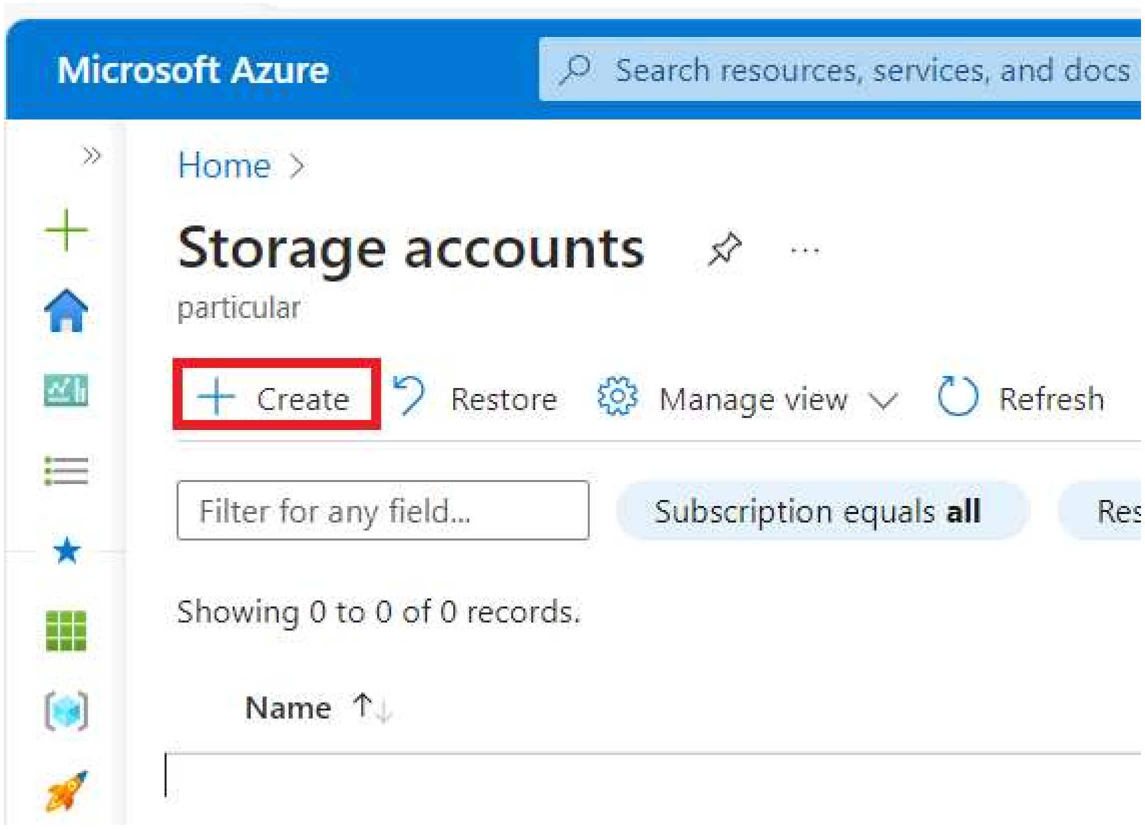
How to use DefaultAzureCredential() for uploading a file to an Azure Blob container

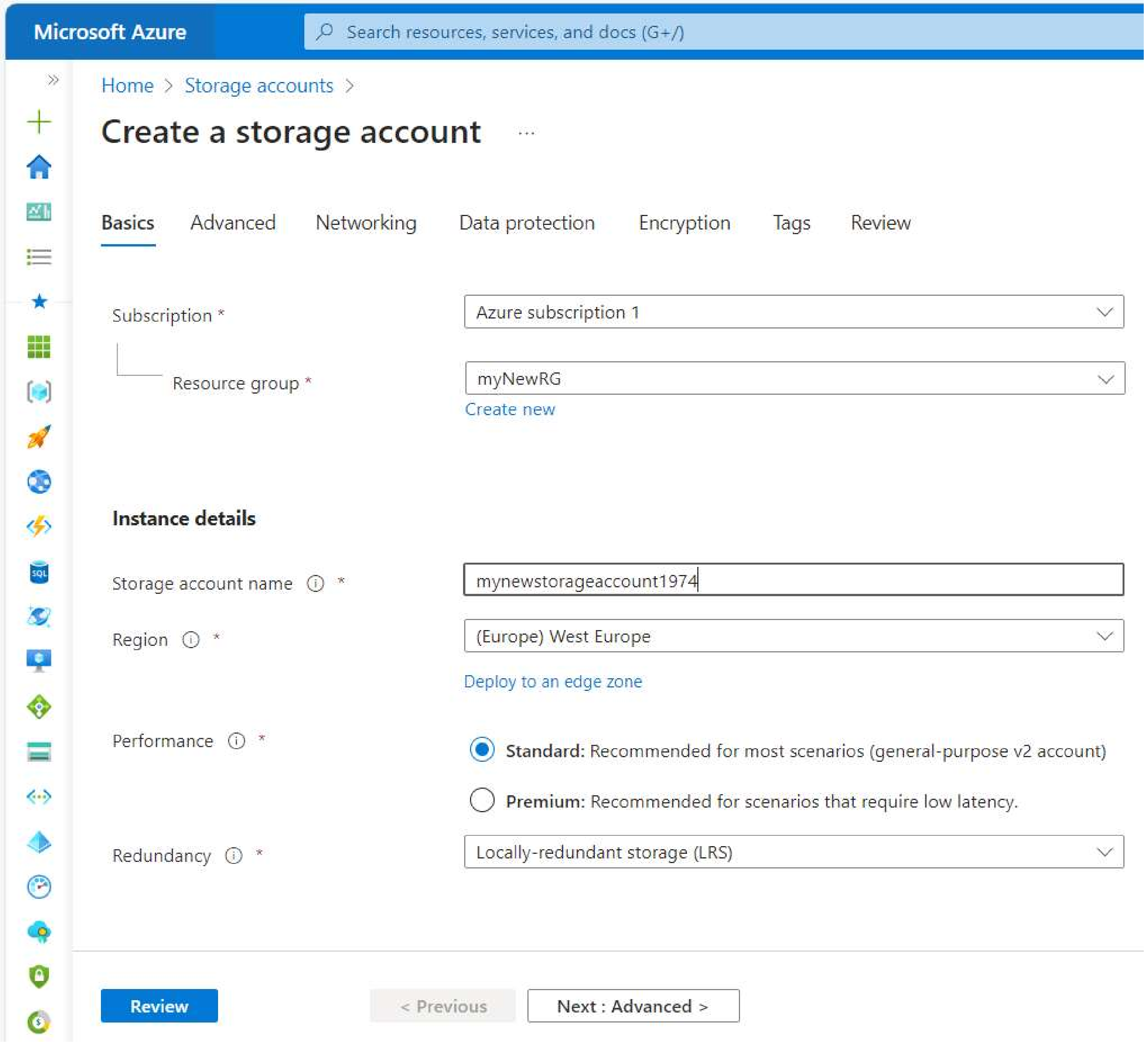
0. Prerequisites

Login in Azure Portal and create a new Azure Storage account and inside create a new Blob container.

First we create the Azure Storage account



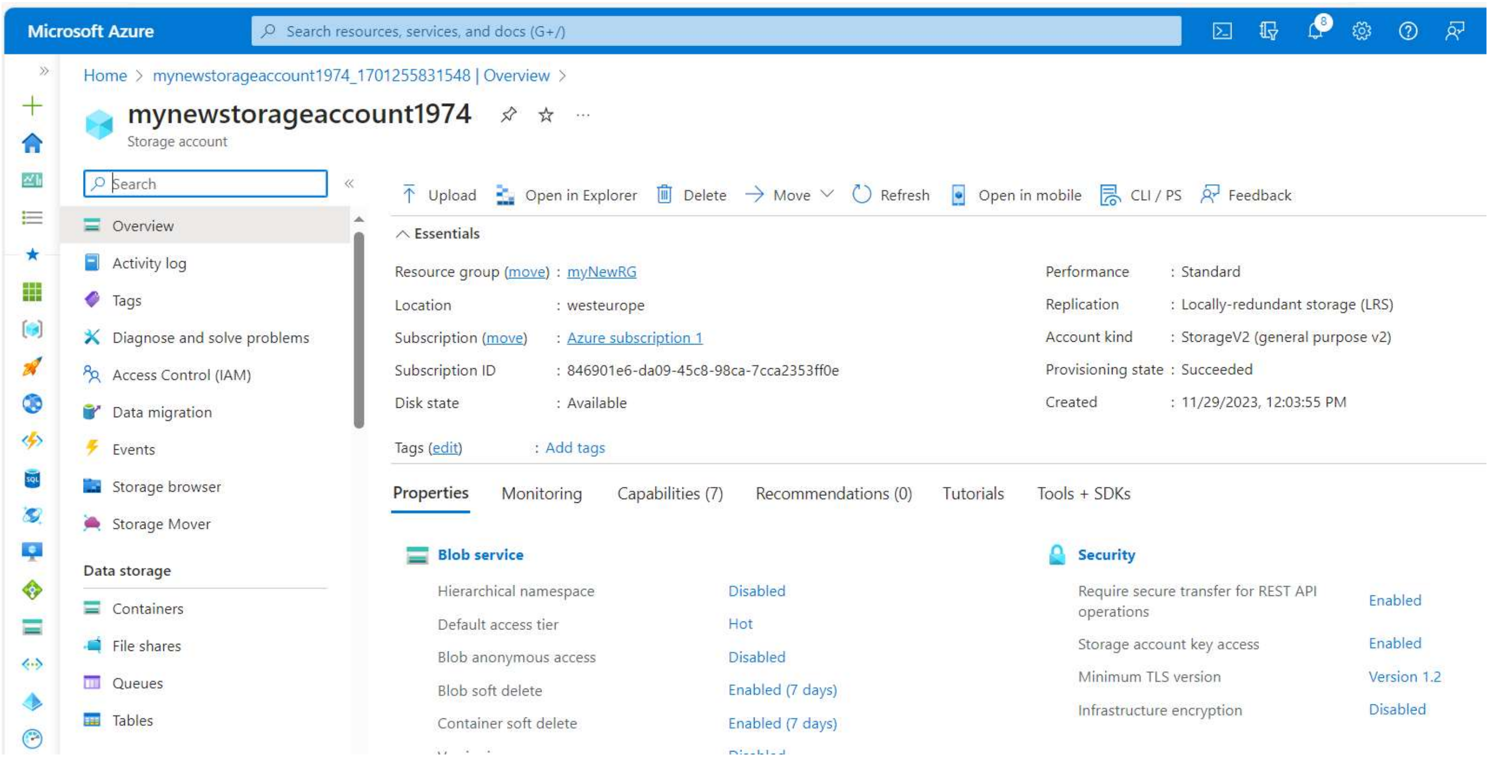
We input the new Storage account data



Subscription * : Azure subscription 1
Resource group * : myNewRG
Storage account name * : mynewstorageaccount1974
Region * : (Europe) West Europe
Performance * : [✓ **Standard:** Recommended for most scenarios (general-purpose v2 account)] **Premium:** Recommended for scenarios that require low latency.
Redundancy * : Locally-redundant storage (LRS)

We press the "Review" button and then the "Create" button

Go to the new Storage account:



Resource group (move)	: myNewRG	Performance	: Standard
Location	: westeurope	Replication	: Locally-redundant storage (LRS)
Subscription (move)	: Azure subscription 1	Account kind	: StorageV2 (general purpose v2)
Subscription ID	: 846901e6-da09-45c8-98ca-7cca2353ff0e	Provisioning state	: Succeeded
Disk state	: Available	Created	: 11/29/2023, 12:03:55 PM
Tags (edit)	: Add tags		

Blob service

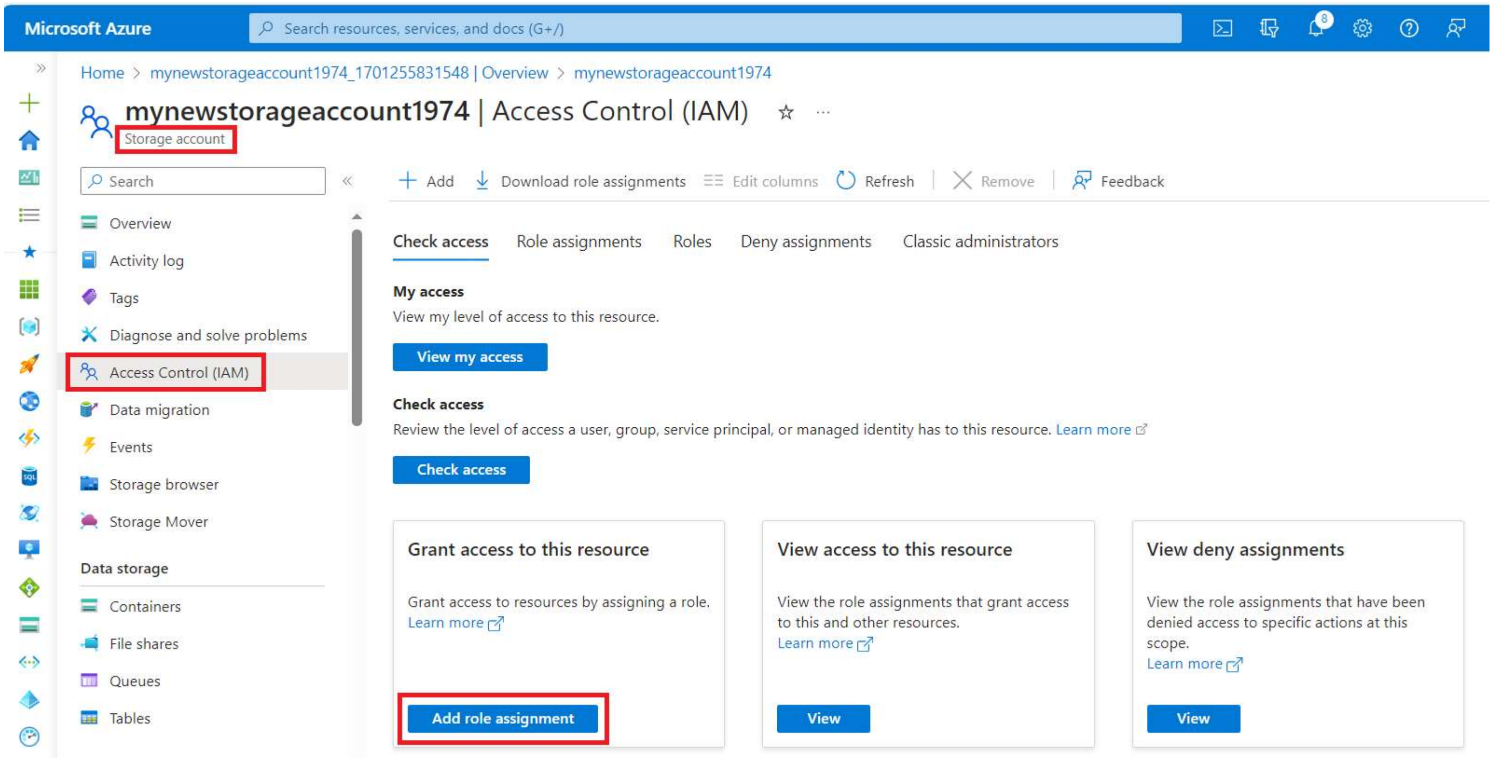
Hierarchical namespace	Disabled
Default access tier	Hot
Blob anonymous access	Disabled
Blob soft delete	Enabled (7 days)
Container soft delete	Enabled (7 days)

Security

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

Now we have to grant permission to the storage account as "**Storage Blob Data Contributor**"

In the Storage account left menu we select "**Access Control (IAM)**" and then we press the button "**Add role assignment**"



My access
View my level of access to this resource.

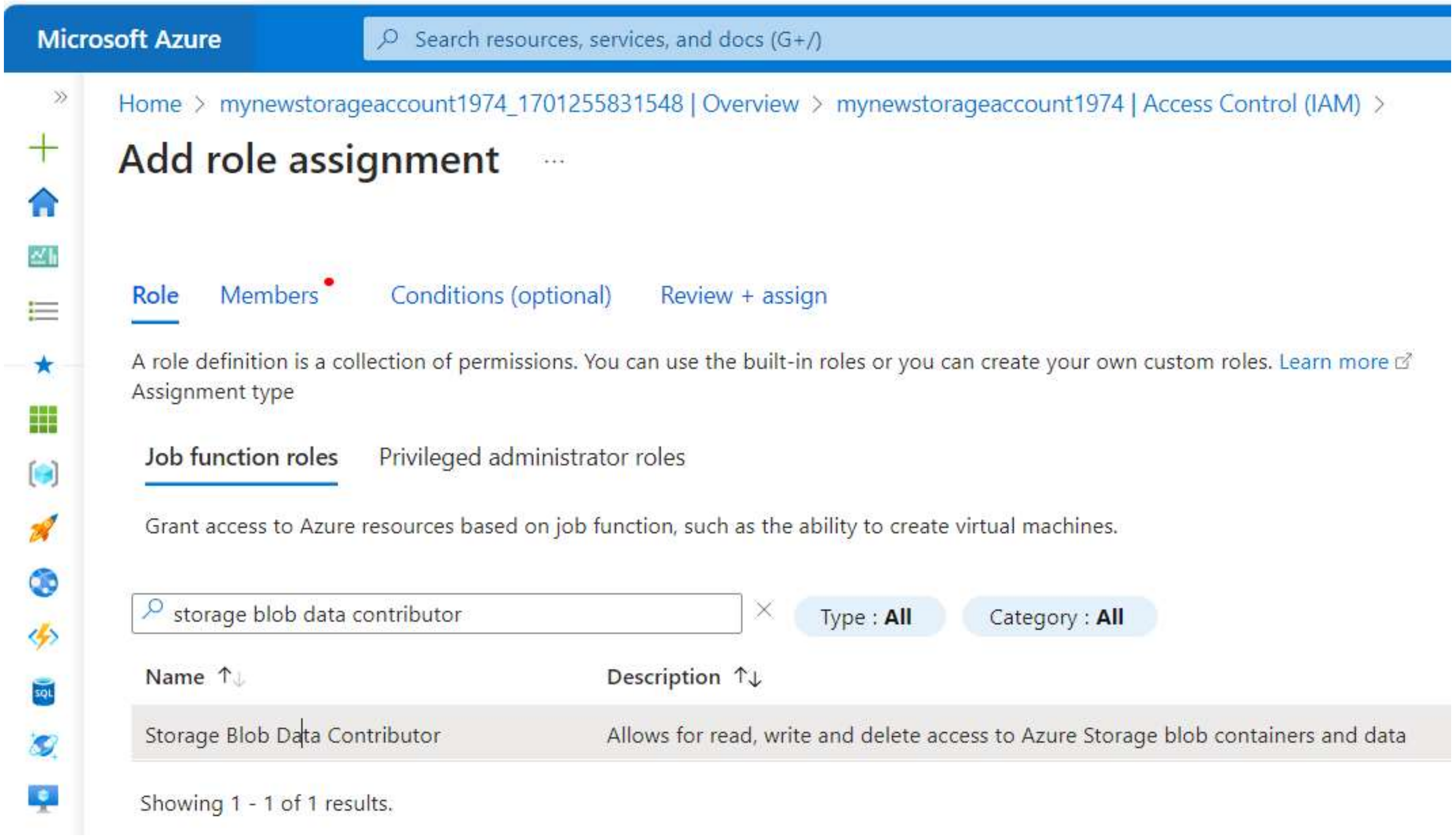
Check access
Review the level of access a user, group, service principal, or managed identity has to this resource. Learn more

Grant access to this resource
Grant access to resources by assigning a role. Learn more

View access to this resource
View the role assignments that grant access to this and other resources. Learn more

View deny assignments
View the role assignments that have been denied access to specific actions at this scope. Learn more

We select the "**Storage Blob Data Contributor**" role



Microsoft Azure
Search resources, services, and docs (G+/)
Home > mynewstorageaccount1974_1701255831548 | Overview > mynewstorageaccount1974 | Access Control (IAM) >
Role
Members
Conditions (optional)
Review + assign
Job function roles
Privileged administrator roles
Type : All
Category : All

Add role assignment

A role definition is a collection of permissions. You can use the built-in roles or you can create your own custom roles. Learn more

Assignment type

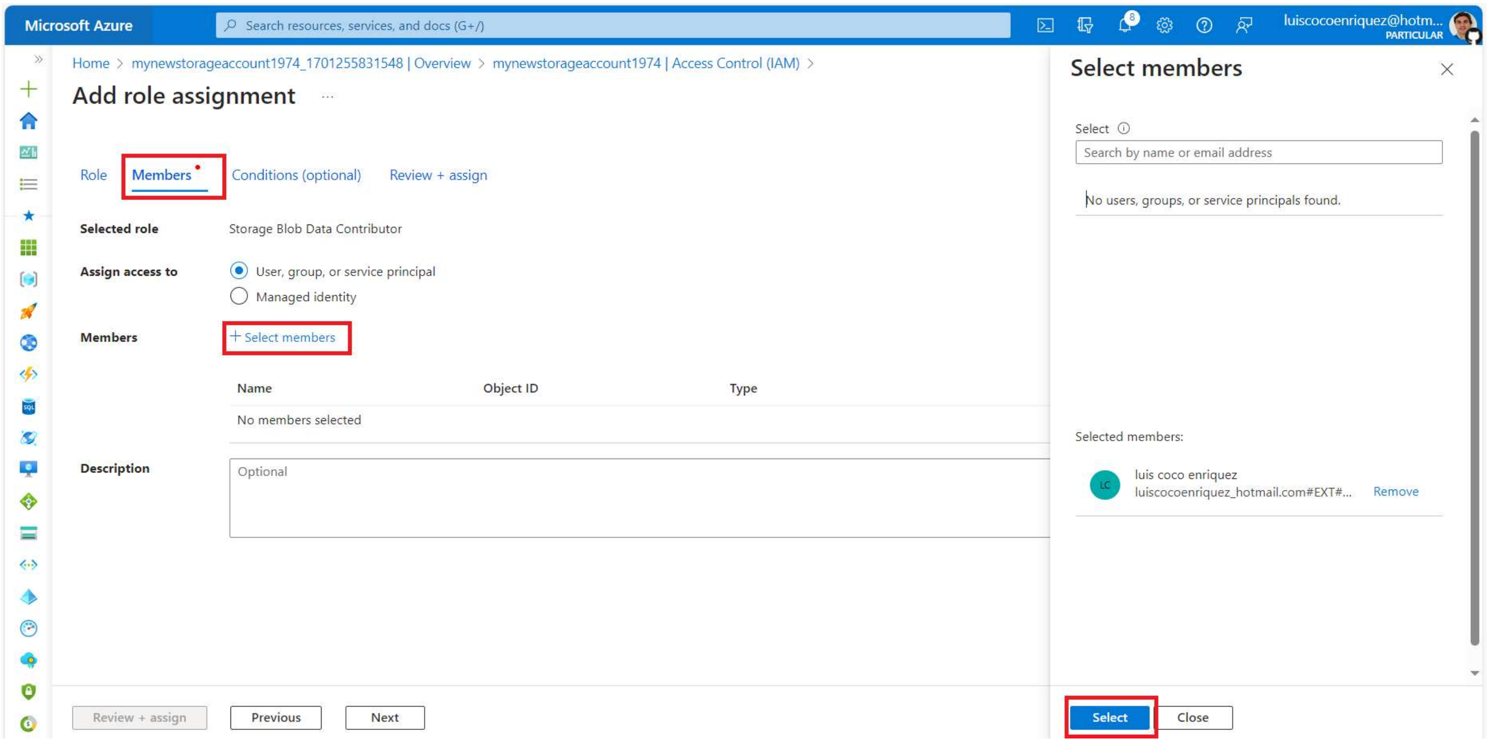
Grant access to Azure resources based on job function, such as the ability to create virtual machines.

storage blob data contributor

Name	Description
Storage Blob Data Contributor	Allows for read, write and delete access to Azure Storage blob containers and data

Showing 1 - 1 of 1 results.

Then we select **user**



Add role assignment

Selected role: Storage Blob Data Contributor

Assign access to: [✓ User, group, or service principal] Managed identity

Members

Name	Object ID	Type
No members selected		

Description: Optional

Select members

Select

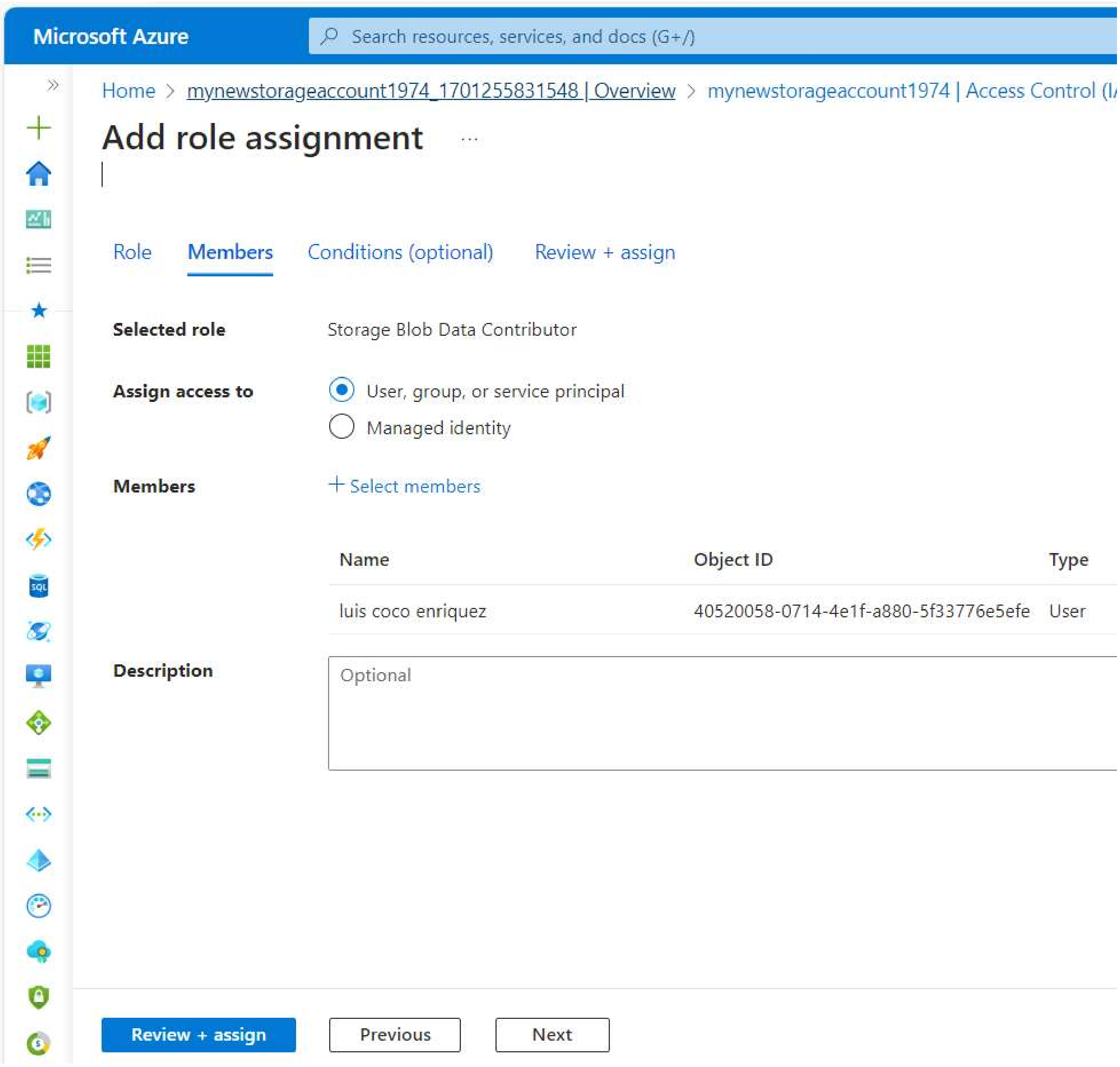
Search by name or email address

No users, groups, or service principals found.

Selected members:

luis coco enriquez
luiscocoenriquez_hotmail.com#EXT#... Remove

Finally, we press the button "**Review + assign**"


Microsoft Azure
Search resources, services, and docs (G+/)
Home > mynewstorageaccount1974_1701255831548 | Overview > mynewstorageaccount1974 | Access Control (I

Add role assignment

Role
Members
Conditions (optional)
Review + assign

Selected role: Storage Blob Data Contributor

Assign access to: [✓ User, group, or service principal] Managed identity

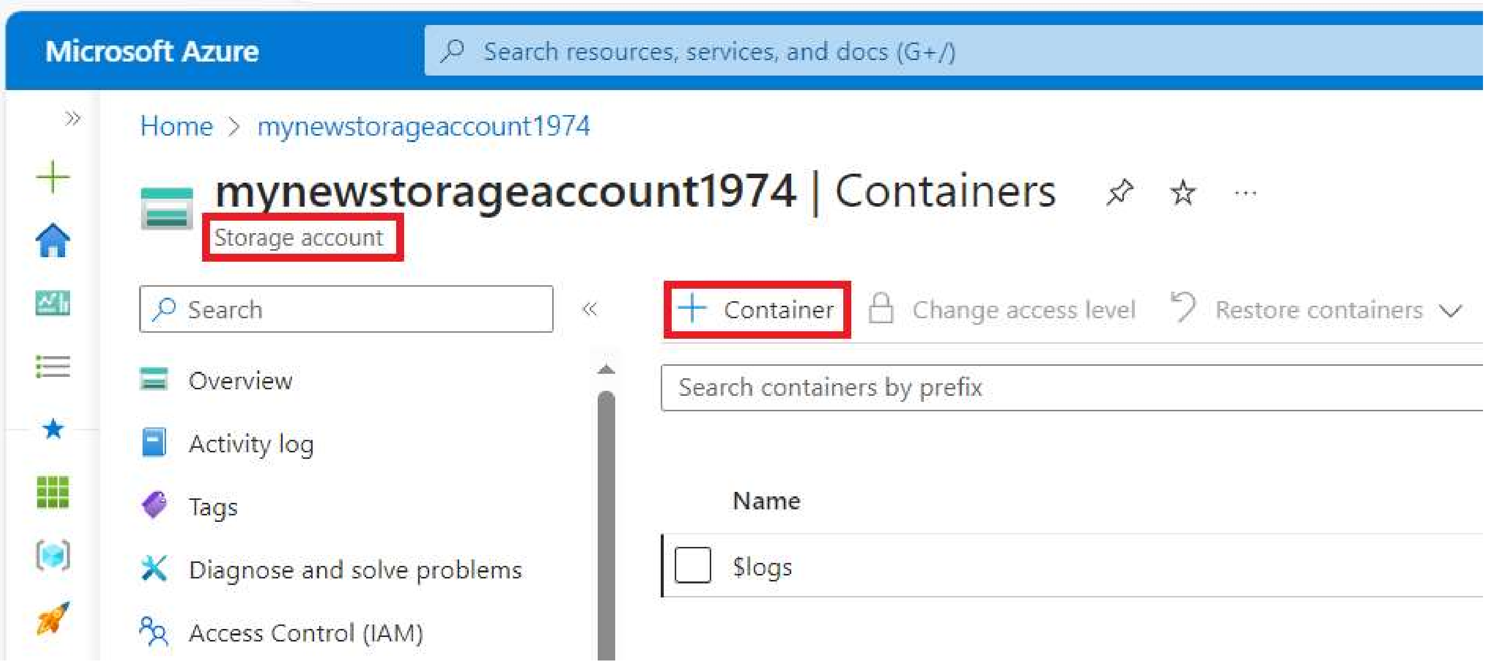
Members: + Select members

Name	Object ID	Type
luis coco enriquez	40520058-0714-4e1f-a880-5f33776e5efe	User

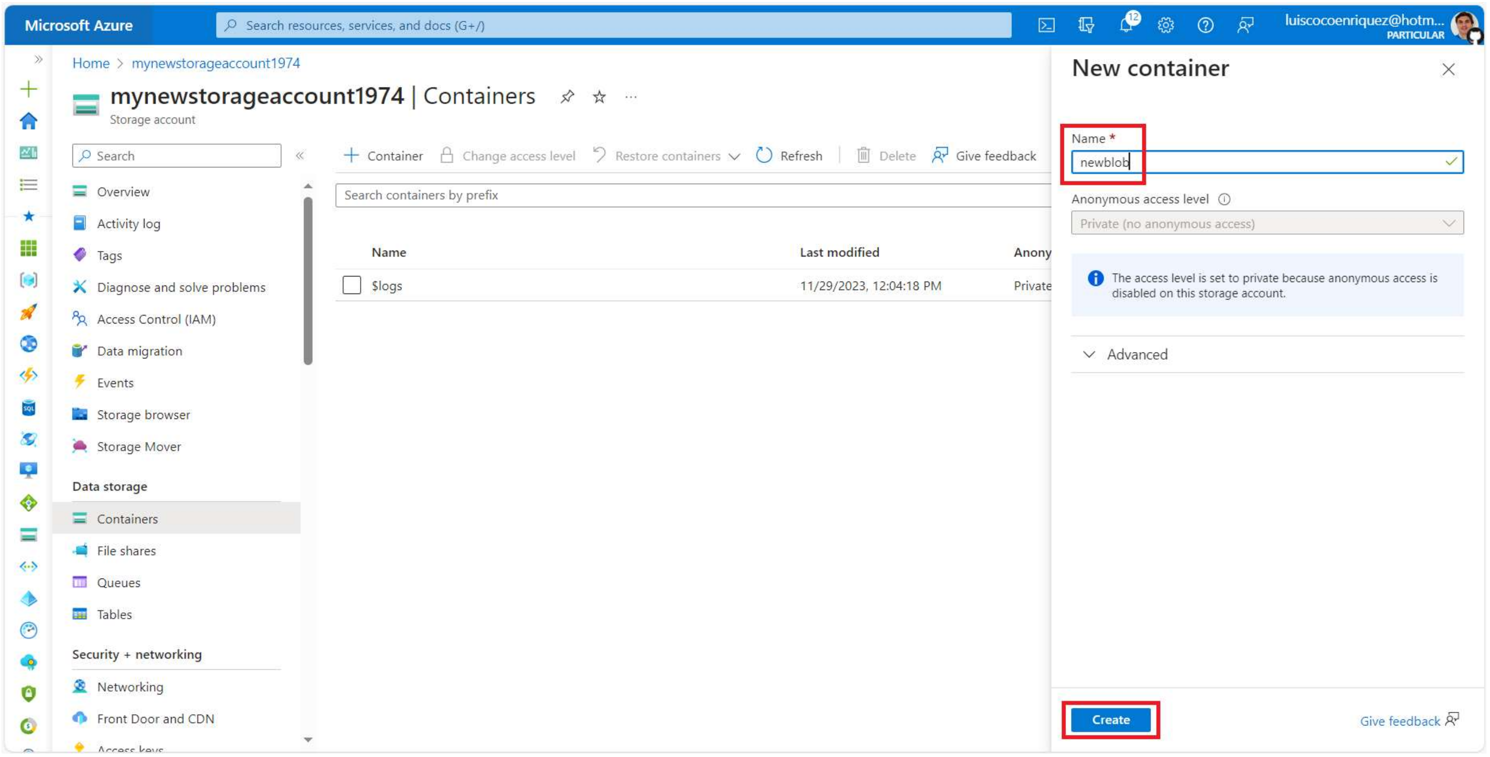
Description: Optional

Review + assign
Previous
Next

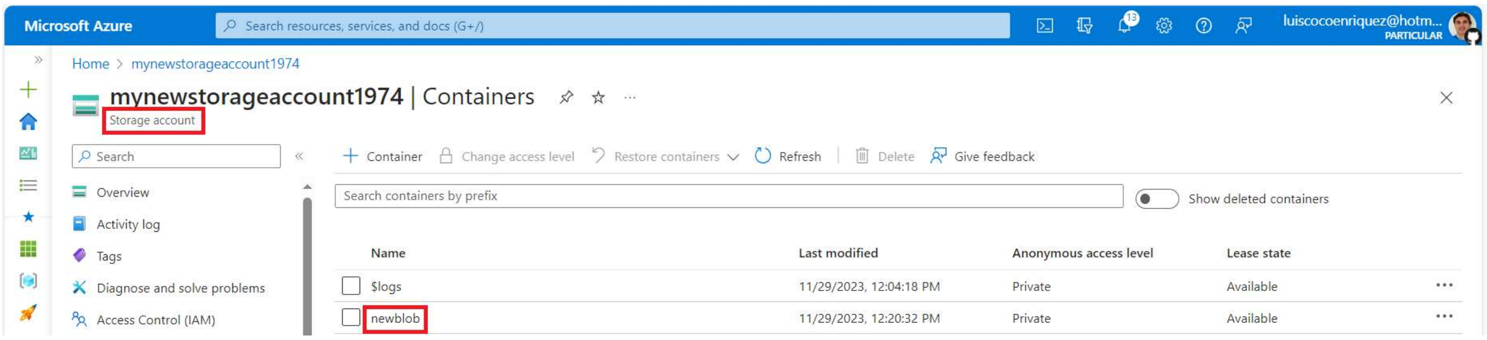
Now we have to create the Blob container:


Microsoft Azure
Search resources, services, and docs (G+/)
Home > mynewstorageaccount1974
mynewstorageaccount1974 | Containers
Storage account
Search
Overview
Activity log
Tags
Diagnose and solve problems
Access Control (IAM)
+ Container
Change access level
Restore containers
Search containers by prefix
Name
$logs

Then we set the Blob container name:


Microsoft Azure
Home > mynewstorageaccount1974
mynewstorageaccount1974 | Containers
Storage account
+ Container
Change access level
Restore containers
Refresh
Delete
Give feedback
New container
Name *
newblob
Anonymous access level
Private (no anonymous access)
The access level is set to private because anonymous access is disabled on this storage account.
Advanced
Create

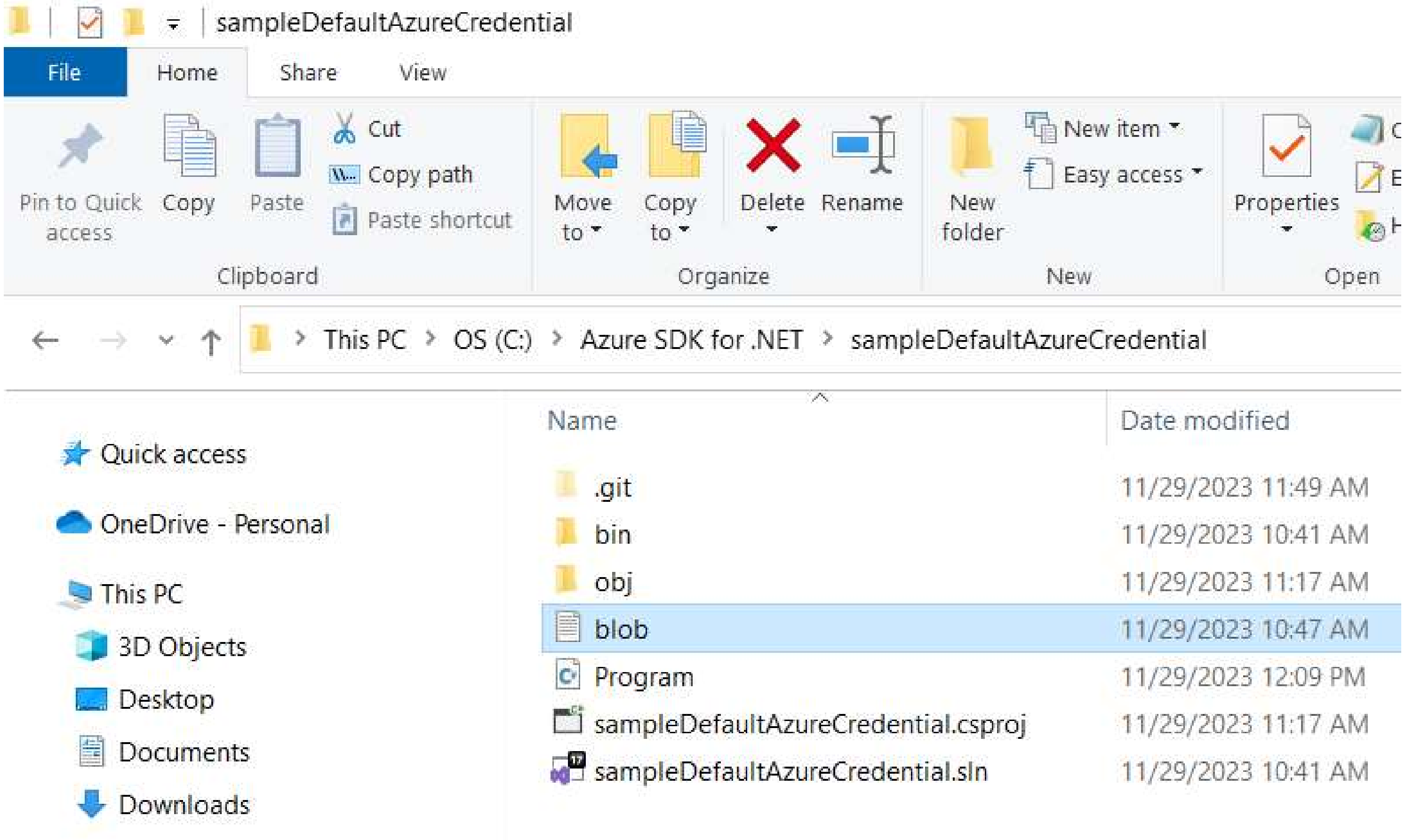
See the new Blob container:



mynewstorageaccount1974 | Containers

Name	Last modified	Anonymous access level	Lease state
$logs	11/29/2023, 12:04:18 PM	Private	Available
newblob	11/29/2023, 12:20:32 PM	Private	Available

We also have to place the "blob.txt" file in the application folder to upload it:



Name	Date modified
.git	11/29/2023 11:49 AM
bin	11/29/2023 10:41 AM
obj	11/29/2023 11:17 AM
blob	11/29/2023 10:47 AM
Program	11/29/2023 12:09 PM
sampleDefaultAzureCredential.csproj	11/29/2023 11:17 AM
sampleDefaultAzureCredential.sln	11/29/2023 10:41 AM

1. Create a new console C# application in VSCode

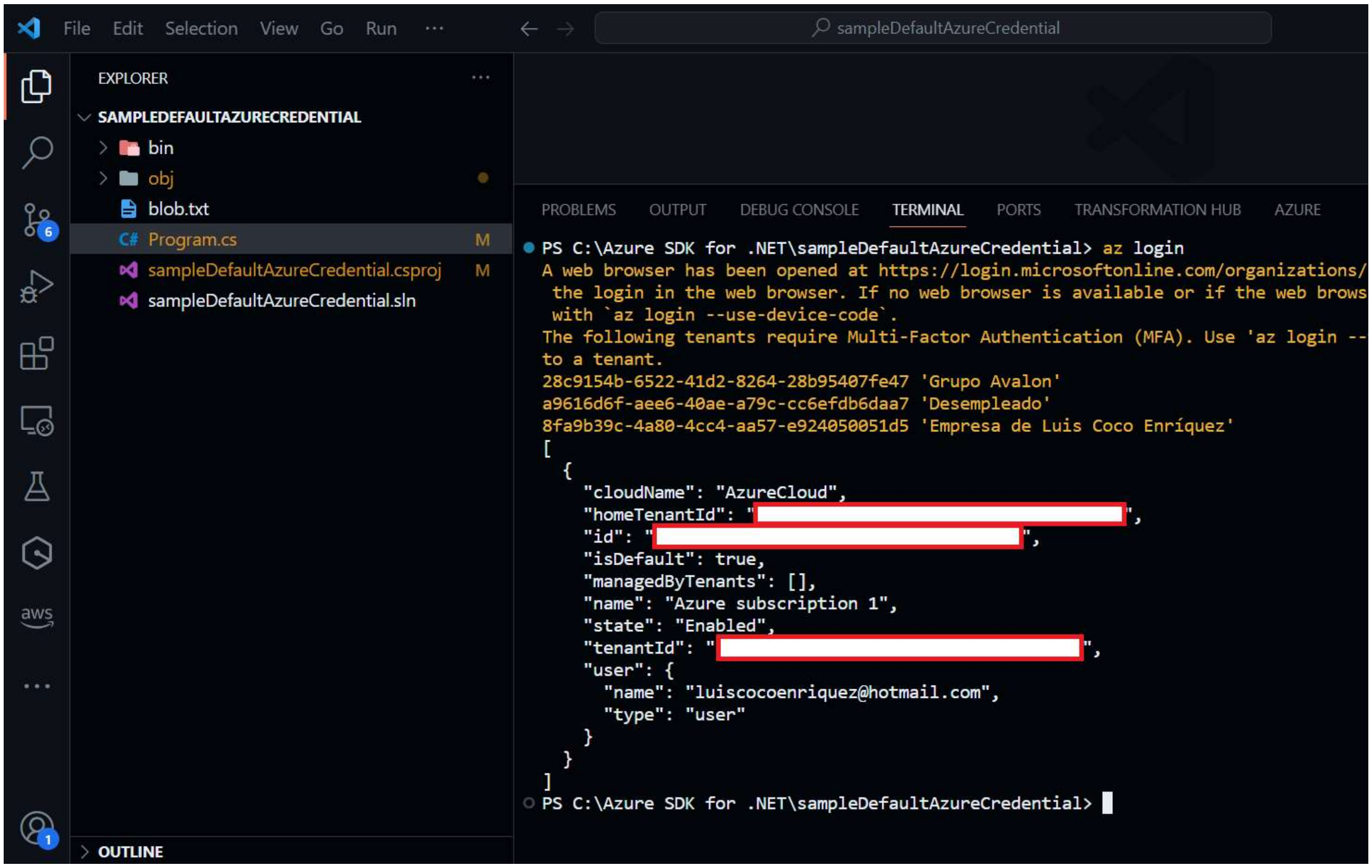
Open VSCode and run this command to create a new C# console application with .NET 8:

```
dotnet new console --framework net8.0
```

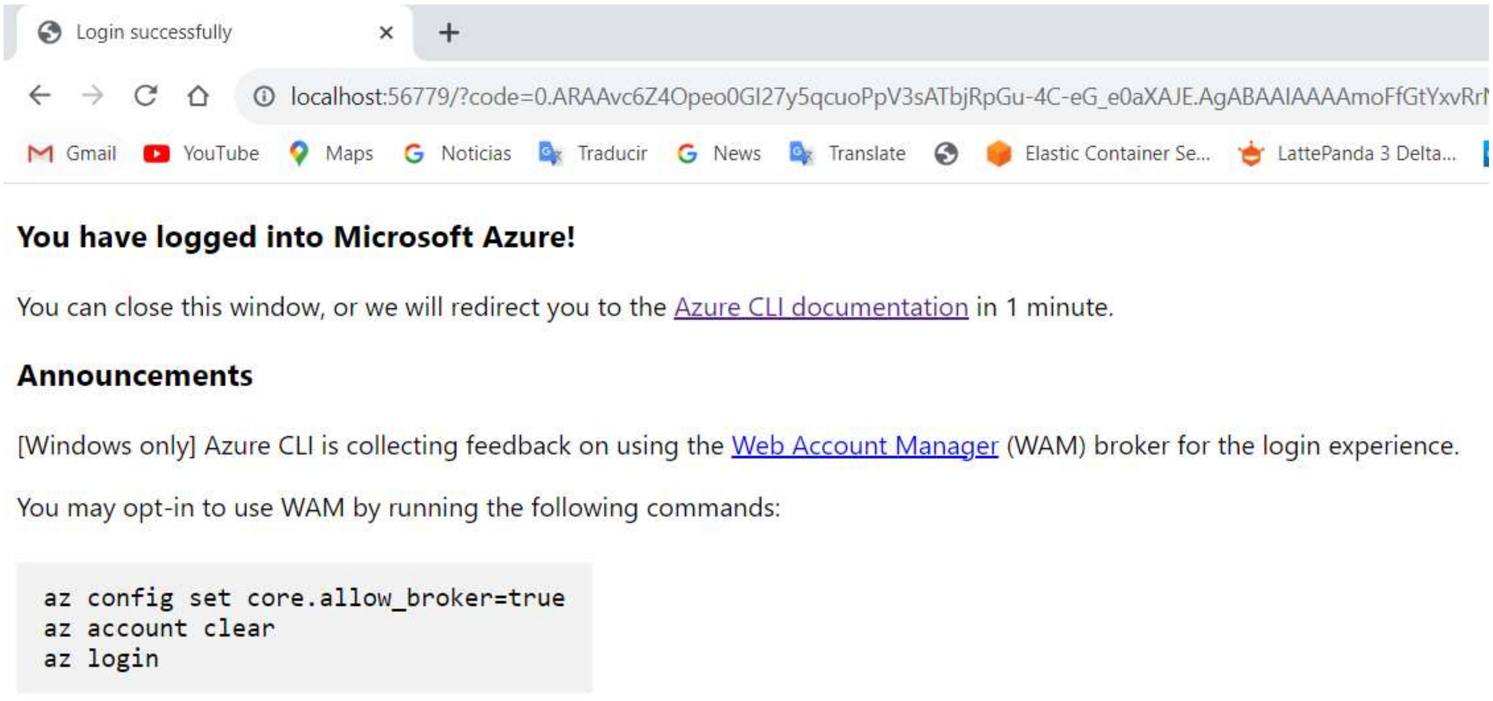
2. Login in Azure with VSCode Terminal window

In the VSCode Terminal window run the command

```
az login
```



```
PS C:\Azure SDK for .NET\sampleDefaultAzureCredential> az login
A web browser has been opened at https://login.microsoftonline.com/organizations/
 the login in the web browser. If no web browser is available or if the web brows
 with `az login --use-device-code`.
The following tenants require Multi-Factor Authentication (MFA). Use 'az login --
to a tenant.
28c9154b-6522-41d2-8264-28b95407fe47 'Grupo Avalon'
a9616d6f-aee6-40ae-a79c-cc6efdb6daa7 'Desempleado'
8fa9b39c-4a80-4cc4-aa57-e924050051d5 'Empresa de Luis Coco Enríquez'
[
  {
    "cloudName": "AzureCloud",
    "homeTenantId": "[redacted]",
    "id": "[redacted]",
    "isDefault": true,
    "managedByTenants": [],
    "name": "Azure subscription 1",
    "state": "Enabled",
    "tenantId": "[redacted]",
    "user": {
      "name": "luiscocoenriquez@hotmail.com",
      "type": "user"
    }
  }
]
PS C:\Azure SDK for .NET\sampleDefaultAzureCredential>
```



You have logged into Microsoft Azure!

You can close this window, or we will redirect you to the Azure CLI documentation in 1 minute.

Announcements

[Windows only] Azure CLI is collecting feedback on using the Web Account Manager (WAM) broker for the login experience.

You may opt-in to use WAM by running the following commands:

```
az config set core.allow_broker=true
az account clear
az login
```

3. Load the libraries

In your internet browser navigate to the Nuget package web page: https://www.nuget.org/packages, and look for the libraries.

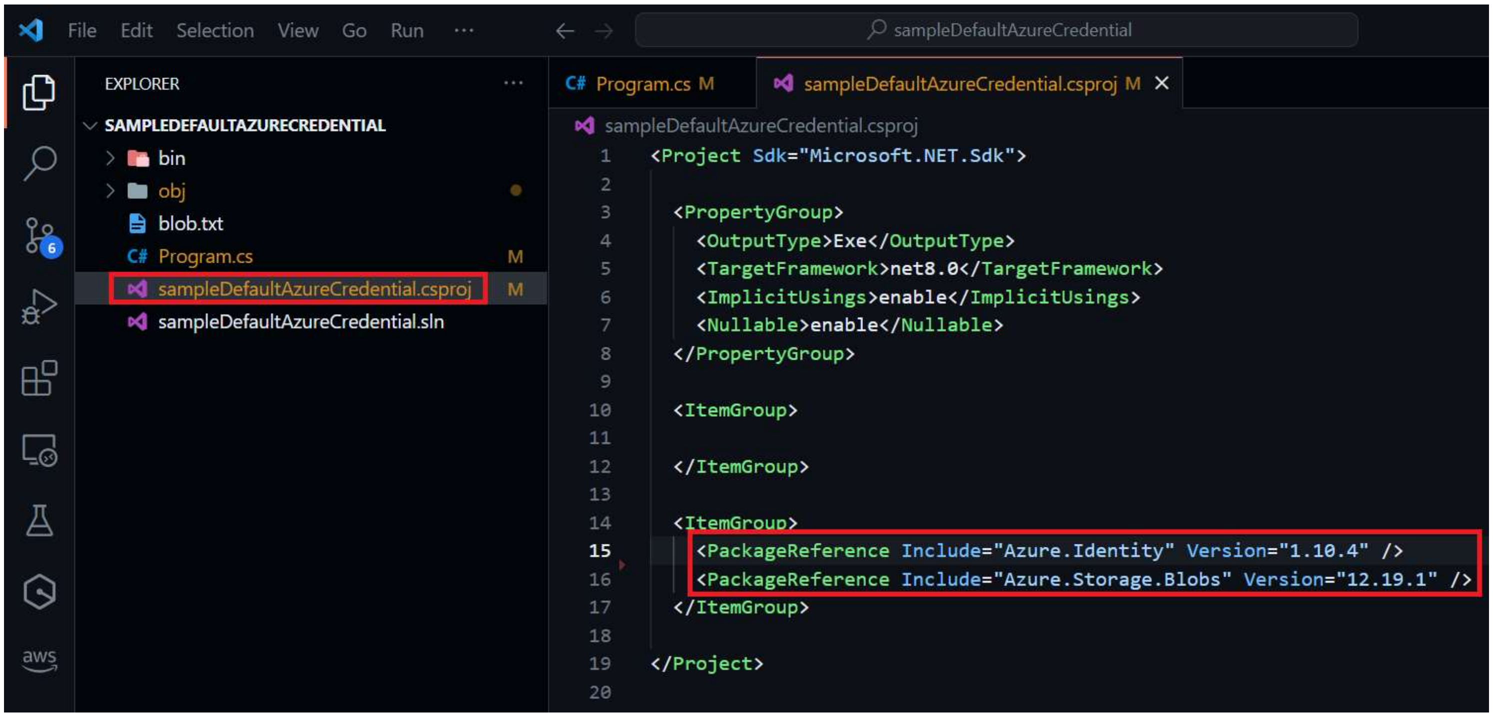
Run these commands to load the "Azure.Identity" and the "Azure.Storage.Blobs" libraries:

```
dotnet add package Azure.Identity --version 1.10.4
dotnet add package Azure.Storage.Blobs --version 12.19.1
```

After then run the command:

```
dotnet restore
```

And confirm in the **csproj** file the libraries are included:


File Edit Selection View Go Run ···
sampleDefaultAzureCredential
EXPLORER
SAMPLEDEFAULTAZURECREDENTIAL
bin
obj
blob.txt
Program.cs M
sampleDefaultAzureCredential.csproj M
sampleDefaultAzureCredential.sln
Program.cs M
sampleDefaultAzureCredential.csproj M

```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>net8.0</TargetFramework>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>enable</Nullable>
  </PropertyGroup>

  <ItemGroup>

  </ItemGroup>

  <ItemGroup>
    <PackageReference Include="Azure.Identity" Version="1.10.4" />
    <PackageReference Include="Azure.Storage.Blobs" Version="12.19.1" />
  </ItemGroup>

</Project>
```

4. Input the application source code

```
using System;
using System.Threading.Tasks;

using Azure.Identity;
using Azure.Storage.Blobs;

//Prerequisite: create an Azure Storage Account and inside create a new Azure Blob container
string storageAccountName = "mynewstorageaccount1999";
string blobContainerName = "newblob";
```

```
var uri = new Uri("https://"+ storageAccountName + ".blob.core.windows.net/" + blobContainerName)
var cred = new DefaultAzureCredential();

// Create a BlobContainerClient
var containerClient = new BlobContainerClient(uri, cred);

// (OPTIONAL) Create the blob container if it doesn't exist
await containerClient.CreateIfNotExistsAsync();

// Create a BlobClient and set the blob name
var blobClient = containerClient.GetBlobClient("blob.txt");

// Upload the file into the blob
await blobClient.UploadAsync("blob.txt");

Console.WriteLine("File uploaded to Blob conatiner successfully!");
```

5. Build and run the application

In the Terminal Window in VSCode type the command:

```
dotnet run
```