API

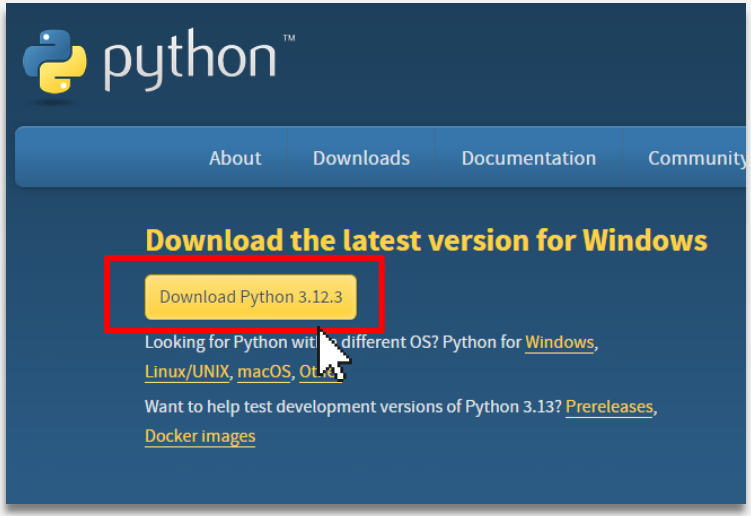
Share Point Online
connect Azure Blob Storage

by: Ítalo Magalhães
contact: linkedin.com/in/magalha7/

SUMÁRIO

1. Pré requisitos para acessar a API

Antes de começar a acessar a API você deve garantir que na sua máquina esteja instalado a versão mais recente do Python. [Clique aqui]

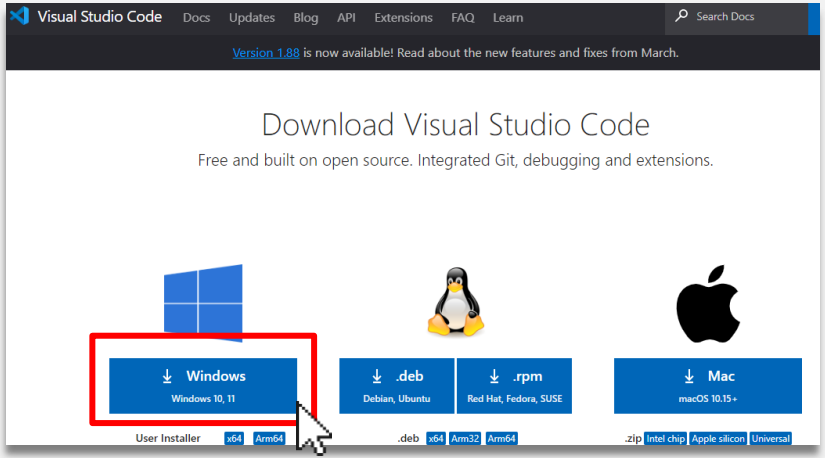


1) Clique em Download Python...

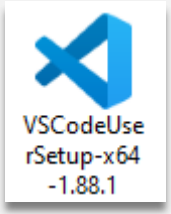
2) Execute o instalador e clique em **Next** para todas etapas

1. Pré requisitos para acessar a API

Certifique-se também de possui instalado o **VS Code.** Para mais informações baixe e instale-o [clicando aqui]



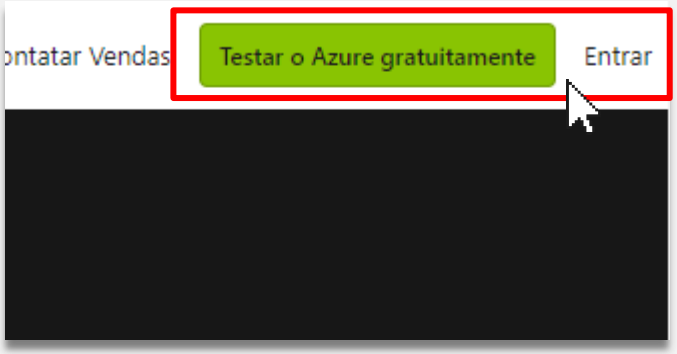
1) Clique em Download para o Windows



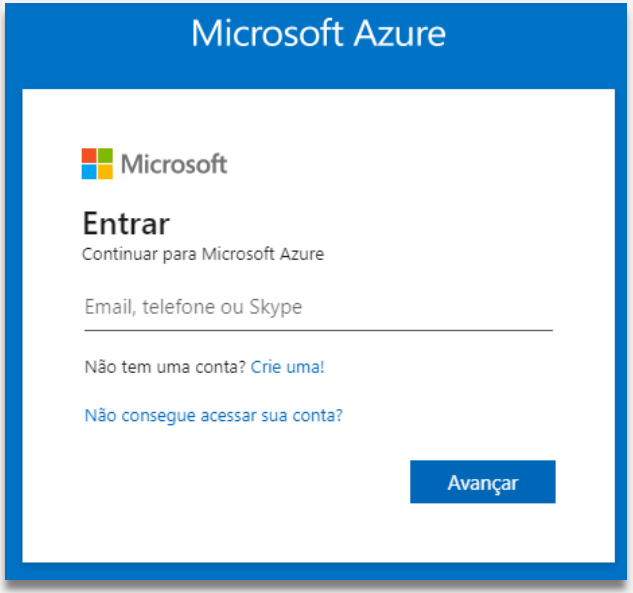
2) Execute o instalador e clique em **Next** para todas etapas

1. Pré requisitos para acessar a API

Certifique-se de possuir uma conta no Azure Cloud. Para mais informações acesse: [link]



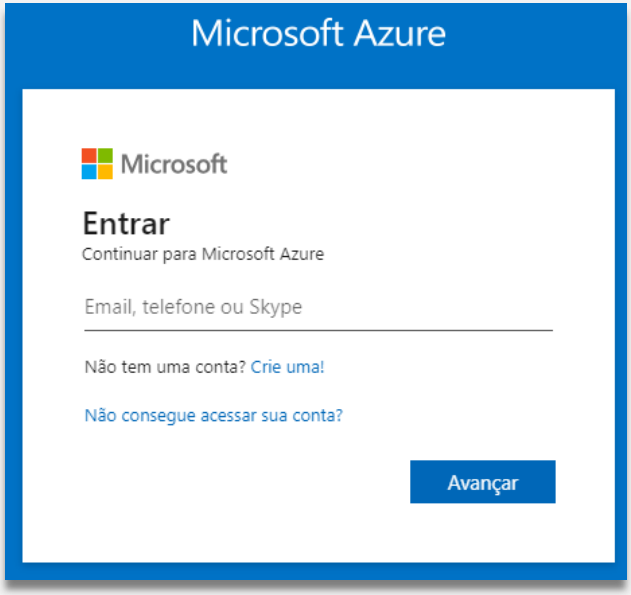
1) Clique em Entrar caso já tenha uma conta ou clique em Testar para criar uma conta



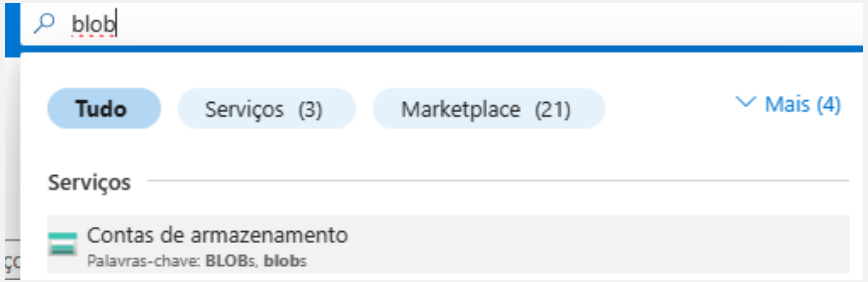
2) Ao clicar em entrar, preencha sua informações e faça Login no portal

1. Pré requisitos para acessar a API

Certifique-se de possuir um serviço ativo do Azure Blob Storage. Para mais informações acesse: [link]



1) Acesse sua conta no **Azure Cloud**

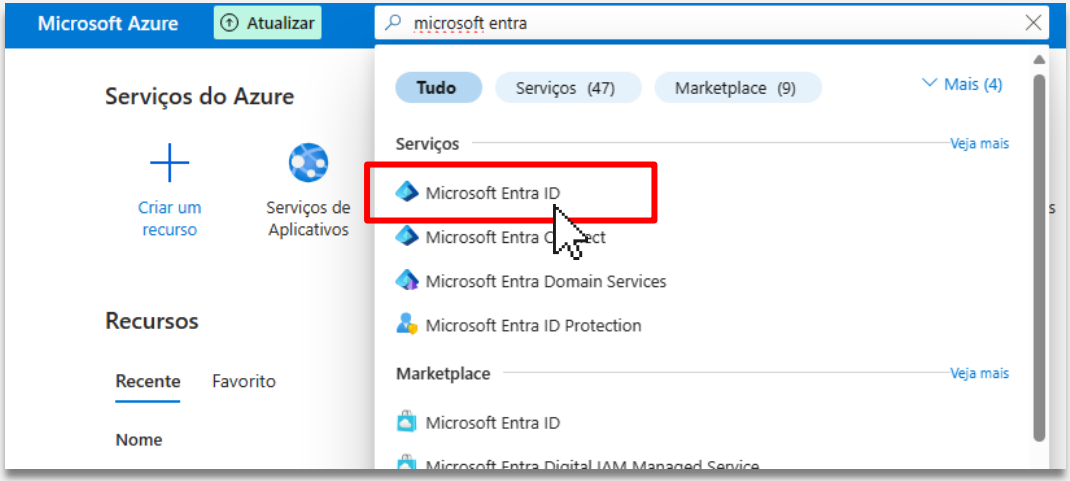


2) Navegue até **Contas de Armazenamento** caso possuir o serviço ativo ou crie um serviço

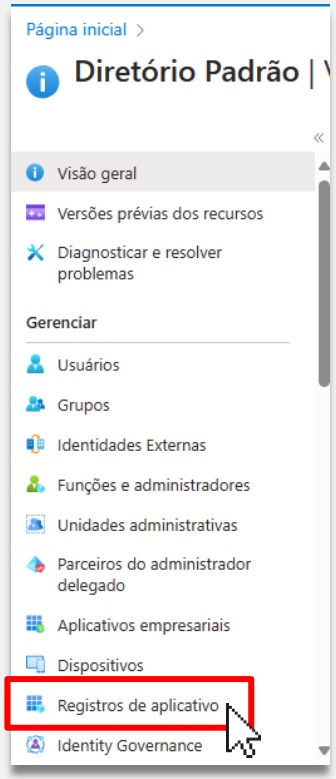
SUMÁRIO

2. Configurando um aplicativo no Azure AD

Ao acessar seu painel no Azure Cloud, siga as etapas abaixo:



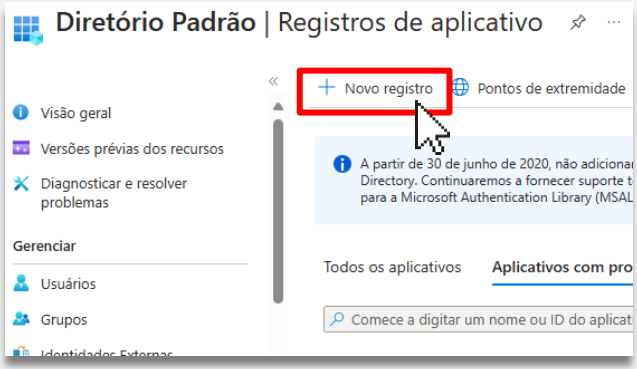
1) Navegue até **Microsoft Entra ID**



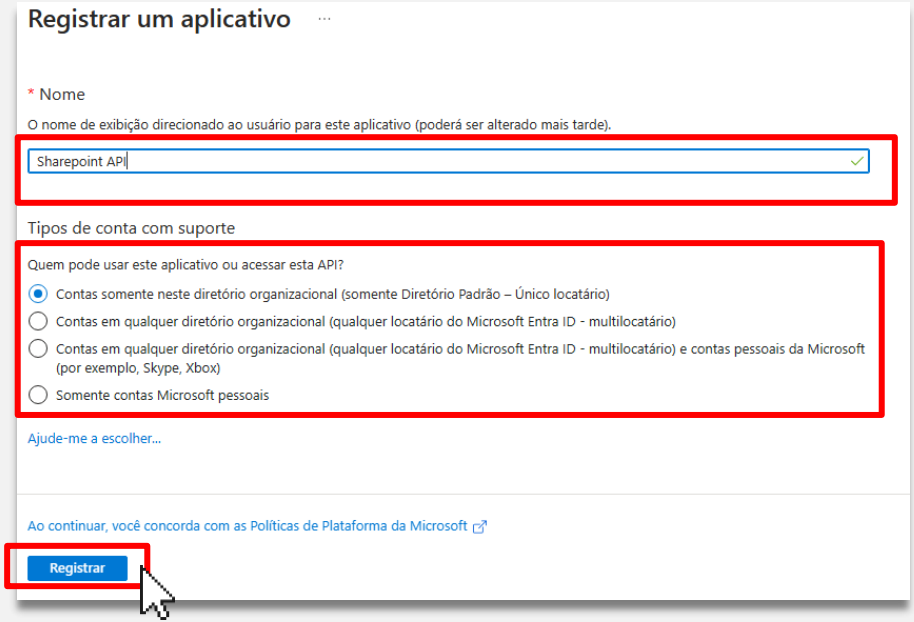
2) Clique em **Registro de aplicativo**

2. Configurando um aplicativo no Azure AD

Registre um aplicativo:



3) Clique em **Novo Registro**

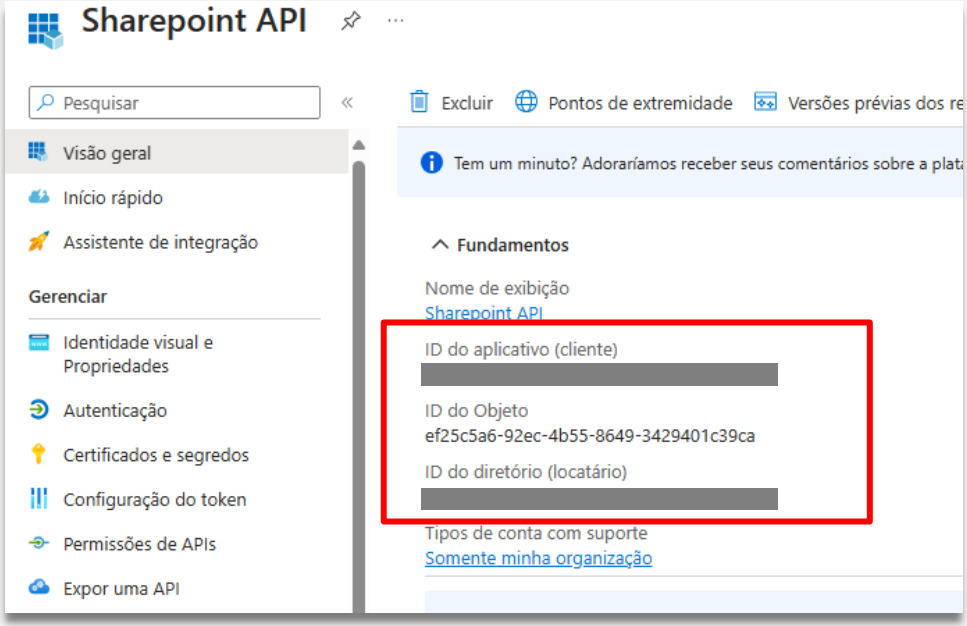


4) Preencha o nome de seu aplicativo selecione quem poderá acessar seu aplicativo e clique em **Registrar**

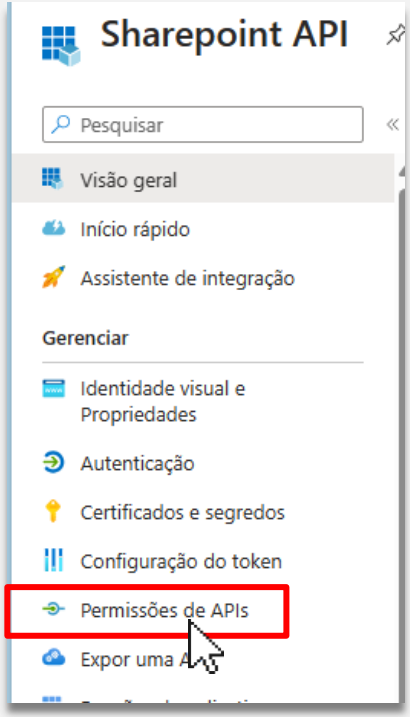
OBS: O tipo de acesso irá depender do uso, recomenda-se para um único usuário a opção de único locatário caso seja um email de uma organização recomenda-se multilocatario

2. Configurando um aplicativo no Azure AD

Conceda permissões para o aplicativo criado:

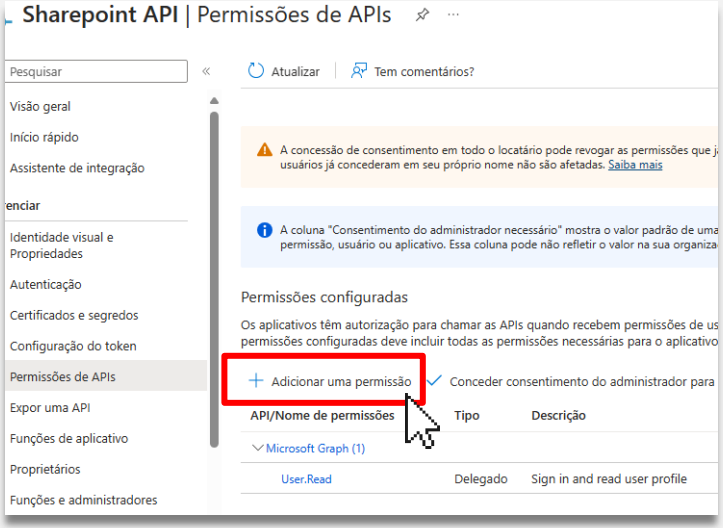


5) Anote o **ID do Apllicativo (cliente)** e **o ID do diretório (locatário)**

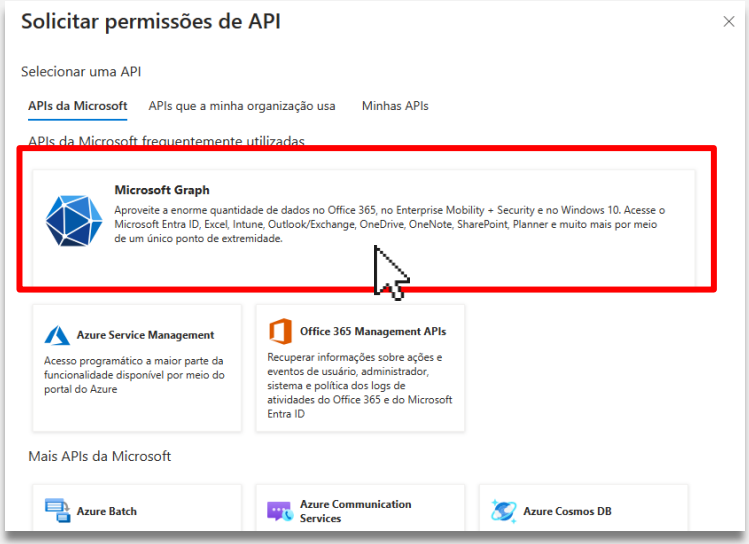


6) Clique em **Permissões de APIS**

2. Configurando um aplicativo no Azure AD

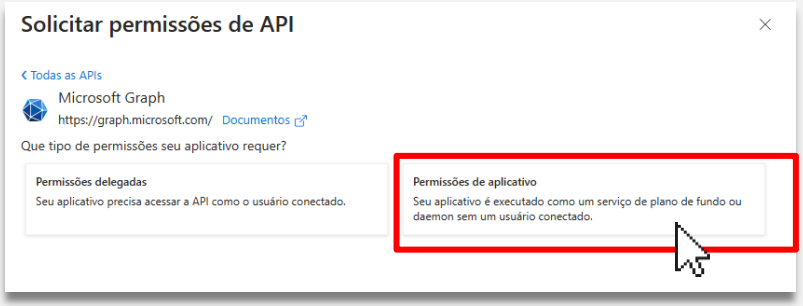


6) Clique em **Adicionar uma permissão**

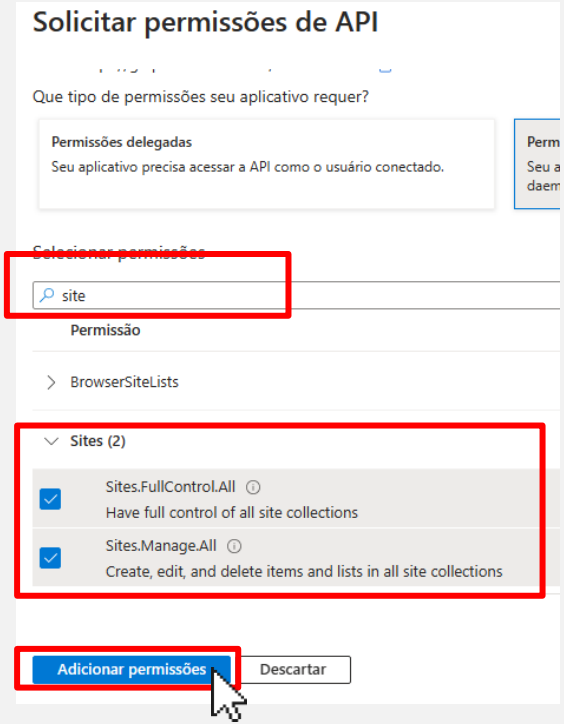


7) Nessa tela iremos utilizar a permissão do **Microsoft Graph** que é uma API da Microsoft que permite nos conectarmos a diversos serviços (ex: SharePoint, OneDrive etc) para mais informações **clique aqui**

2. Configurando um aplicativo no Azure AD

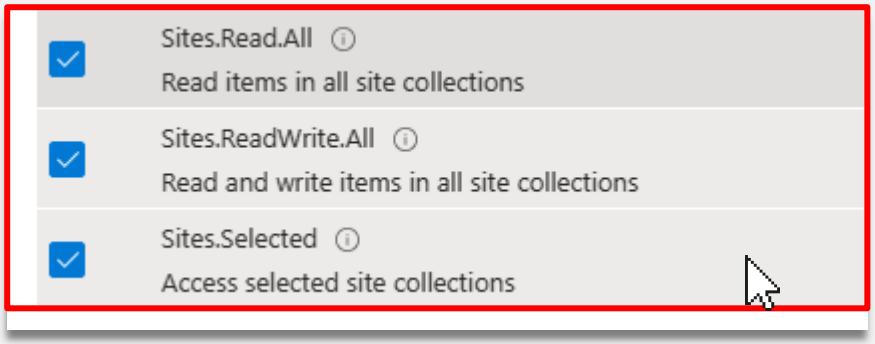


8) Nessa tela iremos clique em **Permissões de aplicativo** que é a permissão que iremos utilizar na API. A **permissão delegada** é para quando formos utilizar alguma autenticação que necessite de um usuário para fazer login, já a **permissão de aplicativo** não necessita de um usuário para fazer login, ou seja, o próprio código fará o mesmo.

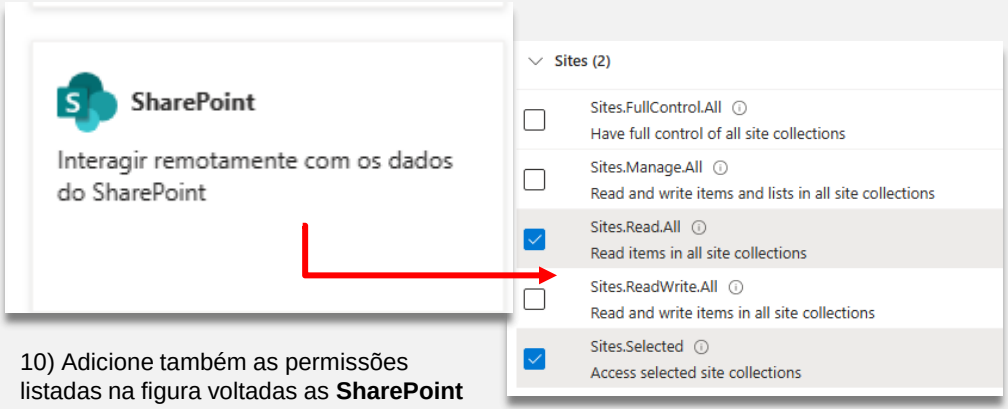


9) Como queremos ter todo controle de um site do Share Point, procure por site e navegue até **sites** selecione as permissões e adicione. Para saber mais sobe permissões [acesse aqui]

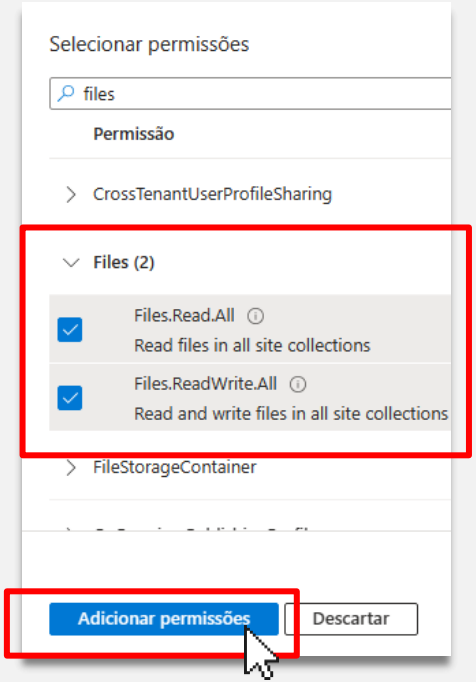
2. Configurando um aplicativo no Azure AD



10) Adicione também as permissões listadas na figura

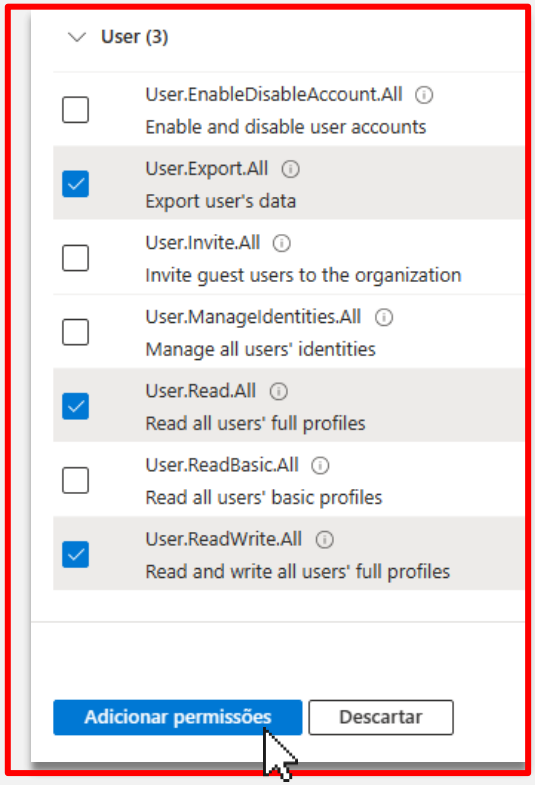


10) Adicione também as permissões listadas na figura voltadas as **SharePoint permissões de aplicativo**

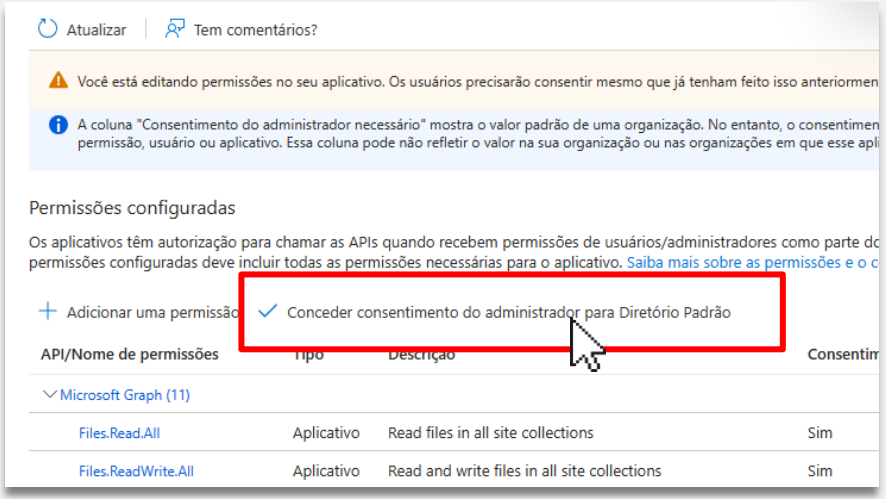


11) Como queremos que nossa API busque arquivos do **SharePoint Online** e salve no **Azure Blob Storage** devemos conceder permissões de **files**

2. Configurando um aplicativo no Azure AD



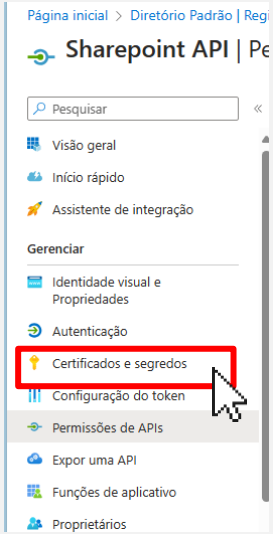
13) Adicione também permissões de usuários, como mostra a figura



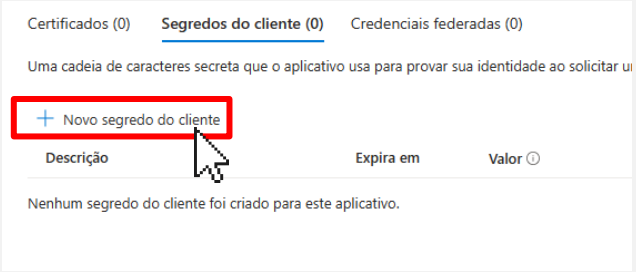
14) Após volte na lista de permissões adicionadas e clique em **Conceder consentimento do administrador....** É importante lembrar que se você for um usuário comum (não organizacional) você irá conseguir conceder as permissões, do contrário não. Para os casos em que seu usuário pertença a uma organização, solicite do administrador da sua organização que conceda essas permissões

2. Configurando um aplicativo no Azure AD

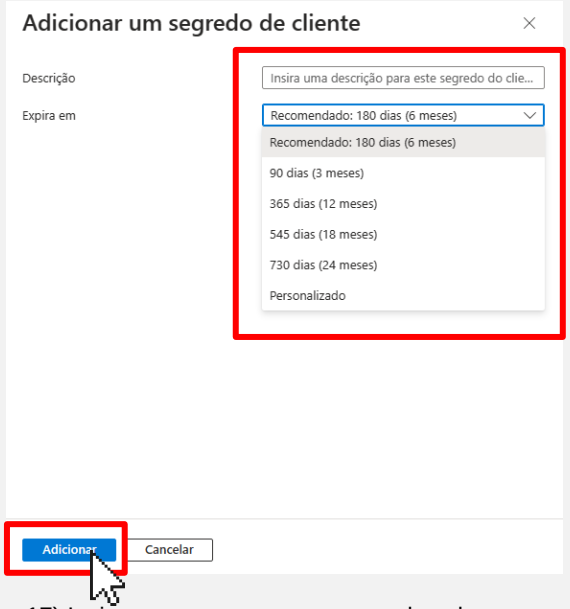
Crie um certificado e um segredo para nossa API conseguir acessar seu aplicativo



15) Clique em **Certificados e Segredos**

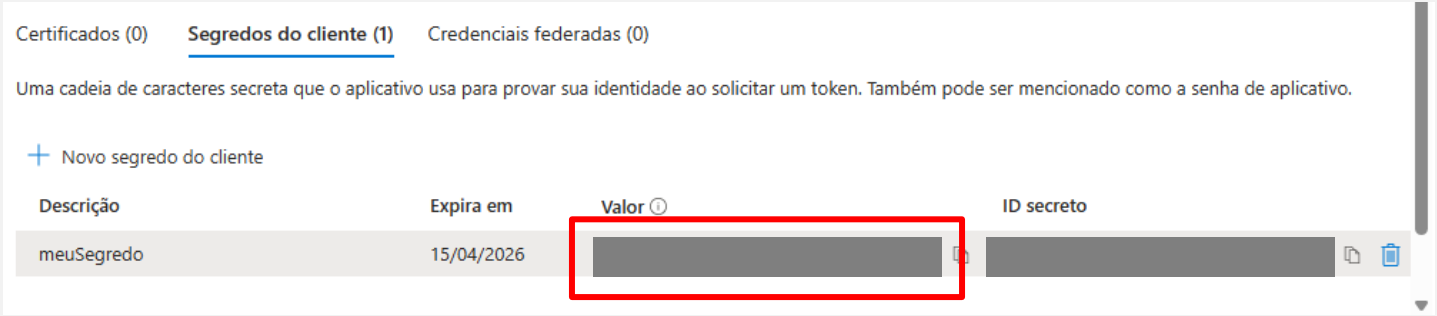


16) Clique em **Novo segredo do Cliente**



17) Insira um nome para seu segredo coloque um prazo de expiração e clique em Adicionar

2. Configurando um aplicativo no Azure AD



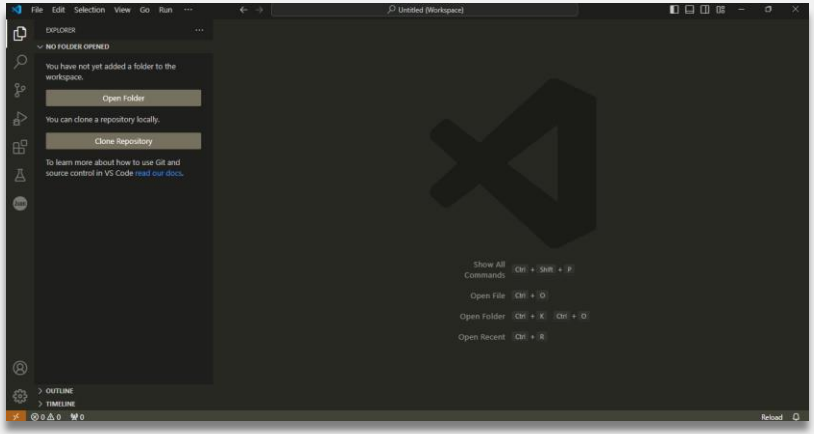
18) Anote o valor do segredo pois esse valor irá sumir.

Pronto seu aplicativo está configurado para acessar a API

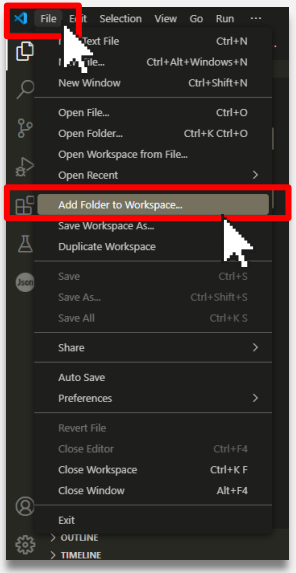
SUMÁRIO

3. Configurando as permissões na API para acessar o aplicativo

Para acessar a API, primeiro abra a pasta **api_SharePoint_To_BlobStorage** no seu VsCode, para isso siga os passos abaixo:



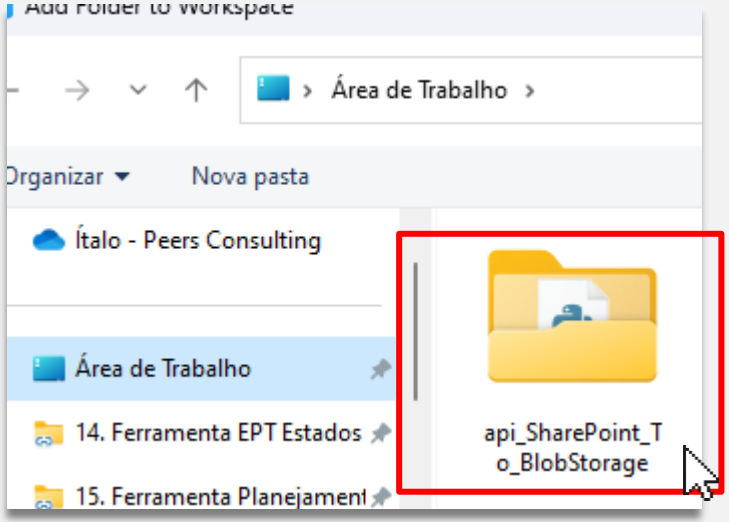
1) Abra seu VsCode



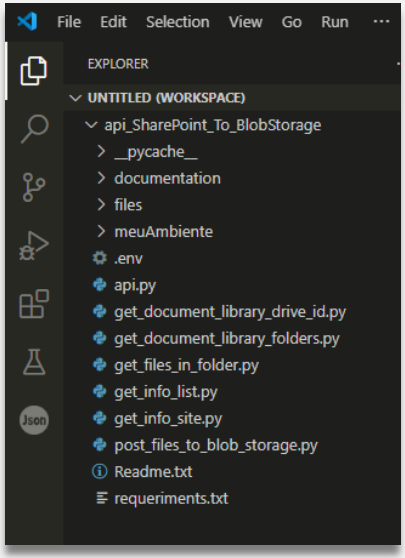
2) Clique em **File**, depois em **Add Folder to Workspace**

3. Configurando as permissões na API para acessar o aplicativo

Para acessar a API, primeiro abra a pasta **api_SharePoint_To_BlobStorage** no seu VsCode, para isso siga os passos abaixo:

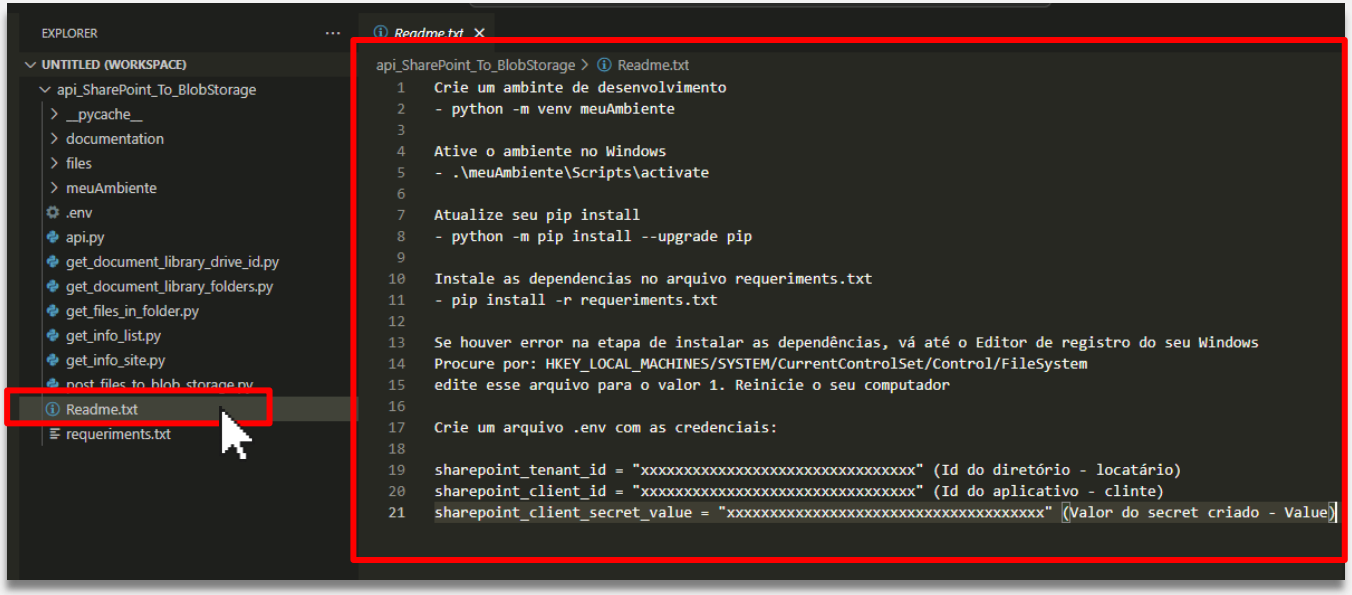


3) Localize a pasta da api e adicione no VSCode



4) Visão geral da pasta da API adicionada no VsCode

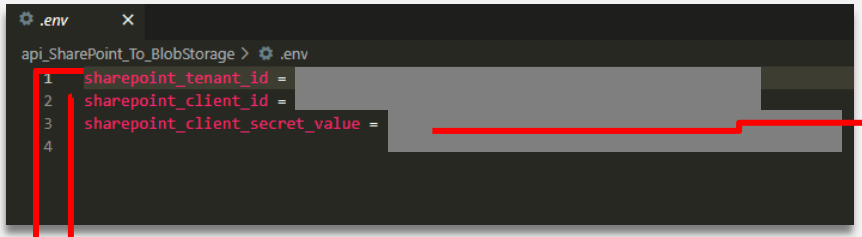
3. Configurando as permissões na API para acessar o aplicativo



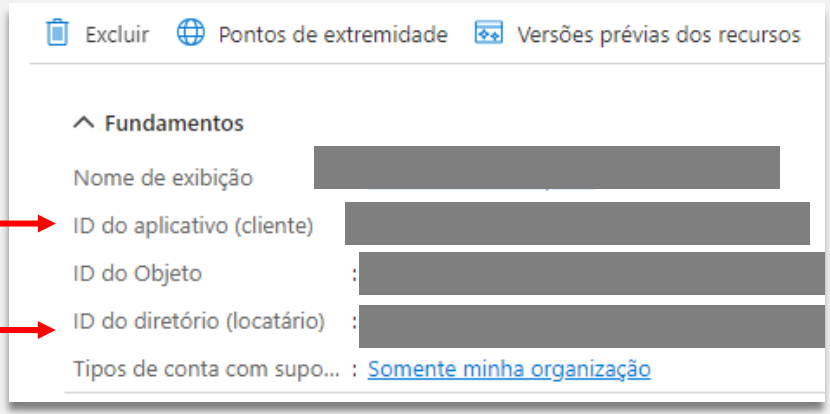
5) Abra o arquivo **Readme.**txt e siga os passos para criar um ambiente para começar a trabalhar com API

3. Configurando as permissões na API para acessar o aplicativo

Crie um arquivo **.env** para configurar as permissões, cole agora os códigos copiados.



O código value secret que você copiou do Slide 16



Feito isso, você está pronto para começar usar a API

SUMÁRIO

4. Conhecendo os métodos da API

A API Share Point to Blob Storage possui alguns arquivos principais vale a pena conhecermos:

- api.py **(Vamos começar por esse)**
- get_document_library_drive_id.py
- get_document_library_folders.py
- get_files_in_folder.py
- get_info_list.py
- get_info_site.py
- post_files_to_blob_storage.py

4. Conhecendo os métodos da API

O arquivo **api.py** é o arquivo principal da API é nele onde a autenticação está implementada. Esse arquivo não iremos executa-lo, pois ele irá servir como arquivo de base para os outros.

```python
def auth():

    msal_authority = f"https://login.microsoftonline.com/{TENANT_ID}"
    msal_scope = ["https://graph.microsoft.com/.default"]

    msal_app = ConfidentialClientApplication(
        client_id=CLIENT_ID,
        client_credential=CLIENT_SECRET,
        authority=msal_authority
    )

    result = msal_app.acquire_token_silent(
        scopes=msal_scope,
        account=None
    )

    if not result:
        result = msal_app.acquire_token_for_client(scopes=msal_scope)

    if "access_token" in result:
        access_token = result["access_token"]
        return access_token
    else:
        raise Exception("No Access Token found or Token has expired in Azure AD")
```

No código desse arquivo é possível visualizar que ele faz uma requisição nos parâmetros definidos no **.env.** Asssim ele define as variáveis de ambiente através do arquivo **. env** e utiliza no código abaixo para fazer a autenticação

4. Conhecendo os métodos da API : get_info_site.py

Iremos começar utilizando o arquivo **get_info_site.py.** Esse arquivo retorna as informações principais (ex: id do site, nome do site, data criação site etc) de um site do SharePoint.

Podemos executar esse arquivo passando como parâmetro o **nomeDoSite** assim:

```
python .\get_info_site.py 'nomeDoSite'
```

Como resposta obtemos um json assim:


```
{
  "@odata.context": "contexto",
  "value":[
    {
      "createdDateTime":"2023-00-00T00:00:10Z",
      "description":"descricao",
      "id":"id do site",
      "lastModifiedDateTime":"2023-00-00T00:00:10Z",
      "name":"nome do site",
      "webUrl":"url do site",
      "displayName":"display name",
      "root":{

      },
      "siteCollection":{
        "hostname":"host name"
      }
    }
  ]
}
```

4. Conhecendo os métodos da API : get_info_list.py

O próximo método da API se chama **get_info_list.py** ele pode ser utilizado para obter informações sobre uma **lista criada** em um site do Share Point Online.

Podemos executar esse arquivo passando
como parâmetro o **nomeDaLista** e o **idDoSite** assim:

```
python .\get_ info_list.py 'nomeDaLista' 'idDoSite'
```

Como resposta obtemos um json: Próximo slide

4. Conhecendo os métodos da API : get_info_list.py


```
{
  "@odata.context":"contexto",
  "@odata.etag":"identificador",
  "createdDateTime":"2024-04-24T00:00:00Z",
  "description":"descricao",
  "eTag":"tag",
  "id":"id da lista",
  "lastModifiedDateTime":"2024-04-24T00:00:00Z",
  "name":"nome da lista",
  "webUrl":"url da lista",
  "displayName":"display name",
  "createdBy":{
    "user":{
      "email":"email usuario criou a lista",
      "id":"id do usuario que criou a lista",
      "displayName":"display name"
    }
  },
  "lastModifiedBy":{
    "user":{
      "email":"email do usuario que fez ultima modificação",
      "id":"id do usuário que modificou a lista",
      "displayName":"display name"
    }
  },
  "parentReference":{
    "siteId":"site id que contem a lista"
  },
  "list":{
    "contentTypesEnabled":false,
    "hidden":false,
    "template":"nome template"
  }
}
```

4. Conhecendo os métodos da API : get_document_library_drive_id.py

O próximo método da API se chama **get_document_library_drive_id.py** ele pode ser utilizado para obter o **id de uma biblioteca de documentos** do site do Share Point Online.

Podemos executar esse arquivo passando
como parâmetro o **nomeDaBibliotecaDeDocumentos** e o **nomeDoSite** assim:

```
python .\get_ document_library_drive_id.py 'nomeDaBibliotecaDeDocumentos' 'nomeDoSite'
```

Como resposta obtemos uma mensagem:

```
Para a biblioteca de documentos Documentos seu drive_id é: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

4. Conhecendo os métodos da API : get_document_library_folders.py

O próximo método da API se chama **get_document_library_folders.py** ele pode ser utilizado para obter **todas as pastas e subpastas de uma biblioteca de documentos** do site do Share Point Online.

Podemos executar esse arquivo passando
como parâmetro o **idBibliotecaDeDocumentos** e o **caminhoDaPastaOuSubpasta** assim:

```
python .\get_ document_library_folder.py 'idDaBibliotecaDeDocumentos' 'caminhoDaPastaOuSubpasta'
```

Obs: O **caminhoDaPastaOuSubpasta** pode começar depois do nome da biblioteca de documentos

Como resposta obtemos um json: Próximo slide

4. Conhecendo os métodos da API : get_document_library_folders.py


```
{
  "@odata.context":"contexto",
  "value":[
    {
      "@microsoft.graph.Decorator":"nome decorator",
      "createdBy":{
        "user":{
          "email":"email do usuario criador",
          "id":"id do usuario criador",
          "displayName":"display name"
        }
      },
      "createdDateTime":"2023-07-06T11:09:32Z",
      "eTag":"nome da tag",
      "id":"id da pasta pu subpasta",
      "lastModifiedBy":{
        "user":{
          "email":"email do usuario que modificou",
          "id":"id do usuario que modificou",
          "displayName":"display name"
        }
      },
```

```
      "lastModifiedDateTime":"2023-07-12T19:32:21Z",
      "name":"nome da pasta ou subpasta",
      "parentReference":{
        "driveType":"tipo drive",
        "driveId":"id do drive",
        "id":"id do drive pai",
        "name":"nome do pai",
        "path":"caminho",
        "siteId":"url da pasta ou subpasta pai"
      },
      "webUrl":"url da subpasta ou pasta",
      "cTag":"tag",
      "fileSystemInfo":{
        "createdDateTime":"2023-07-06T11:09:32Z",
        "lastModifiedDateTime":"2023-07-12T19:32:21Z"
      },
      "folder":{
        "childCount":1
      },
      "shared":{
        "scope":"users"
      },
      "size":"tamanho da pasta ou subpasta"
    }
  ]
}
```

4. Conhecendo os métodos da API : get_files_in_folder.py

O próximo método da API se chama **get_files_in_folder.py** ele pode ser utilizado para obter **todos os arquivos de uma pasta ou subpasta** do site do Share Point Online.

Podemos executar esse arquivo passando
como parâmetro o **idDaBibliotecaDocumentos** e o **caminhoDaPastaOuSubpasta** assim:

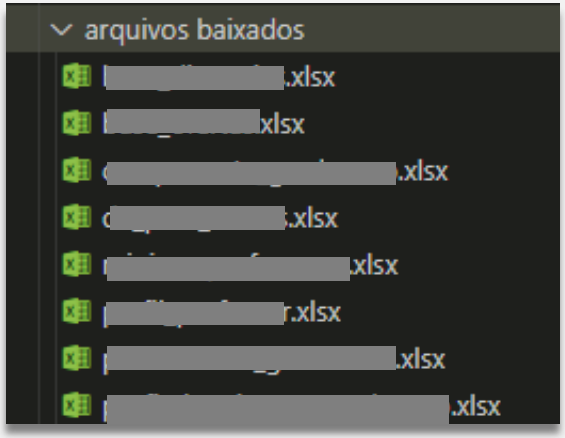
```
python .\get_files_in_folder.py 'idDaBibliotecaDocumentos' 'caminhoDaPastaOuSubpasta'
```

Obs: O **caminhoDaPastaOuSubpasta** pode começar depois do nome da biblioteca de documentos

Como resposta obtemos os arquivos baixados na pasta **arquivos baixados**:

Próximo slide

4. Conhecendo os métodos da API : get_files_in_folder.py



1) Arquivos baixados na pasta **arquivos baixados**

4. Conhecendo os métodos da API : post_files_to_blob_storage.py

O próximo método da API se chama **post_files_to_blob_storage.py** ele pode ser utilizado para pegar todos arquivos baixados e salvos na pasta **arquivos baixados** e fazer upload para um container do Blob Storage.

Podemos executar esse arquivo passando
como parâmetro o **conectionString**, o **nameContainerNoBlob**, **caminhoContainerNoBlobNome Arquivo** e o **caminhoLocalArquivoNome**:

```
python .\get_files_in_folder.py 'conectionString' 'nameContainerNoBlob' 'caminhoContainerNoBlobNome Arquivo' 'caminhoLocalArquivoNome'
```