

Configurando meu primeiro repositório GitHub

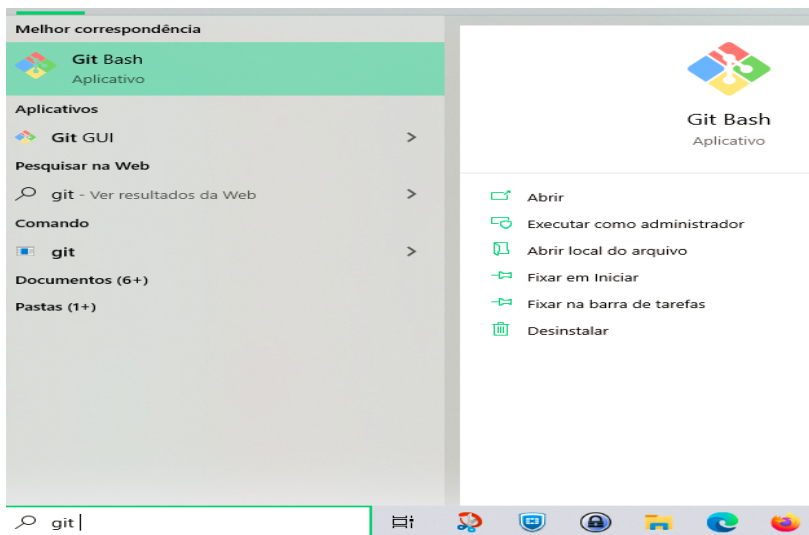
e clonando-o com o Git

Por Marcelo Soares

Passo 1) Gerando um par de chaves SSH:

Vamos gerar um par de chaves SSH com a finalidade de configurar o acesso ao GitHub. Considerando que o Git já esteja instalado na máquina local (os procedimentos foram realizados no Windows 10).

1.1) No menu iniciar do seu Windows pesquise por **Git Bash**. Clique em “Abrir”, para iniciar o prompt do Git:



1.2) Com o bash aberto vamos gerar um par de chaves SSH. Digite o comando abaixo:

\$ ssh-keygen -t ed25519 -C seuemail@com.br

```
$ ssh-keygen -t ed25519 -C [redacted]@gmail.com
Generating public/private ed25519 key pair.
Enter file in which to save the key (/c/Users/[redacted]/.ssh/id_ed25519):
Created directory '/c/Users/[redacted]/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/[redacted]/.ssh/id_ed25519
Your public key has been saved in /c/Users/[redacted]/.ssh/id_ed25519.pub
The key's fingerprint is:
SHA256:[redacted]
The key's randomart image is:
+--[ED25519 256]--+
|      . .+=B|
|      . o .oB+|
|      . . o o +.X|
|      . . . *.X+|
|      + S   .o@ +|
|      + +   .+=+o+|
|      . o   oo= |
|      E   + . |
|      .o.. |
+-----[SHA256]-----+
```

Em **ssh-key** as flags “-t” define o tipo de criptografia (no exemplo ed25519) e “-C ” define um comentário (no exemplo o e-mail).

Após rodar o comando, será solicitado o local onde deve ser salvo o par de chaves (“**Enter file in which to save the key**”), e caso deseje manter o padrão apenas tecla <Enter> - o que é fortemente recomendado, a menos que você realmente saiba o que está fazendo. Em seguida, será solicitado inserir uma senha (“**Enter passphrase (empty for no passphrase):**”), caso não deseje colocar senha apenas tecla <Enter>, e depois confirme sua escolha digitando novamente a senha (“**Enter same passphrase again:**”)

1.3) Agora navegue até o diretório onde as chaves foram salvas e em seguida abra o arquivo *.pub. Use os comandos abaixo:

```
$ cd /c/Users/"SeuUser"/.ssh/  
$ cat id_ed25519.pub
```



OBS: Este procedimento é necessário para configurar seu GitHub (Passo 2)

1.4) Agora vamos iniciar o Git e adicionar a chave privada “id_ed25519”. Observe que não é a chave “id_ed25519.pub”. Para Tal, use os comandos abaixo:

```
$ eval $(ssh-agent -s)
```

> Agent pid 424 # Esta saída informa o PID do processo iniciado

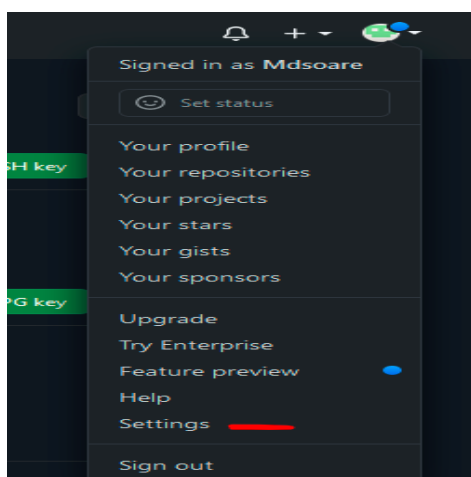
```
$ ssh-add id_ed25519 # Este comando adiciona a chave privada. Será solicitado sua senha
```

> Enter passphrase for id_ed25519:

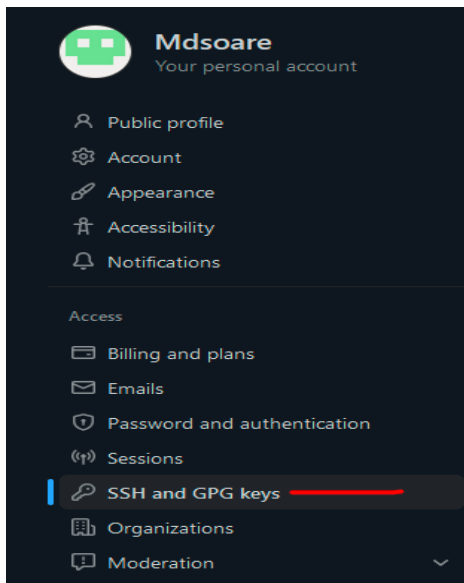
> Identity added: id_ed25519 (seuemail@com.br)

Passo 2) Configurando o GitHub.

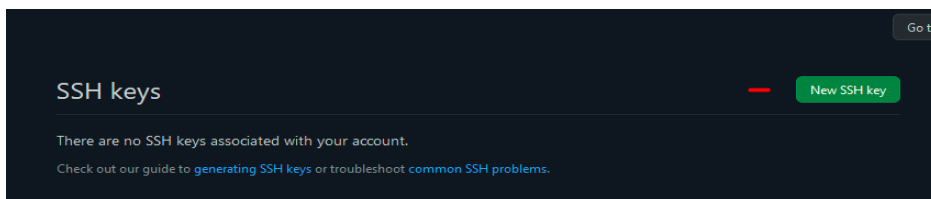
2.1) Após realizar o login no GitHub, vá em “**Settings**”:



2.2) Em seguida clique na opção “SSH and GPG Keys”:



2.3) Agora verifique a opção “SSH Keys” e depois a opção “New SSH Keys”:



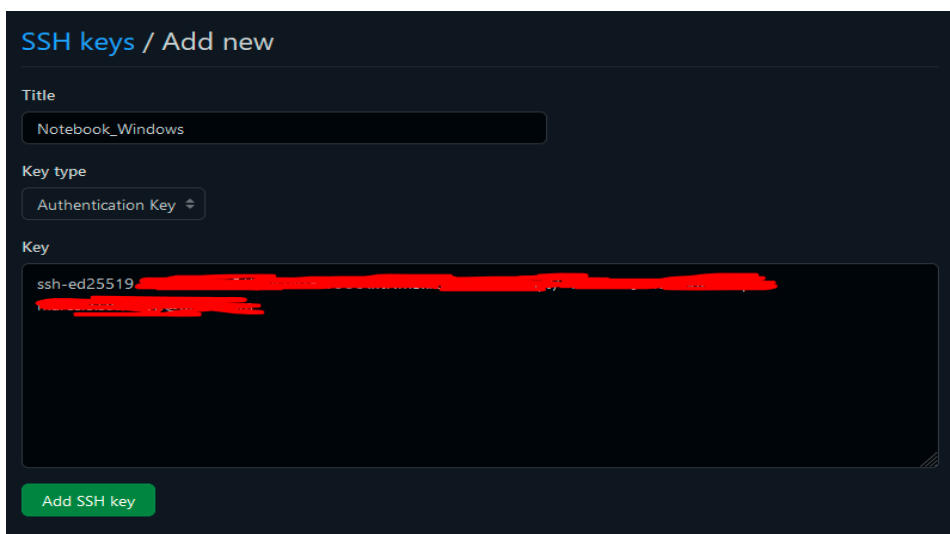
2.4) Na nova tela preencha as informações de acordo com suas necessidades. Segue meu exemplo:

Title ⇒ Um título que servirá de identificação da chave cadastrada. Escolhi “**Notebook_Windows**”

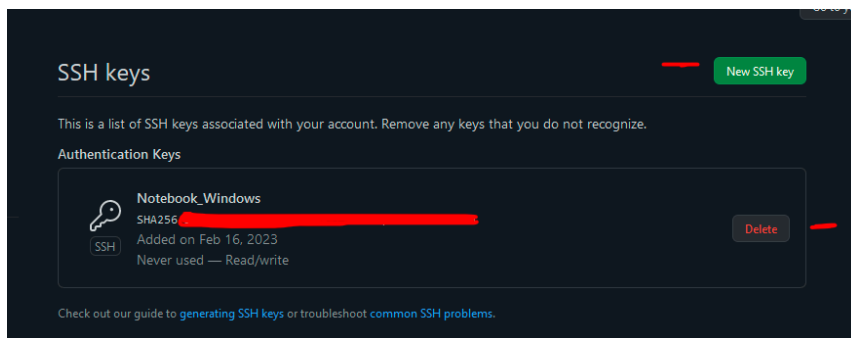
Key type ⇒ Não há necessidade de alteração para este exemplo.

Key ⇒ Cole o conteúdo da chave *.pub obtida na saída do comando do procedimento anterior 1.3.

E por fim, clique em “**Add SSH Key**”



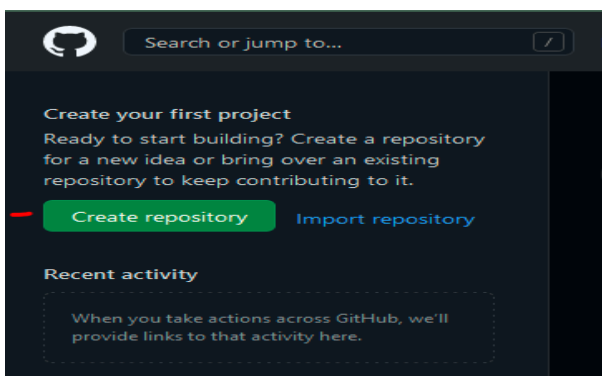
Pronto! É possível visualizar que a chave SSH foi configurada com sucesso. O GitHub está configurado para o acesso remoto sem a necessidade de inserir senha ao usar o Git.



NOTA: O GitHub pode ser configurado para ter várias chaves cadastradas, basta seguir os mesmos passos e clicar em **“New SSH Keys”**. Também é possível remover uma chave com a opção **“Delete”** neste mesmo lugar.

Passo 3) Criando um repositório no GitHub

3.1) No seu GitHub selecione a opção **“Create repository”**



3.2) Siga as instruções. Segue meu exemplo:

Repository Name ⇒ Nome do seu repositório

Public ⇒ Repositório será aberto (acessível) a outros usuários. Opção que escolhi.

Private ⇒ Seria o oposto ao Public. Existem algumas características associadas a estas opções, mas isso extrapola o propósito deste pequeno tutorial.

Add a README file ⇒ Para gerar um arquivo **README.md** que será exibido no início do acesso ao repositório. Não verifiquei essa opção porque desejo exemplificar como fazê-lo pelo Git.

Add .gitignore ⇒ Opção para ignorar determinados arquivos. Não se aplica para este tutorial (None = Nenhum).

Choose a license ⇒ Escolha o tipo de licença, eu escolhi **“MIT License”**. Caso deseje se informar sobre o assunto basta realizar uma pesquisa no seu buscador preferido.

Após o preenchimento clique em **“Create repository”**. Segue abaixo o print das opções:

Owner * Mdsoare / Repository name * Bootcamp_dio ✓

Great repository names are short and memorable. Need inspiration? How about [expert-octo-fiesta](#)?

Description (optional)

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)
.gitignore template: None ▼

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)
License: MIT License ▼

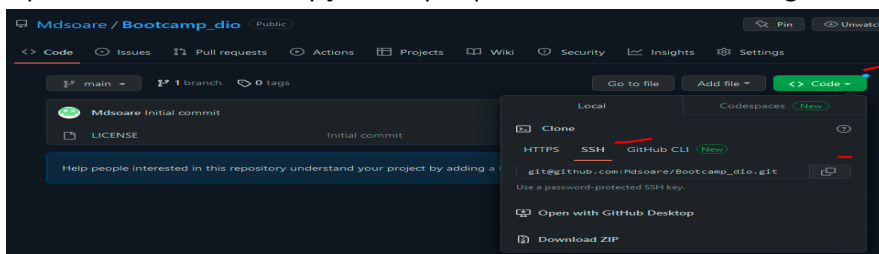
i You are creating a public repository in your personal account.

Create repository

Repositório foi criado com sucesso!

Passo 4) Clonando o repositório pelo Git:

4.1) Com seu repositório já criado, ainda no GitHub vá em **“Code”** e escolha a forma de como copiar o link do repositório. Eu escolhi a opção SSH porque foi a forma como configurei o acesso nos passos anteriores:



4.2) No Git Bash criei um diretório em **c/** chamado **“workspace”**. É ele que vou usar como o local para clonar o repositório criado no GitHub. Segue:

```
MINGW64 ~/.ssh
$ cd c:
MINGW64 /c
$ pwd
/c
MINGW64 /c
$ mkdir workspace
MINGW64 /c
$ cd workspace/
MINGW64 /c/workspace
$ pwd
/c/workspace
```

4.3) Ao entrar no diretório criado, para executar o comando abaixo para clonar o repositório do GitHub, no meu caso o **“Bootcamp_dio”**:

\$ git clone git@github.com:Mdsoare/Bootcamp_dio.git

```
MINGW64 /c/workspace
$ git clone git@github.com:Mdsoare/Bootcamp_dio.git
Cloning into 'Bootcamp_dio'...
The authenticity of host 'github.com (20.201.28.151)' can't be established.
ED25519 key fingerprint is SHA256:
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
MINGW64 /c/workspace
$ ls
Bootcamp_dio/
```

OBS: Como foi a primeira vez que realizei o acesso via SSH, precisei confirmar (**yes**). É possível verificar que um diretório novo foi criado chamado pelo mesmo nome do repositório GitHub. Ao entrar no diretório, podemos listar seu conteúdo:

\$ ls -a

```
MINGW64 /c/workspace/Bootcamp_dio
$ ls -a
./ ../ .git/ LICENSE
```

NOTA: Após realizar os passos acima para clonar um repositório, podemos observar que este também foi adicionado um **repositório remoto** com o alias **“origin”**. Observe o comando abaixo, após entrar no diretório **Bootcamp_dio**:

\$ git remote -v

```
MINGW64 /c/workspace/Bootcamp_dio
$ git remote -v
origin git@github.com:Mdsoare/Bootcamp_dio.git (fetch)
origin git@github.com:Mdsoare/Bootcamp_dio.git (push)
```

Este comando exibe seus diretórios remotos e para **adicionar outro repositório remoto**, use:

\$ git remote add “seualias” “seulinkremotogithub” # sem as aspas.

Também é possível **remover um repositório remoto**, usando:

\$ git remote remove "seualias" # sem as aspas .

Passo 5) Criando o arquivo **README.md** e sincronizando ao GitHub.

NOTA: Um arquivo *.md segue/usa a "linguagem" markdown. Para maiores informações realize suas pesquisas no seu buscador preferido.

5.1) No diretório **Bootcamp_dio**, crie o arquivo **README.md**, use:

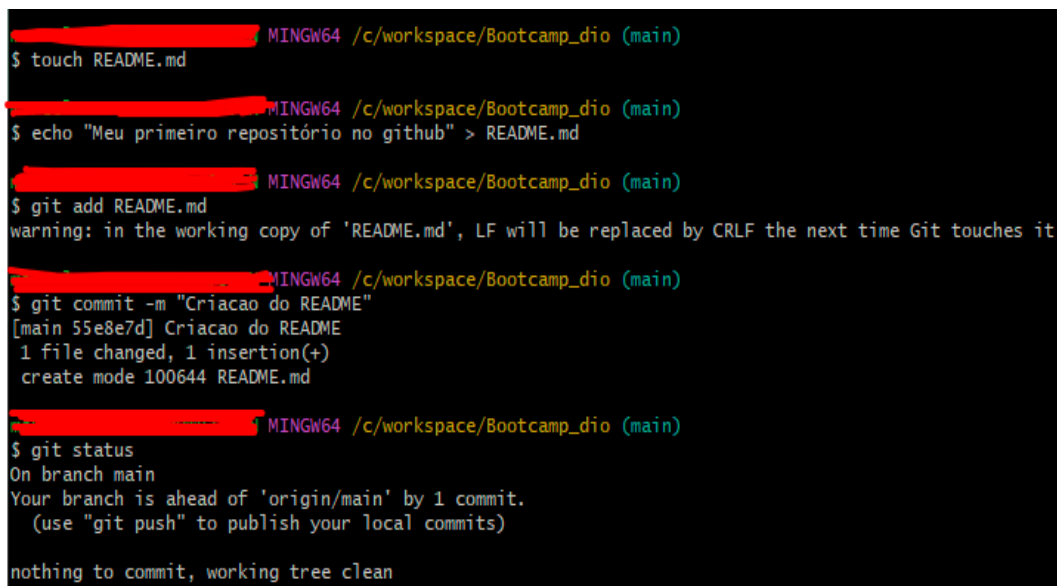
\$ touch README.md # O comando cria um arquivo vazio.

\$ echo "Meu primeiro repositório no GitHub" > README.md # Este comando seria suficiente para criar o arquivo **README.md**, não sendo necessário o uso do comando **touch**, mas quis exemplificar.

5.2) "Comitando" e adicionando ao GitHub:

\$ git add *

\$ git commit -m "Criacao do README.md"



```
MINGW64 /c/workspace/Bootcamp_dio (main)
$ touch README.md

MINGW64 /c/workspace/Bootcamp_dio (main)
$ echo "Meu primeiro repositório no github" > README.md

MINGW64 /c/workspace/Bootcamp_dio (main)
$ git add README.md
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

MINGW64 /c/workspace/Bootcamp_dio (main)
$ git commit -m "Criacao do README"
[main 55e8e7d] Criacao do README
1 file changed, 1 insertion(+)
create mode 100644 README.md

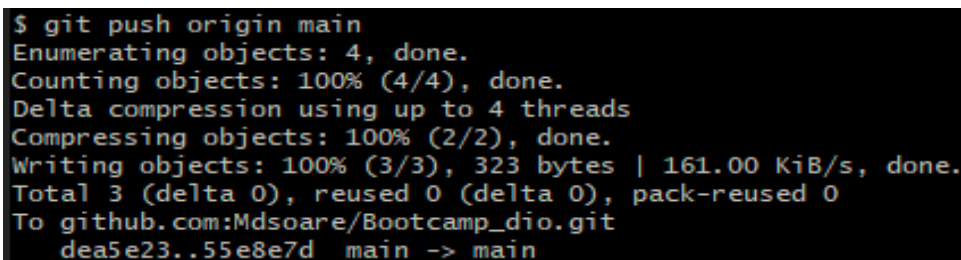
MINGW64 /c/workspace/Bootcamp_dio (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

nothing to commit, working tree clean
```

NOTA: Usei o comando **"git status"** para conferir se houve algum erro/ pendência no commit.

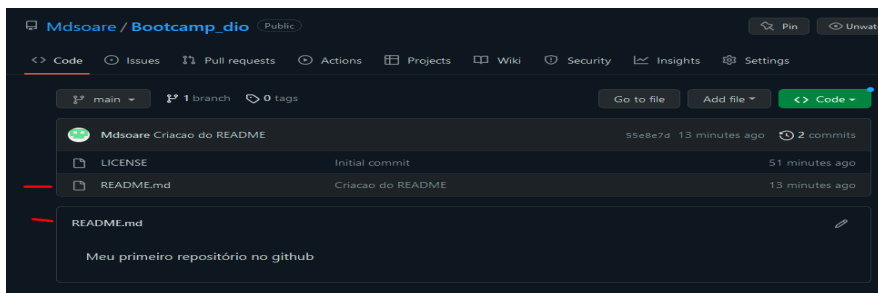
Com tudo certo, vamos sincronizar o repositório local com o remoto, usando:

\$ git push origin main

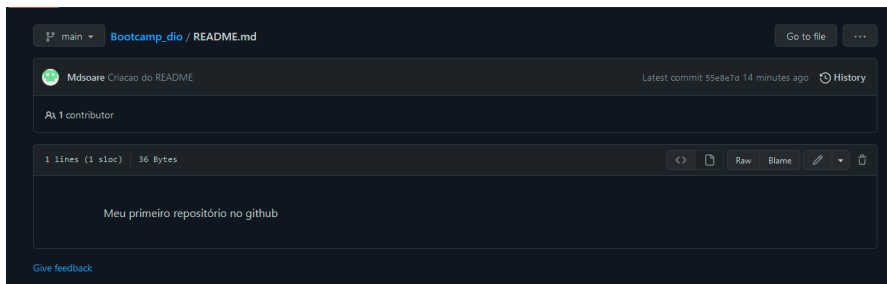


```
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 323 bytes | 161.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:Mdsoare/Bootcamp_dio.git
dea5e23..55e8e7d main -> main
```

5.3) Podemos conferir no GitHub que o arquivo **README.md** foi realmente adicionado:



5.4) Ao clicar em README.md, teremos o histórico de seu versionamento:

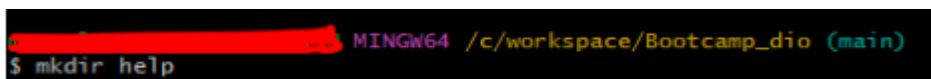


Passo 6) Criando um novo diretório e sincronizando no GitHub.

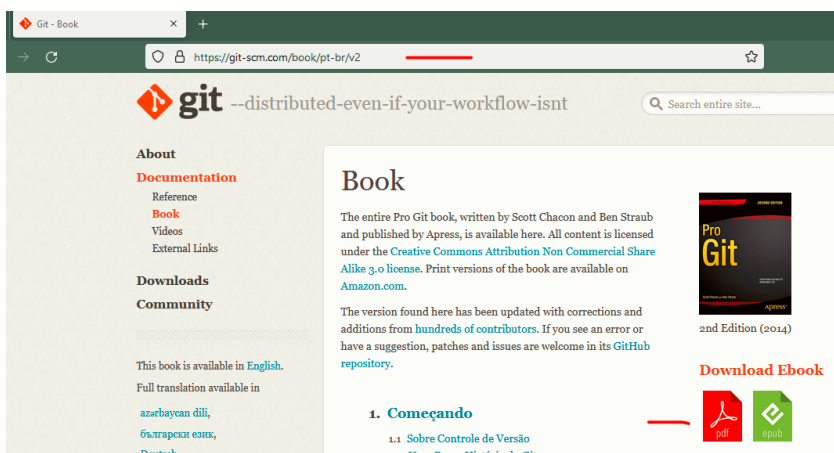
Apenas para reforçar o que foi feito até aqui, vamos adicionar um subdiretório e acrescentar um arquivo *.pdf, nele.

6.1) Criando o diretório “help”. Dentro do diretório **Bootcamp_dio**, use:

\$ mkdir help



6.2) Vamos acessar o link <https://git-scm.com/book/pt-br/v2> para baixar o livro **progit.pdf**, que é free e contém um manual do Git em português (na sua maior parte, rs):



6.3) Vou mover o arquivo **progit.pdf** baixado em **Downloads** para o diretório **help**, e depois atualizar o GitHub. Seguem os passos no Git Bash:

\$ pwd # Exibe o diretório atual

```
$ pwd
/c/workspace/Bootcamp_dio/help
```

\$ cd /c/Users/seuuser/Downloads/ # Acessando o diretório de Download onde o arquivo pdf foi baixado

```
$ cd /c/Users/[redacted]/Downloads/
```

\$ mv progit.pdf /c/workspace/Bootcamp_dio/help # Move o arquivo pdf para o diretório help

```
$ mv progit.pdf /c/workspace/Bootcamp_dio/help
```

\$ cd - # Acessa o diretório anterior, neste caso o help.

\$ ls # lista o conteúdo do diretório

```
$ cd -
/c/workspace/Bootcamp_dio/help
MINGW64 /c/workspace/Bootcamp_dio/help (main)
$ ls
progit.pdf
```

Gravando as alterações. Observe que foi necessário voltar para o diretório Bootcamp_dio para “comitarmos”:

\$ cd ..

\$ git add *

\$ git commit -m “Adicionando o diretório help ao github”

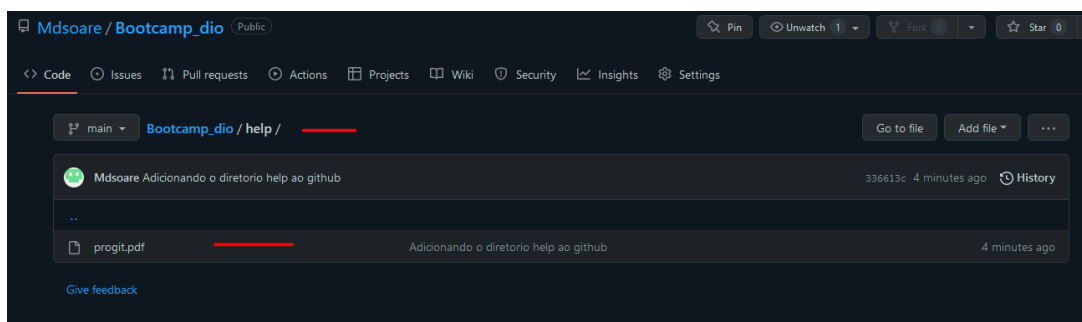
```
$ cd ..
MINGW64 /c/workspace/Bootcamp_dio (main)
$ pwd
/c/workspace/Bootcamp_dio
MINGW64 /c/workspace/Bootcamp_dio (main)
$ git add *
MINGW64 /c/workspace/Bootcamp_dio (main)
$ git commit -m "Adicionando o diretorio help ao github"
[main 336613c] Adicionando o diretorio help ao github
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 help/progit.pdf
```

Vamos sincronizarmos com o repositório remoto:

\$ git push origin main

```
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 12.08 MiB | 2.78 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:Mdsoare/Bootcamp_dio.git
 55e8e7d..336613c  main -> main
```

6.4) No GitHub, podemos observar que tanto a pasta help quanto o progit.pdf foram disponibilizados com sucesso:



Com isso finalizamos os passos para a criação de um repositório no GitHub e a clonagem do mesmo pelo Git. Existem outras formas de clonarmos um repositório do GitHub, escolhendo a que melhor atenda às suas necessidades.

Clonando um repositório pelo Visual Studio Code

Se seu editor default do Git for o VS Code, também podemos clonar um repositório por ele em vez.

NOTA: Esta configuração dispensa o uso de chaves SSH. Vejamos:

Passo 1) Crie um repositório no GitHub:

1.1) Para este exemplo vou criar outro repositório, desta vez chamado de **“Bootcamp-dio-java-basico”**. As ações são bem parecidas com as já mencionadas ao criar o repositório **Bootcamp_dio** no GitHub.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner * / Repository name *
Mdsoare / Bootcamp-dio-java-basico ✓

Great repository names are short and memorable. Need inspiration? How about [ubiquitous-memory?](#)

Description (optional)

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Java

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License

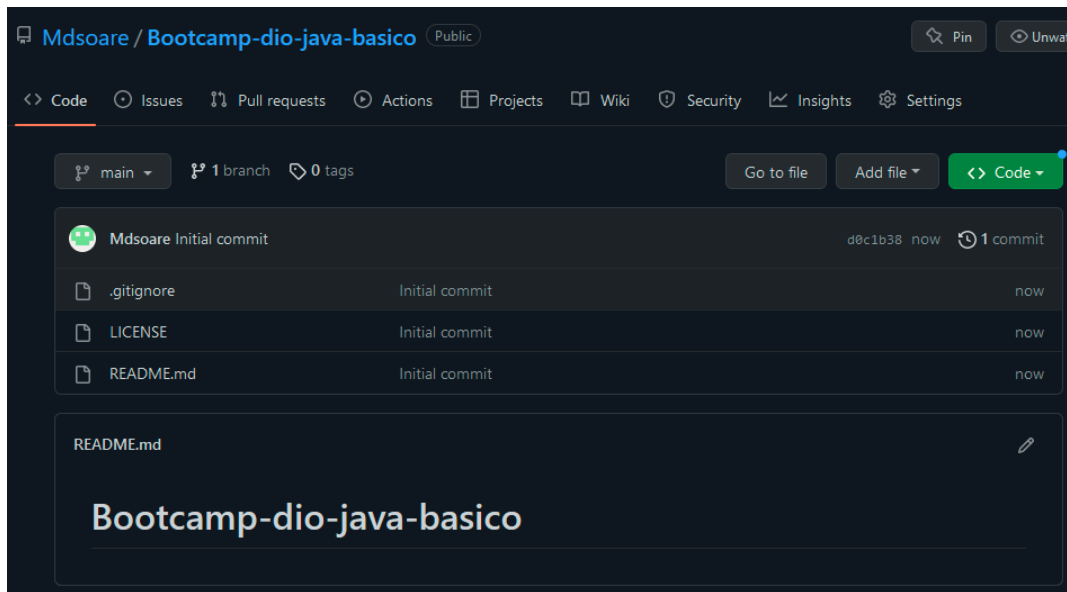
This will set `main` as the default branch. Change the default name in your [settings](#).

i You are creating a public repository in your personal account.

Create repository

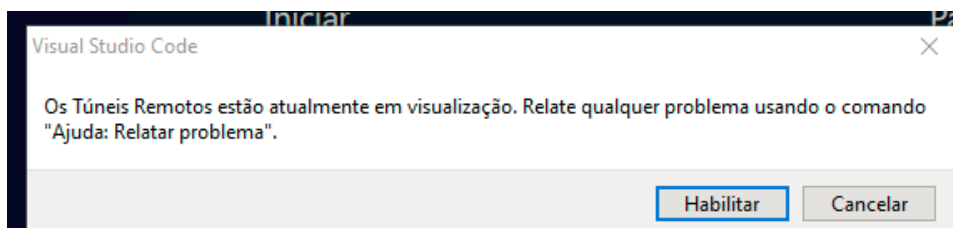
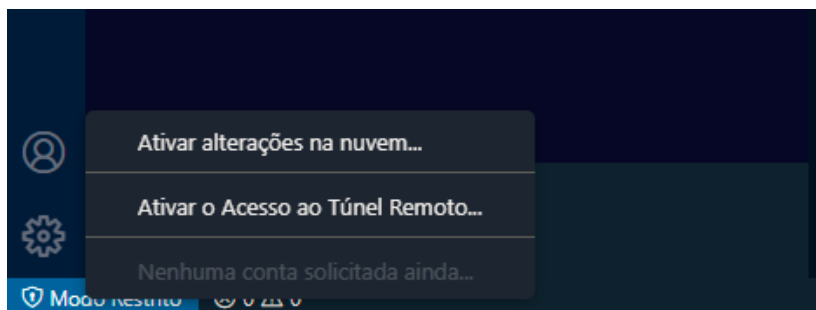
NOTA: Desta vez, já verifiquei a opção **“Add a README file”** e em **“Add .gitignore”** selecionei a opção **java**, para ignorar arquivos desnecessários no versionamento

Novo repositório Criado:

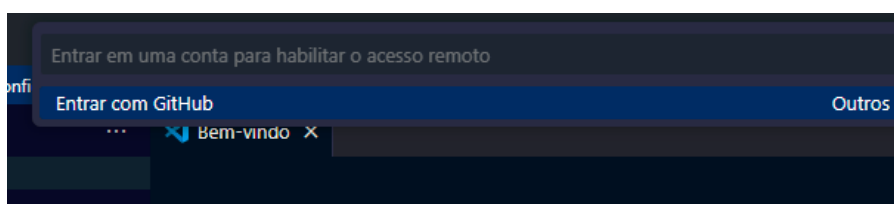


Passo 2) Configurando seu VS Code.



2.1) Abra seu VS Code e no ícone acima da engrenagem clique em **“Ativar o Acesso ao Túnel Remoto...”** e em seguida **“Habilitar”**:



2.2) Selecione **“Entrar com GitHub”**:



2.3) No seu navegador padrão será exigida suas credenciais GitHub:

Sign in to GitHub
to continue to GitHub for VS Code

Username or email address


Password [Forgot password?](#)


Sign in


New to GitHub? [Create an account.](#)

2.4) Após inserir suas credenciais GitHub clique em **“Authorize Visual-Studio-Code”**:

Authorize GitHub for VS Code

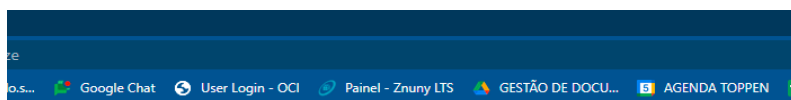
 **GitHub for VS Code by Visual Studio Code**
wants to access your Mdsore account

 **Organizations and teams**
Read-only access

 **Personal user data**
Email addresses (read-only)

Cancel **Authorize Visual-Studio-Code**

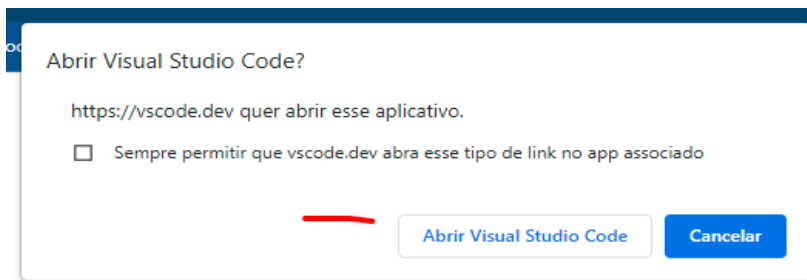
Authorizing will redirect to
<https://vscode.dev>



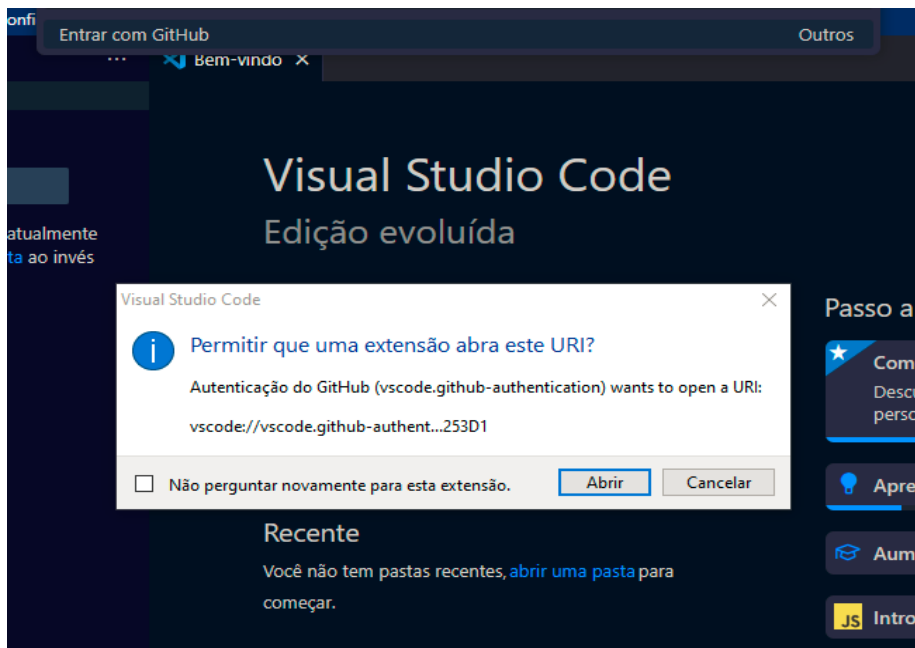
You are being redirected to the authorized application.

If your browser does not redirect you back, please [click here](#) to continue.

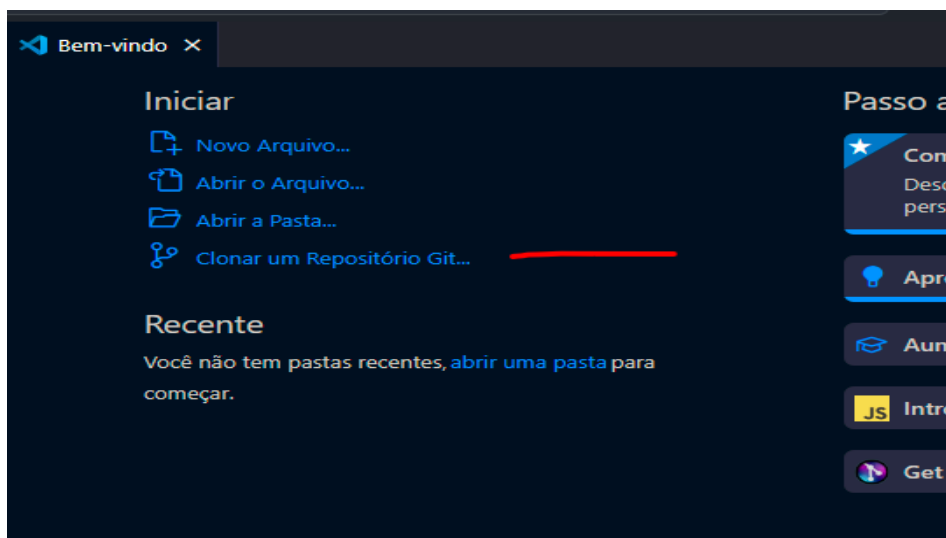
2.5) Você será redirecionado a abrir o VS Code através do pop-up abaixo, clique em **“Abrir Visual Studio Code”**:



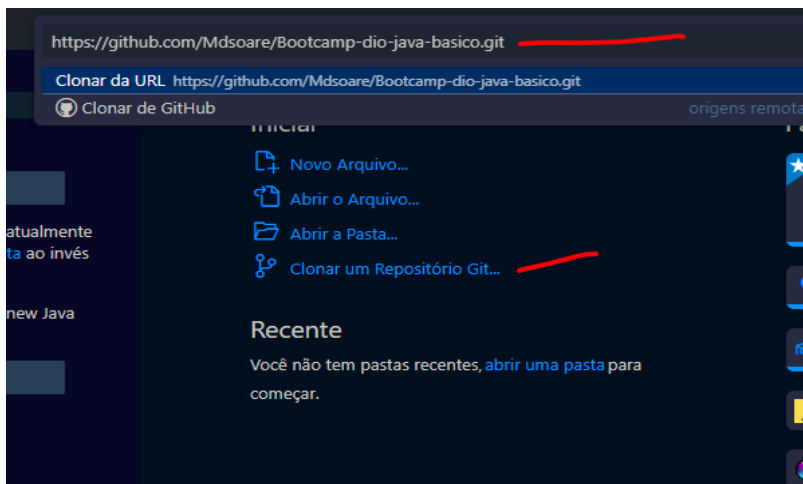
2.6) Agora, no VS Code clique em **“Abrir”**:



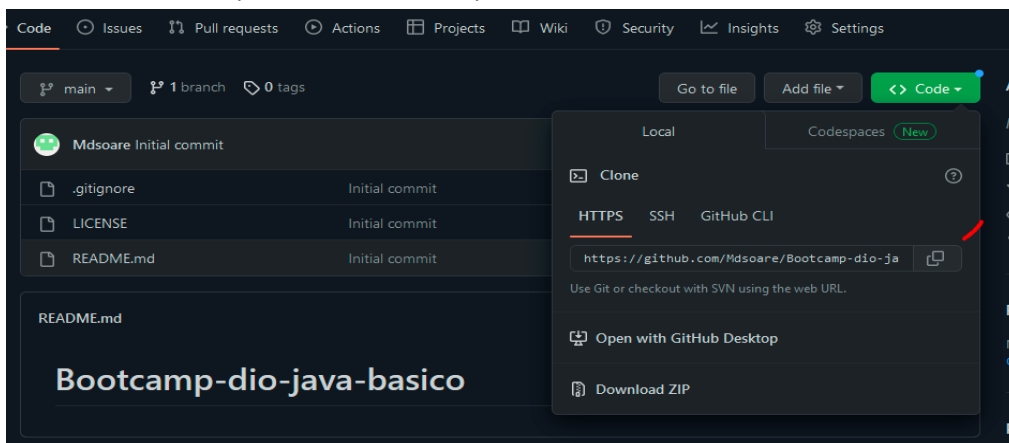
2.7) Agora clique em **“Clonar um Repositório Git...”**:



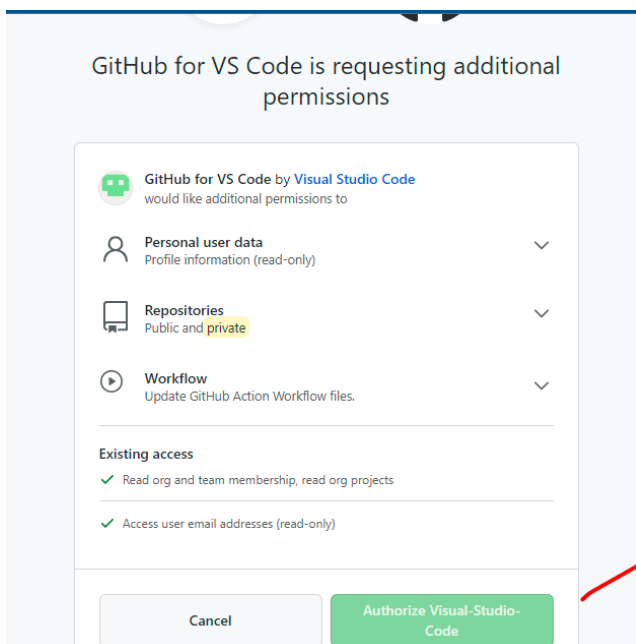
2.8) Basta inserir a url do github e clicar em “Clonar de GitHub”:



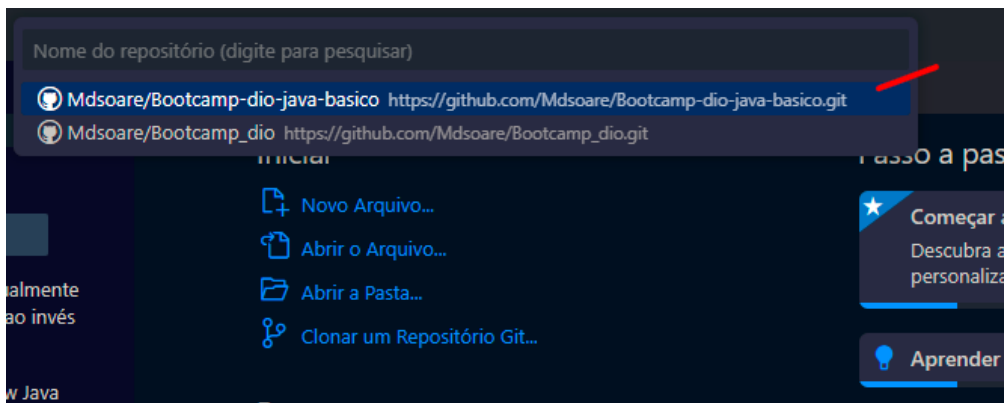
NOTA: Lembrando que a url deve ser copiada do seu GitHub:



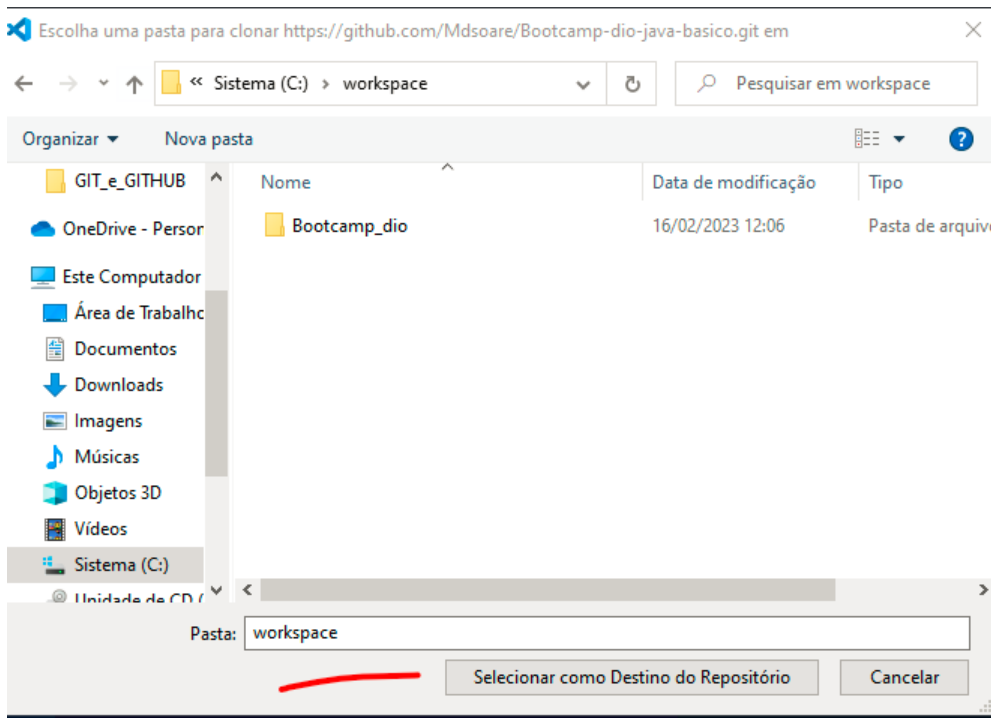
2.9) Uma nova url será aberta no seu navegador padrão. Clique em “Authorize Visual-Studio-Code”



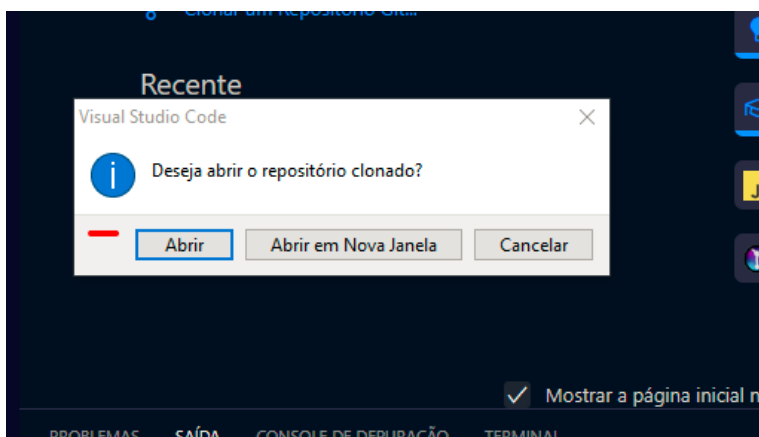
2.10) No VS Code basta escolher o repositório a ser clonado, no nosso caso “**Bootcamp-dio-java-basico**”



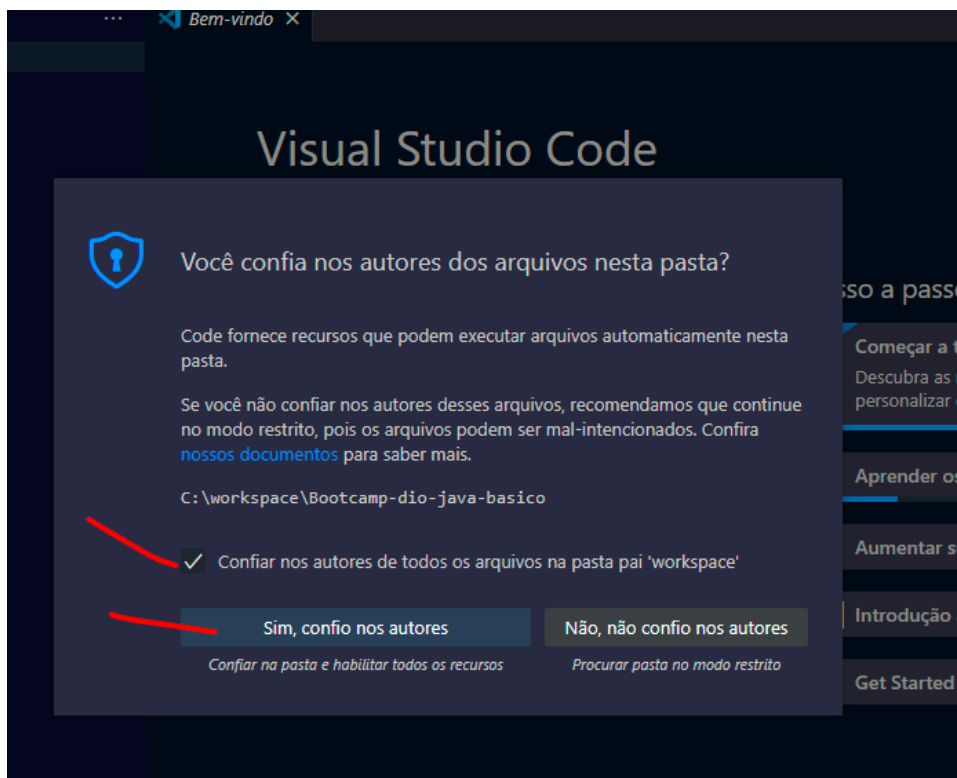
2.11) Na próxima tela será solicitado o local onde será clonado seu repositório remoto. Clique em “**Selecionar como Destino de Repositório**”:



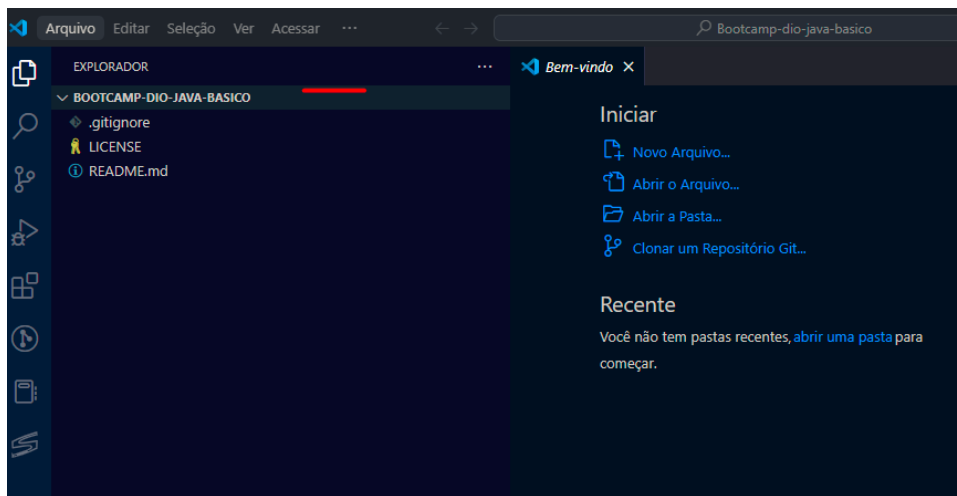
2.12) Agora clique em “**Abrir**”:



2.13) Verifique as opções marcadas abaixo:



Pronto! Seu repositório foi clonado com sucesso.



Fonte: <https://docs.github.com/pt/get-started/quickstart/create-a-repo?tool=cli>