KAZAKH KBTU BRITISH
TECHNICAL
UNIVERSITY

Cloud Development, 2021 fall.
Lab3

Report

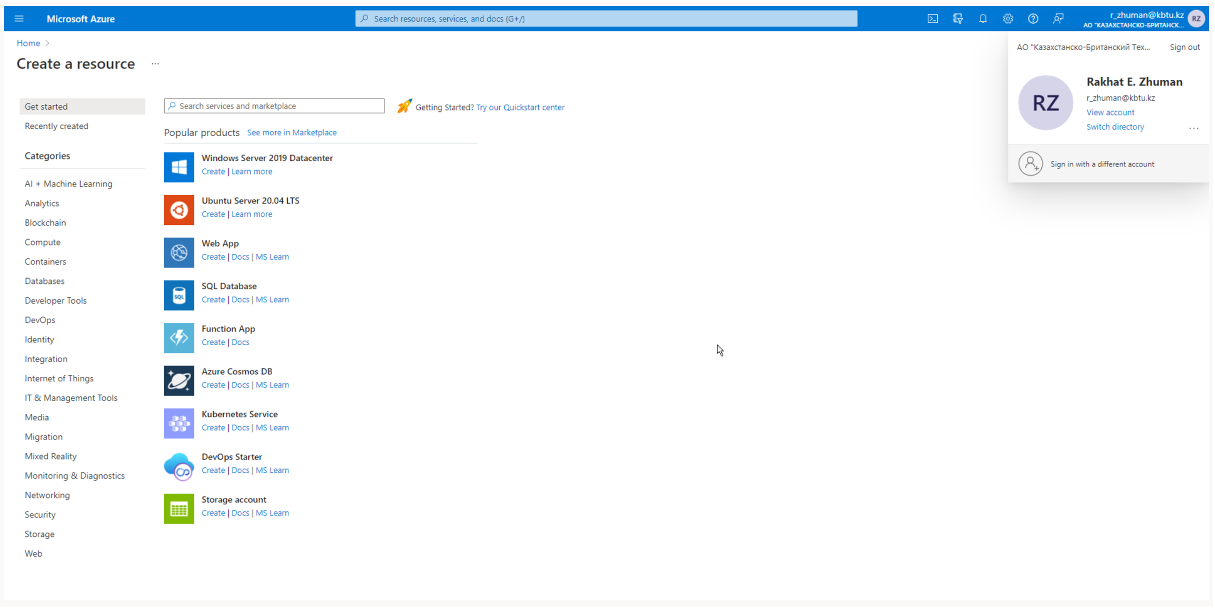
Retrieving Azure Storage resources and metadata by using the Azure Storage SDK for .NET

made by Zhuman Rakhat

Almaty 2021

Exercise 1: Create Azure resources

Task 1: Open the Azure portal


Microsoft Azure
Search resources, services, and docs (G+/)
r_zhuman@kbtu.kz
AO "КАЗАХСТАНСКО-БРИТАНСК...
Home
АО "Казахстанско-Британский Тех...
Sign out
RZ
Rakhat E. Zhuman
r_zhuman@kbtu.kz
View account
Switch directory
Sign in with a different account

Create a resource

Get started
Recently created

Categories

AI + Machine Learning
Analytics
Blockchain
Compute
Containers
Databases
Developer Tools
DevOps
Identity
Integration
Internet of Things
IT & Management Tools
Media
Migration
Mixed Reality
Monitoring & Diagnostics
Networking
Security
Storage
Web

Search services and marketplace
Getting Started? Try our Quickstart center

Popular products See more in Marketplace

Windows Server 2019 Datacenter
Create | Learn more

Ubuntu Server 20.04 LTS
Create | Learn more

Web App
Create | Docs | MS Learn

SQL Database
Create | Docs | MS Learn

Function App
Create | Docs

Azure Cosmos DB
Create | Docs | MS Learn

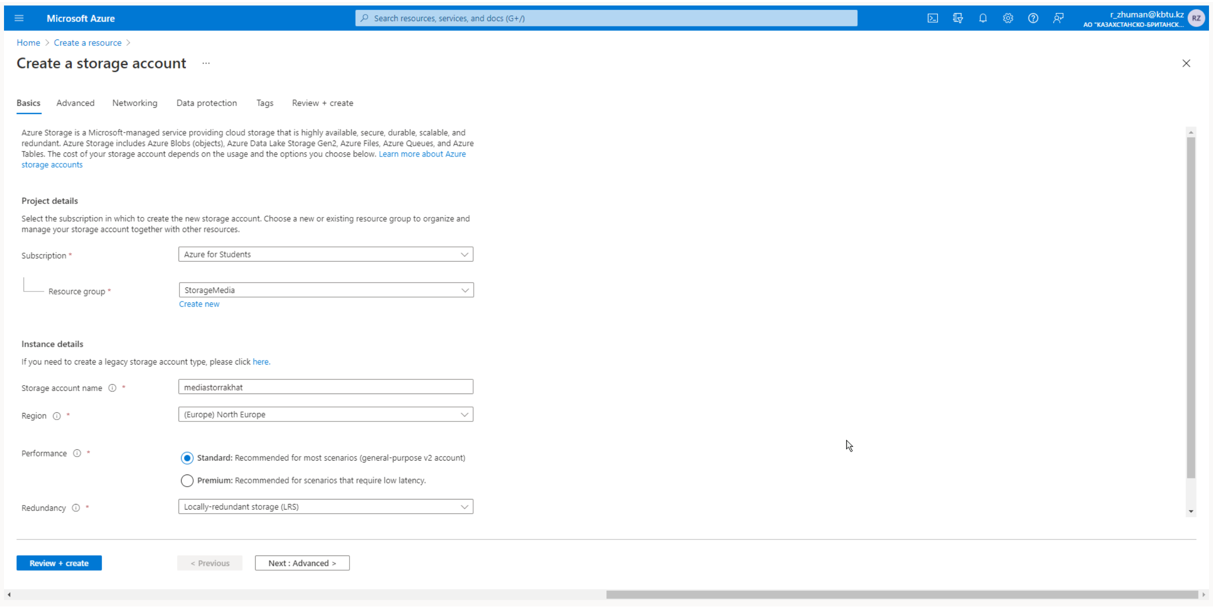
Kubernetes Service
Create | Docs | MS Learn

DevOps Starter
Create | Docs | MS Learn

Storage account
Create | Docs | MS Learn

Task 2: Create an Azure Storage account

Create a new storage account with the following details:



Create a storage account

Azure Storage is a Microsoft-managed service providing cloud storage that is highly available, secure, durable, scalable, and redundant. Azure Storage includes Azure Blobs (objects), Azure Data Lake Storage Gen2, Azure Files, Azure Queues, and Azure Tables. The cost of your storage account depends on the usage and the options you choose below. Learn more about Azure storage accounts

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 * : Azure for Students

Resource group * : StorageMedia
Create new

Instance details

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

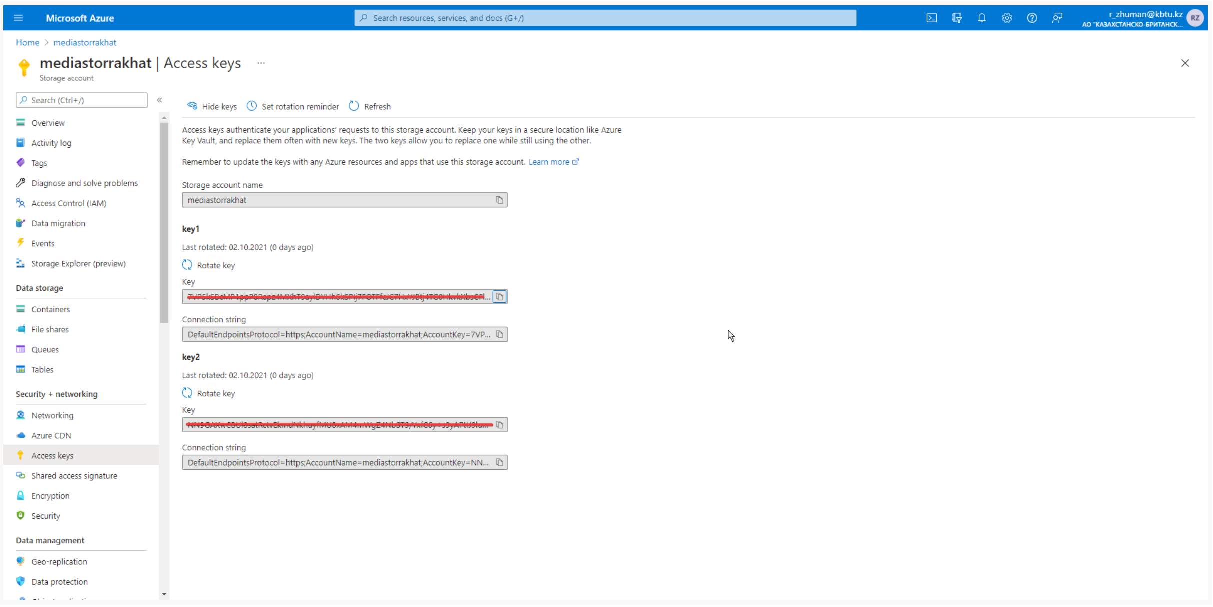
Storage account name * : mediastorrakhat

Region * : (Europe) North Europe

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

Redundancy * : Locally-redundant storage (LRS)

Record the value in the Storage account name text box and any of the Key text boxes. You'll use these values later in this lab.


Microsoft Azure
Search resources, services, and docs (G+/)
r_zhuman@kbtu.kz
АО "КАЗАХСТАНСКО-БРИТАНСК...
Home > mediastorrakhat

mediastorrakhat | Access keys

Storage account

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: mediastorrakhat

key1

Last rotated: 02.10.2021 (0 days ago)

Rotate key

Key: [redacted]

Connection string: DefaultEndpointsProtocol=https;AccountName=mediastorrakhat;AccountKey=7VP...

key2

Last rotated: 02.10.2021 (0 days ago)

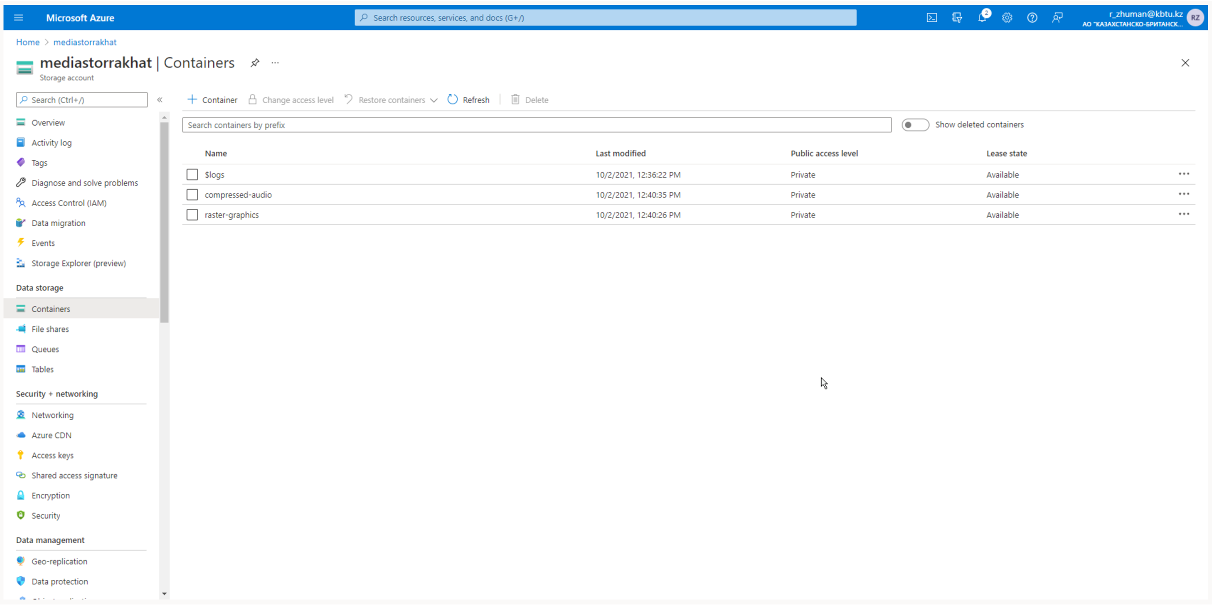
Rotate key

Key: [redacted]

Connection string: DefaultEndpointsProtocol=https;AccountName=mediastorrakhat;AccountKey=NN...

Exercise 2: Upload a blob into a container

Task 1: Create storage account containers

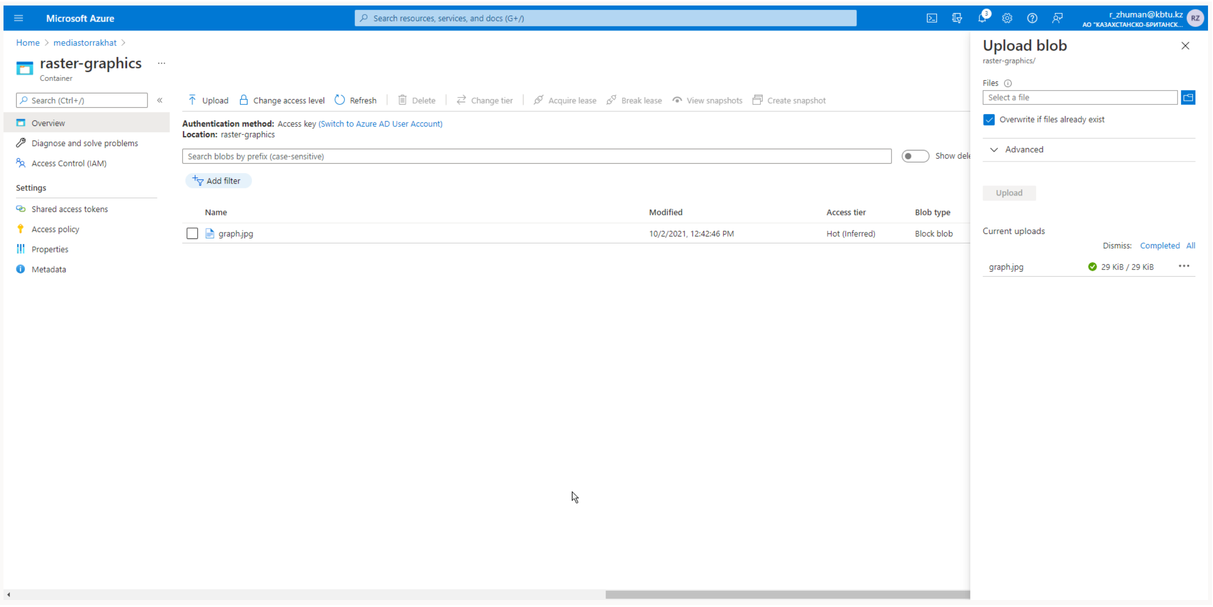

Microsoft Azure
Search resources, services, and docs (G+/)
r_zhuman@kbtu.kz
АО "КАЗАХСТАНСКО-БРИТАНСК...
Home > mediastorrakhat

mediastorrakhat | Containers

Storage account

Name	Last modified	Public access level	Lease state
$logs	10/2/2021, 12:36:22 PM	Private	Available
compressed-audio	10/2/2021, 12:40:35 PM	Private	Available
raster-graphics	10/2/2021, 12:40:26 PM	Private	Available

Task 2: Upload a storage account blob



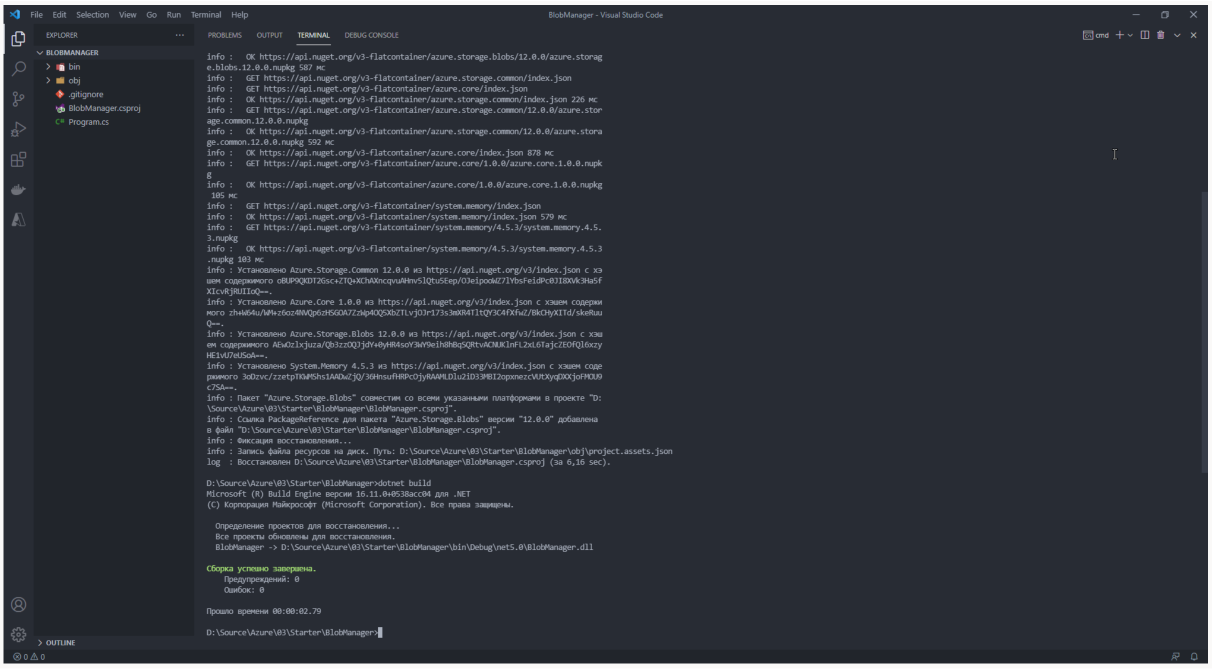
Authentication method: Access key (Switch to Azure AD User Account)
Location: raster-graphics

Name	Modified	Access tier	Blob type
graph.jpg	10/2/2021, 12:42:46 PM	Hot (Inferred)	Block blob

graph.jpg ✓ 29 KiB / 29 KiB

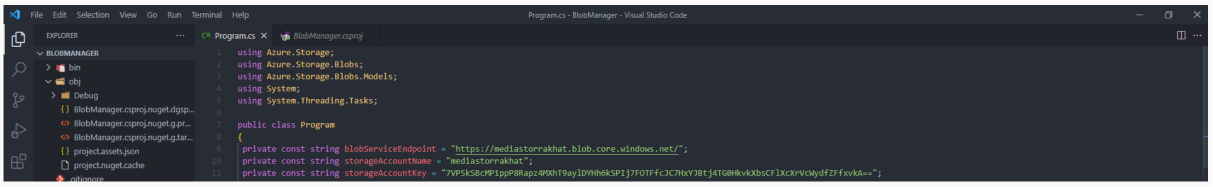
Exercise 3: Access containers by using the .NET SDK

Task 1: Create .NET project



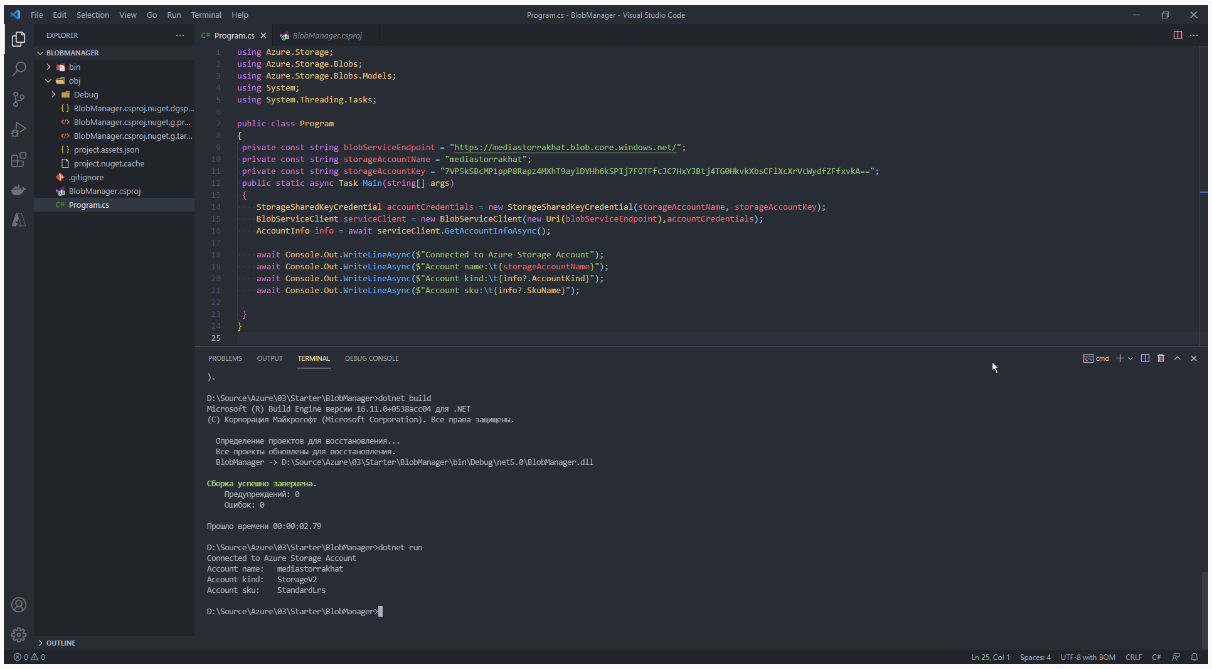
```
info :   OK https://api.nuget.org/v3-flatcontainer/azure.storage.blobs/12.0.0/azure.storag
e.blobs.12.0.0.nupkg 587 мс
info :   GET https://api.nuget.org/v3-flatcontainer/azure.storage.common/index.json
info :   GET https://api.nuget.org/v3-flatcontainer/azure.core/index.json
info :   OK https://api.nuget.org/v3-flatcontainer/azure.storage.common/index.json 226 мс
info :   GET https://api.nuget.org/v3-flatcontainer/azure.storage.common/12.0.0/azure.stor
age.common.12.0.0.nupkg
info :   OK https://api.nuget.org/v3-flatcontainer/azure.storage.common/12.0.0/azure.stora
ge.common.12.0.0.nupkg 592 мс
info :   OK https://api.nuget.org/v3-flatcontainer/azure.core/index.json 878 мс
info :   GET https://api.nuget.org/v3-flatcontainer/azure.core/1.0.0/azure.core.1.0.0.nupk
g
info :   OK https://api.nuget.org/v3-flatcontainer/azure.core/1.0.0/azure.core.1.0.0.nupkg
 105 мс
info :   GET https://api.nuget.org/v3-flatcontainer/system.memory/index.json
info :   OK https://api.nuget.org/v3-flatcontainer/system.memory/index.json 579 мс
info :   GET https://api.nuget.org/v3-flatcontainer/system.memory/4.5.3/system.memory.4.5.
3.nupkg
info :   OK https://api.nuget.org/v3-flatcontainer/system.memory/4.5.3/system.memory.4.5.3
.nupkg 103 мс
info : Установлено Azure.Storage.Common 12.0.0 из https://api.nuget.org/v3/index.json с хэ
шем содержимого oBUP9QKDT2Gsc+ZTQ+XChAXncqvuAHnv5lQtu5Eep/OJeipooWZ7lYbsFeidPc0JI8XVk3Ha5f
XIcvRjRUIIoQ==.
info : Установлено Azure.Core 1.0.0 из https://api.nuget.org/v3/index.json с хэшем содержи
мого zh+W64u/WM+z6oz4NVQp6zHSGOA7ZzWp4OQ5XbZTLvjOJr173s3mXR4TltQY3C4fXfwZ/BkCHyXITd/skeRuu
Q==.
info : Установлено Azure.Storage.Blobs 12.0.0 из https://api.nuget.org/v3/index.json с хэш
ем содержимого AEwOzlxjuza/Qb3zzOQJjdY+0yHR4soY3WY9eih8hBqSQRtvACNUKlnFL2xL6TajcZEOfQl6xzy
HE1vU7eUSoA==.
info : Установлено System.Memory 4.5.3 из https://api.nuget.org/v3/index.json с хэшем соде
ржимого 3oDzvc/zzetpTKWMShs1AADwZjQ/36HnsufHRPcOjyRAAMLDlu2iD33MBI2opxnezcVUtXyqDXXjoFMOU9
c7SA==.
info : Пакет "Azure.Storage.Blobs" совместим со всеми указанными платформами в проекте "D:
\Source\Azure\03\Starter\BlobManager\BlobManager.csproj".
info : Ссылка PackageReference для пакета "Azure.Storage.Blobs" версии "12.0.0" добавлена
в файл "D:\Source\Azure\03\Starter\BlobManager\BlobManager.csproj".
info : Фиксация восстановления...
info : Запись файла ресурсов на диск. Путь: D:\Source\Azure\03\Starter\BlobManager\obj\project.assets.json
log  : Восстановлен D:\Source\Azure\03\Starter\BlobManager\BlobManager.csproj (за 6,16 sec).

D:\Source\Azure\03\Starter\BlobManager>dotnet build
Microsoft (R) Build Engine версии 16.11.0+0538acc04 для .NET
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

  Определение проектов для восстановления...
  Все проекты обновлены для восстановления.
  BlobManager -> D:\Source\Azure\03\Starter\BlobManager\bin\Debug\net5.0\BlobManager.dll

Сборка успешно завершена.
    Предупреждений: 0
    Ошибок: 0

Прошло времени 00:00:02.79

D:\Source\Azure\03\Starter\BlobManager>
```

Task 2: Modify the Program class to access Storage


Program.cs - BlobManager - Visual Studio Code
EXPLORER
BLOBMANAGER
bin
obj
Debug
BlobManager.csproj.nuget.dgsp...
BlobManager.csproj.nuget.g.pr...
BlobManager.csproj.nuget.g.tar...
project.assets.json
project.nuget.cache
Program.cs
BlobManager.csproj

```
using Azure.Storage;
using Azure.Storage.Blobs;
using Azure.Storage.Blobs.Models;
using System;
using System.Threading.Tasks;

public class Program
{
 private const string blobServiceEndpoint = "https://mediastorrakhat.blob.core.windows.net/";
 private const string storageAccountName = "mediastorrakhat";
 private const string storageAccountKey = "7VP5kSBcMP1ppP8Rapz4MXhT9aylDYHh6kSPIj7FOTFfcJC7HxYJBtj4TG0HkvkXbsCFlXcXrVcWydfZFfxvkA==";
```

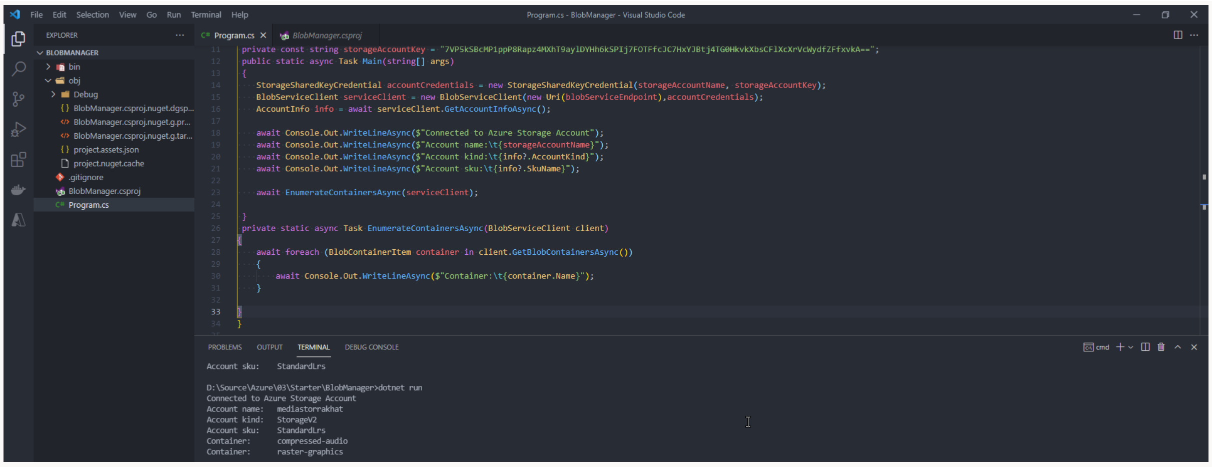
Task 3: Connect to the Azure Storage blob service endpoint



```csharp
using Azure.Storage;
using Azure.Storage.Blobs;
using Azure.Storage.Blobs.Models;
using System;
using System.Threading.Tasks;

public class Program
{
 private const string blobServiceEndpoint = "https://mediastorrakhat.blob.core.windows.net/";
 private const string storageAccountName = "mediastorrakhat";
 private const string storageAccountKey = "7VP5kSBcMP1ppP8Rapz4MXhT9aylDYHh6kSPIj7FOTFfcJC7HxYJBtj4TG0HkvkXbsCFlXcXrVcWydfZFfxvkA==";
 public static async Task Main(string[] args)
 {
    StorageSharedKeyCredential accountCredentials = new StorageSharedKeyCredential(storageAccountName, storageAccountKey);
    BlobServiceClient serviceClient = new BlobServiceClient(new Uri(blobServiceEndpoint),accountCredentials);
    AccountInfo info = await serviceClient.GetAccountInfoAsync();

    await Console.Out.WriteLineAsync($"Connected to Azure Storage Account");
    await Console.Out.WriteLineAsync($"Account name:\t{storageAccountName}");
    await Console.Out.WriteLineAsync($"Account kind:\t{info?.AccountKind}");
    await Console.Out.WriteLineAsync($"Account sku:\t{info?.SkuName}");

 }
}
```

```
).

D:\Source\Azure\03\Starter\BlobManager>dotnet build
Microsoft (R) Build Engine версии 16.11.0+0538acc04 для .NET
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

  Определение проектов для восстановления...
  Все проекты обновлены для восстановления.
  BlobManager -> D:\Source\Azure\03\Starter\BlobManager\bin\Debug\net5.0\BlobManager.dll

Сборка успешно завершена.
    Предупреждений: 0
    Ошибок: 0

Прошло времени 00:00:02.79

D:\Source\Azure\03\Starter\BlobManager>dotnet run
Connected to Azure Storage Account
Account name:   mediastorrakhat
Account kind:   StorageV2
Account sku:    StandardLrs

D:\Source\Azure\03\Starter\BlobManager>
```

Task 4: Enumerate the existing containers

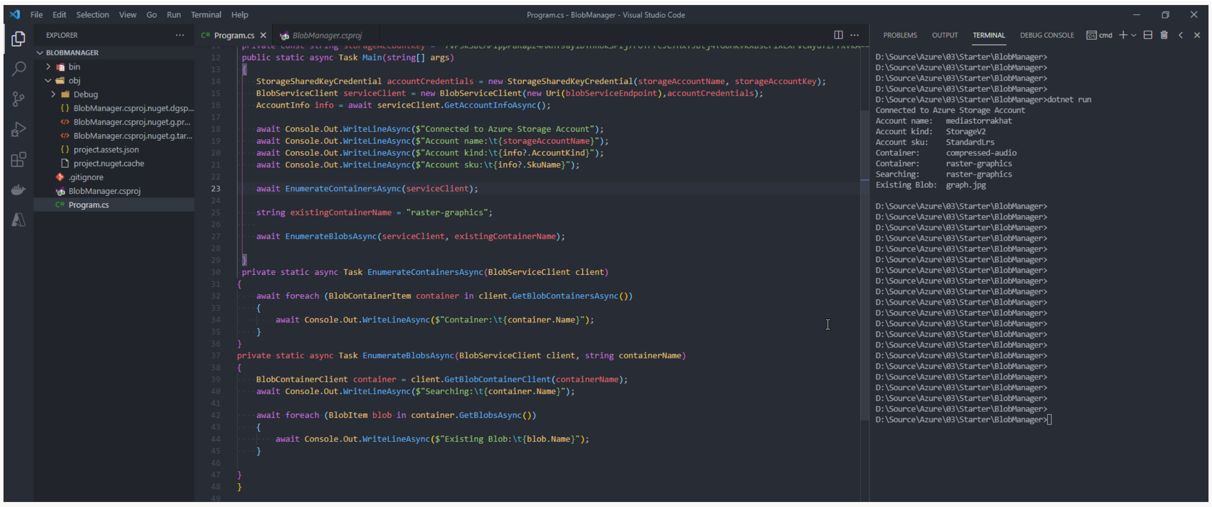


```csharp
 private const string storageAccountKey = "7VP5kSBcMP1ppP8Rapz4MXhT9aylDYHh6kSPIj7FOTFfcJC7HxYJBtj4TG0HkvkXbsCFlXcXrVcWydfZFfxvkA==";
 public static async Task Main(string[] args)
 {
    StorageSharedKeyCredential accountCredentials = new StorageSharedKeyCredential(storageAccountName, storageAccountKey);
    BlobServiceClient serviceClient = new BlobServiceClient(new Uri(blobServiceEndpoint),accountCredentials);
    AccountInfo info = await serviceClient.GetAccountInfoAsync();

    await Console.Out.WriteLineAsync($"Connected to Azure Storage Account");
    await Console.Out.WriteLineAsync($"Account name:\t{storageAccountName}");
    await Console.Out.WriteLineAsync($"Account kind:\t{info?.AccountKind}");
    await Console.Out.WriteLineAsync($"Account sku:\t{info?.SkuName}");

    await EnumerateContainersAsync(serviceClient);

 }
 private static async Task EnumerateContainersAsync(BlobServiceClient client)
{
    await foreach (BlobContainerItem container in client.GetBlobContainersAsync())
    {
        await Console.Out.WriteLineAsync($"Container:\t{container.Name}");
    }

}
}
```

```
Account sku:    StandardLrs

D:\Source\Azure\03\Starter\BlobManager>dotnet run
Connected to Azure Storage Account
Account name:   mediastorrakhat
Account kind:   StorageV2
Account sku:    StandardLrs
Container:      compressed-audio
Container:      raster-graphics
```

Exercise 4: Retrieve blob Uniform Resource Identifiers (URIs) by using the .NET SDK

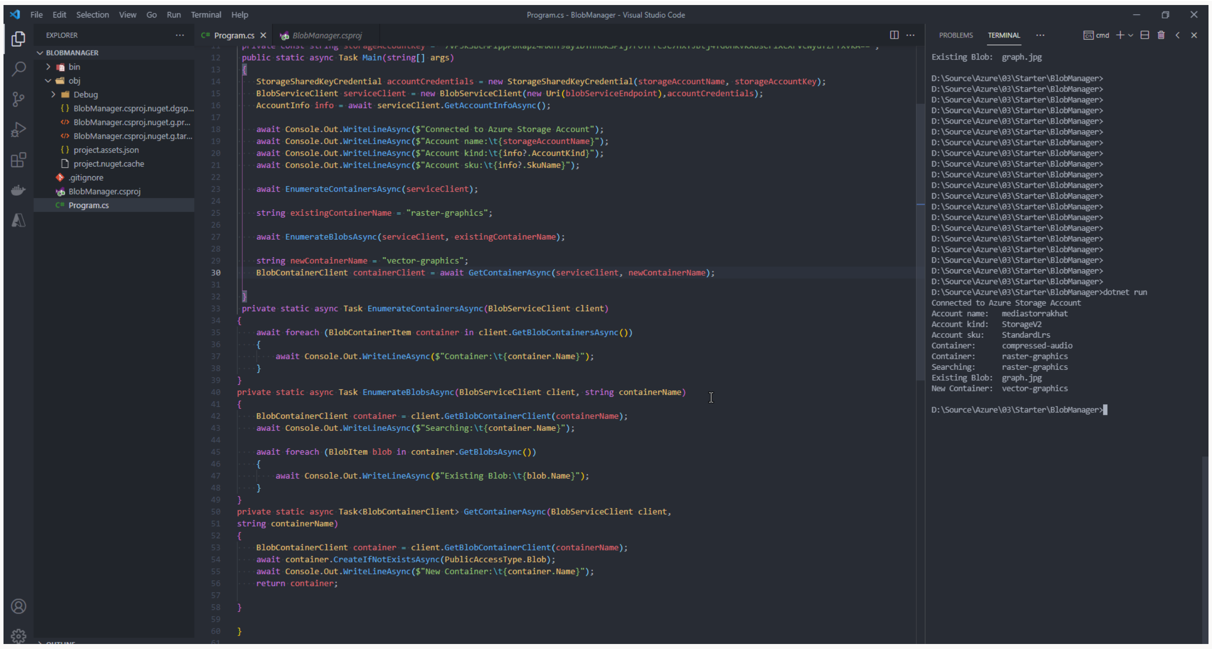
Task 1: Enumerate the blobs in an existing container by using the SDK



```csharp
 public static async Task Main(string[] args)
 {
    StorageSharedKeyCredential accountCredentials = new StorageSharedKeyCredential(storageAccountName, storageAccountKey);
    BlobServiceClient serviceClient = new BlobServiceClient(new Uri(blobServiceEndpoint),accountCredentials);
    AccountInfo info = await serviceClient.GetAccountInfoAsync();

    await Console.Out.WriteLineAsync($"Connected to Azure Storage Account");
    await Console.Out.WriteLineAsync($"Account name:\t{storageAccountName}");
    await Console.Out.WriteLineAsync($"Account kind:\t{info?.AccountKind}");
    await Console.Out.WriteLineAsync($"Account sku:\t{info?.SkuName}");

    await EnumerateContainersAsync(serviceClient);

    string existingContainerName = "raster-graphics";

    await EnumerateBlobsAsync(serviceClient, existingContainerName);

 }
 private static async Task EnumerateContainersAsync(BlobServiceClient client)
{
    await foreach (BlobContainerItem container in client.GetBlobContainersAsync())
    {
        await Console.Out.WriteLineAsync($"Container:\t{container.Name}");
    }
}
private static async Task EnumerateBlobsAsync(BlobServiceClient client, string containerName)
{
    BlobContainerClient container = client.GetBlobContainerClient(containerName);
    await Console.Out.WriteLineAsync($"Searching:\t{container.Name}");

    await foreach (BlobItem blob in container.GetBlobsAsync())
    {
        await Console.Out.WriteLineAsync($"Existing Blob:\t{blob.Name}");
    }

}
}
```

```
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>dotnet run
Connected to Azure Storage Account
Account name:   mediastorrakhat
Account kind:   StorageV2
Account sku:    StandardLrs
Container:      compressed-audio
Container:      raster-graphics
Searching:      raster-graphics
Existing Blob:  graph.jpg

D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
```

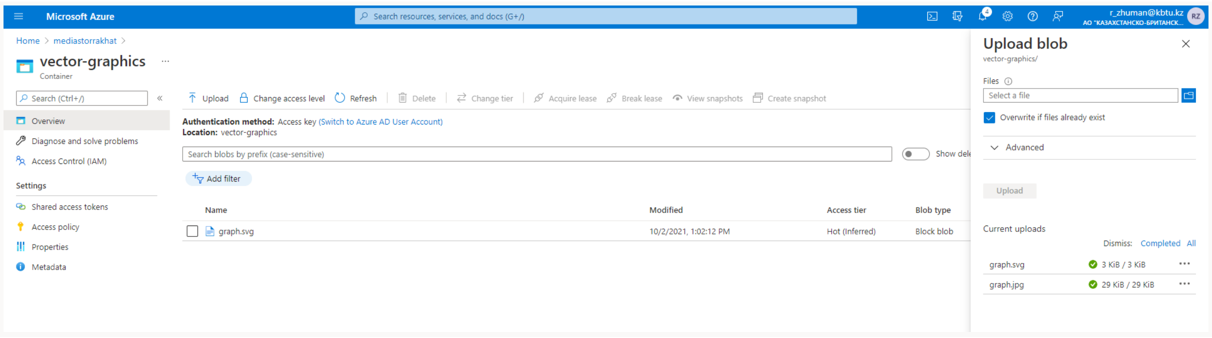
Task 2: Create a new container by using the SDK



```csharp
public static async Task Main(string[] args)
{
    StorageSharedKeyCredential accountCredentials = new StorageSharedKeyCredential(storageAccountName, storageAccountKey);
    BlobServiceClient serviceClient = new BlobServiceClient(new Uri(blobServiceEndpoint),accountCredentials);
    AccountInfo info = await serviceClient.GetAccountInfoAsync();

    await Console.Out.WriteLineAsync($"Connected to Azure Storage Account");
    await Console.Out.WriteLineAsync($"Account name:\t{storageAccountName}");
    await Console.Out.WriteLineAsync($"Account kind:\t{info?.AccountKind}");
    await Console.Out.WriteLineAsync($"Account sku:\t{info?.SkuName}");

    await EnumerateContainersAsync(serviceClient);

    string existingContainerName = "raster-graphics";

    await EnumerateBlobsAsync(serviceClient, existingContainerName);

    string newContainerName = "vector-graphics";
    BlobContainerClient containerClient = await GetContainerAsync(serviceClient, newContainerName);

}
private static async Task EnumerateContainersAsync(BlobServiceClient client)
{
    await foreach (BlobContainerItem container in client.GetBlobContainersAsync())
    {
        await Console.Out.WriteLineAsync($"Container:\t{container.Name}");
    }
}
private static async Task EnumerateBlobsAsync(BlobServiceClient client, string containerName)
{
    BlobContainerClient container = client.GetBlobContainerClient(containerName);
    await Console.Out.WriteLineAsync($"Searching:\t{container.Name}");

    await foreach (BlobItem blob in container.GetBlobsAsync())
    {
        await Console.Out.WriteLineAsync($"Existing Blob:\t{blob.Name}");
    }
}
private static async Task<BlobContainerClient> GetContainerAsync(BlobServiceClient client,
string containerName)
{
    BlobContainerClient container = client.GetBlobContainerClient(containerName);
    await container.CreateIfNotExistsAsync(PublicAccessType.Blob);
    await Console.Out.WriteLineAsync($"New Container:\t{container.Name}");
    return container;

}

}
```

```
Existing Blob:  graph.jpg

D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>dotnet run
Connected to Azure Storage Account
Account name:   mediastorrakhat
Account kind:   StorageV2
Account sku:    StandardLrs
Container:      compressed-audio
Container:      raster-graphics
Searching:      raster-graphics
Existing Blob:  graph.jpg
New Container:  vector-graphics

D:\Source\Azure\03\Starter\BlobManager>
```

Task 3: Upload a new blob by using the portal



Home > mediastorrakhat >

vector-graphics
Container

Authentication method: Access key (Switch to Azure AD User Account)
Location: vector-graphics

Name	Modified	Access tier	Blob type
graph.svg	10/2/2021, 1:02:12 PM	Hot (Inferred)	Block blob

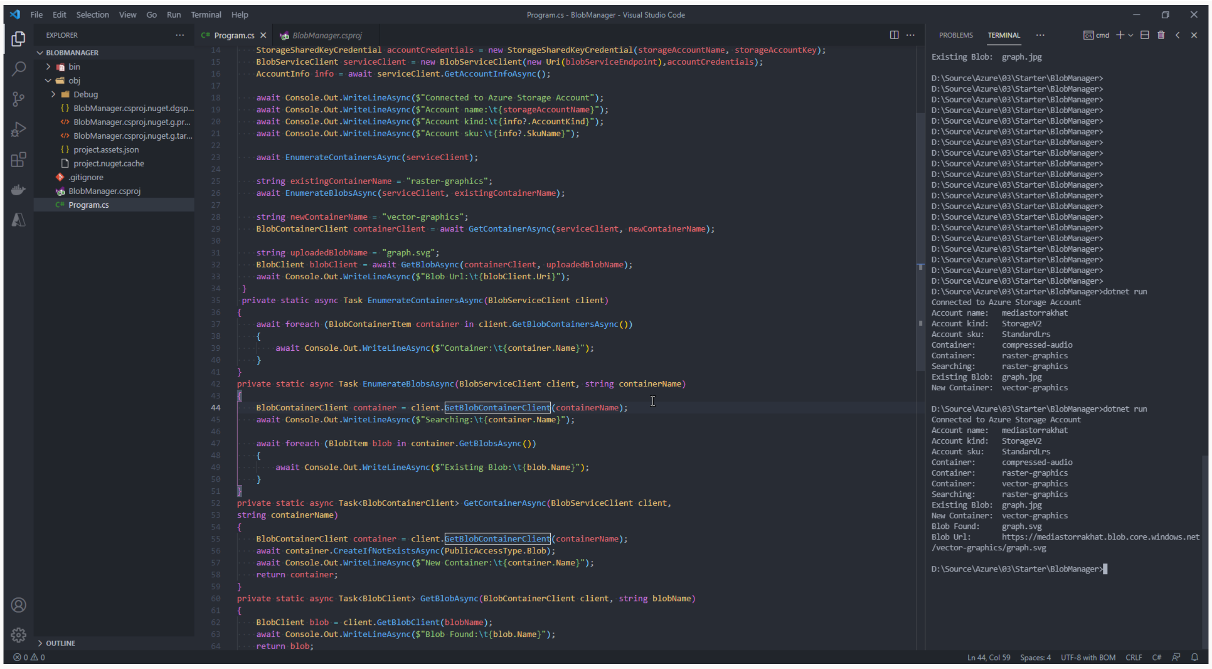
Upload blob
vector-graphics/

Files: Select a file
[✓] Overwrite if files already exist

Current uploads

graph.svg	3 KiB / 3 KiB
graph.jpg	29 KiB / 29 KiB

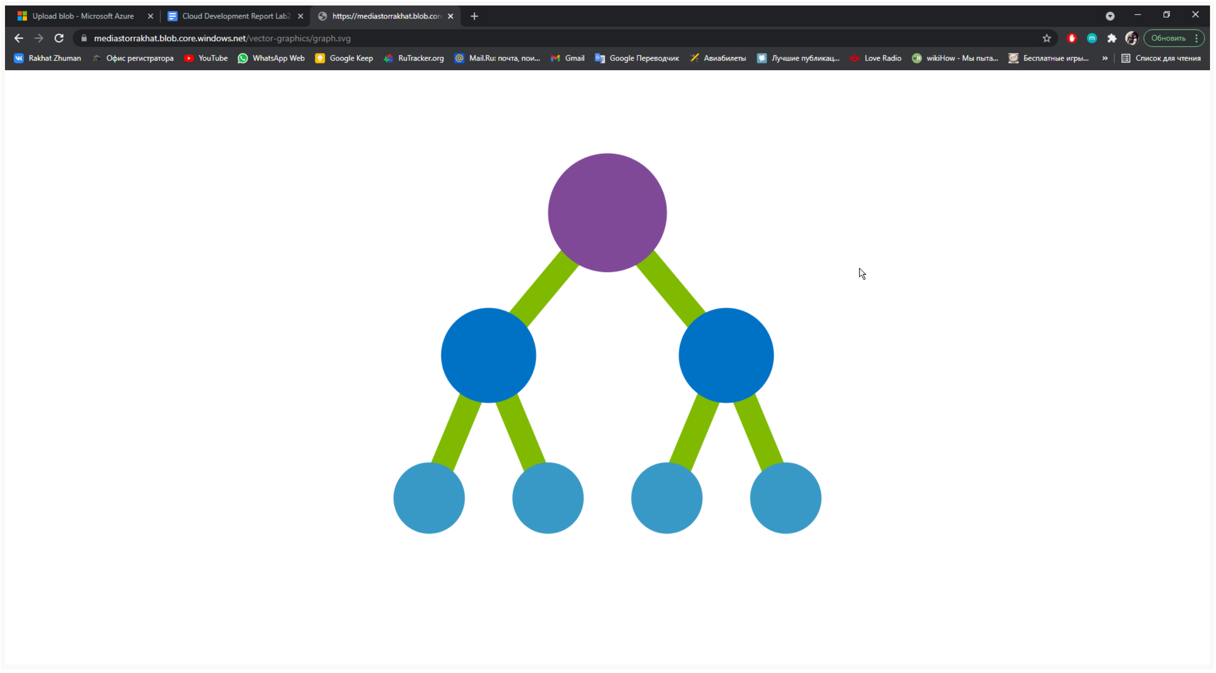
Task 4: Access blob URI by using the SDK



```csharp
    StorageSharedKeyCredential accountCredentials = new StorageSharedKeyCredential(storageAccountName, storageAccountKey);
    BlobServiceClient serviceClient = new BlobServiceClient(new Uri(blobServiceEndpoint),accountCredentials);
    AccountInfo info = await serviceClient.GetAccountInfoAsync();

    await Console.Out.WriteLineAsync($"Connected to Azure Storage Account");
    await Console.Out.WriteLineAsync($"Account name:\t{storageAccountName}");
    await Console.Out.WriteLineAsync($"Account kind:\t{info?.AccountKind}");
    await Console.Out.WriteLineAsync($"Account sku:\t{info?.SkuName}");

    await EnumerateContainersAsync(serviceClient);

    string existingContainerName = "raster-graphics";
    await EnumerateBlobsAsync(serviceClient, existingContainerName);

    string newContainerName = "vector-graphics";
    BlobContainerClient containerClient = await GetContainerAsync(serviceClient, newContainerName);

    string uploadedBlobName = "graph.svg";
    BlobClient blobClient = await GetBlobAsync(containerClient, uploadedBlobName);
    await Console.Out.WriteLineAsync($"Blob Url:\t{blobClient.Uri}");
}
private static async Task EnumerateContainersAsync(BlobServiceClient client)
{
    await foreach (BlobContainerItem container in client.GetBlobContainersAsync())
    {
        await Console.Out.WriteLineAsync($"Container:\t{container.Name}");
    }
}
private static async Task EnumerateBlobsAsync(BlobServiceClient client, string containerName)
{
    BlobContainerClient container = client.GetBlobContainerClient(containerName);
    await Console.Out.WriteLineAsync($"Searching:\t{container.Name}");

    await foreach (BlobItem blob in container.GetBlobsAsync())
    {
        await Console.Out.WriteLineAsync($"Existing Blob:\t{blob.Name}");
    }
}
private static async Task<BlobContainerClient> GetContainerAsync(BlobServiceClient client,
string containerName)
{
    BlobContainerClient container = client.GetBlobContainerClient(containerName);
    await container.CreateIfNotExistsAsync(PublicAccessType.Blob);
    await Console.Out.WriteLineAsync($"New Container:\t{container.Name}");
    return container;
}
private static async Task<BlobClient> GetBlobAsync(BlobContainerClient client, string blobName)
{
    BlobClient blob = client.GetBlobClient(blobName);
    await Console.Out.WriteLineAsync($"Blob Found:\t{blob.Name}");
    return blob;
```

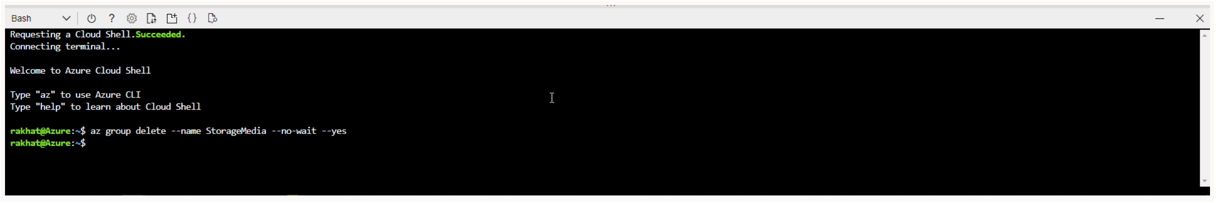
```
Existing Blob:  graph.jpg

D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>
D:\Source\Azure\03\Starter\BlobManager>dotnet run
Connected to Azure Storage Account
Account name:   mediastorrakhat
Account kind:   StorageV2
Account sku:    StandardLrs
Container:      compressed-audio
Container:      raster-graphics
Searching:      raster-graphics
Existing Blob:  graph.jpg
New Container:  vector-graphics

D:\Source\Azure\03\Starter\BlobManager>dotnet run
Connected to Azure Storage Account
Account name:   mediastorrakhat
Account kind:   StorageV2
Account sku:    StandardLrs
Container:      compressed-audio
Container:      raster-graphics
Container:      vector-graphics
Searching:      raster-graphics
Existing Blob:  graph.jpg
New Container:  vector-graphics
Blob Found:     graph.svg
Blob Url:       https://mediastorrakhat.blob.core.windows.net
/vector-graphics/graph.svg

D:\Source\Azure\03\Starter\BlobManager>
```

Task 5: Test the URI by using a browser


Upload blob - Microsoft Azure
Cloud Development Report Lab
https://mediastorrakhat.blob.cor
mediastorrakhat.blob.core.windows.net/vector-graphics/graph.svg

Exercise 5: Clean up your subscription

Task 2: Delete a resource group



```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell

rakhat@Azure:~$ az group delete --name StorageMedia --no-wait --yes
rakhat@Azure:~$
```