recommended)

Click on Publish button to push our function to our Function App.



FileUploadFunApp - Zip Deploy.pubxml
Azure Function App (Windows)

Ready to publish.

Settings

Configuration	Release
Target Runtime	Portable

Show all settings

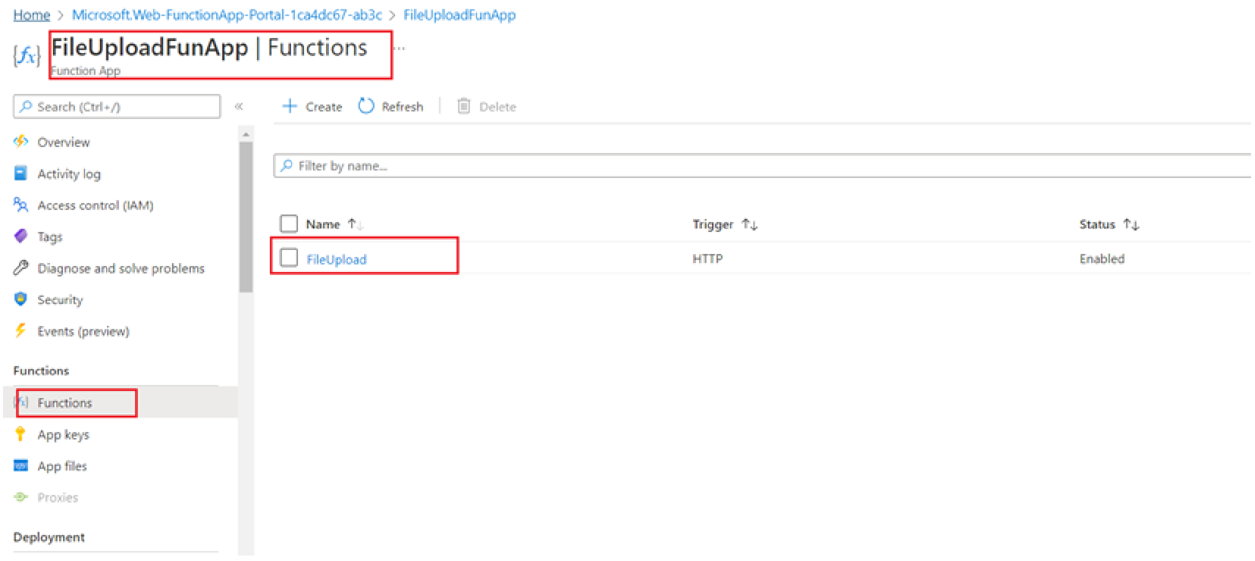
Hosting

Subscription	[redacted]
Resource group	azuretraining
Resource name	FileUploadFunApp
User name	$FileUploadFunApp
Password	**********

Site: http://fileuploadfunapp.azurewebsites.net

Service Dependencies

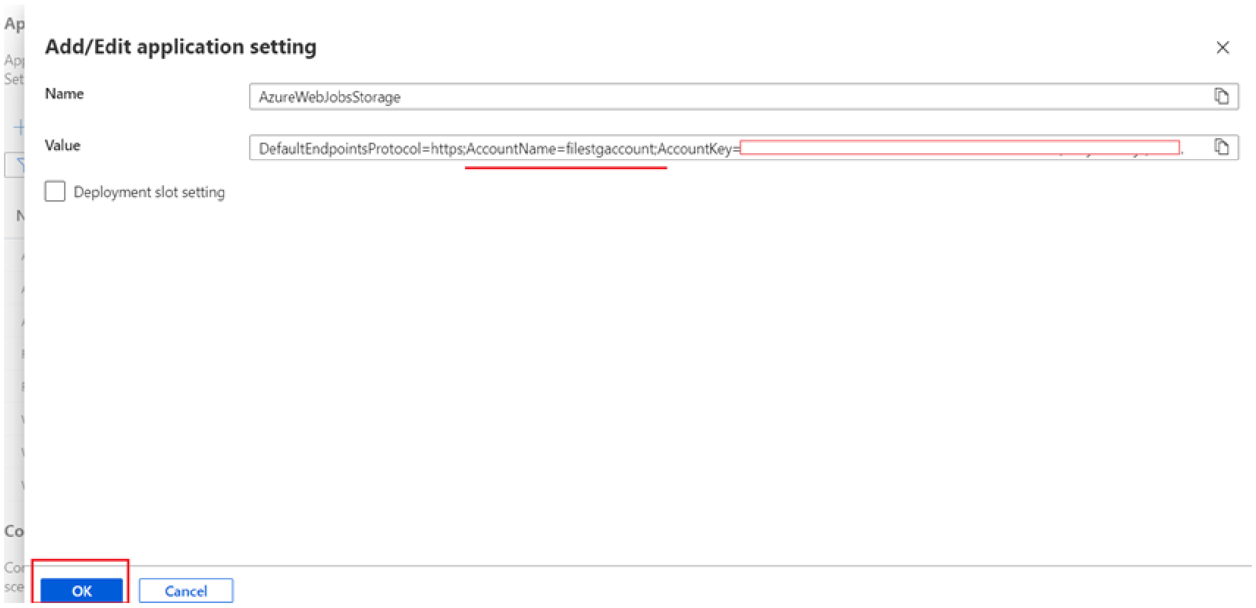
Once our app is successfully published on Azure, go to the Azure portal and search for our function app. Select Functions from the left sidebar to see our deployed function.



Name	Trigger	Status
FileUpload	HTTP	Enabled

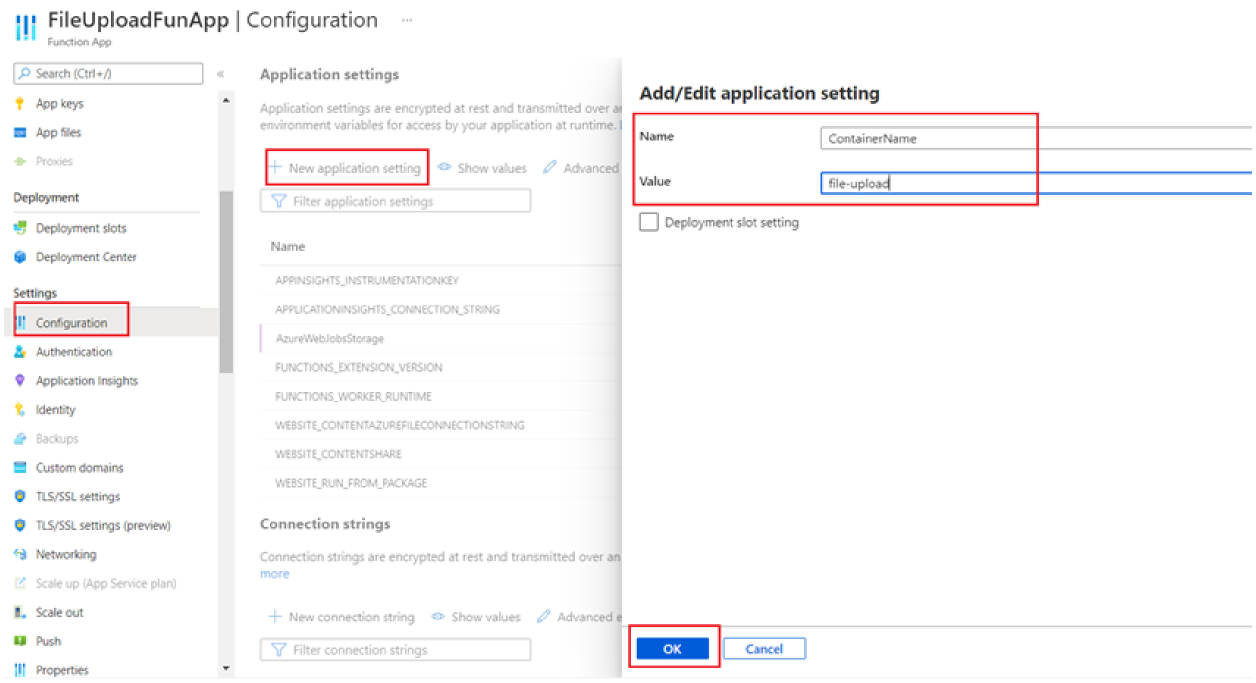
Configure the storage account connection string into Configuration setting of Function app

Now we have deployed our function into Function app so next step is to configure the settings. So click on Configuration. For this app we have to set value of Storage connection string and container name in configuration. So seach for "AzureWebJobsStorage" and click on Edit button and paste your storage connection string.



Name: AzureWebJobsStorage

Value: DefaultEndpointsProtocol=https;AccountName=filestgaccount;AccountKey=[redacted]



Application settings are encrypted at rest and transmitted over a... environment variables for access by your application at runtime.

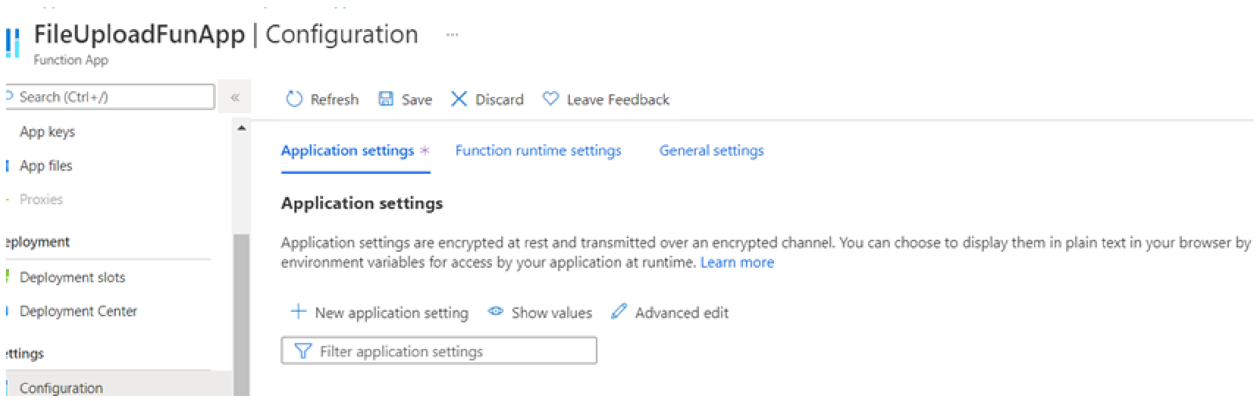
APPINSIGHTS_INSTRUMENTATIONKEY
APPLICATIONINSIGHTS_CONNECTION_STRING
AzureWebJobsStorage
FUNCTIONS_EXTENSION_VERSION
FUNCTIONS_WORKER_RUNTIME
WEBSITE_CONTENTAZUREFILECONNECTIONSTRING
WEBSITE_CONTENTSHARE
WEBSITE_RUN_FROM_PACKAGE

Connection strings are encrypted at rest and transmitted over an... more

Name: ContainerName

Value: file-upload

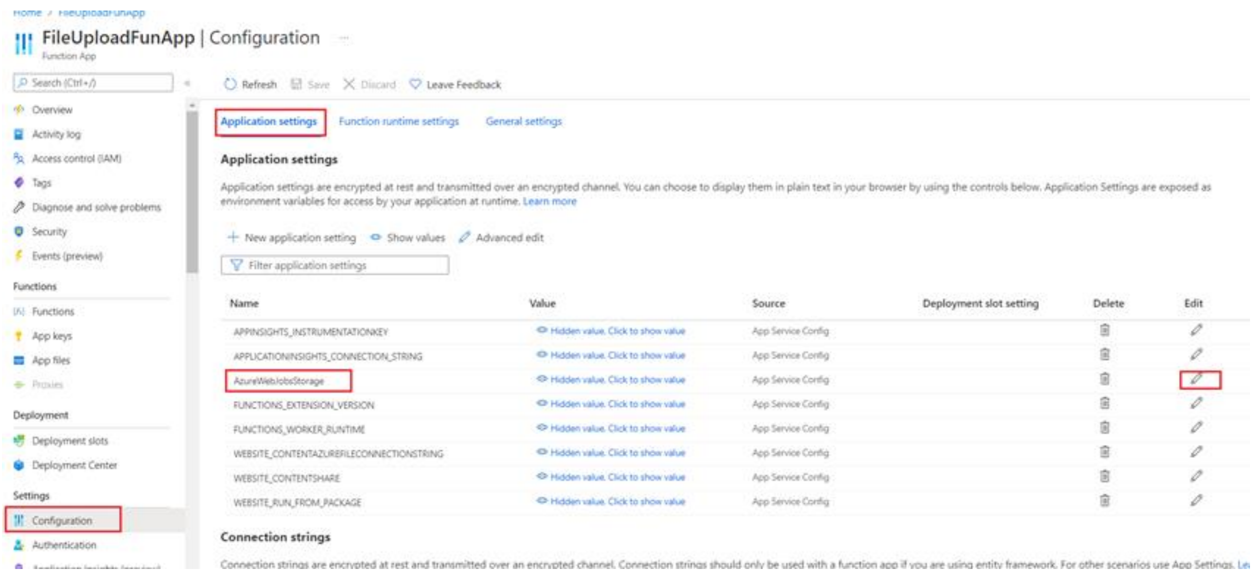
Click on "+ New application setting" and give name as "ContainerName" and Value as "file-upload".


FileUploadFunApp | Configuration

Application settings

Application settings are encrypted at rest and transmitted over an encrypted channel. You can choose to display them in plain text in your browser by environment variables for access by your application at runtime. Learn more

Finally click on Save button to save changes.


FileUploadFunApp | Configuration

Application settings

Application settings are encrypted at rest and transmitted over an encrypted channel. You can choose to display them in plain text in your browser by using the controls below. Application Settings are exposed as environment variables for access by your application at runtime. Learn more

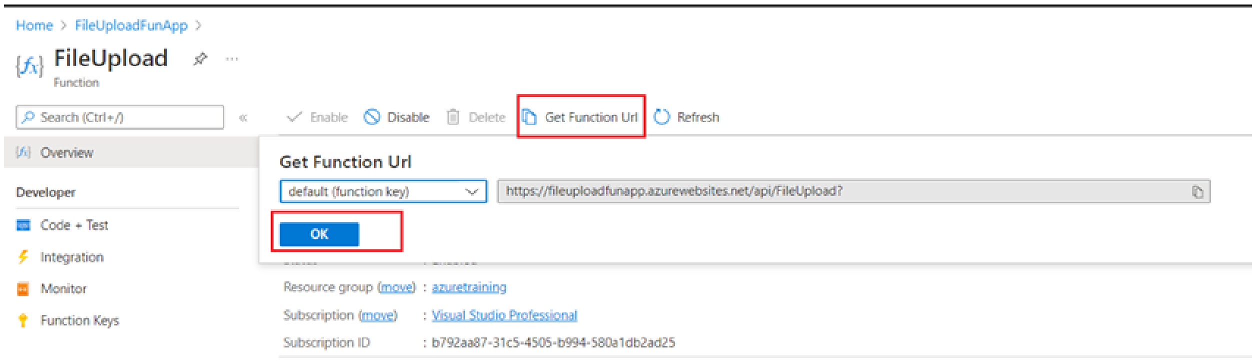
Name	Value	Source	Deployment slot setting	Delete	Edit
APPINSIGHTS_INSTRUMENTATIONKEY	Hidden value. Click to show value	App Service Config			
APPLICATIONINSIGHTS_CONNECTION_STRING	Hidden value. Click to show value	App Service Config			
AzureWebJobsStorage	Hidden value. Click to show value	App Service Config			
FUNCTIONS_EXTENSION_VERSION	Hidden value. Click to show value	App Service Config			
FUNCTIONS_WORKER_RUNTIME	Hidden value. Click to show value	App Service Config			
WEBSITE_CONTENTAZUREFILECONNECTIONSTRING	Hidden value. Click to show value	App Service Config			
WEBSITE_CONTENTSHARE	Hidden value. Click to show value	App Service Config			
WEBSITE_RUN_FROM_PACKAGE	Hidden value. Click to show value	App Service Config			

Connection strings

Connection strings are encrypted at rest and transmitted over an encrypted channel. Connection strings should only be used with a function app if you are using entity framework. For other scenarios use App Settings. Le

Test publish app using postman

To ge the url of deployed function click on Functions -> Select File Upload function -> Click on Get Function Url -> Copy url and paste in postman.


FileUpload

Get Function Url

default (function key)

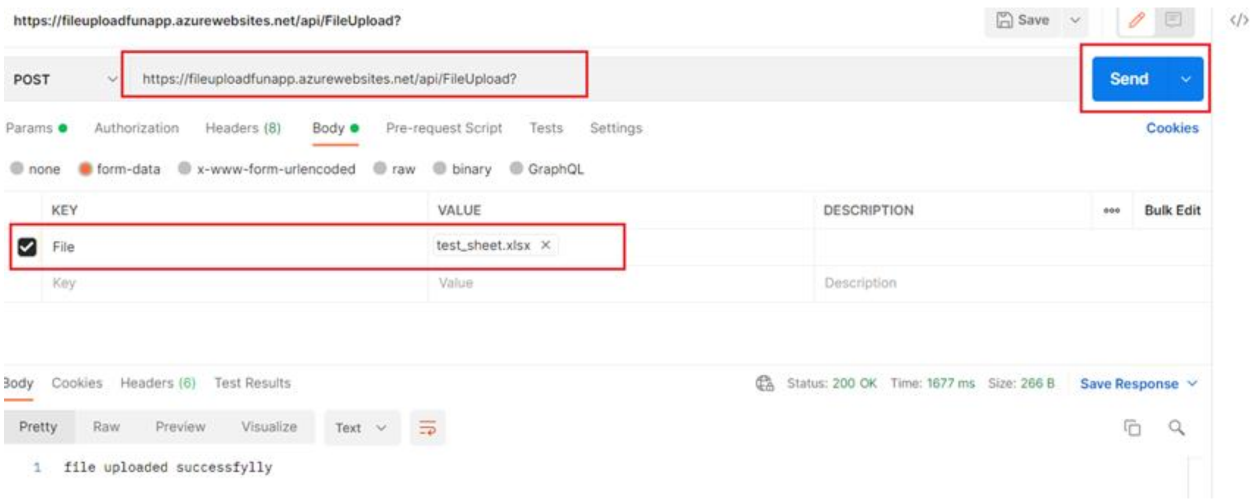
https://fileuploadfunapp.azurewebsites.net/api/FileUpload?

Resource group (move) : azuretraining

Subscription (move) : Visual Studio Professional

Subscription ID : b792aa87-31c5-4505-b994-580a1db2ad25

Now as we test locally in form-data add a key "File" and select file you want to upload.

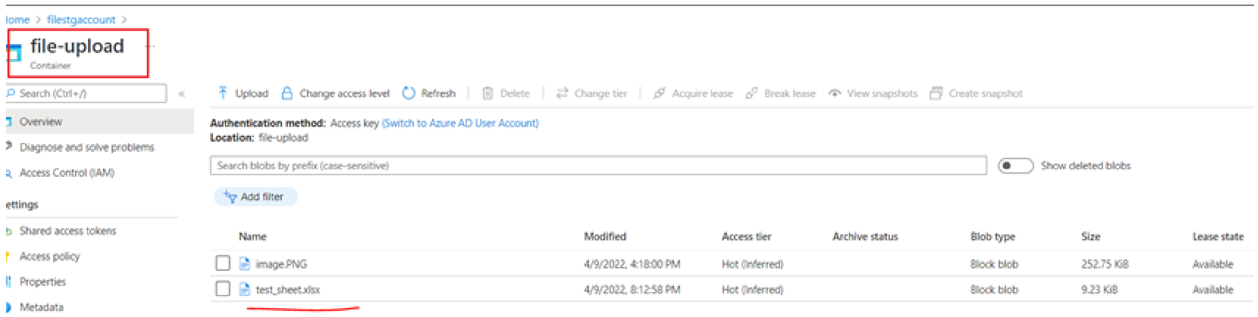


KEY	VALUE	DESCRIPTION
File	test_sheet.xlsx	

Status: 200 OK Time: 1677 ms Size: 266 B

```
file uploaded successfylly
```

Now if we check container we can see the file uploaded into it.



Authentication method: Access key (Switch to Azure AD User Account)
Location: file-upload

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
image.PNG	4/9/2022, 4:18:00 PM	Hot (Inferred)		Block blob	252.75 KiB	Available
test_sheet.xlsx	4/9/2022, 8:12:58 PM	Hot (Inferred)		Block blob	9.23 KiB	Available