

# BLOBS DO AZURE ARMAZENADOS NA WEB API DO ASP.NET CORE

---

Murilo das Dores Alves  
Sistemas Operacionais 2 - Noite

# 1) INTRODUÇÃO

- Esta apresentação demonstrará o uso de API para adicionar, excluir e baixar arquivos armazenados em container BLOB da Nuvem Microsoft Azure com linguagem ASP.NET (C#).
- Também haverá referencial teórico sobre as ferramentas e recursos utilizados neste trabalho.

## 2) FERRAMENTAS

- Microsoft Visual Studio 2022 (.NET 6) para construir API
- Portal da Azure (Container Blob armazenado na Nuvem, onde ambos são da Microsoft Azure)
- Swagger para testar API

### 3) RECURSOS

- **API:** ASP.NET Core(C#)
- **Container:** Azure Blob
- **Nuvem:** Portal da Azure
- **Pacote:** NuGet (Azure Blob e Container, e Serilog)

## 4)MICROSOFT VISUAL STUDIO



## 4)MICROSOFT VISUAL STUDIO

- Permite que todo o ciclo de desenvolvimento seja concluído em um único lugar. Por exemplo, edição, depuração, teste, controle da versão e implantar na nuvem.
- Com recursos e linguagens no Visual Studio diversificados, pode-se escrever seu primeiro trecho de código a desenvolver em vários tipos de projeto. Por exemplo, criar aplicativos de área de trabalho e Web com aplicativos .NET, móveis e de jogos com C++.

## 4)MICROSOFT VISUAL STUDIO

- Para desenvolver qualquer tipo de aplicativo ou aprender um idioma, trabalha-se no Ambiente de Desenvolvimento Integrado (IDE) do Visual Studio.
- Além da edição de código, a IDE do Visual Studio reúne designers gráficos, compiladores, ferramentas de conclusão de código, controle do código-fonte, extensões e outros recursos em um só lugar.

# 4)MICROSOFT VISUAL STUDIO

The image shows the Microsoft Visual Studio IDE interface with several callouts highlighting key features:

- Create a new project**: Points to the **File** menu.
- Run your code**: Points to the **Run** button (a green play icon) in the toolbar.
- Launch Live Share**: Points to the **Live Share** button in the toolbar.
- Send feedback**: Points to the **Send Feedback** button in the top right corner.
- Manage your Azure resources**: Points to the **Server Explorer** pane on the left.
- Add controls to your UI**: Points to the **Toolbox** pane on the left.
- Manage files, projects, and solutions**: Points to the **Solution Explorer** pane on the right.
- Collaborate on code projects with your team**: Points to the **Git Changes** pane on the right.

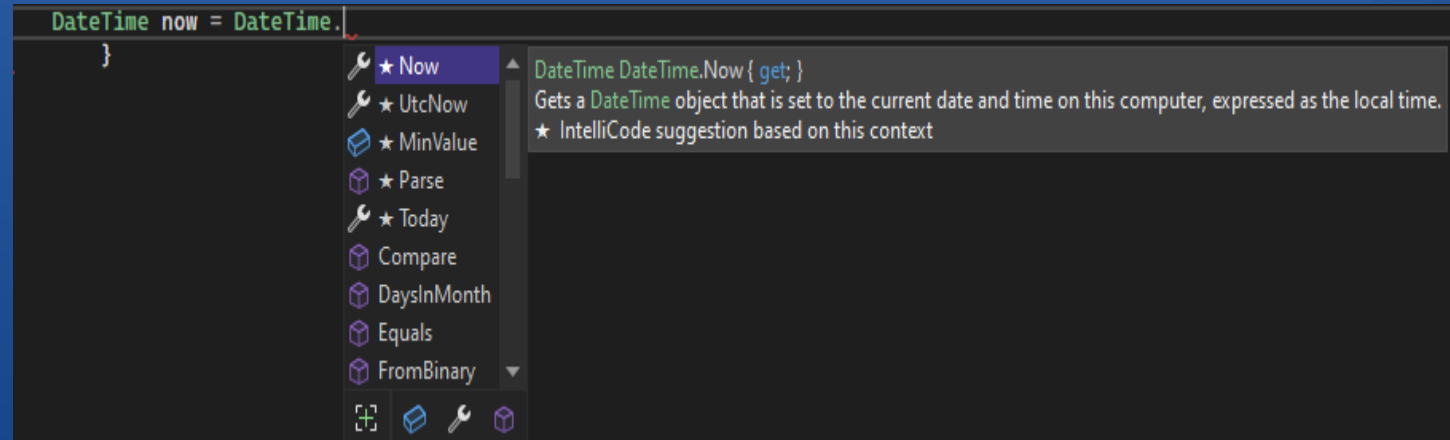
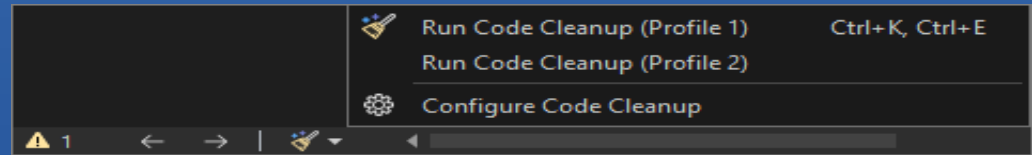
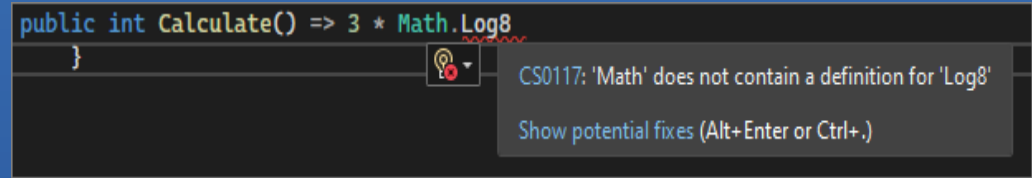
The main editor window displays the following C# code:

```
1 using System;
2 using CalculatorLibrary;
3
4 namespace CalculatorProgram
5 {
6
7     0 references
8     class Program
9     {
10
11         0 references
12         static void Main(string[] args)
13         {
14
15             bool endApp = false;
16             // Display title as the C# console calculator app.
17             Console.WriteLine("Console Calculator in C#\r");
18             Console.WriteLine("-----\n");
19
20             Calculator calculator = new Calculator();
21             while (!endApp)
22             {
23                 // Declare variables and set to empty.
24                 string numInput1 = "";
25                 string numInput2 = "";
26                 double result = 0;
27
28                 // Ask the user to type the first number.
29                 Console.Write("Type a number, and then press Enter: ");
30                 numInput1 = Console.ReadLine();
31
32                 double cleanNum1 = 0;
33                 while (!double.TryParse(numInput1, out cleanNum1))
34                 {
35                     Console.WriteLine("This is not valid input. Please enter an integer");
36                 }
37             }
38         }
39     }
40 }
```



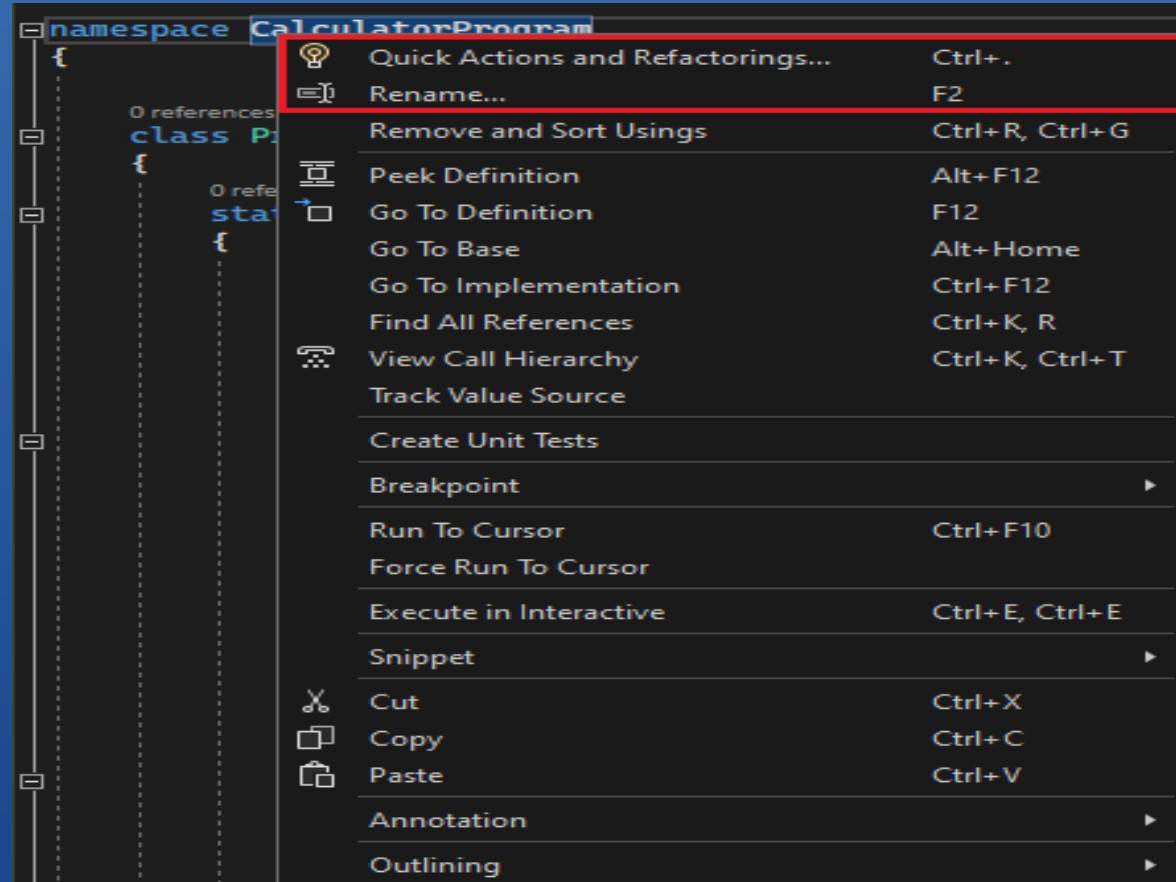
## 4.1) Recursos de Produtividade Populares

- Rabiscos e Ações Rápidas
- Limpeza de código
- IntelliSense



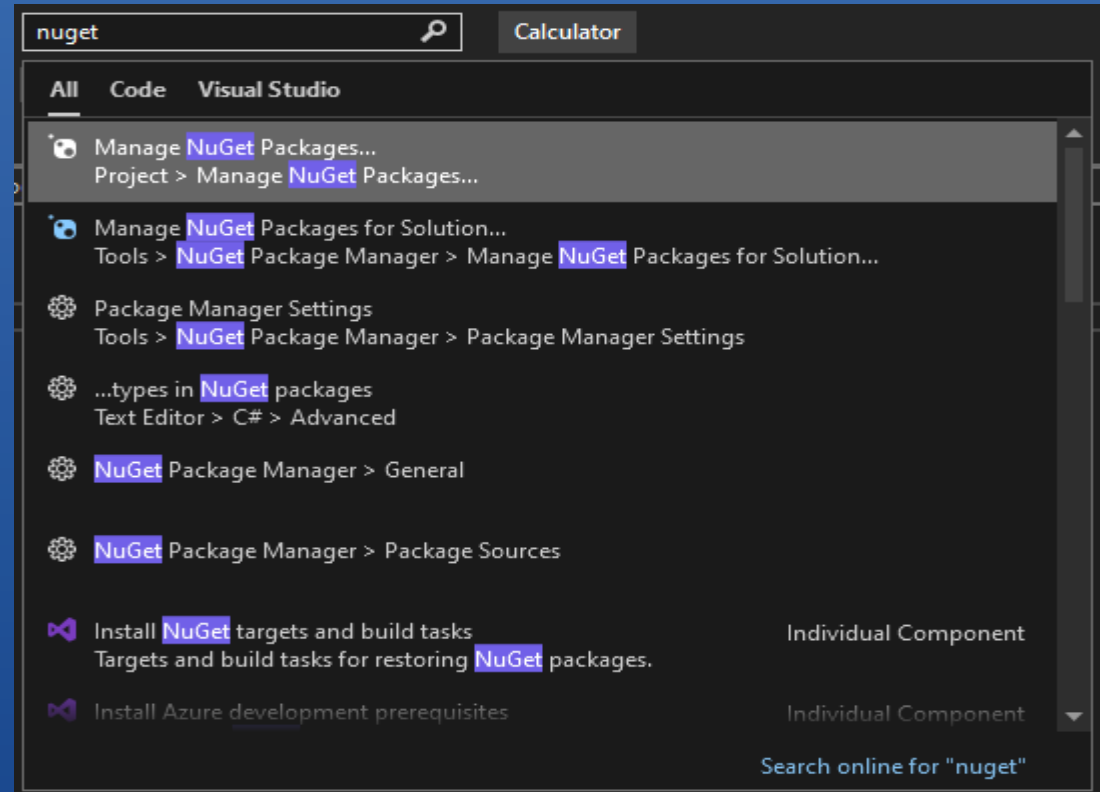
## 4.1) Recursos de Produtividade Populares

- Refatoração



## 4.1) Recursos de Produtividade Populares

- Pesquisa do Visual Studio



## 4.1) Recursos de Produtividade Populares

- Hierarquia de chamadas

The screenshot shows the 'Call Hierarchy' window in Visual Studio. The left pane displays a tree view of the call hierarchy for 'My Solution'. The right pane shows the 'Call Sites' and 'Location' for the selected method.

Call Sites	Location
while (!double.TryParse(numInput1, out	Program.cs - (29, 32)
while (!double.TryParse(numInput2, out	Program.cs - (40, 32)

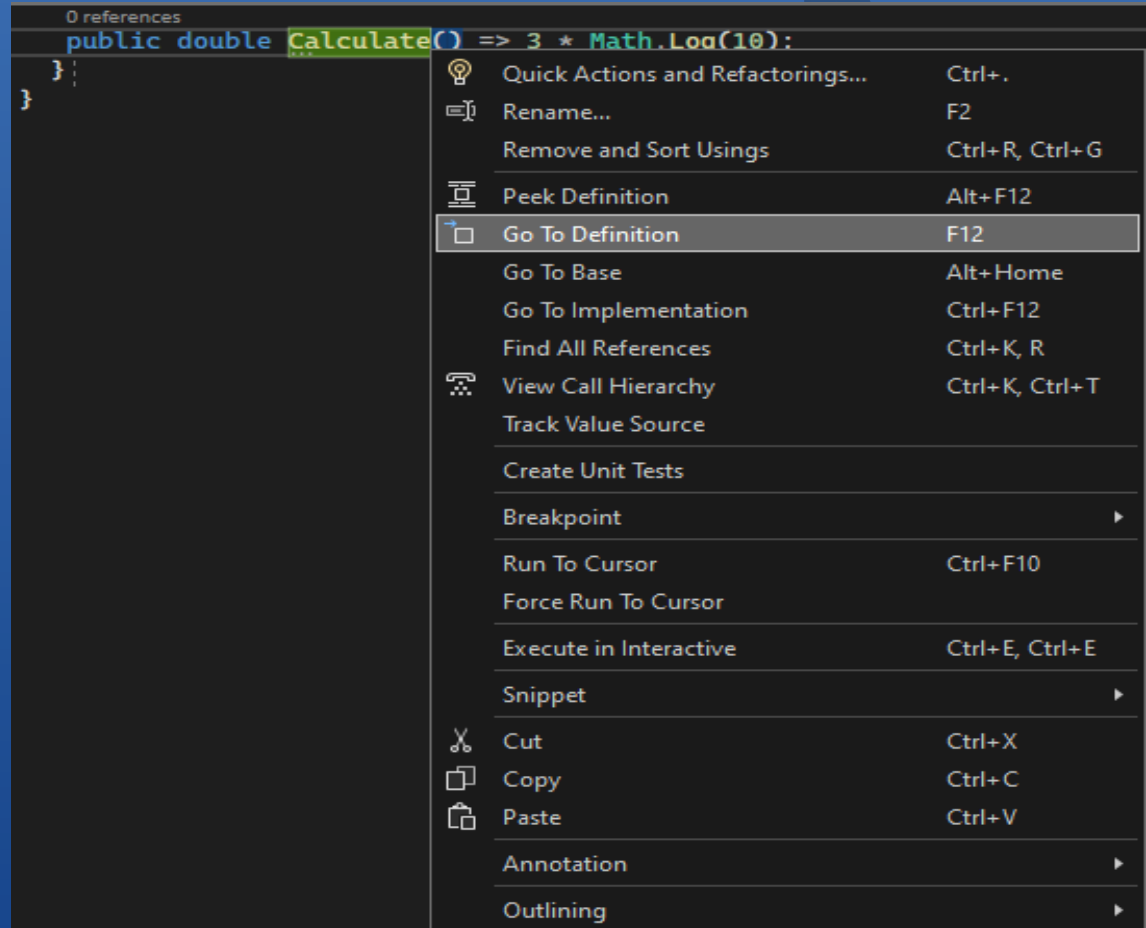
- CodeLens

The screenshot shows the 'CodeLens' feature in Visual Studio. The left pane displays the 'CalculatorLibrary\Program.cs' file with a 'Collapse All' button. The right pane shows the 'CalculatorLibrary\CalculatorLibrary.cs' file with a 'writer' variable. A tooltip is visible over the 'Calculator' class, showing the 'public class Calculator' definition.

```
public class Calculator
{
    // ...
}
```

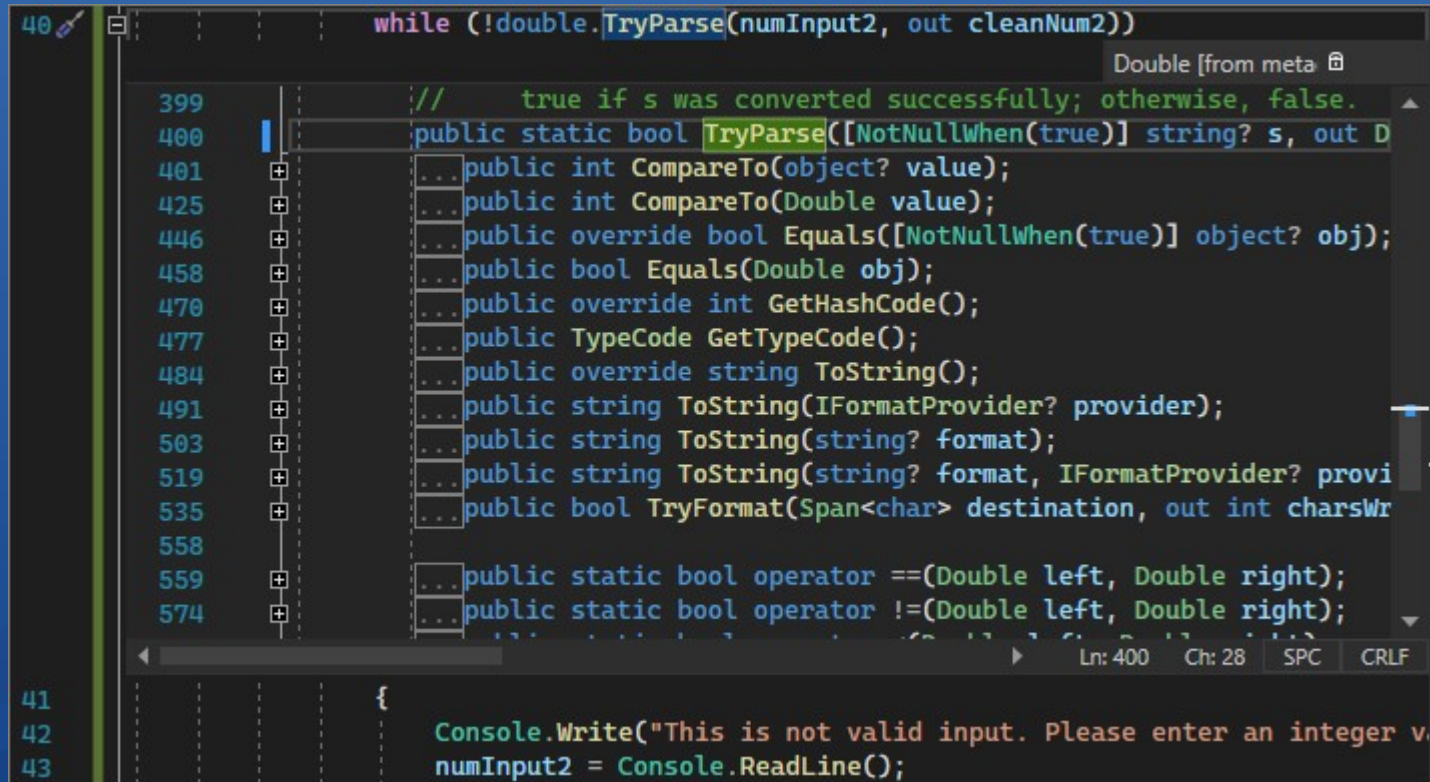
## 4.1) Recursos de Produtividade Populares

- Ir para Definição



## 4.1) Recursos de Produtividade Populares

- Espiar Definição



The screenshot shows an IDE window with a code editor. The top part of the editor displays a `while` loop: `while (!double.TryParse(numInput2, out cleanNum2))`. Below this, the definition of the `TryParse` method is shown, starting with `public static bool TryParse([NotNullWhen(true)] string? s, out Double value)`. The method body includes several `public` methods: `CompareTo(object? value)`, `CompareTo(Double value)`, `Equals([NotNullWhen(true)] object? obj)`, `Equals(Double obj)`, `GetHashCode()`, `GetTypeCode()`, `ToString()`, `ToString(IFormatProvider? provider)`, `ToString(string? format)`, `ToString(string? format, IFormatProvider? provider)`, `TryFormat(Span<char> destination, out int charsWritten)`, `operator ==(Double left, Double right)`, and `operator !=(Double left, Double right)`. The code is color-coded, with comments in green and method signatures in blue. The bottom part of the editor shows a `{` block with a `Console.WriteLine` statement: `Console.WriteLine("This is not valid input. Please enter an integer value.");` and a `numInput2 = Console.ReadLine();` statement.

```
40  while (!double.TryParse(numInput2, out cleanNum2))  
    Double [from meta]  
399  // true if s was converted successfully; otherwise, false.  
400  public static bool TryParse([NotNullWhen(true)] string? s, out D  
401  ... public int CompareTo(object? value);  
425  ... public int CompareTo(Double value);  
446  ... public override bool Equals([NotNullWhen(true)] object? obj);  
458  ... public bool Equals(Double obj);  
470  ... public override int GetHashCode();  
477  ... public TypeCode GetTypeCode();  
484  ... public override string ToString();  
491  ... public string ToString(IFormatProvider? provider);  
503  ... public string ToString(string? format);  
519  ... public string ToString(string? format, IFormatProvider? provi  
535  ... public bool TryFormat(Span<char> destination, out int charsWr  
558  
559  ... public static bool operator ==(Double left, Double right);  
574  ... public static bool operator !=(Double left, Double right);  
41  {  
42  Console.WriteLine("This is not valid input. Please enter an integer v  
43  numInput2 = Console.ReadLine();
```

## 4.1) Recursos de Produtividade Populares

- Live Share

Edite e depure de forma colaborativa com outras pessoas em tempo real, independentemente do tipo de aplicativo ou da linguagem de programação.

Compartilhamento seu projeto instantaneamente e com segurança.

Compartilhar sessões de depuração, instâncias de terminal, aplicativos Web localhost, chamadas de voz e muito mais.

## 4.2) Instalação



### **.NET desktop development**



Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET and .NET Frame...



## 5) .NET

Ao usar o Visual Studio e o .NET, desenvolve-se aplicativos para área de trabalho, Web, celulares, jogos e IoT. Você pode escrever aplicativos .NET nas linguagens C#, F# ou Visual Basic. Exemplos de aplicações:

- Desenvolvimento da área de trabalho usando o .NET Core
- Desenvolvimento na Web ou Nuvem usando o ASP.NET Core
- Desenvolvimento de jogos usando o Unity
- Aprendizado de máquina usando ML.NET

## 6) ASP.NET CORE

ASP.NET Core é uma estrutura de software livre multiplataforma, de alto desempenho e software livre para criar aplicativos modernos, habilitados para nuvem e conectados à Internet. Com o ASP.NET Core, você pode:

- Criar aplicativos e serviços Web, aplicativos de Internet das Coisas (IoT) e back-ends móveis.
- Usar suas ferramentas de desenvolvimento favoritas no Windows, macOS e Linux.
- Implantar na nuvem ou local.
- Execução no .NET Core.

## 6.1) Por que Escolher o ASP.NET Core?

- Uma história unificada para a criação da interface do usuário da Web e das APIs Web.
- Projetado para capacidade de teste.
- Razor As páginas tornam os cenários de codificação focados em páginas mais fáceis e produtivos.
- Blazor permite que você use C# no navegador ao lado do JavaScript. Compartilhe a lógica de aplicativo do lado do cliente e do servidor toda escrita com o .NET.
- Capacidade de desenvolver e executar no Windows, macOS e Linux.
- De software livre e voltado para a comunidade.
- Integração de estruturas modernas do lado do cliente e fluxos de trabalho de desenvolvimento.
- Suporte para hospedagem de serviços RPC (chamada de procedimento remoto) usando gRPC.
- Um sistema de configuração pronto para a nuvem, baseado no ambiente.
- Injeção de dependência interna.
- Um pipeline de solicitação HTTP leve, modular e de alto desempenho.
- Capacidade de hospedar no seguinte: Kestrel, IIS, HTTP.sys, Nginx, Apache, Docker
- Controle de versão lado a lado.
- Ferramentas que simplificam o moderno desenvolvimento para a Web.

## 6.2) Compilar APIs Web e uma Interface do Usuário da Web Usando o ASP.NET Core MVC

- O padrão MVC (Model-View-Controller) ajuda a tornar as APIs e os aplicativos Webs testáveis.
- Razor Pages, cujo modelo de programação baseado em página, torna-se a criação da interface do usuário da Web mais fácil e produtiva.
- Marcação fornece uma sintaxe produtiva para Razor exibições de Páginas e MVC.
- Os Auxiliares de Marcação permitem que o código do servidor participe da criação e renderização de elementos HTML em arquivos do Razor.
- O suporte interno para vários formatos de dados e negociação de conteúdo permite que as APIs Web alcancem uma ampla gama de clientes, incluindo navegadores e dispositivos móveis.
- O model binding mapeia os dados de solicitações HTTP para os parâmetros de método de ação automaticamente.
- A Validação de Modelos executa automaticamente a validação no lado do cliente e do servidor.

## 6.3) Desenvolvimento do Lado do Cliente

ASP.NET Core integra-se com a estruturas e bibliotecas populares do lado do cliente, incluindo Blazor, Angular, React e Bootstrap.

## 6.4) ASP.NET Core Estruturas de Destino

- Multiplataforma. É executado no Windows, macOS e Linux.
- Desempenho aprimorado
- Controle de versão lado a lado
- Novas APIs
- Software livre

## 7) C#

Produtivas, multiuso, fortemente tipadas, orientadas a objeto, código aberto

C# e Visual Basic são linguagens de programação projetadas para a criação de uma variedade de aplicativos executados no .NET. Essas linguagens são poderosas, fortemente tipadas e orientadas a objeto.

Elas se baseiam no .NET Compiler Platform “Roslyn”, que fornece APIs de análise de código avançadas e são de código aberto no GitHub.

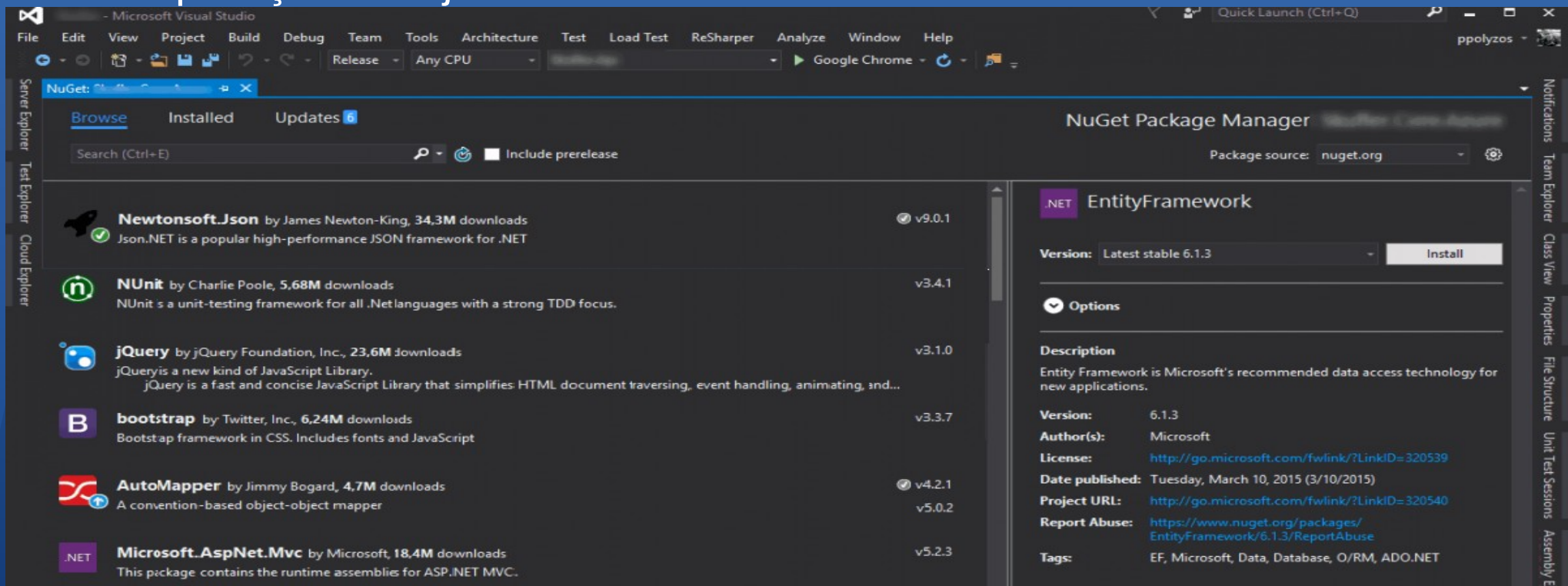
## 7) C#

- Aplicativos para áreas de trabalho
- Aplicativos Android/iOS
- Desenvolvimento de jogos



## 8)NuGet

- NuGet Packages serve para instalar pacotes específicos de acordo com a aplicação desejada.



## 9) AZURE BLOB STORAGE NO .NET 6.0

A solução Azure Blob Storage é utilizada para armazenar objetos binários em nuvem. A velocidade, escalabilidade, facilidade de acesso e segurança (contra acidentes e ataques) tornam esse armazenamento em nuvem atraente para as organizações, sendo também uma ferramenta à um custo muito baixo.

A vantagem em utilizá-lo principalmente, é de não salvar arquivos do tipo IMAGE ou Base64 no banco de dados relacional, o que deixa uma operação muito custosa e lenta dependendo do tamanho e quantidade.

## 9) AZURE BLOB

O Blob Storage é ideal para:

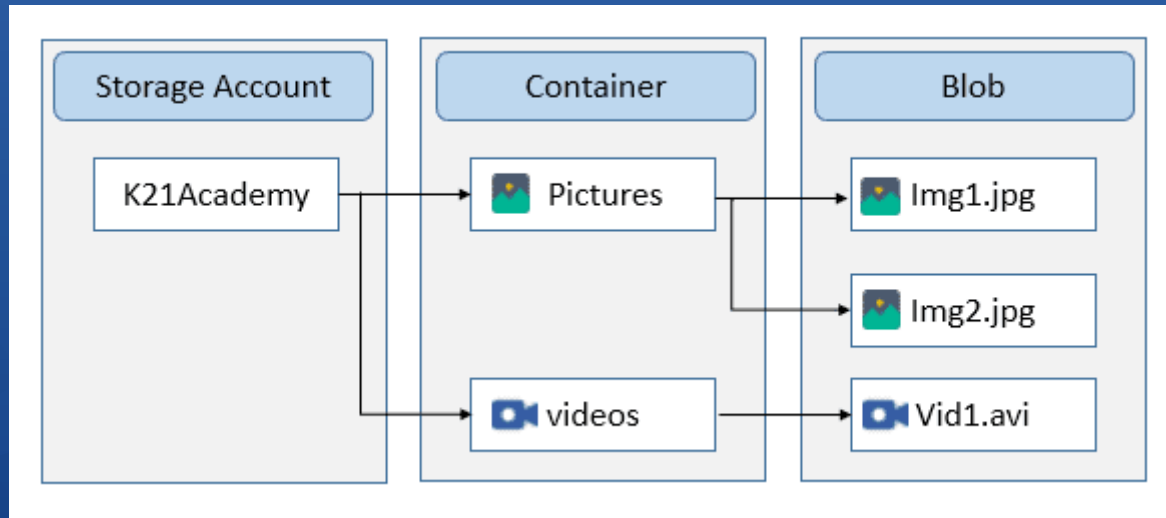
- Fornecer imagens ou documentos no navegador diretamente.
- Streaming de vídeo ou áudio.
- Armazenamento de dados para backup, restauração e arquivamento.
- Armazenar arquivos para acesso distribuído.
- Gravar arquivos de log.

## 9.1) Pré-requisitos

- Uma conta no Azure
- Visual Studio 2022 (.NET 6.0)
- Nuget package Microsoft.Azure.Storage.Blob

## 9.2) Configurando o Azure Blob Storage

- **Storage Account:** Conta de armazenamento do Azure.
- **Container:** Atua como um diretório para organizar e gerenciar os objetos.
- **Blob:** Representação de um objeto não estruturado, podendo ser texto, áudio, imagem, vídeo, etc.



## 10) PORTAL DA AZURE

- O portal do Azure é um console unificado baseado na Web alternativo para as ferramentas de linha de comando.
- Pode-se gerenciar sua assinatura do Azure usando uma interface gráfica do usuário.
- Criar, gerenciar e monitorar tudo, desde aplicativos Web simples a implantações em nuvem complexas no portal.
- O portal do Azure é projetado para ter resiliência e disponibilidade contínua, sendo presente em todos os datacenters do Azure.
- Essa configuração o torna resiliente a falhas em datacenters individuais, evitando lentidão na rede por manter a proximidade com os usuários.
- O portal é atualizado constantemente e não requer nenhum tempo de inatividade para atividades de manutenção.

## 10.1) Soluções

- **Durável e altamente disponível:** o Azure Storage tem redundância ativada padrão, garantindo que nossos dados estejam seguros em caso de falhas inesperadas de hardware. Há também opcionalmente, a replicação de dados entre data centers ou regiões geográficas para proteção adicional contra percalços locais ou desastres naturais.

## 10.1) SOLUÇÕES

- **Seguro:** O serviço de armazenamento criptografa todos os dados que armazena. Também há o fornecimento de controle fino de quem possa acessar esses dados.
- **Escalável:** O Armazenamento escalável e foi projetado para atender às necessidades modernas de salvamento e desempenho de dados dos aplicativos.
- **Gerenciado:** Sendo um serviço em nuvem, o Azure lida com a manutenção de hardware, atualizações e problemas críticos para os usuários que não precisam de preocupar com nenhum deles.



## 10.1) SOLUÇÕES

- **Acessível:** Os dados armazenados no Azure Storage estão em acesso em qualquer local do mundo em http. A Microsoft fornece SDKs para armazenamento Azure em diversos idiomas como .NET, Java, Node.js, Python, PHP, Ruby, Go, etc. Também há o acesso dos serviços de armazenamento através de uma API REST, Azure PowerShell ou Azure CLI. Também o portal Azure e o Aplicativo Azure Storage Explorer fornecem uma interface visual para trabalhar com os dados armazenados neste.

## 10.2) tipos de serviços de armazenamento azure

- **Azure Blob Storage:** oferece lojas de objetos massivamente escaláveis para armazenar dados de texto e binários. Ele também inclui suporte para análise de big data através do Data Lake Storage Gen2.
- **Azure Queues:** a Microsoft oferece o Azure Queue Storage como um serviço para armazenar um grande número de mensagens. Eles podem habilitar mensagens confiáveis entre aplicativos e componentes.

## 10.2) Tipos de Serviços de Armazenamento da Azure

- **Azure Tables:** é uma excelente escolha para armazenar dados estruturados do NoSQL na nuvem, e valores de chave e atributos sem definir um esquema.
- **Azure Disks:** oferecem volumes de armazenamento gerenciados em blocos que são anexados às Máquinas Virtuais do Azure.
- **Azure Files:** um compartilhamento de arquivos em nuvem totalmente gerenciado, onde são acessados através de serviços de nuvem e no local.

## 11) SWAGGER

- É uma ferramenta para testar API's
- O Swagger é um framework composto por diversas ferramentas que auxilia a descrição, consumo e visualização de serviços de uma API REST, independentemente da linguagem.

## 11.1) Tipos de Tarefas

- 1) A especificação da API determina os modelos de dados que serão entendidos pela API e as funcionalidades presentes na mesma. Para cada funcionalidade, precisa-se especificar o seu nome, os parâmetros que devem ser passados no momento de sua invocação e os valores que irão ser retornados aos usuários da API. O OpenAPI Specification é um exemplo de ferramenta.
- 2) Após especificar a API, a implementação é facilitada pelo framework, com a ferramenta Swagger Codegen, onde é possível montar o código inicial nas principais linguagem de programação automaticamente.
- 3) Os testes de API são importantes, pois ajudam a garantir o funcionamento, o desempenho e a confiabilidade da sua aplicação. O Swagger oferece ferramentas para testes manuais, automatizados e de desempenho
- 4) Para auxiliar na utilização da API, o Swagger dispõe de ferramenta para deixar a visualização mais intuitiva, onde há interação com a API.

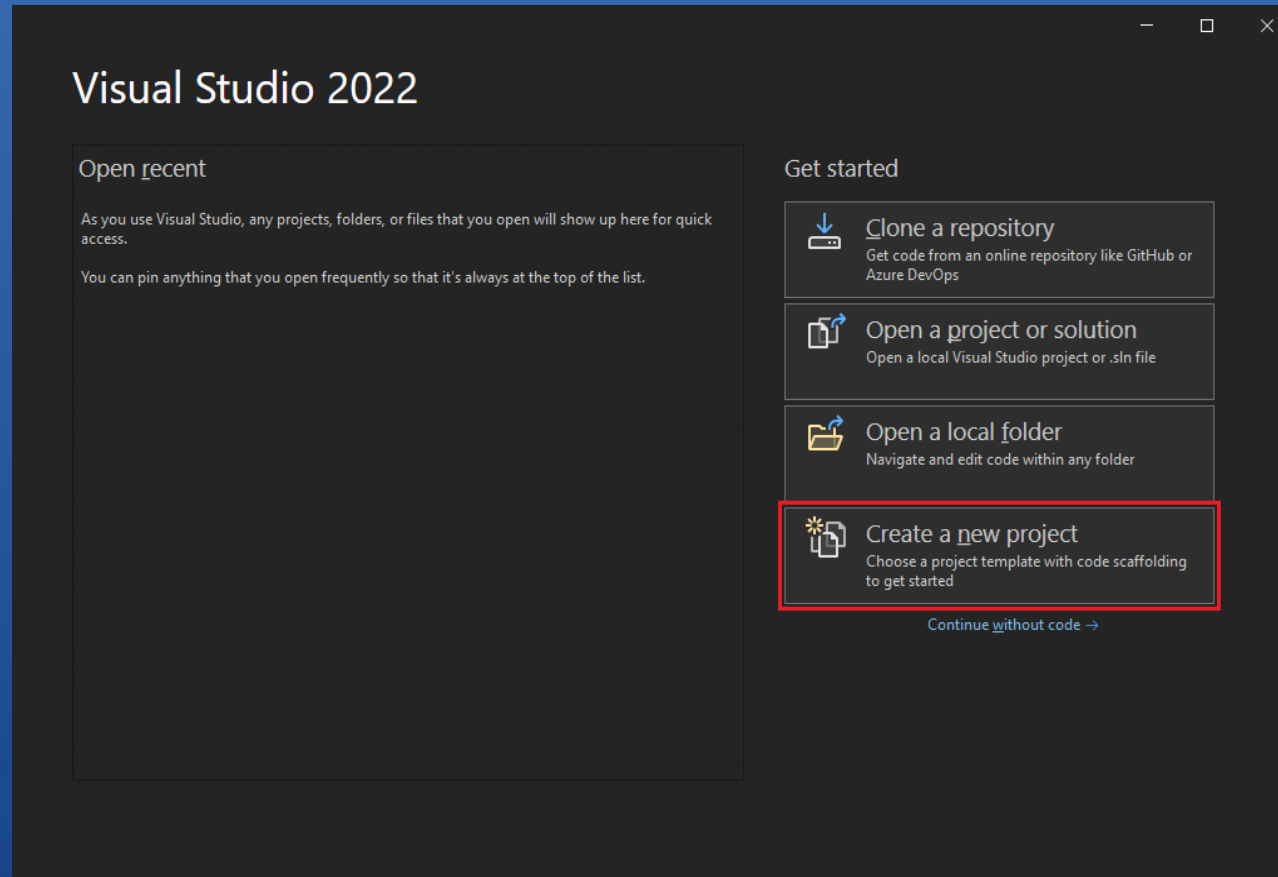
## 11.2) Criação de Tarefas

- 1) Automaticamente:** No desenvolvimento da API é gerada a documentação simultaneamente.
- 2) Manualmente:** O desenvolvedor escreve livremente as especificações da API e as publicam posteriormente em seu próprio servidor.
- 3) Codegen:** Conversão de todas as anotações contidas no código fonte das APIs REST em documentação.

## 12) IMPLEMENTAÇÃO DA API

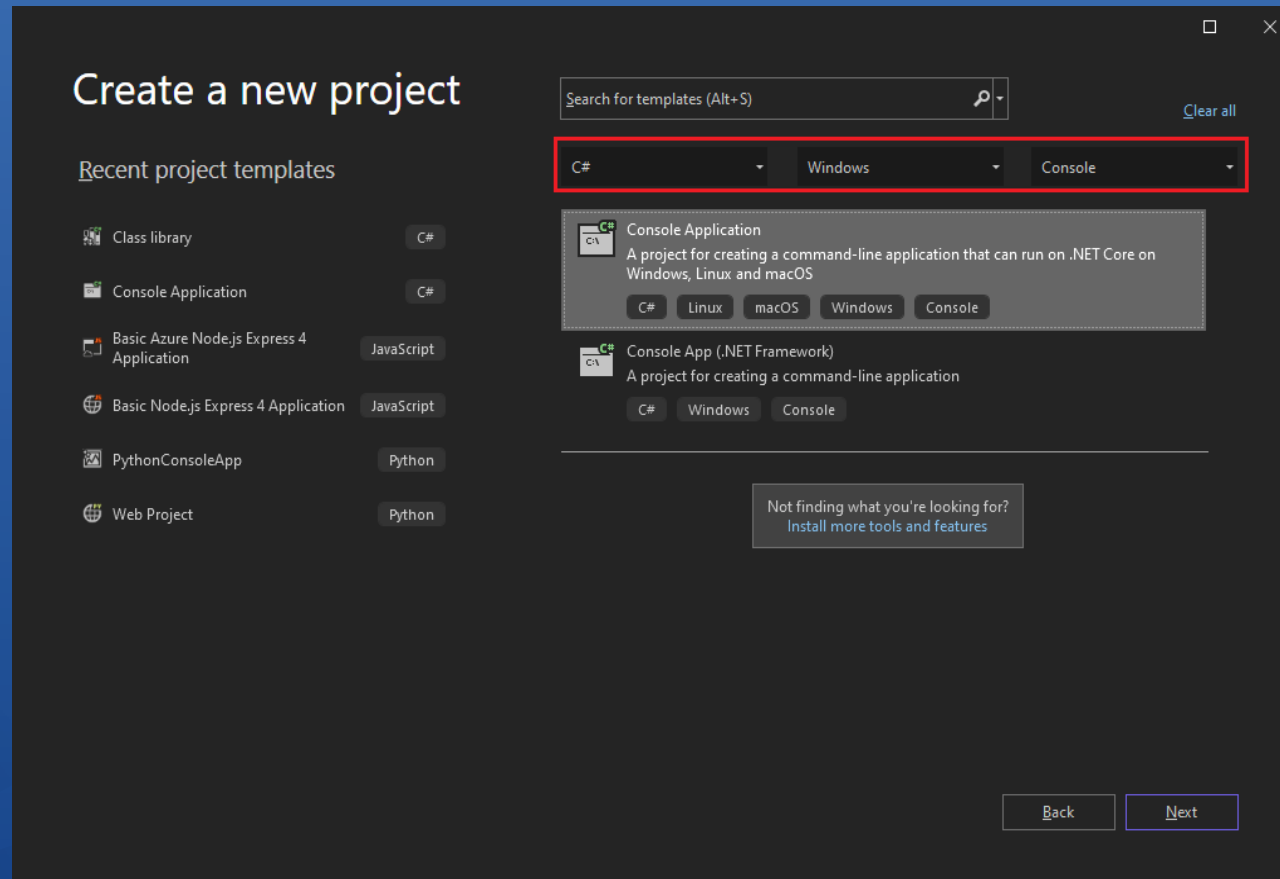
- 12.1) Criação de um Programa
- 12.2) Programa e seus Arquivos
- 12.3) Execução
- 12.4) Contêiner no Portal da Azure
- 12.5) Teste da API no Swagger

## 12.1) Criação de um Programa





# 12.1) Criação de um Programa



## 12.1) Criação de um Programa

□ ×

### Configure your new project

Console Application C# Linux macOS Windows Console

Project name

Location

 ...

Solution name ⓘ

☐ Place solution and project in the same directory

Back Next

## 12.1) Criação de um Programa

□ ×

### Additional information

Console Application C# Linux macOS Windows Console

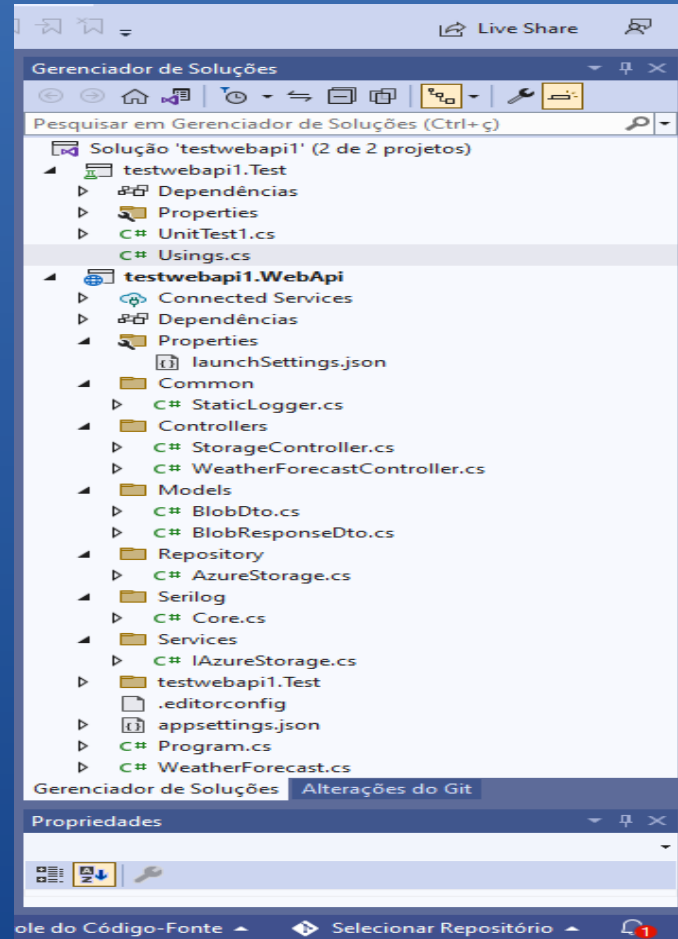
Framework ⓘ

.NET 6.0 (Preview)

▼

Back Create

## 12.2) Programa e seus Arquivos



## 12.2) Programa e seus Arquivos



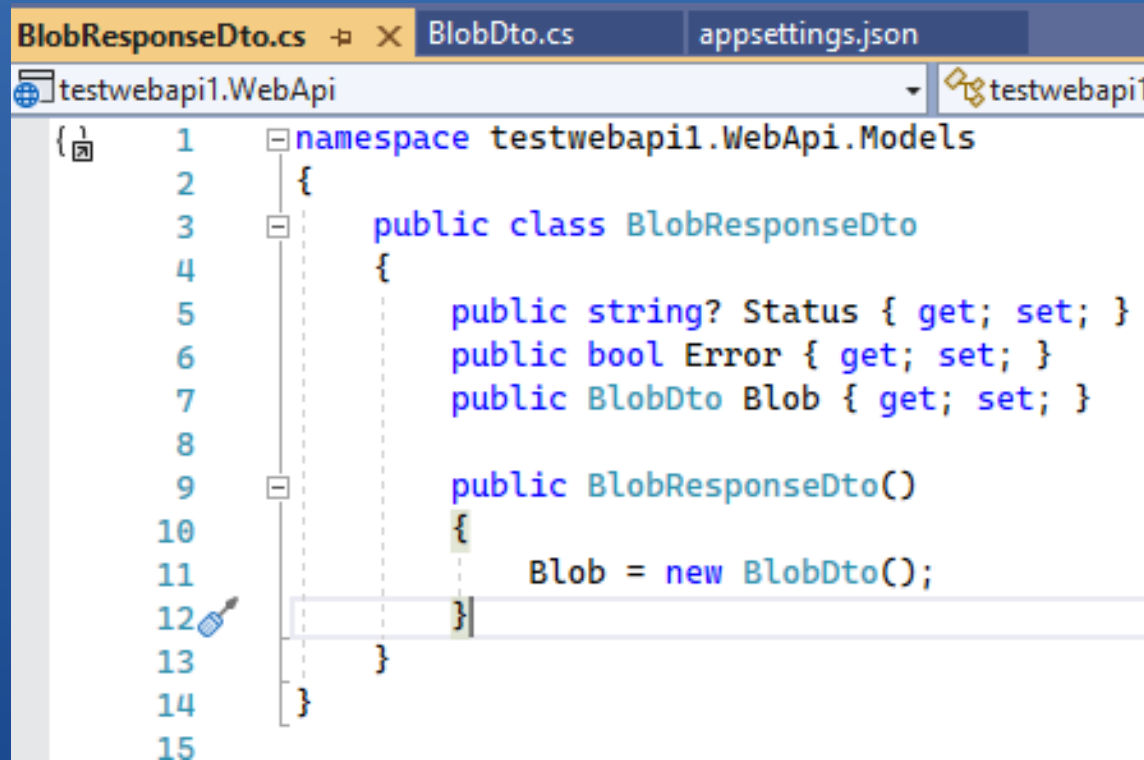
```
appsettings.json X WeatherForecast.cs X
Esquema: https://json.schemastore.org/appsettings.json
1 {
2   "Logging": {
3     "LogLevel": {
4       "Default": "Information",
5       "Microsoft.AspNetCore": "Warning"
6     }
7   },
8   "AllowedHosts": "*",
9   "BlobConnectionString": "DefaultEndpointsProtocol=https;AccountName=testwebapi1;AccountKey=FJI8HWivk67ihjmdGqu5G6qtFuggXRThe2qOPUlee15bzriVaNGyeekquQ9TolssB+RCgh5tK7no+AstBTS1/g==;EndpointSuffix=core.windows.net",
10  "BlobContainerName": "container"
11 }
```

## 12.2) Programa e seus Arquivos



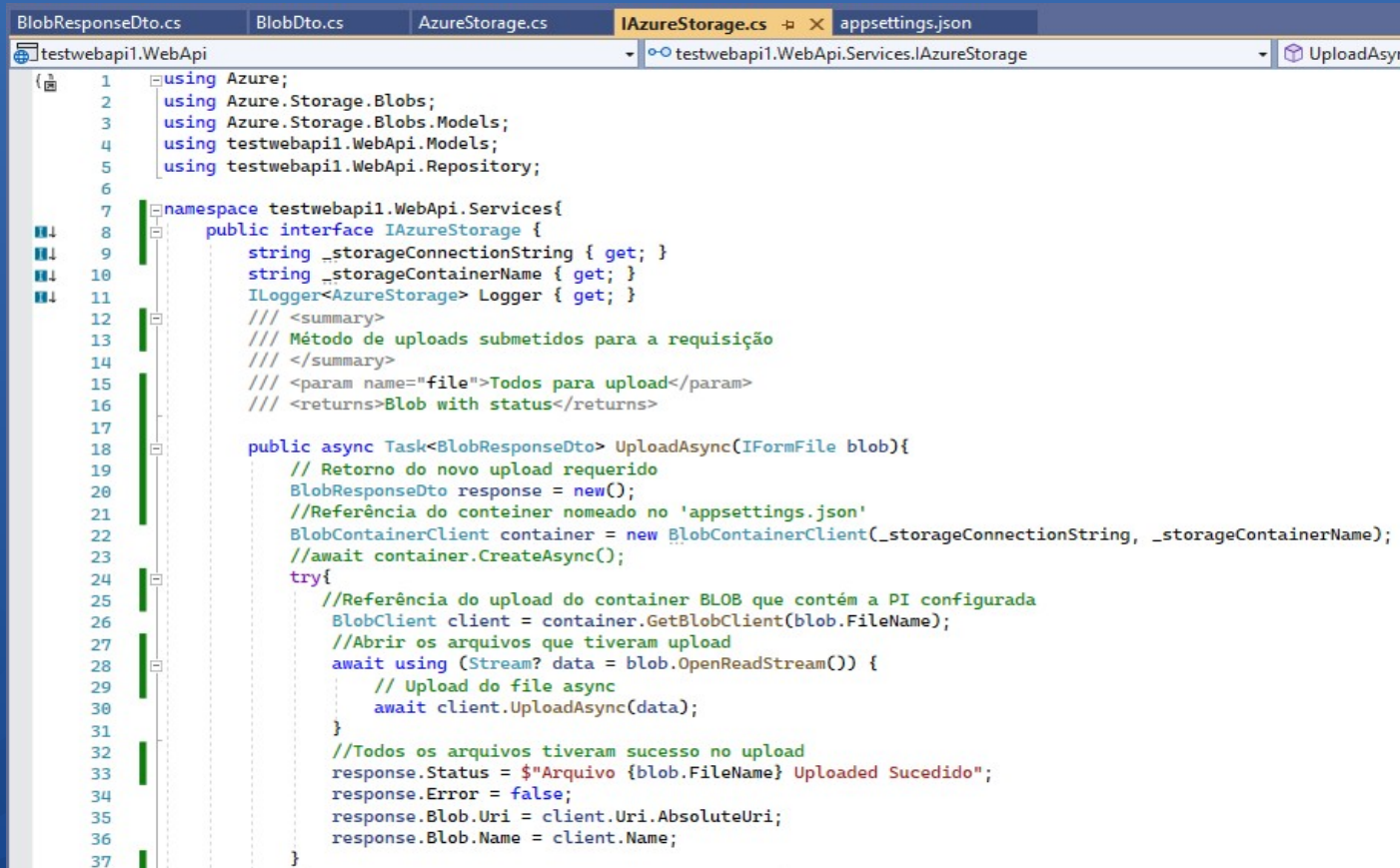
```
BlobDto.cs  X appsettings.json
testwebapi1.WebApi
1 namespace testwebapi1.WebApi.Models
2 {
3     public class BlobDto
4     {
5         public string? Uri { get; set; }
6         public string? Name { get; set; }
7         public string? ContentType { get; set; }
8         public Stream? Content { get; set; }
9     }
10 }
11
```

## 12.2) Programa e seus Arquivos



```
1 namespace testwebapi1.WebApi.Models
2 {
3     public class BlobResponseDto
4     {
5         public string? Status { get; set; }
6         public bool Error { get; set; }
7         public BlobDto Blob { get; set; }
8
9         public BlobResponseDto()
10        {
11            Blob = new BlobDto();
12        }
13    }
14 }
15
```

## 12.2) Programa e seus Arquivos



The screenshot displays a Visual Studio code editor with the following tabs: BlobResponseDto.cs, BlobDto.cs, AzureStorage.cs, IAzureStorage.cs, and appsettings.json. The active file is IAzureStorage.cs, which is part of the testwebapi1.WebApi.Services namespace. The code defines the IAzureStorage interface and implements the UploadAsync method.

```
1 using Azure;
2 using Azure.Storage.Blobs;
3 using Azure.Storage.Blobs.Models;
4 using testwebapi1.WebApi.Models;
5 using testwebapi1.WebApi.Repository;
6
7 namespace testwebapi1.WebApi.Services{
8     public interface IAzureStorage {
9         string _storageConnectionString { get; }
10        string _storageContainerName { get; }
11        ILogger<IAzureStorage> Logger { get; }
12
13        /// <summary>
14        /// Método de uploads submetidos para a requisição
15        /// </summary>
16        /// <param name="file">Todos para upload</param>
17        /// <returns>Blob with status</returns>
18
19        public async Task<BlobResponseDto> UploadAsync(IFormFile blob){
20            // Retorno do novo upload requerido
21            BlobResponseDto response = new();
22            //Referência do container nomeado no 'appsettings.json'
23            BlobContainerClient container = new BlobContainerClient(_storageConnectionString, _storageContainerName);
24            //await container.CreateAsync();
25            try{
26                //Referência do upload do container BLOB que contém a PI configurada
27                BlobClient client = container.GetBlobClient(blob.FileName);
28                //Abrir os arquivos que tiveram upload
29                await using (Stream? data = blob.OpenReadStream()) {
30                    // Upload do file async
31                    await client.UploadAsync(data);
32                }
33                //Todos os arquivos tiveram sucesso no upload
34                response.Status = $"Arquivo {blob.FileName} Uploaded Sucedido";
35                response.Error = false;
36                response.Blob.Uri = client.Uri.AbsoluteUri;
37                response.Blob.Name = client.Name;
38            }
39        }
40    }
41 }
```

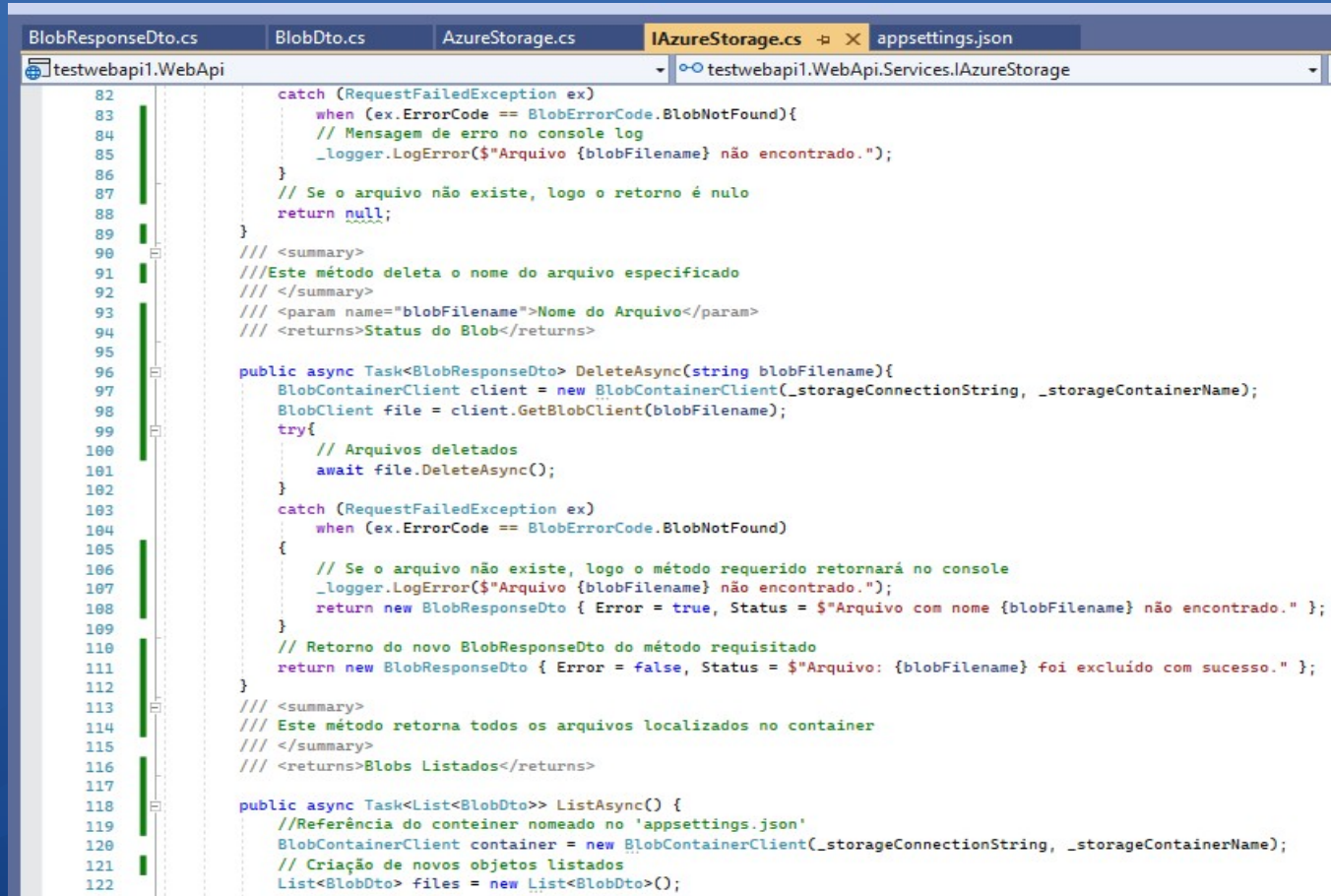


## 12.2) Programa e seus Arquivos

```
BlobResponseDto.cs  BlobDto.cs  AzureStorage.cs  IAzureStorage.cs  appsettings.json  WeatherForecast.cs
testwebapi1.WebApi  testwebapi1.WebApi.Services.IAzureStorage  DownloadAsync(string blobFilename)

38 //Se o arquivo existe, logo o upload não poderá ser executado
39 catch (RequestFailedException ex)
40 {
41     when (ex.ErrorCode == BlobErrorCodes.BlobAlreadyExists){
42         _logger.LogError($"Nome do Arquivo {blob.FileName} já existe no container. Defina outro nome para armazenar o arquivo no contêiner: '{_storageContainerName}'.");
43         response.Status = $"Nome do Arquivo {blob.FileName} já existe. Por favor, use outro nome para armazenar seu arquivo.";
44         response.Error = true;
45         return response;
46     }
47     // Existe erro inesperado, logo retornará a mensagem de erro
48     catch (RequestFailedException ex){
49         //Mensagem de erro no console para retornar o método requerido
50         _logger.LogError($"Exceção não tratada. ID: {ex.StackTrace} - Mensagem: {ex.Message}");
51         response.Status = $"Erro inesperado: {ex.StackTrace}. Verifique o registro com o ID do StackTrace.";
52         response.Error = true;
53         return response;
54     }
55     // Retorno do objeto do BlobUploadResponse
56     return response;
57 }
58 /// <summary>
59 /// Método de downloads especificados para todos os nomes dos arquivos
60 /// </summary>
61 /// <param name="blobFilename">Nome do Arquivo</param>
62 /// <returns>Blob</returns>
63
64 public async Task<BlobDto> DownloadAsync(string blobFilename){
65     //Referência do container nomeado no 'appsettings.json'
66     BlobContainerClient client = new BlobContainerClient(_storageConnectionString, _storageContainerName);
67     try{
68         //Referência do upload do container BLOB que contém a PI configurada
69         BlobClient file = client.GetBlobClient(blobFilename);
70         // Verificar se o arquivo existe no container
71         if (await file.ExistsAsync()){
72             var data = await file.OpenReadAsync();
73             Stream blobContent = data;
74             // Download dos arquivos detalhados no async
75             var content = await file.DownloadContentAsync();
76             //Adicionar dados nas variáveis ordenadamente para retornar no BlobDto
77             string name = blobFilename;
78             string contentType = content.Value.Details.ContentType;
79             //Criar novo BlobDto com as variáveis nos dados do blob
80             return new BlobDto { Content = blobContent, Name = name, ContentType = contentType };
81         }
82     }
83 }
```

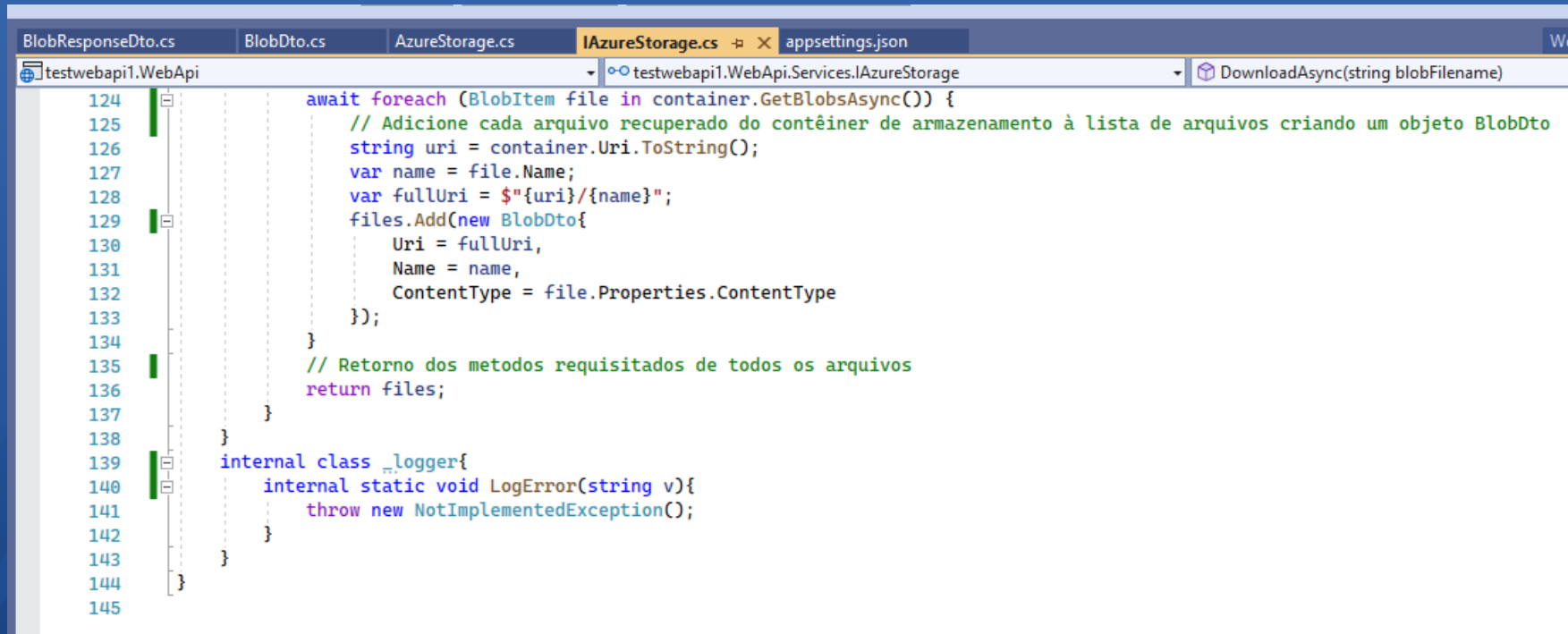
## 12.2) Programa e seus Arquivos



```
BlobResponseDto.cs  BlobDto.cs  AzureStorage.cs  IAzureStorage.cs  appsettings.json
testwebapi1.WebApi  testwebapi1.WebApi.Services.IAzureStorage

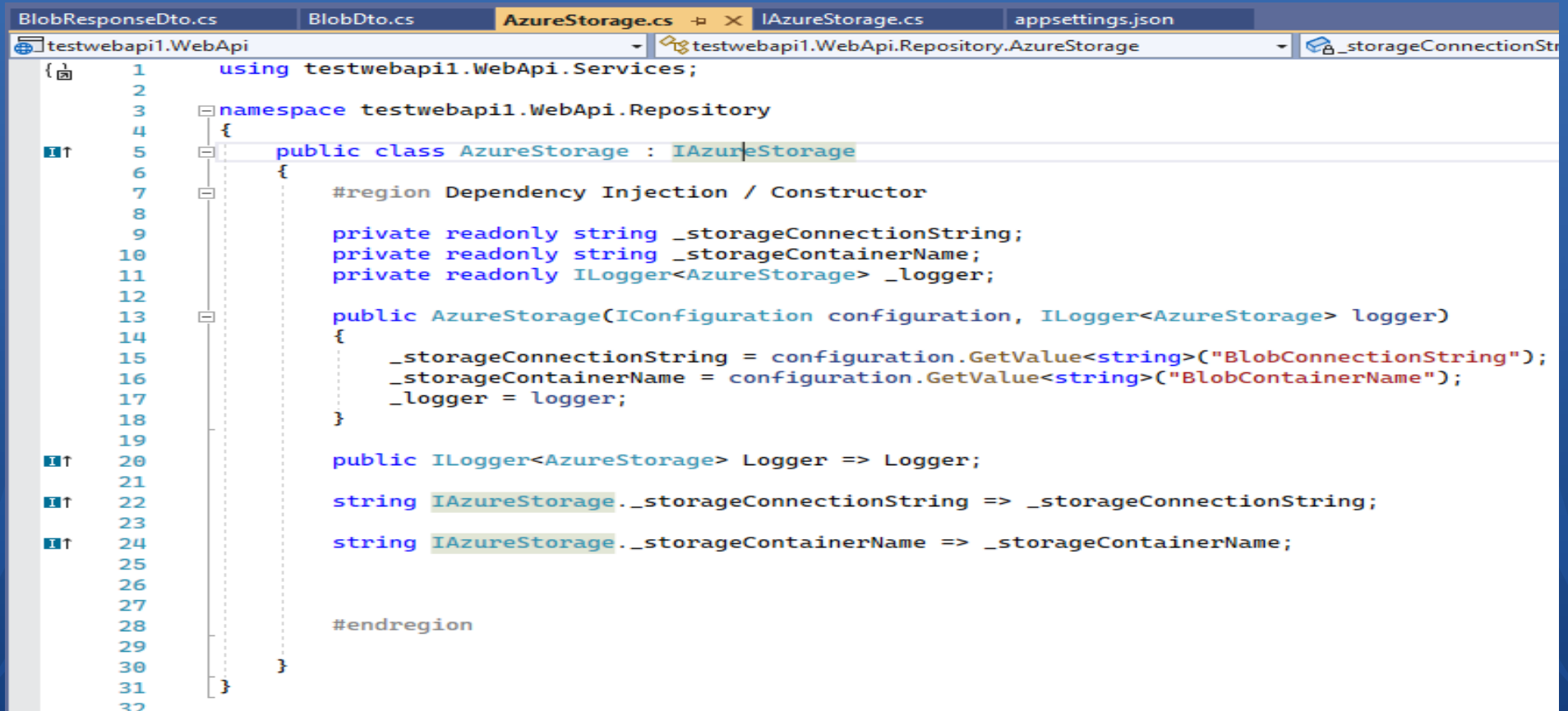
82      catch (RequestFailedException ex)
83      {
84          when (ex.ErrorCode == BlobErrorCode.BlobNotFound){
85              // Mensagem de erro no console log
86              _logger.LogError($"Arquivo {blobFilename} não encontrado.");
87          }
88          // Se o arquivo não existe, logo o retorno é nulo
89          return null;
90      }
91      /// <summary>
92      /// Este método deleta o nome do arquivo especificado
93      /// </summary>
94      /// <param name="blobFilename">Nome do Arquivo</param>
95      /// <returns>Status do Blob</returns>
96      public async Task<BlobResponseDto> DeleteAsync(string blobFilename){
97          BlobContainerClient client = new BlobContainerClient(_storageConnectionString, _storageContainerName);
98          BlobClient file = client.GetBlobClient(blobFilename);
99          try{
100              // Arquivos deletados
101              await file.DeleteAsync();
102          }
103          catch (RequestFailedException ex)
104          {
105              when (ex.ErrorCode == BlobErrorCode.BlobNotFound)
106              {
107                  // Se o arquivo não existe, logo o método requerido retornará no console
108                  _logger.LogError($"Arquivo {blobFilename} não encontrado.");
109                  return new BlobResponseDto { Error = true, Status = $"Arquivo com nome {blobFilename} não encontrado." };
110              }
111              // Retorno do novo BlobResponseDto do método requisitado
112              return new BlobResponseDto { Error = false, Status = $"Arquivo: {blobFilename} foi excluído com sucesso." };
113          }
114      }
115      /// <summary>
116      /// Este método retorna todos os arquivos localizados no container
117      /// </summary>
118      /// <returns>Blobs Listados</returns>
119      public async Task<List<BlobDto>> ListAsync() {
120          //Referência do container nomeado no 'appsettings.json'
121          BlobContainerClient container = new BlobContainerClient(_storageConnectionString, _storageContainerName);
122          // Criação de novos objetos listados
123          List<BlobDto> files = new List<BlobDto>();
```

## 12.2) Programa e seus Arquivos



```
124     await foreach (BlobItem file in container.GetBlobsAsync()) {
125         // Adicione cada arquivo recuperado do contêiner de armazenamento à lista de arquivos criando um objeto BlobDto
126         string uri = container.Uri.ToString();
127         var name = file.Name;
128         var fullUri = $"{uri}/{name}";
129         files.Add(new BlobDto{
130             Uri = fullUri,
131             Name = name,
132             ContentType = file.Properties.ContentType
133         });
134     }
135     // Retorno dos metodos requisitados de todos os arquivos
136     return files;
137 }
138 }
139 internal class _logger{
140     internal static void LogError(string v){
141         throw new NotImplementedException();
142     }
143 }
144 }
145
```

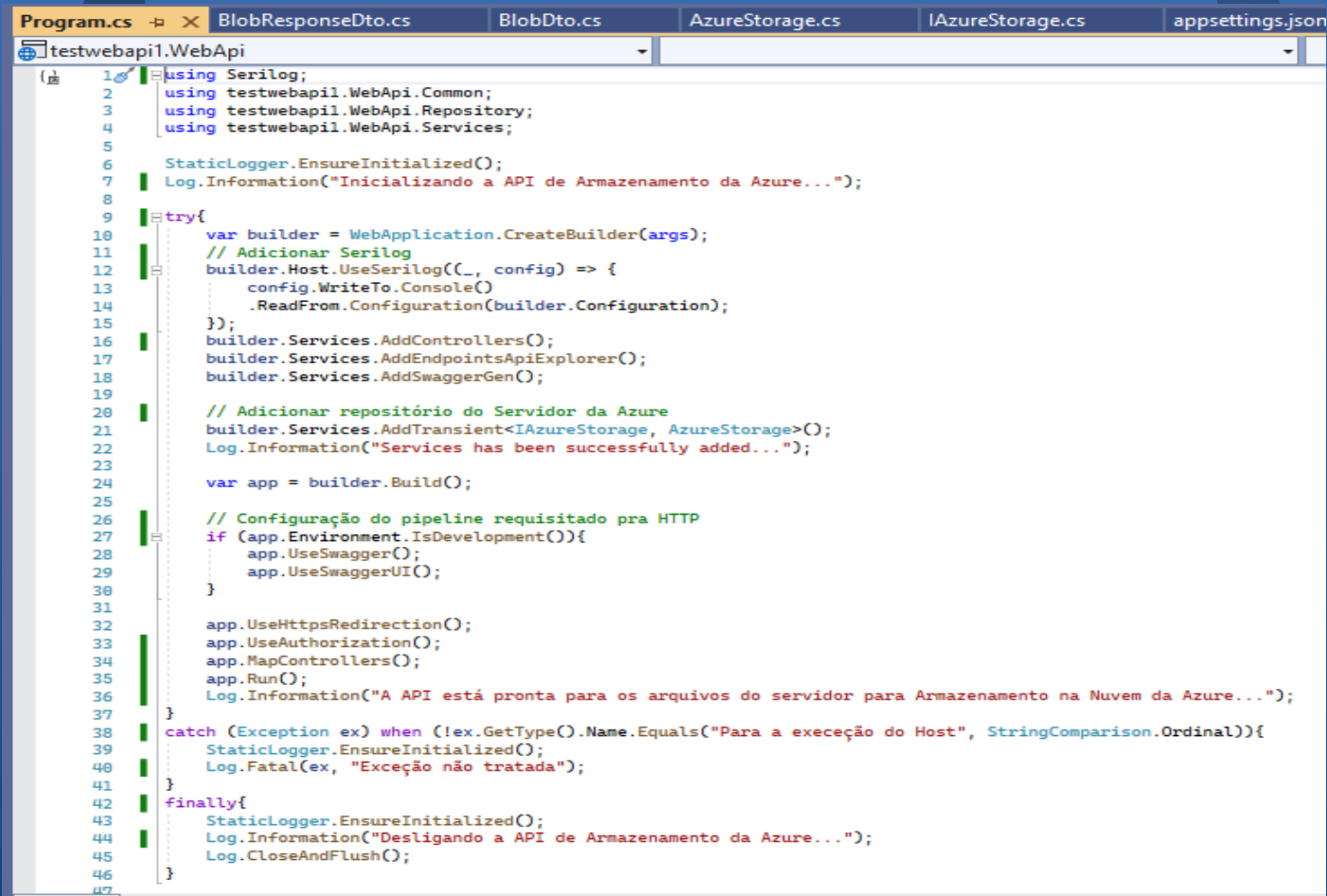
## 12.2) Programa e seus Arquivos



```
BlobResponseDto.cs  BlobDto.cs  AzureStorage.cs  IAzureStorage.cs  appsettings.json
testwebapi1.WebApi  testwebapi1.WebApi.Repository.AzureStorage  _storageConnectionStr

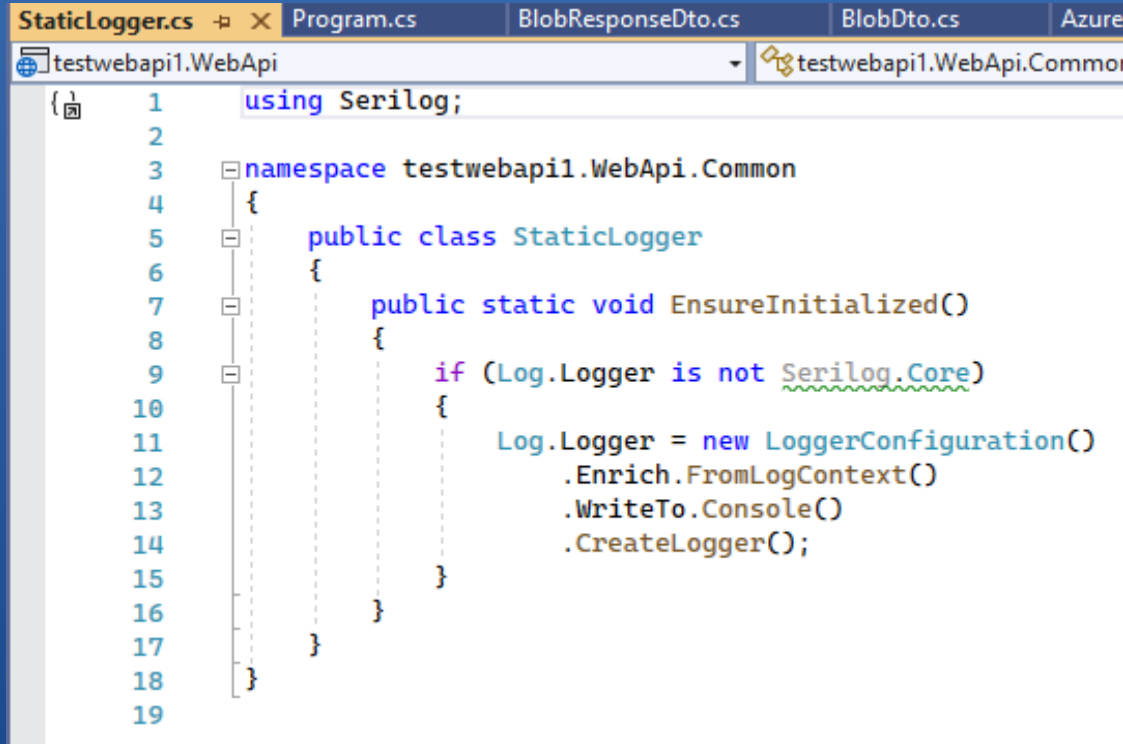
1  using testwebapi1.WebApi.Services;
2
3  namespace testwebapi1.WebApi.Repository
4  {
5      public class AzureStorage : IAzureStorage
6      {
7          #region Dependency Injection / Constructor
8
9          private readonly string _storageConnectionString;
10         private readonly string _storageContainerName;
11         private readonly ILogger<AzureStorage> _logger;
12
13         public AzureStorage(IConfiguration configuration, ILogger<AzureStorage> logger)
14         {
15             _storageConnectionString = configuration.GetValue<string>("BlobConnectionString");
16             _storageContainerName = configuration.GetValue<string>("BlobContainerName");
17             _logger = logger;
18         }
19
20         public ILogger<AzureStorage> Logger => _logger;
21
22         string IAzureStorage._storageConnectionString => _storageConnectionString;
23
24         string IAzureStorage._storageContainerName => _storageContainerName;
25
26
27
28         #endregion
29     }
30
31
32
```

## 12.2) Programa e seus Arquivos



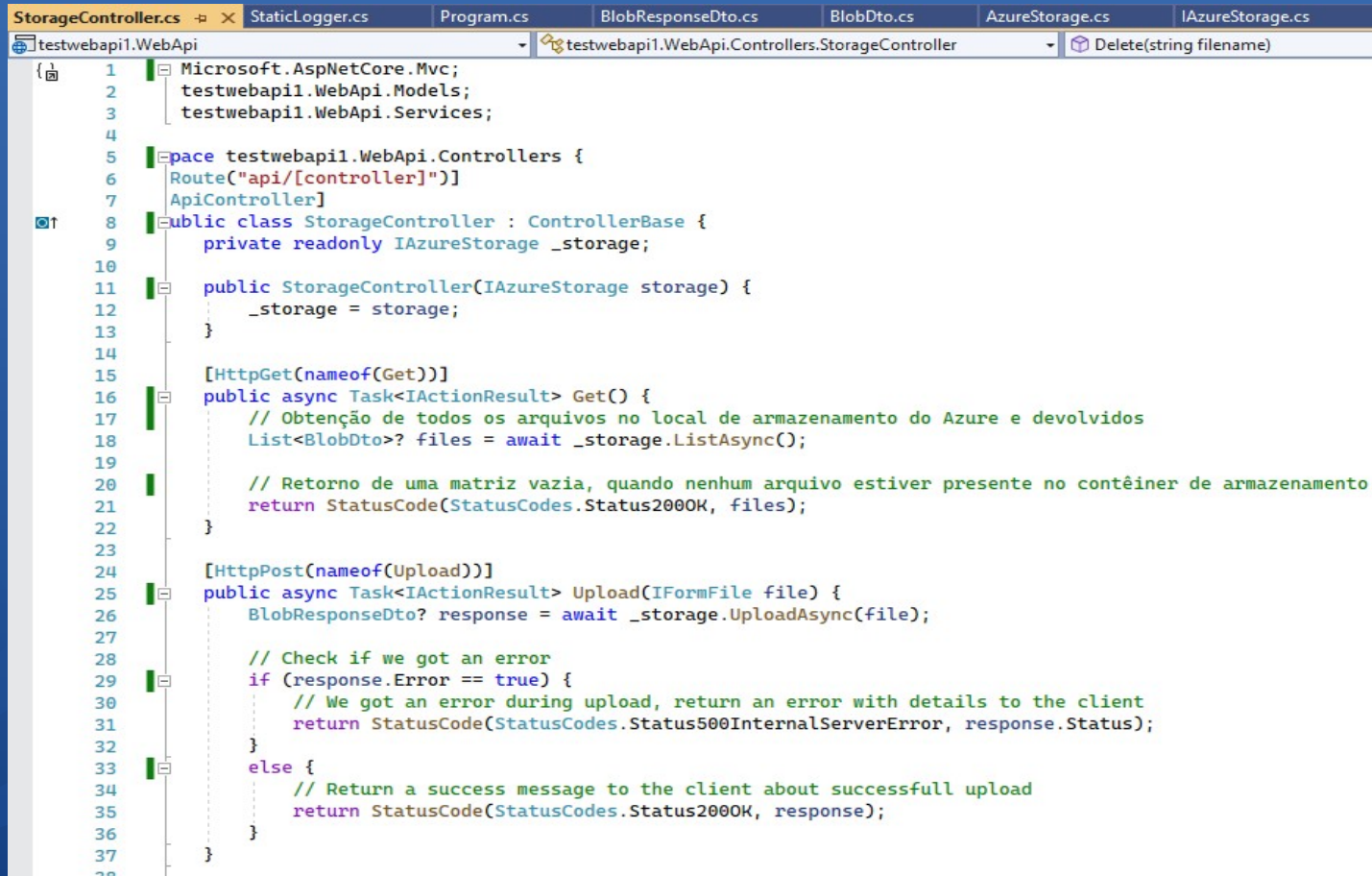
```
Program.cs  BlobResponseDto.cs  BlobDto.cs  AzureStorage.cs  IAzureStorage.cs  appsettings.json
testwebapi1.WebApi
1 using Serilog;
2 using testwebapi1.WebApi.Common;
3 using testwebapi1.WebApi.Repository;
4 using testwebapi1.WebApi.Services;
5
6 StaticLogger.EnsureInitialized();
7 Log.Information("Iniciando a API de Armazenamento da Azure...");
8
9 try{
10     var builder = WebApplication.CreateBuilder(args);
11     // Adicionar Serilog
12     builder.Host.UseSerilog( (_, config) => {
13         config.WriteTo.Console()
14             .ReadFrom.Configuration(builder.Configuration);
15     });
16     builder.Services.AddControllers();
17     builder.Services.AddEndpointsApiExplorer();
18     builder.Services.AddSwaggerGen();
19
20     // Adicionar repositório do Servidor da Azure
21     builder.Services.AddTransient<IAzureStorage, AzureStorage>();
22     Log.Information("Services has been successfully added...");
23
24     var app = builder.Build();
25
26     // Configuração do pipeline requisitado pra HTTP
27     if (app.Environment.IsDevelopment()){
28         app.UseSwagger();
29         app.UseSwaggerUI();
30     }
31
32     app.UseHttpsRedirection();
33     app.UseAuthorization();
34     app.MapControllers();
35     app.Run();
36     Log.Information("A API está pronta para os arquivos do servidor para Armazenamento na Nuvem da Azure...");
37 }
38 catch (Exception ex) when (!ex.GetType().Name.Equals("Para a execução do Host", StringComparison.Ordinal)){
39     StaticLogger.EnsureInitialized();
40     Log.Fatal(ex, "Exceção não tratada");
41 }
42 finally{
43     StaticLogger.EnsureInitialized();
44     Log.Information("Desligando a API de Armazenamento da Azure...");
45     Log.CloseAndFlush();
46 }
47
```

## 12.2) Programa e seus Arquivos



```
1  using Serilog;
2
3  namespace testwebapi1.WebApi.Common
4  {
5      public class StaticLogger
6      {
7          public static void EnsureInitialized()
8          {
9              if (Log.Logger is not Serilog.Core)
10             {
11                 Log.Logger = new LoggerConfiguration()
12                     .Enrich.FromLogContext()
13                     .WriteTo.Console()
14                     .CreateLogger();
15             }
16         }
17     }
18 }
19
```

## 12.2) Programa e seus Arquivos

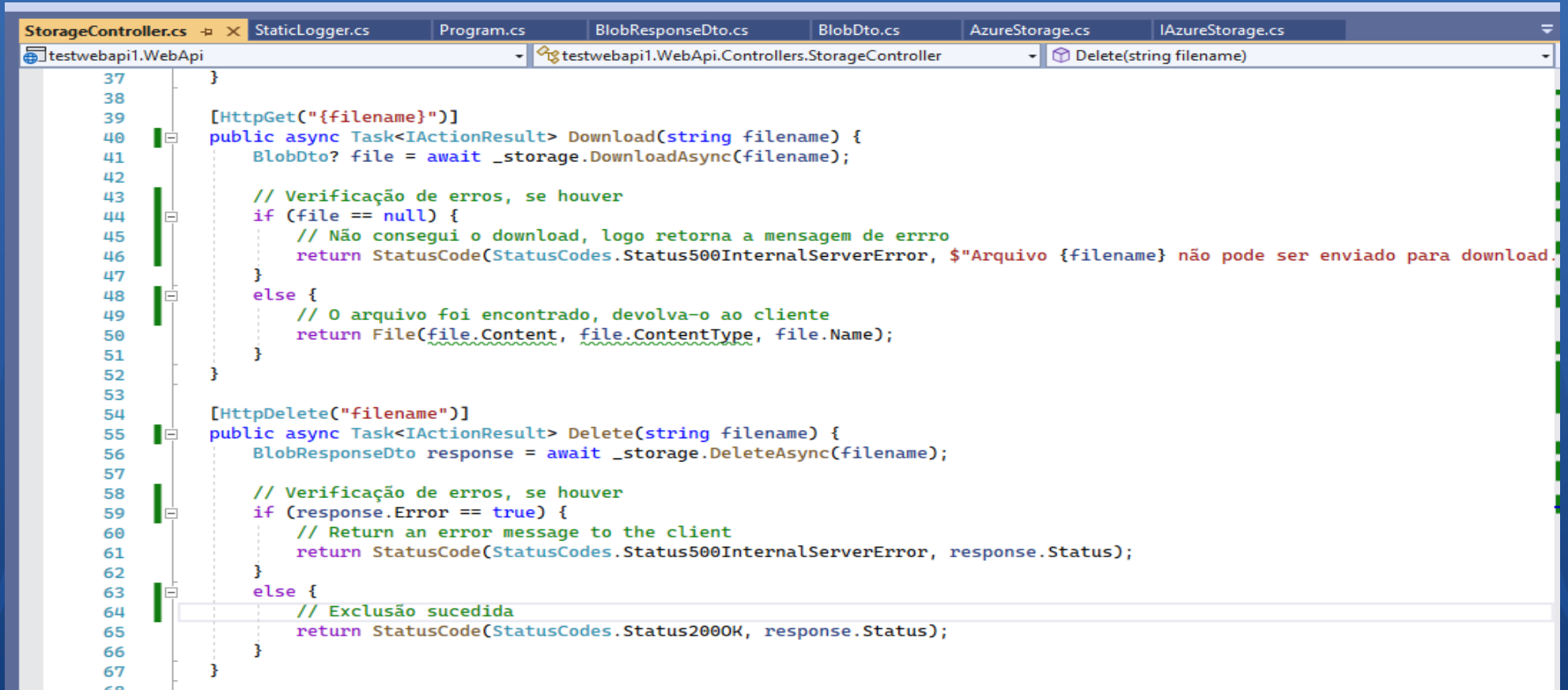


```
StorageController.cs  StaticLogger.cs  Program.cs  BlobResponseDto.cs  BlobDto.cs  AzureStorage.cs  IAzureStorage.cs
testwebapi1.WebApi  testwebapi1.WebApi.Controllers.StorageController  Delete(string filename)

1  Microsoft.AspNetCore.Mvc;
2  testwebapi1.WebApi.Models;
3  testwebapi1.WebApi.Services;
4
5  namespace testwebapi1.WebApi.Controllers {
6      Route("api/[controller]")
7      ApiController
8  public class StorageController : ControllerBase {
9      private readonly IAzureStorage _storage;
10
11     public StorageController(IAzureStorage storage) {
12         _storage = storage;
13     }
14
15     [HttpGet(nameof(Get))]
16     public async Task<IActionResult> Get() {
17         // Obtenção de todos os arquivos no local de armazenamento do Azure e devolvidos
18         List<BlobDto>? files = await _storage.ListAsync();
19
20         // Retorno de uma matriz vazia, quando nenhum arquivo estiver presente no contêiner de armazenamento
21         return StatusCode(StatusCodes.Status200OK, files);
22     }
23
24     [HttpPost(nameof(Upload))]
25     public async Task<IActionResult> Upload(IFormFile file) {
26         BlobResponseDto? response = await _storage.UploadAsync(file);
27
28         // Check if we got an error
29         if (response.Error == true) {
30             // We got an error during upload, return an error with details to the client
31             return StatusCode(StatusCodes.Status500InternalServerError, response.Status);
32         }
33         else {
34             // Return a success message to the client about successfull upload
35             return StatusCode(StatusCodes.Status200OK, response);
36         }
37     }
38 }
```



## 12.2) Programa e seus Arquivos

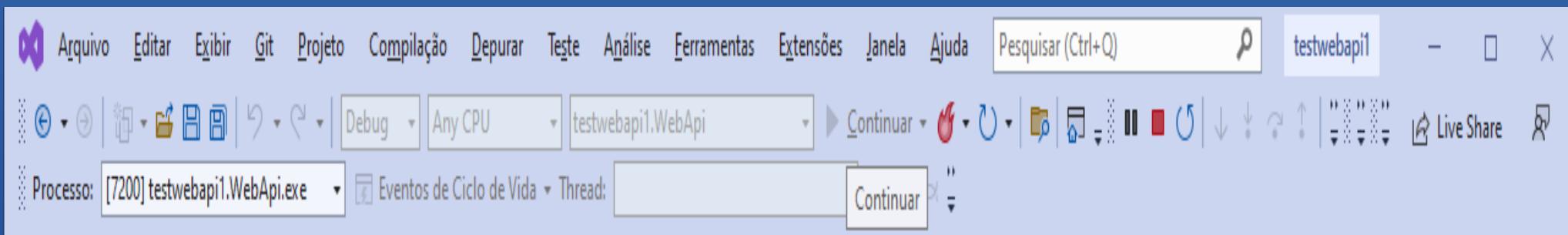


```
StorageController.cs x StaticLogger.cs Program.cs BlobResponseDto.cs BlobDto.cs AzureStorage.cs IAzureStorage.cs
testwebapi1.WebApi testwebapi1.WebApi.Controllers.StorageController Delete(string filename)

37     }
38
39     [HttpGet("{filename}")]
40     public async Task<IActionResult> Download(string filename) {
41         BlobDto? file = await _storage.DownloadAsync(filename);
42
43         // Verificação de erros, se houver
44         if (file == null) {
45             // Não consegui o download, logo retorna a mensagem de erro
46             return StatusCode(StatusCode.Status500InternalServerError, $"Arquivo {filename} não pode ser enviado para download.");
47         }
48         else {
49             // O arquivo foi encontrado, devolva-o ao cliente
50             return File(file.Content, file.ContentType, file.Name);
51         }
52     }
53
54     [HttpDelete("filename")]
55     public async Task<IActionResult> Delete(string filename) {
56         BlobResponseDto response = await _storage.DeleteAsync(filename);
57
58         // Verificação de erros, se houver
59         if (response.Error == true) {
60             // Return an error message to the client
61             return StatusCode(StatusCode.Status500InternalServerError, response.Status);
62         }
63         else {
64             // Exclusão sucedida
65             return StatusCode(StatusCode.Status200OK, response.Status);
66         }
67     }
68 }
```



## 12.3) Execução

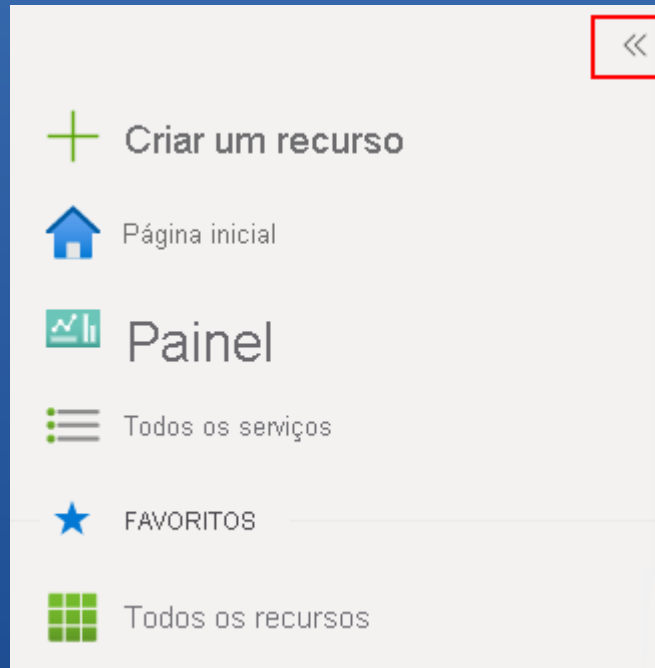


## 12.3) Execução

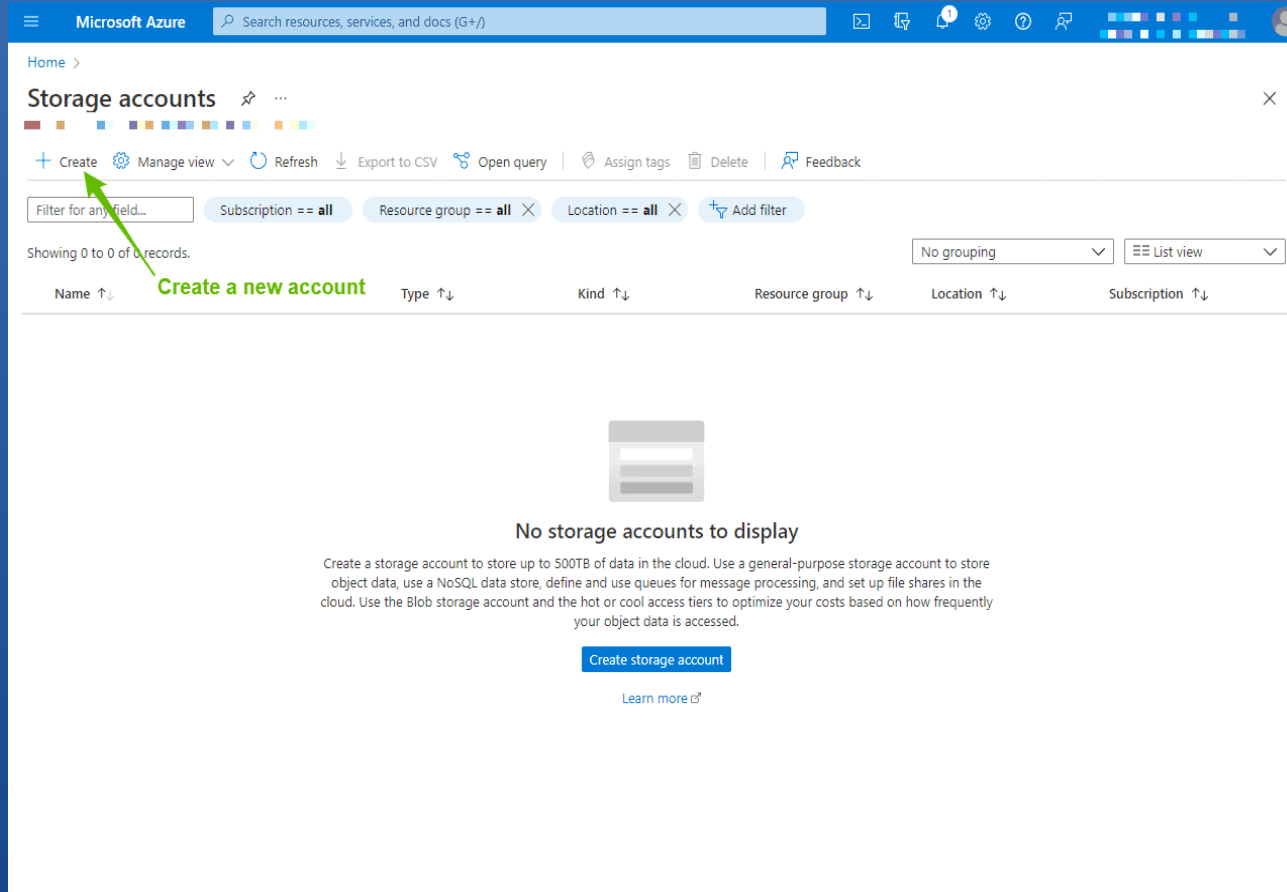
C:\Users\Novo\Desktop\testwebapi1\bin\Debug\net6.0\testwebapi1.WebApi.exe

```
[11:00:12 INF] Inicializando a API de Armazenamento da Azure...
[11:00:15 INF] Services has been successfully added...
[11:00:16 INF] Now listening on: https://localhost:7247
[11:00:16 INF] Now listening on: http://localhost:5247
[11:00:16 INF] Application started. Press Ctrl+C to shut down.
[11:00:16 INF] Hosting environment: Development
[11:00:16 INF] Content root path: C:\Users\Novo\Desktop\testwebapi1\
[11:00:18 INF] Request starting HTTP/2 GET https://localhost:7247/swagger/index.html - -
[11:00:19 INF] Request finished HTTP/2 GET https://localhost:7247/swagger/index.html - - - 200 - text/html; charset=utf-8 853.9968ms
[11:00:19 INF] Request starting HTTP/2 GET https://localhost:7247/_vs/browserLink - -
[11:00:19 INF] Request starting HTTP/2 GET https://localhost:7247/_framework/aspnetcore-browser-refresh.js - -
[11:00:19 INF] Request finished HTTP/2 GET https://localhost:7247/_framework/aspnetcore-browser-refresh.js - - - 200 12006 application/javascript; charset=utf-8 9.3033ms
[11:00:19 INF] Request finished HTTP/2 GET https://localhost:7247/_vs/browserLink - - - 200 - text/javascript; charset=UTF-8 210.9972ms
[11:00:20 INF] Request starting HTTP/2 GET https://localhost:7247/swagger/favicon-16x16.png - -
[11:00:20 INF] Sending file. Request path: '/favicon-16x16.png'. Physical path: 'N/A'
[11:00:20 INF] Request finished HTTP/2 GET https://localhost:7247/swagger/favicon-16x16.png - - - 0 665 image/png 12.8413ms
[11:00:20 INF] Request starting HTTP/2 GET https://localhost:7247/swagger/v1/swagger.json - -
[11:00:20 INF] Request finished HTTP/2 GET https://localhost:7247/swagger/v1/swagger.json - - - 200 - application/json; charset=utf-8 120.1760ms
```

## 12.4) Contêiner no Portal da Azure



# 12.4) Contêiner no Portal da Azure



The screenshot displays the Microsoft Azure portal interface. At the top, the navigation bar includes the 'Microsoft Azure' logo, a search bar, and various utility icons. The main content area is titled 'Storage accounts' and features a '+ Create' button, which is highlighted by a green arrow and the text 'Create a new account'. Below the button, there are filters for 'Subscription == all', 'Resource group == all', and 'Location == all', along with an 'Add filter' button. The table below the filters shows 'Showing 0 to 0 of 0 records.' and a 'No grouping' dropdown. The table headers are 'Name', 'Type', 'Kind', 'Resource group', 'Location', and 'Subscription'. The main content area is empty, displaying a message: 'No storage accounts to display'. Below this message, there is a 'Create storage account' button and a 'Learn more' link.

Home >

## Storage accounts

+ Create Manage view Refresh Export to CSV Open query Assign tags Delete Feedback

Filter for any field... Subscription == all Resource group == all Location == all Add filter

Showing 0 to 0 of 0 records. No grouping List view

Name	Type	Kind	Resource group	Location	Subscription
------	------	------	----------------	----------	--------------


**No storage accounts to display**

Create a storage account to store up to 500TB of data in the cloud. Use a general-purpose storage account to store object data, use a NoSQL data store, define and use queues for message processing, and set up file shares in the cloud. Use the Blob storage account and the hot or cool access tiers to optimize your costs based on how frequently your object data is accessed.

[Create storage account](#)

[Learn more](#)

# 12.4) Contêiner no Portal da Azure

 Microsoft Azure

[Home](#) > [Storage accounts](#) >

## Create a storage account ...


[Basics](#) [Advanced](#) [Networking](#) [Data protection](#) [Encryption](#) [Tags](#) [Review + create](#)

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 \*



Resource group \*

Test

Create new

### Instance details

If you need to create a legacy storage account type, please click [here](#).

Storage account name ⓘ \*

testwebapi

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)

## 12.4) Contêiner no Portal da Azure

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

Home > Storage accounts >

### Create a storage account

Validation passed

Basics Advanced Networking Data protection Encryption Tags Review + create

**Basics**

Subscription	Test
Resource Group	westeurope
Location	testwebapi
Storage account name	Resource manager
Deployment model	Standard
Performance	Locally-redundant storage (LRS)
Replication	

**Advanced**

Secure transfer	Enabled
Allow storage account key access	Enabled
Allow cross-tenant replication	Enabled
Default to Azure Active Directory authorization in the Azure portal	Disabled
Blob public access	Enabled
Minimum TLS version	Version 1.2
Enable hierarchical namespace	Disabled
Enable network file system v3	Disabled
Access tier	Hot
Enable SFTP	Disabled
Large file shares	Disabled

**Networking**

Network connectivity	Public endpoint (all networks)
Default routing tier	Microsoft network routing

**Data protection**

	Disabled
--	----------

**Create storage account**

Create < Previous Next > Download a template for automation

## 12.4) Contêiner no Portal da Azure

Microsoft Azure Upgrade Search resources, services, and docs (G+/)

muriloda.408@outlook.c... DIRETÓRIO PADRÃO

### Azure services

- Create a resource
- App Configuration
- Static Web Apps
- Function App
- All resources
- Quickstart Center
- Virtual machines
- App Services
- Storage accounts
- More services

### Resources

Recent Favorite

Name	Type	Last Viewed
testwebapi1	Storage account	4 days ago
Test	Resource group	5 days ago
apialturas	Static Web App	2 weeks ago

See all

## 12.4) Contêiner no Portal da Azure

The screenshot displays the Azure Portal interface for a storage account named 'webapiazurestorage\_1648209044654'. The left sidebar shows the navigation menu with 'Containers' highlighted under 'Data storage'. The main area shows the 'Containers' view with a '+ Container' button and a table of existing containers. A 'New container' dialog is open on the right, showing the 'Name' field set to 'files-container' and the 'Public access level' set to 'Private (no anonymous access)'.

**1. Select Containers**

**2. Create new Container**

**3. Give it a name**

**New container**

Name \*  
files-container ✓

Public access level ⓘ  
Private (no anonymous access) ▼

Advanced

Encryption scope  
Select from existing account scopes ▼

☐ Use this encryption scope for all blobs in the container

☐ Enable version-level immutability support ⓘ

**i** In order to enable version-level immutability support, your storage account must have versioning turned on.



## 12.4) Contêiner no Portal da Azure

The screenshot displays the Microsoft Azure portal interface. The browser tab is titled 'testwebapi1 - Microsoft Azure'. The address bar shows the URL: <https://portal.azure.com/#@muriloda408outlook.onmicrosoft.com/resource/subscriptions/056819ce-7d4e-46ef-9950-c1a>. The page title is 'testwebapi1 | Access keys' for a 'Storage account'. The left sidebar contains a navigation menu with categories: Overview, Activity log, Tags, Diagnose and solve problems, Access Control (IAM), Data migration, Events, Storage browser, Data storage (highlighted), and Security + networking. Under 'Data storage', 'Containers' is selected. The main content area shows instructions on access keys, a search bar, and a 'Set rotation reminder' button. It lists the storage account name as 'testwebapi1' and displays two keys, 'key1' and 'key2', both rotated on 10/25/2022 (4 days ago). Each key has a 'Key' field (partially obscured by dots) and a 'Connection string' field (partially obscured by dots). Buttons for 'Hide', 'Show', and 'Rotate key' are present for each key.

Microsoft Azure

Search resources, services, and docs (G+)

Home > testwebapi1

**testwebapi1** | Access keys ☆ ...  
Storage account

Search

Overview  
Activity log  
Tags  
Diagnose and solve problems  
Access Control (IAM)  
Data migration  
Events  
Storage browser

Data storage

Containers  
File shares  
Queues  
Tables

Security + networking

Networking  
Azure CDN  
Access keys  
Shared access signature

Set rotation reminder Refresh

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 about managing storage account access keys](#)

Storage account name  
testwebapi1

**key1** Rotate key  
Last rotated: 10/25/2022 (4 days ago)  
Key  
FJl8HWivk67ihjmDGqu5G6qtFUggXRThe2qOPUleei5bzriVaNGyeekquQ9TolssB+R... Hide  
Connection string  
DefaultEndpointsProtocol=https;AccountName=testwebapi1;AccountKey=FJl8H... Hide

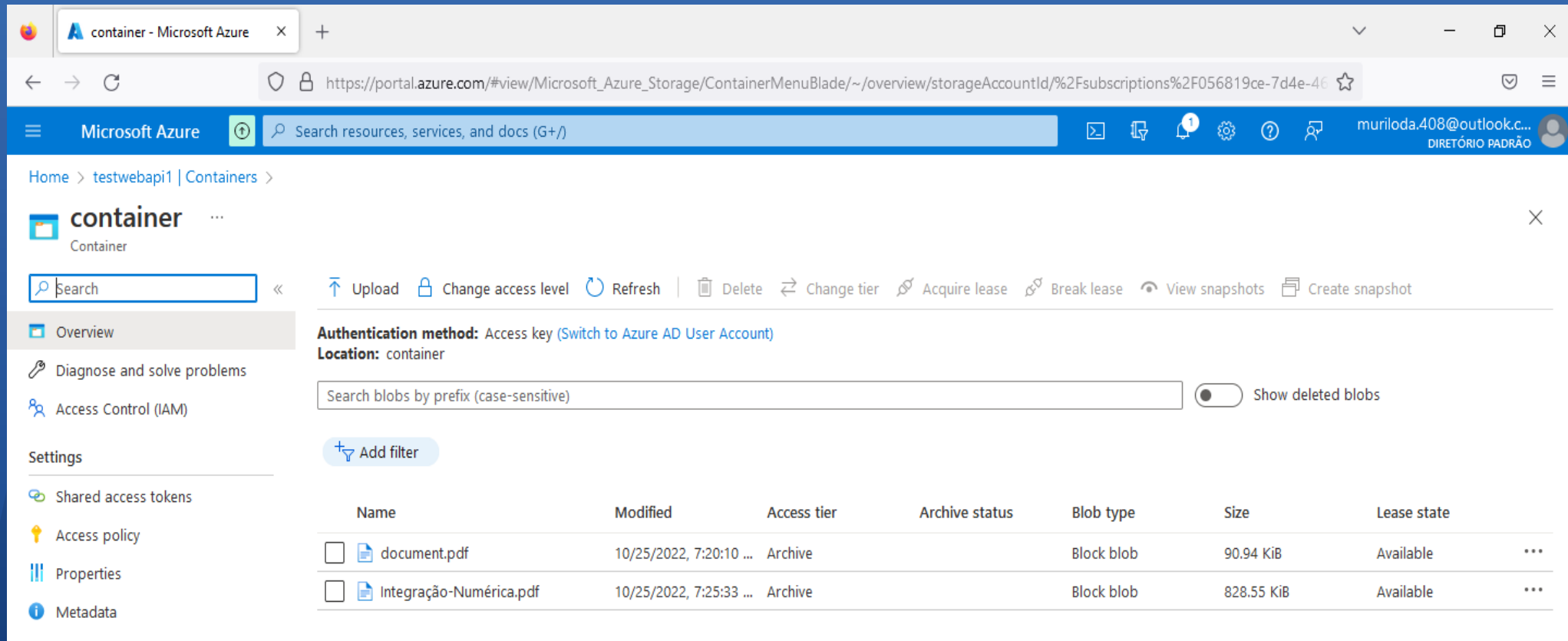
**key2** Rotate key  
Last rotated: 10/25/2022 (4 days ago)  
Key  
..... Show  
Connection string  
..... Show

# 12.4) Contêiner no Portal da Azure

The screenshot displays the Microsoft Azure portal interface. The top navigation bar includes the 'Microsoft Azure' logo, a search bar, and user information for 'muriloda.408@outlook.c...' with the role 'DIRETÓRIO PADRÃO'. The breadcrumb trail shows 'Home > testwebapi1'. The main heading is 'testwebapi1 | Containers', with 'Storage account' indicated below it. A left-hand sidebar lists various services: Overview, Activity log, Tags, Diagnose and solve problems, Access Control (IAM), Data migration, Events, Storage browser, Data storage, Containers (selected), File shares, Queues, and Tables. The main content area features a search bar labeled 'Search containers by prefix' and a toggle for 'Show deleted containers'. Below this is a table listing containers.

	Name	Last modified	Public access level	Lease state	
<input type="checkbox"/>	\$logs	10/25/2022, 3:11:29 PM	Private	Available	...
<input type="checkbox"/>	container	10/25/2022, 3:16:28 PM	Private	Available	...

## 12.4) Contêiner no Portal da Azure





The screenshot shows the Microsoft Azure portal interface. The browser tab is 'container - Microsoft Azure'. The URL is 'https://portal.azure.com/#view/Microsoft\_Azure\_Storage/ContainerMenuBlade/~/overview/storageAccountId/%2Fsubscriptions%2F056819ce-7d4e-46...'. The user is logged in as 'muriloda.408@outlook.c...' with the role 'DIRETÓRIO PADRÃO'.

The page title is 'container' with a subtitle 'Container'. The left sidebar contains the following links: Overview (selected), Diagnose and solve problems, Access Control (IAM), Settings, Shared access tokens, Access policy, Properties, and Metadata.

The main content area shows the 'Authentication method' as 'Access key (Switch to Azure AD User Account)' and the 'Location' as 'container'. There is a search bar for blobs with the placeholder text 'Search blobs by prefix (case-sensitive)'. A toggle switch for 'Show deleted blobs' is present.

A table lists the blobs in the container:

	Name	Modified	Access tier	Archive status	Blob type	Size	Lease state	
<input type="checkbox"/>	 document.pdf	10/25/2022, 7:20:10 ...	Archive		Block blob	90.94 KiB	Available	...
<input type="checkbox"/>	 Integração-Numérica.pdf	10/25/2022, 7:25:33 ...	Archive		Block blob	828.55 KiB	Available	...

## 12.5) Teste da API no Swagger

The screenshot displays the Swagger UI interface in a web browser. The browser's address bar shows the URL `https://localhost:7247/swagger/index.html`. The Swagger logo and "Supported by SMARTBEAR" are visible in the top left. A dropdown menu in the top right allows selecting a definition, with "testwebapi1.WebApi v1" currently selected. The main content area displays the API title "testwebapi1.WebApi" with version "1.0" and "OAS3" specification. Below the title, the URL `https://localhost:7247/swagger/v1/swagger.json` is shown. A section titled "Storage" is expanded, revealing four API endpoints:

- GET** `/api/Storage/Get`
- POST** `/api/Storage/Upload`
- GET** `/api/Storage/{filename}`
- DELETE** `/api/Storage/filename`

## 12.5) Teste da API no Swagger

POST

/api/Storage/Upload

Cancel

Reset

Parameters

No parameters

Request body

multipart/form-data

file

string(\$binary)

Browse...

Documento[1].docx

☐ Send empty value

Execute

Clear

Responses

Curl

```
curl -X 'POST' \
  https://localhost:7247/api/Storage/Upload \
  -H 'accept: */*' \
  -H 'Content-Type: multipart/form-data' \
  -F 'file=@Documento[1].docx;type=application/vnd.openxmlformats-officedocument.wordprocessingml.document'
```

Request URL

https://localhost:7247/api/Storage/Upload

Server response

Code

Details

200

Response body

```
{
  "status": "Arquivo Documento[1].docx Uploaded Sucedido",
  "error": false,
  "blob": {
    "uri": "https://testwebapi1.blob.core.windows.net/container/DocumentoX581X5D.docx",
    "name": "Documento[1].docx",
    "contentType": null,
    "content": null
  }
}
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Sun, 30 Oct 2022 14:30:28 GMT
server: Kestrel
x-firefox-spy: h2
```

Responses

Code

Description

Links

200

Success

No links

## 12.5) Teste da API no Swagger

Microsoft Azure

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

muriloda.408@outlook.c...  
DIRETÓRIO PADRÃO

Home >

container

Container

Search

Upload Change access level Refresh Delete Change tier Acquire lease Break lease View snapshots Create snapshot

Overview

Diagnose and solve problems

Access Control (IAM)

Settings

Shared access tokens

Access policy

Properties

Metadata

Authentication method: Access key (Switch to Azure AD User Account)

Location: container

Search blobs by prefix (case-sensitive)

Show deleted blobs

Add filter

	Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
<input type="checkbox"/>	document.pdf	10/25/2022, 7:20:10 ...	Archive		Block blob	90.94 KiB	Available ...
<input type="checkbox"/>	Documento[1].docx	10/30/2022, 11:30:28...	Hot (Inferred)		Block blob	9.02 KiB	Available ...
<input type="checkbox"/>	Integração-Numérica.pdf	10/25/2022, 7:25:33 ...	Archive		Block blob	828.55 KiB	Available ...
<input type="checkbox"/>	modelo-recibo-simples-para-imp...	10/30/2022, 11:16:25...	Archive		Block blob	9.55 KiB	Available ...

## 12.5) Teste da API no Swagger

GET

/api/Storage/Get

⌵

Parameters

Cancel

No parameters

Execute

Clear

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7247/api/Storage/Get' \
  -H 'accept: */*'
```

Request URL

https://localhost:7247/api/Storage/Get

Server response

Code

Details

200

Response body

```
[
  {
    "uri": "https://testebapi1.blob.core.windows.net/container/documento[1].docx",
    "name": "Documento[1].docx",
    "contentType": "application/octet-stream",
    "content": null
  },
  {
    "uri": "https://testebapi1.blob.core.windows.net/container/Integracao-Numérica.pdf",
    "name": "Integracao-Numérica.pdf",
    "contentType": "application/octet-stream",
    "content": null
  },
  {
    "uri": "https://testebapi1.blob.core.windows.net/container/document.pdf",
    "name": "document.pdf",
    "contentType": "application/octet-stream",
    "content": null
  },
  {
    "uri": "https://testebapi1.blob.core.windows.net/container/modelo-recibo-simples-para-impressao.docx",
    "name": "modelo-recibo-simples-para-impressao.docx",
    "contentType": "application/octet-stream",
    "content": null
  }
]
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Sun, 30 Oct 2022 14:33:42 GMT
server: Kestrel
x-firefox-spy: h2
```

Responses

Code	Description	Links
200	Success	No links

## 12.5) Teste da API no Swagger

GET

/api/Storage/{filename}

Cancel

Parameters

Name	Description
filename <span>*</span> required	
string	Documento[1].docx
(path)	

ExecuteClear

Responses

Curl

```
curl -X "GET" \
  "https://localhost:7247/api/Storage/Documento%5B1%5D.docx" \
  -H "accept: */*"
```

Request URL

```
https://localhost:7247/api/Storage/Documento%5B1%5D.docx
```

Server response

Code	Details
200	<div><div>Response body</div><div><a href="#">Download file</a></div><div>Response headers</div><div><pre>content-disposition: attachment; filename="Documento[1].docx"; filename="UTF-8'Documento%5B1%5D.docx" content-length: 9239 content-type: application/octet-stream date: Sun, 30 Oct 2022 14:30:50 GMT server: Kestrel x-firefox-spdyn: h2</pre></div></div>

Responses

Code	Description	Links
200	Success	No links



## 12.5) Teste da API no Swagger

**DELETE** /api/Storage/filename

Parameters

Cancel

Name	Description
filename	
string	
(query)	

Execute

Clear

Responses

Curl

```
curl -X 'DELETE' \
  'https://localhost:7247/api/Storage/filename?filename=document.pdf' \
  -H 'accept: */*'
```

Request URL

```
https://localhost:7247/api/Storage/filename?filename=document.pdf
```

Server response

Code	Details
200	<div>Response body</div> <div>Arquivo: document.pdf foi excluído com sucesso.</div> <div>Response headers</div> <pre>content-type: text/plain; charset=utf-8 date: Sun, 30 Oct 2022 14:36:19 GMT server: Restrel x-firefox-spdy: h2</pre>

Responses

Code	Description	Links
200	Success	No links

## 12.5) Teste da API no Swagger

container - Microsoft Azure

https://portal.azure.com/#view/Microsoft\_Azure\_Storage/ContainerMenuBlade/~/overview/storageAccountId/%2F056819ce-7d...

Microsoft Azure Upgrade Search resources, services, and docs (G+)

Home > container Container

Search

Upload Change access level Refresh Delete Change tier Acquire lease Break lease View

**Authentication method:** Access key (Switch to Azure AD User Account)  
**Location:** container

Search blobs by prefix (case-sensitive) Show deleted blobs

Add filter

	Name	Modified	Access tier	Archive status	Blob type	Size	Lease state	
<input type="checkbox"/>	Documento[1].docx	10/30/2022, 11:30:28...	Hot (Inferred)		Block blob	9.02 KiB	Available	...
<input type="checkbox"/>	Integração-Numérica.pdf	10/25/2022, 7:25:33 ...	Archive		Block blob	828.55 KiB	Available	...
<input type="checkbox"/>	modelo-recibo-simples-para-imp...	10/30/2022, 11:16:25...	Archive		Block blob	9.55 KiB	Available	...

**RS868.82 credit remaining**  
Subscription 'Azure subscription 1' has a remaining credit of R\$868.82.  
[Upgrade to a Pay-As-You-Go subscription.](#)

Overview  
Diagnose and solve problems  
Access Control (IAM)  
Settings  
Shared access tokens  
Access policy  
Properties  
Metadata

## 13) CONCLUSÃO

- Em fim, o testes de API's dependem de três ferramentas interligadas que são IDE para inserir códigos, contêineres e serviços em Nuvem.
- Também são dependentes dos pacotes instalados na IDE os contêineres e a nuvem.
- Para a compilação da API ocorrer, não deve haver falhas nos códigos de qualquer arquivo do pacote de todo o projeto.
- Os recursos da IDE facilitaram para que todo o projeto tivesse sucesso.

# 14) REFERÊNCIAS

<https://visualstudio.microsoft.com/pt-br/vs/getting-started/>, acessado em 23/10/2022 às 19:40

<https://learn.microsoft.com/pt-br/visualstudio/get-started/csharp/visual-studio-ide?view=vs-2022>, acessado em 29/10/2022 às 19:03

<https://visualstudio.microsoft.com/pt-br/vs/features/net-development/>, acessado em 29/10/2022 às 19:13

<https://www.c-sharpcorner.com/article/azure-blob-storage-with-asp-net-core-web-api-net-core-3-1/>, acessado em 23/10/2022 às 20:54

<https://learn.microsoft.com/pt-br/azure/azure-portal/azure-portal-overview>, acessado em 23/10/2022 às 19:49

<https://code-maze.com/azure-blob-storage-with-asp-net-core-and-angular/>, acessado em 23/10/2022 às 20:52

<https://blog.christian-schou.dk/how-to-use-azure-blob-storage-with-asp-net-core/>, acessado em 23/10/2022 às 21:05

<https://henriquemaui.net/azure-blob-storage-no-net-6-0/>, acessado em 23/10/2022 às 20:32

<http://www2.decom.ufop.br/terralab/documentando-sua-api-rest-com-swagger/>, acessado em 29/10/2022 às 22:02

## 15) PERGUNTAS

- Quais os recursos da Microsoft foram utilizados? Por quê?
- O projeto foi desenvolvido com quais linguagens de programação? Por quê?