

SESSÃO 2

Primeiros Passos com Git

Objetivos:

- alterar a *prompt* do *Git Bash*;
- inicializar um repositório;
- identificar o estado do repositório e o estado dos ficheiros;
- efetuar *commits*;
- visualizar o histórico de *commits*
- clonar repositórios

1. Alterar a *prompt* do *git*

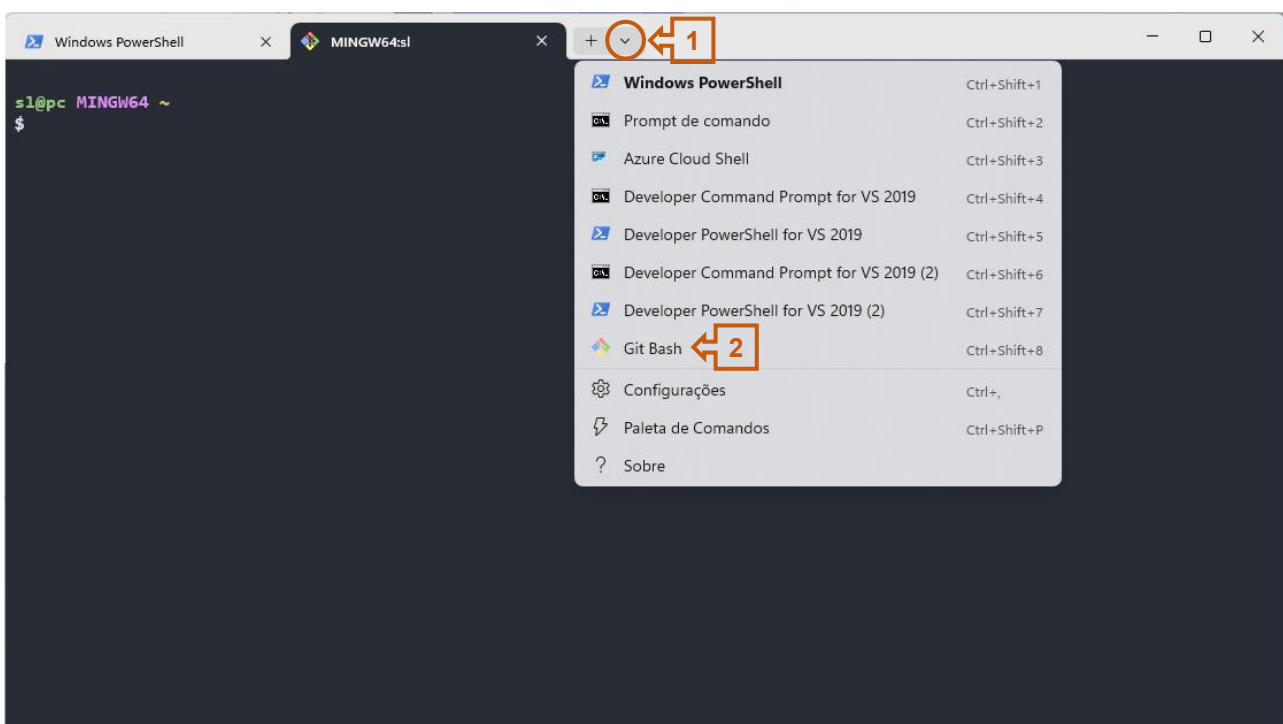
- fazer uma cópia do ficheiro **git-prompt.sh** e gravar a cópia com o nome **git-prompt-old.sh**;
- editar o ficheiro **git-prompt.sh** no **Visual Studio Code** e gravar como administrador.

Nota: <https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/How-to-customize-Git-Bash-Shell-prompt-settings>

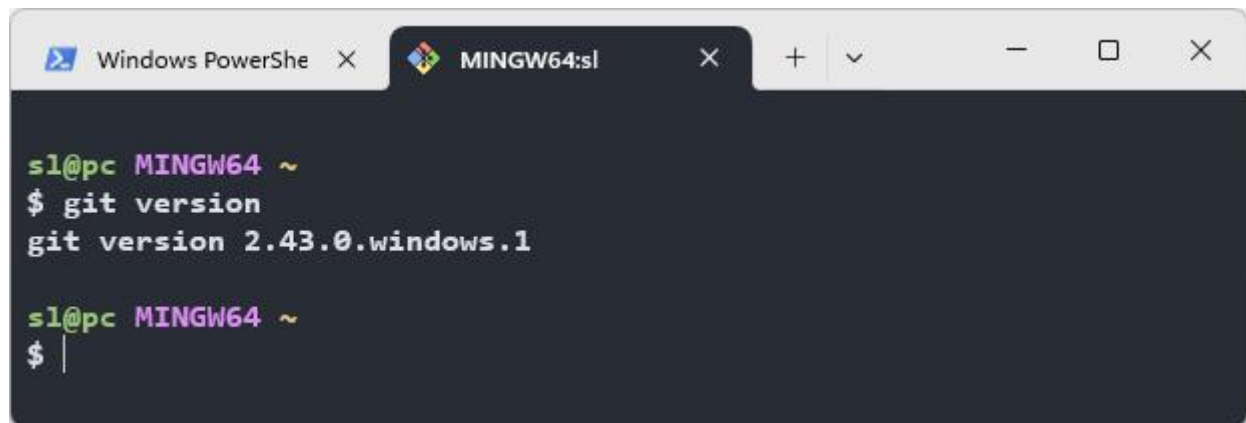
Transcrição _____ **Vídeo 4**

2. Aceder à *shell* do *git* através de

- clique com o botão direito do rato no ambiente de trabalho;
- no menu de atalho **Abrir no terminal**;
- na janela do terminal clicar na seta para baixo e no menu aberto clicar em **Git Bash**;



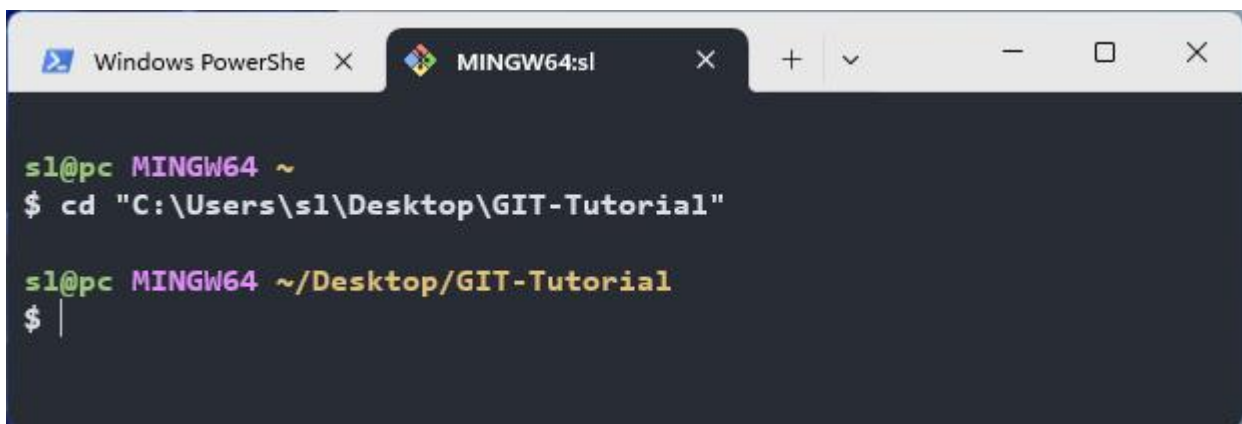
3. Verificar a versão do *git* através d o comando git version

A screenshot of a Windows PowerShell terminal window with a tab labeled 'MINGW64:sl'. The prompt is 'sl@pc MINGW64 ~'. The user enters '\$ git version' and the output is 'git version 2.43.0.windows.1'. The prompt returns to '\$'.

Repositório - é um projeto

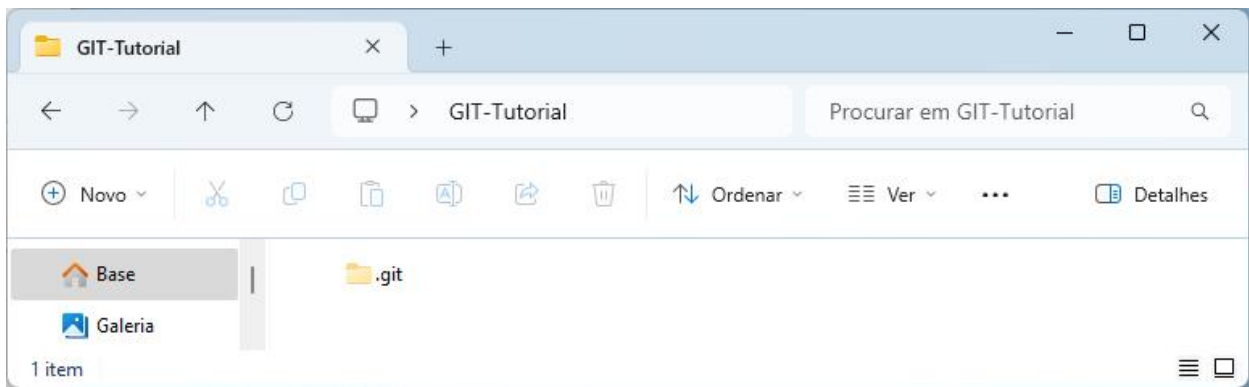
4. Inicializar um repositório. Para isso:

- a) criar uma pasta no ambiente de trabalho com o nome **GIT-Tutorial**;
- b) abrir a pasta no terminal através do comando **cd** [caminho para a pasta].

A screenshot of a Windows PowerShell terminal window with a tab labeled 'MINGW64:sl'. The prompt is 'sl@pc MINGW64 ~'. The user enters '\$ cd "C:\Users\sl\Desktop\GIT-Tutorial"' and the prompt changes to 'sl@pc MINGW64 ~/Desktop/GIT-Tutorial'. The prompt returns to '\$'.

- c) **inicializar o repositório** através do comando **git init**. Verifique que na pasta do projeto surge uma pasta oculta **.git** que não deve ser **alterada / eliminada**.

A screenshot of a Windows PowerShell terminal window with a tab labeled 'MINGW64:sl'. The prompt is 'sl@pc MINGW64 ~/Desktop/GIT-Tutorial'. The user enters '\$ git init' and the output is 'Initialized empty Git repository in C:/Users/sl/Desktop/GIT-Tutorial/.git/'. The prompt returns to '\$'. The next line shows 'sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)'.



5. Verificar o estado do repositório através do comando **git status**

```
sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git status
On branch main

No commits yet

nothing to commit (create/copy files and use "git add" to track)

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$
```

| | |
|--|--|
| On branch main | estamos no <i>main</i> |
| No commits yet | ainda não foram efetuados <i>commits</i> |
| nothing to commit (create/copy files and use "git add" to track) | não existe nada para fazer commit nem ficheiros nem pastas |

- d) clique com o botão direito do rato no ambiente de trabalho;
- d) no menu de atalho **Abrir no terminal**;
- e) na janela do terminal clicar na seta para baixo e no menu aberto clicar em **Git Bash**;

6. Criar conteúdo no projeto

- a) **Abrir** a pasta do projeto no **Visual Studio Code**.
- b) **Criar** um ficheiro HTML, **index.html**, com o seguinte conteúdo.

```

<> index.html > ...
1  <!DOCTYPE html>
2  <html lang="pt-pt">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>GIT-Tutorial :: Início</title>
7  </head>
8  <body>
9
10 </body>
11 </html>

```

c) **Verificar** novamente o **status**.

```

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        index.html

nothing added to commit but untracked files present (use "git add
" to track)

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$

```

| | |
|--------------------|--|
| On branch main | estamos no <i>main</i> |
| No commits yet | ainda não foram efetuados <i>commits</i> |
| Untracked files... | ficheiros não rastreados |

| Os ficheiros de um repositório estão num de três estados: | | |
|---|--|--|
| Untracked | Tracked | Committed |
| ficheiros ignorados pelo git | monitorizados pelo git mas ainda não submetidos a nenhum commit; git registou o conteúdo dos ficheiros que foram adicionados à <i>staging area</i> . | ficheiros incluídos no commit e encontram-se dentro do repositório |

7. Monitorizar o ficheiro, isto é, verificar quaisquer alterações que aconteçam no projeto. Para isso:

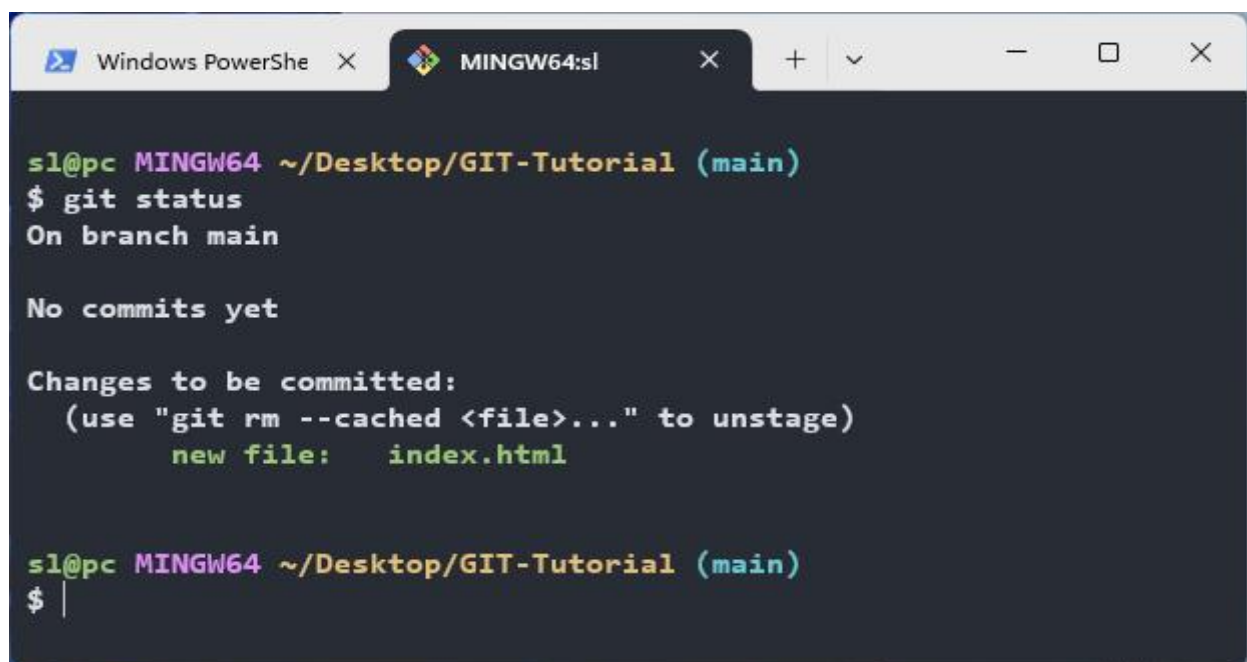
- a) **Aplicar** o comando **git add .** que significa que quaisquer alterações que ocorram na pasta **GIT-Tutorial** passam a ser monitorizados pelo **git**.



```
sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git add .

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$
```

- b) **Verificar o status.**



```
sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$
```

Commit - registo num repositório do estado de um ou vários ficheiros num determinado momento; *snapshot* [fotografia] do estado dos ficheiros / pastas que sofreram o *commit*.

8. Executar *commits*

- a) *Commit* inicial



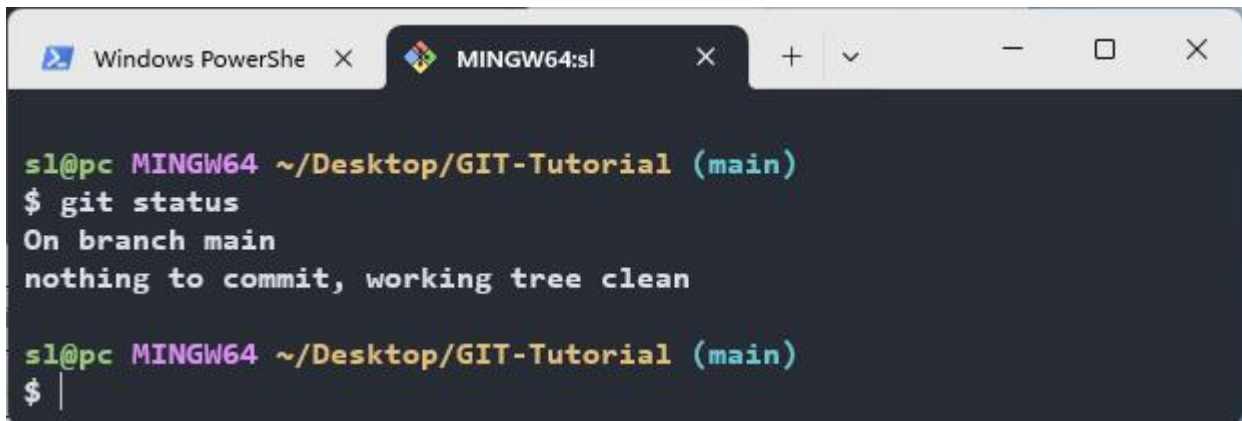
```
sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git commit -m "Commit inicial"
[main (root-commit) 131eceb] Commit inicial
1 file changed, 11 insertions(+)
create mode 100644 index.html

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$
```

git commit - m "Commit inicial"

-m [message] permite especificar uma mensagem; este comando significa que foi efetuada uma primeira fotografia do projeto

b) **Verificar o *status***



```
sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git status
On branch main
nothing to commit, working tree clean

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$
```

É indicado que não há nada para fazer *commit* todo o conteúdo já está registado no repositório.

9. Voltar ao **Visual Studio Code** para alterar o ficheiro **index.html** para incluir uma imagem:

- a) baixar o logótipo;
- b) criar a pasta **img** na pasta **GIT-Tutorial** e copiar a imagem para esta pasta.
- c) alterar a página **index.html** para incluir os elementos **h1** e **img**



```
<> index.html > ...
1  <!DOCTYPE html>
2  <html lang="pt-pt">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>GIT-Tutorial :: Início</title>
7  </head>
8  <body>
9      <h1>Primeiros passos no GIT</h1>
10     
12 </body>
</html>
```

d) **verificar o *status***


```
MINGW64:sl

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        img/

no changes added to commit (use "git add" and/or "git commit -a")

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$
```

A imagem acima indica que o ficheiro index.html que já está a ser monitorizado sofreu alterações e que foi criada uma pasta img que não está a ser monitorizada.

- e) **Digitar** o comando **git add .** para monitorizar todos os ficheiros
- f) **Verificar o estado**
- g) **Efetuar novo commit** com a mensagem “Adicionada imagem com o logotipo Git”

```
MINGW64:sl

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git add .

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   img/Git-logo.png
        modified:   index.html

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git commit -m "Adicionada imagem com o logotipo Git"
[main ac96b21] Adicionada imagem com o logotipo Git
2 files changed, 3 insertions(+), 1 deletion(-)
create mode 100644 img/Git-logo.png

sl@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$
```

10. Verificar os *commits* que foram efetuados.

- a) **Digitar** o comando **git log** para verificar os *commits* executados.

```
MINGW64:sl
s1@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git log
commit ac96b2153852575a7549aa3c155798b01950253f (HEAD -> main)
Author: s1 <156254633+s1bca@users.noreply.github.com>
Date: Thu Mar 7 22:32:31 2024 +0000

    Adicionada imagem com o logotipo Git

commit 131eceb9764e7429ab7318842a6fb9fa2516d11e
Author: s1 <156254633+s1bca@users.noreply.github.com>
Date: Wed Mar 6 08:13:45 2024 +0000

    Commit inicial

s1@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$
```

Na janela acima é possível identificar as seguintes informações:

- 🖥 **ac96.... hash do commit** - identificador único do *commit*;
- 🖥 **Author** nome e *email* do autor do *commit*
- 🖥 **Date** data e hora em que o *commit* foi executado

b) o comando **git log --oneline** devolve uma listagem condensada dos *commits*.

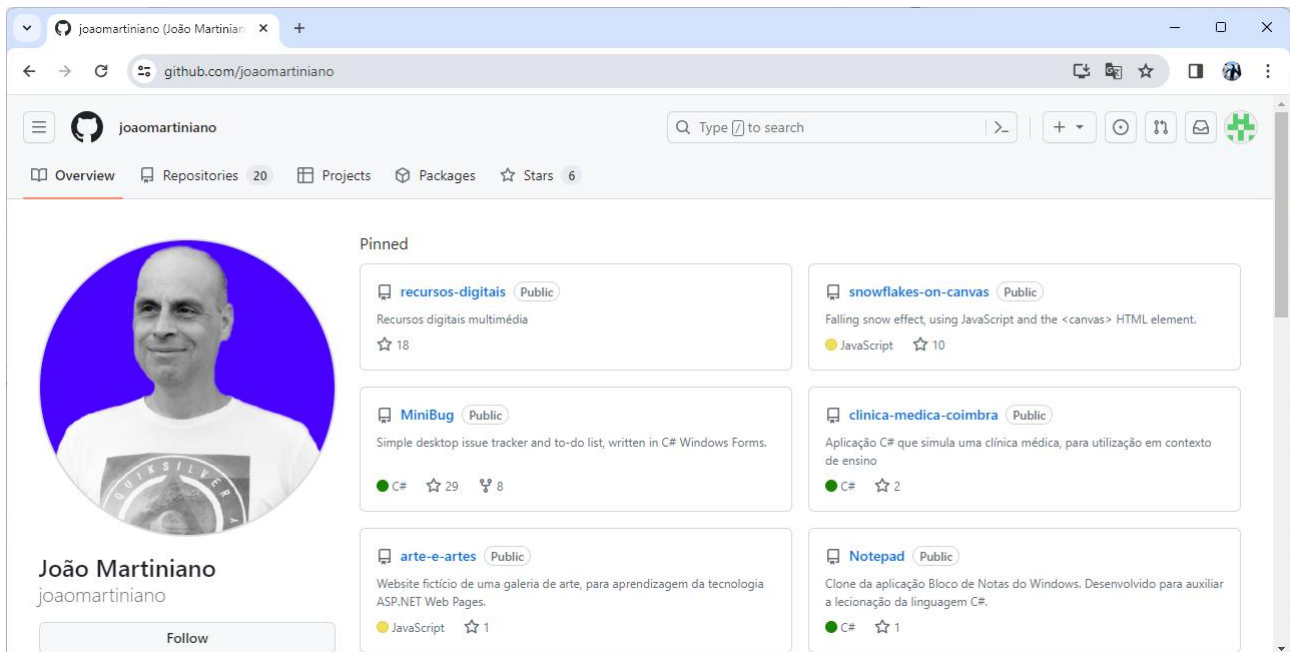
```
MINGW64:sl
s1@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$ git log --oneline
ac96b21 (HEAD -> main) Adicionada imagem com o logotipo Git
131eceb Commit inicial

s1@pc MINGW64 ~/Desktop/GIT-Tutorial (main)
$
```

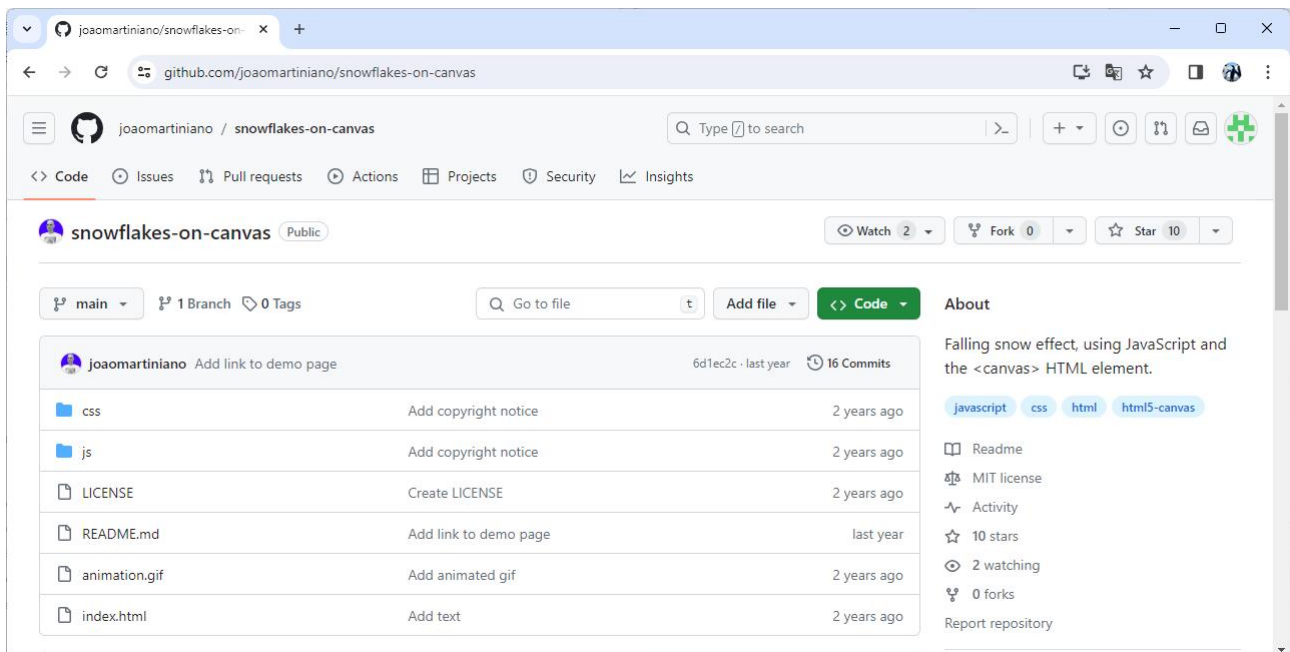

11. Clonar um repositório com o git

a) ir para a página do git hub

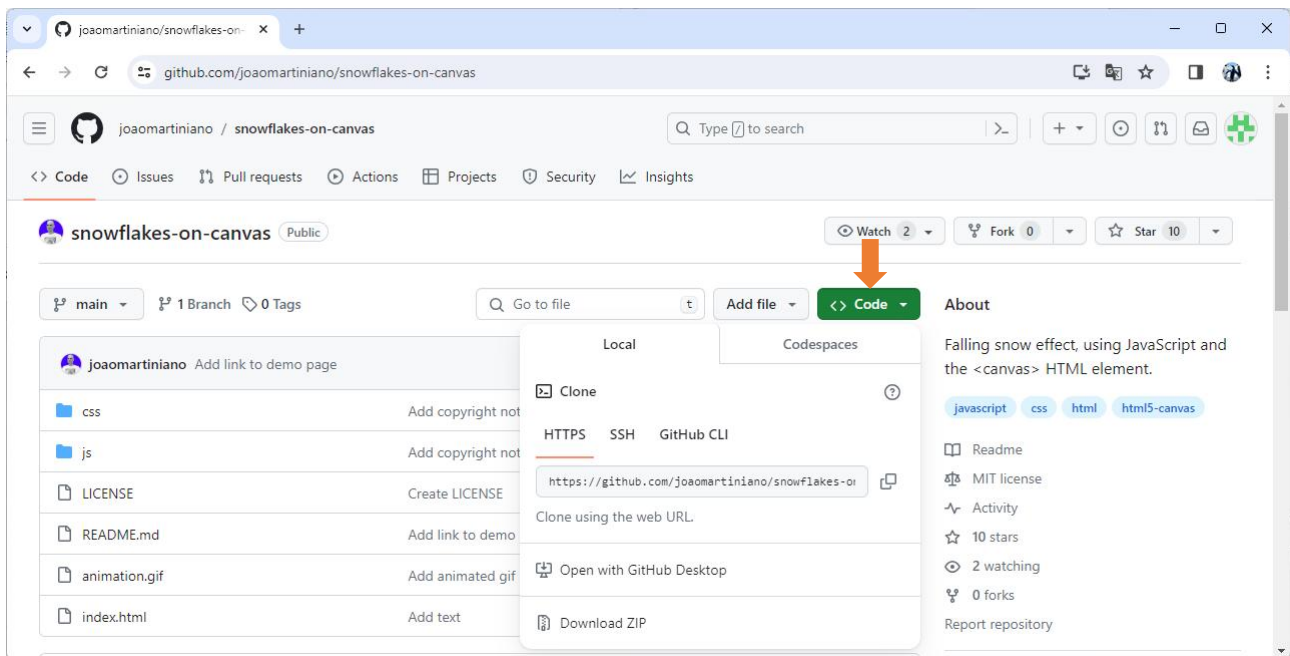
<https://github.com/joao martiniano>



b) clicar no nome do repositório snowflakes-on-canvas, por exemplo;



c) clicar em **code**



Link - pode ser usado para clonar o repositório

Download zip - cria um zip com o repositório

- d) **Criar uma nova pasta** para receber o clone do repositório [snowflakes-clone]
- e) **Abrir numa nova janela da linha de comandos Git Bash** a pasta snowflakes-clone
- f) **Digitar o comando git clone** “link do repositório”

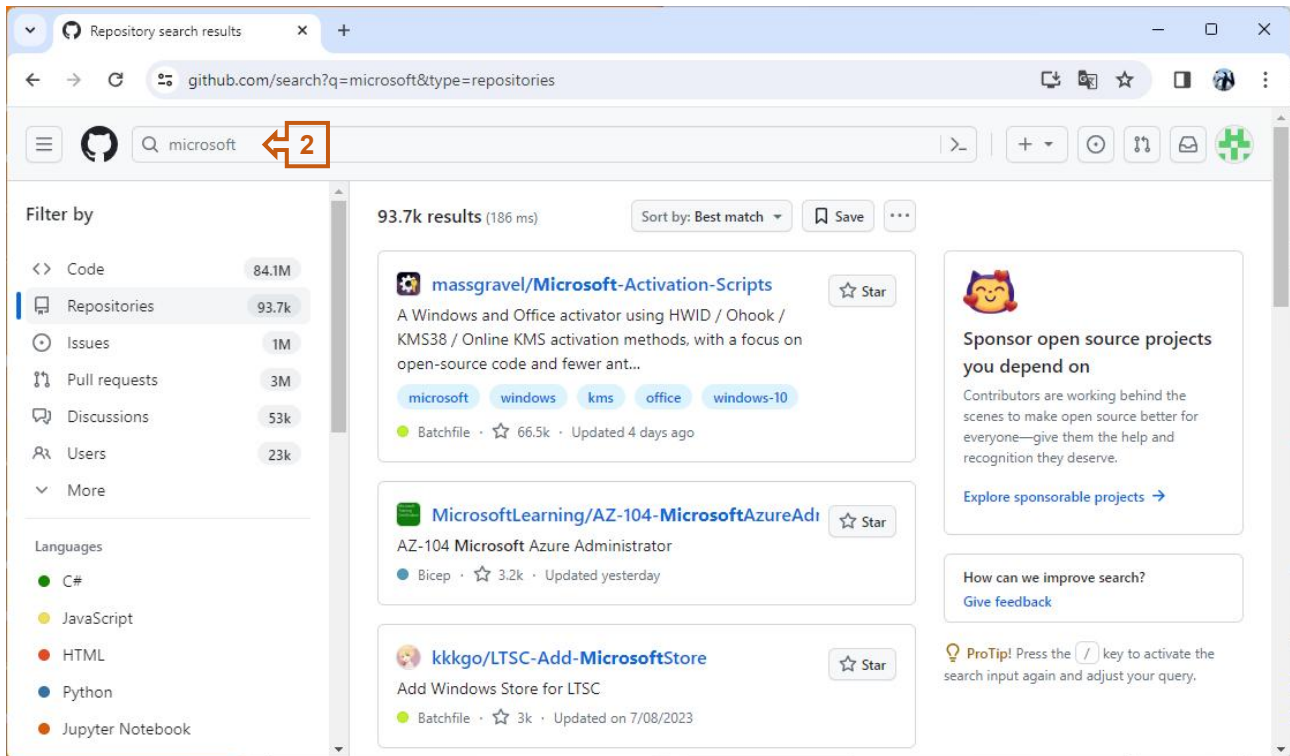
git clone <https://github.com/joaomartiniano/snowflakes-on-canvas.git>

```
MINGW64:sl
s1@pc MINGW64 ~
$ cd "C:\Users\s1\Desktop\snowflakes-clone"

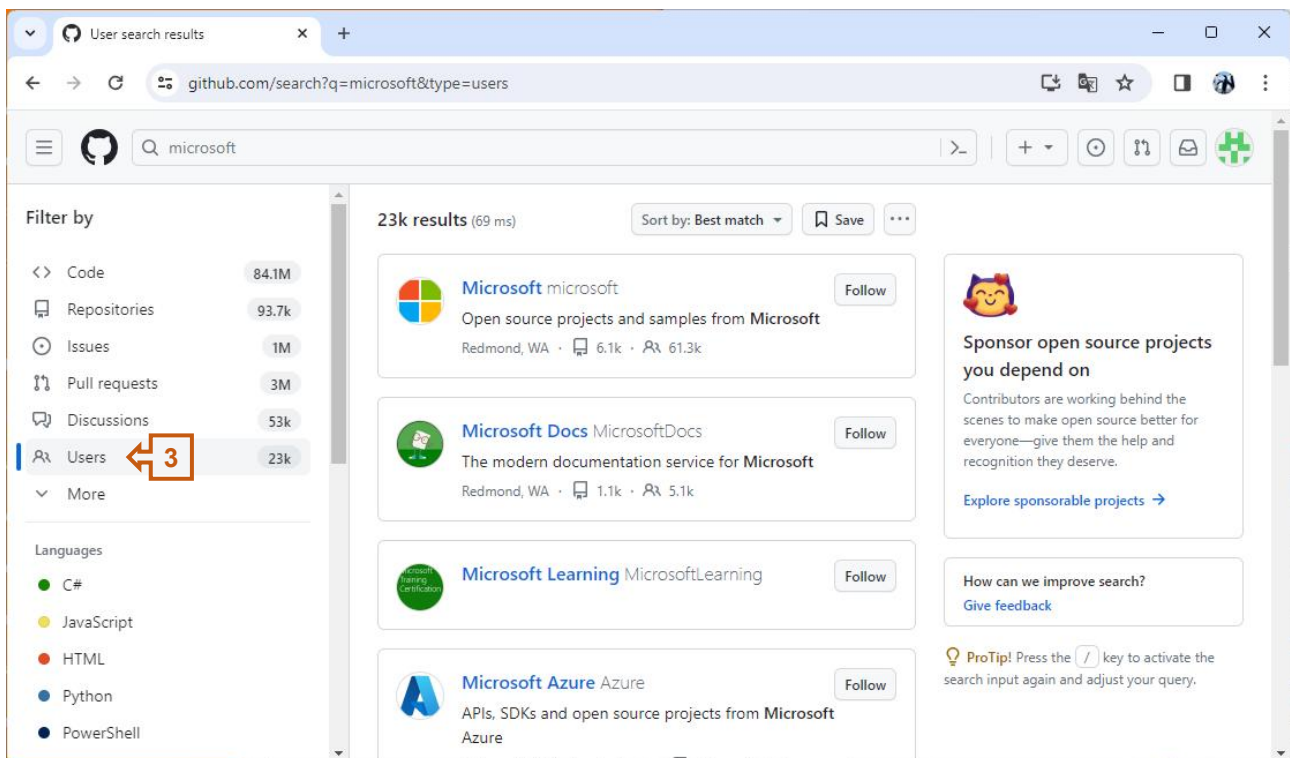
s1@pc MINGW64 ~/Desktop/snowflakes-clone
$ git clone https://github.com/joaomartiniano/snowflakes-on-canvas.git
Cloning into 'snowflakes-on-canvas'...
remote: Enumerating objects: 66, done.
remote: Counting objects: 100% (66/66), done.
remote: Compressing objects: 100% (40/40), done.
remote: Total 66 (delta 22), reused 62 (delta 20), pack-reused 0
Receiving objects: 100% (66/66), 1.40 MiB | 1.24 MiB/s, done.
Resolving deltas: 100% (22/22), done.

s1@pc MINGW64 ~/Desktop/snowflakes-clone
$
```

1. Aceder ao site <https://github.com/>
2. Na caixa de pesquisa escrever **Microsoft**

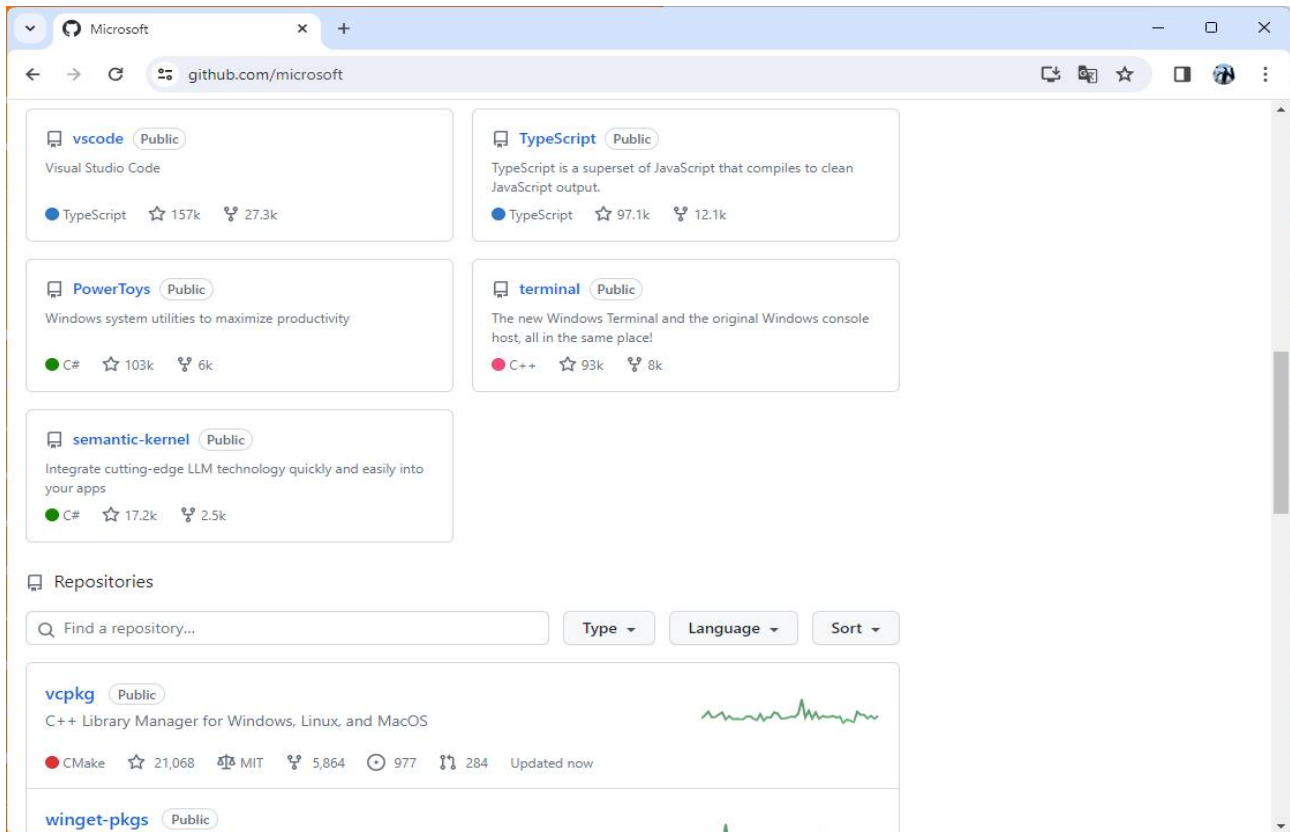


3. Para aceder à página da **Microsoft** clicar em **Users** no menu da esquerda.

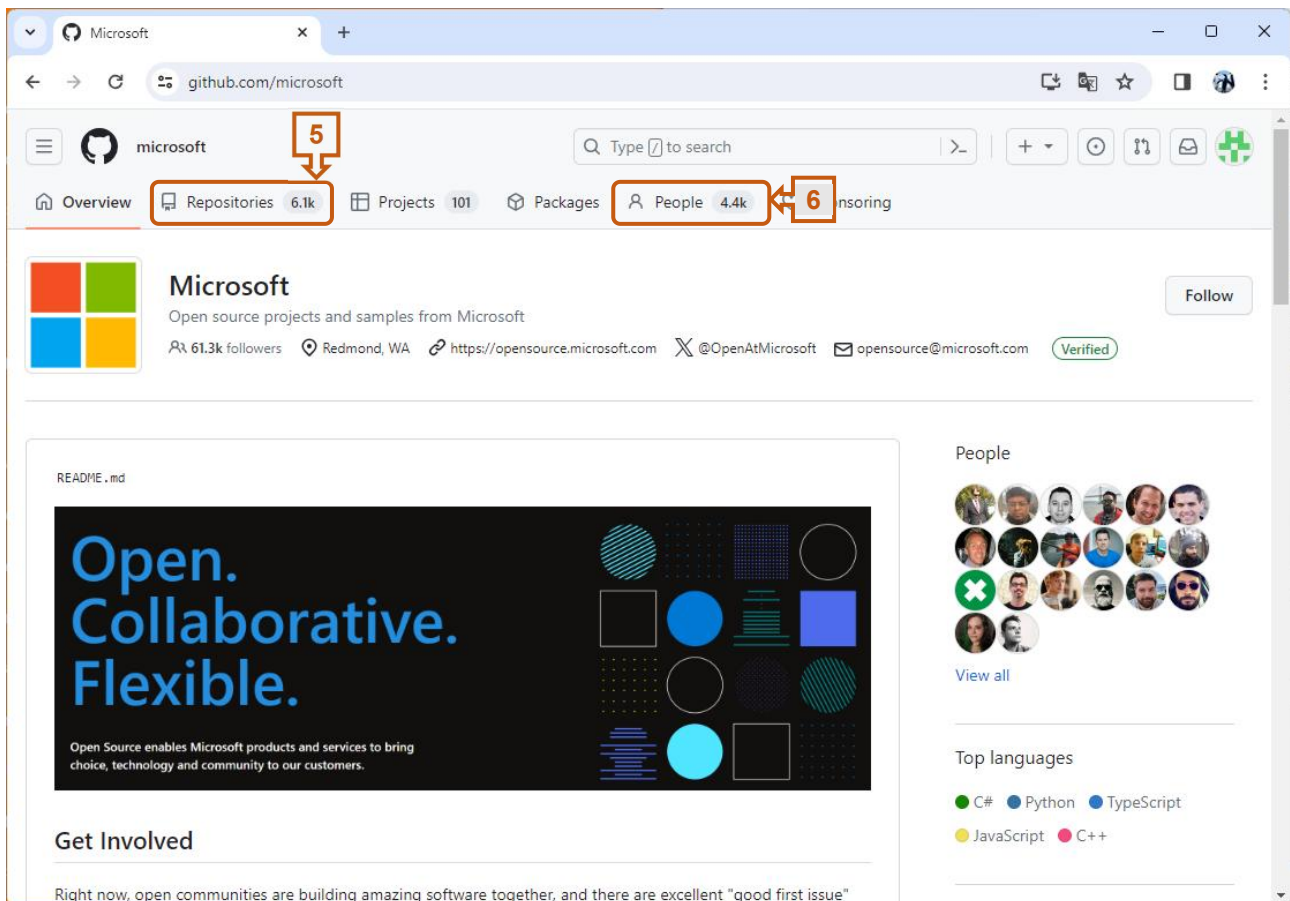


4. Na página principal da **microsoft** destacam-se as seguintes áreas:

a) **Pinned** - repositórios em destaque.

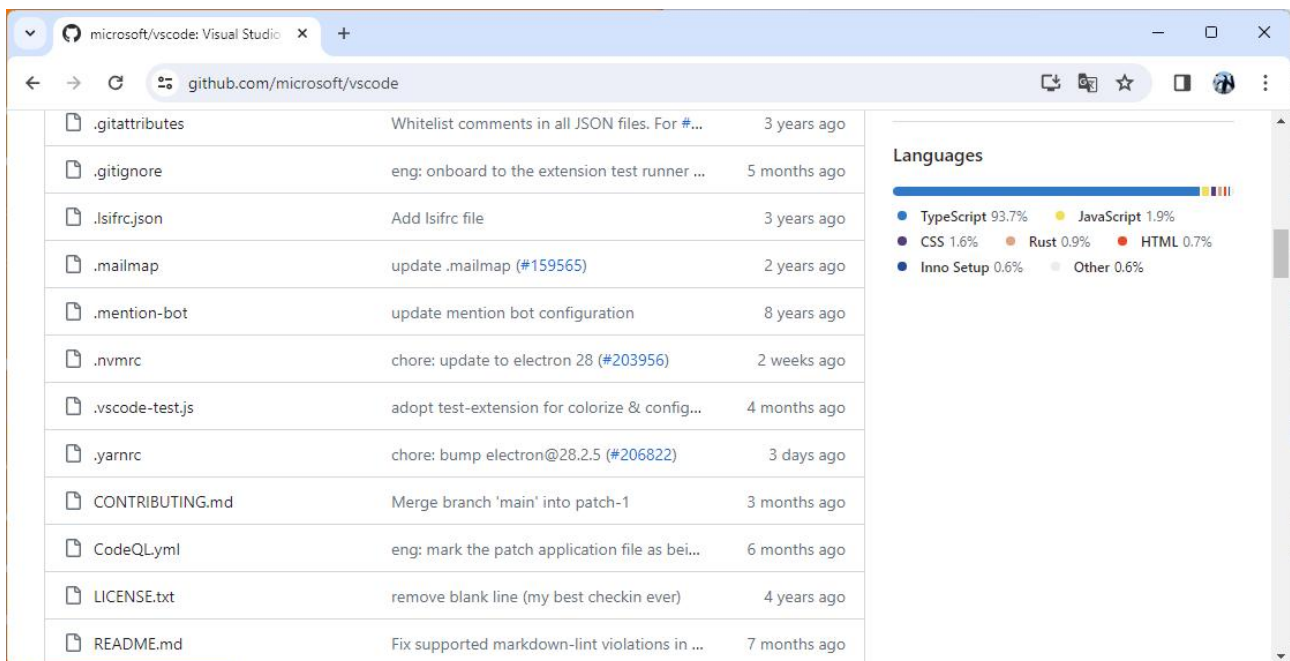
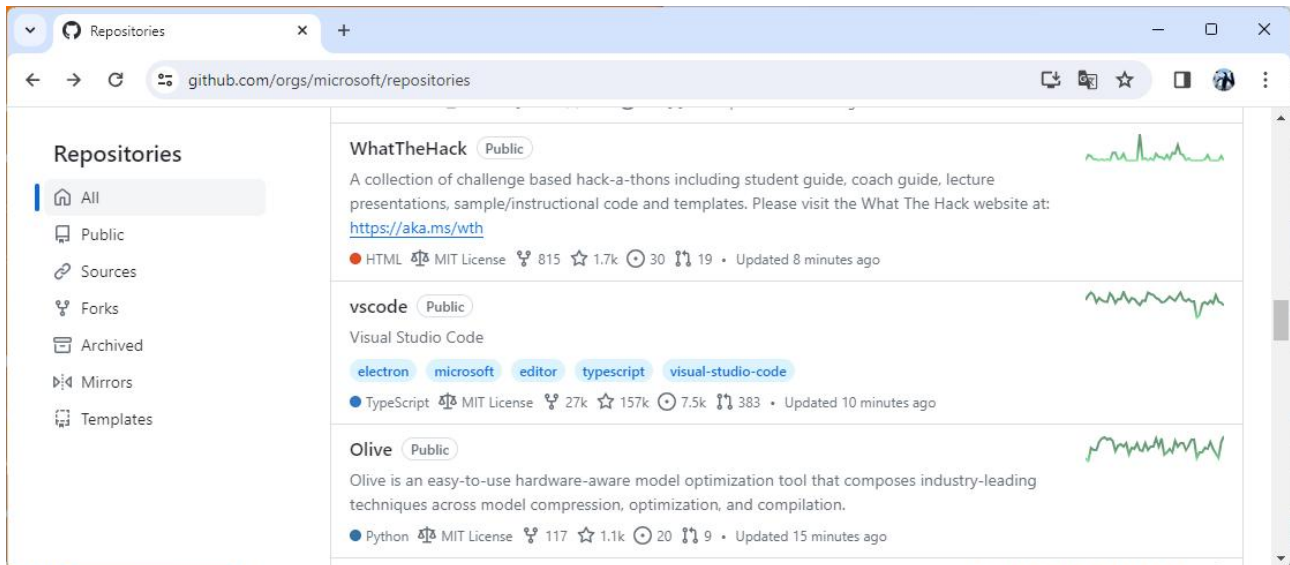


5. Para saber o **total de repositórios da Microsoft** ir para o topo do página e na imagem é possível ver 6.1k (cerca de 6100 repositórios)



6. Ainda nesta página é possível ver o **total de pessoas associadas** People 4.4k.

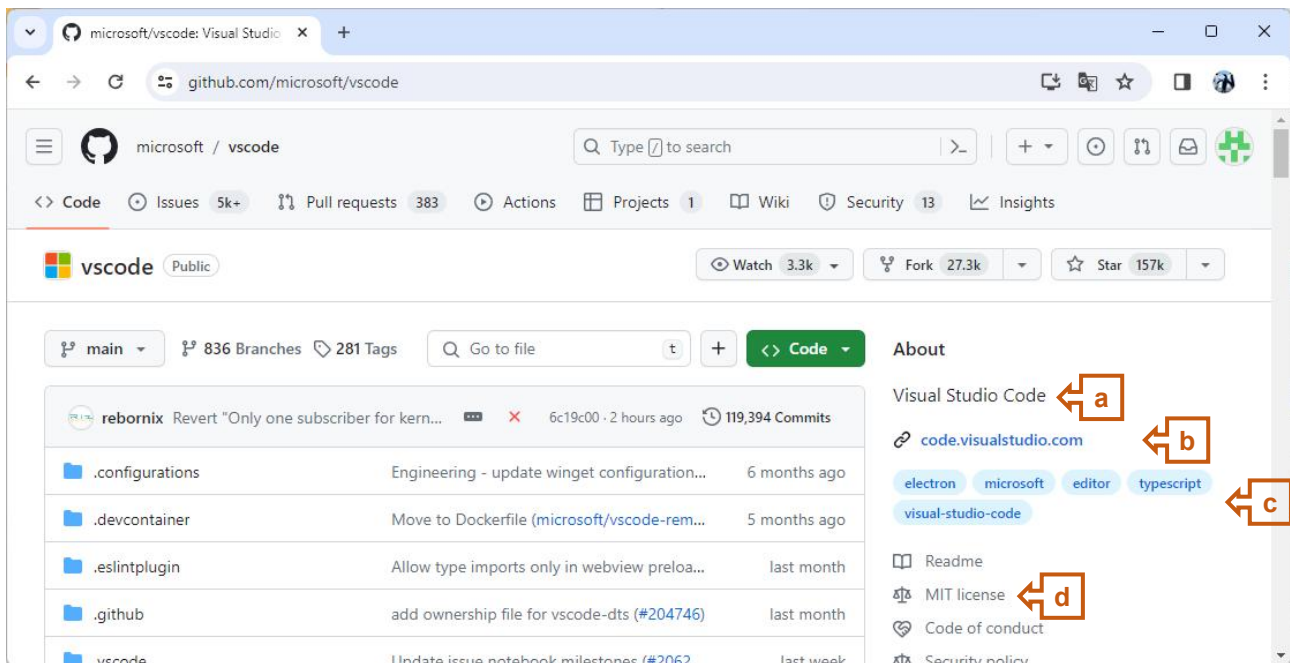
7. Ainda na página **Git Hub** da **Microsoft** aceder ao repositório do **visual studio code**.



Na listagem é possível identificar o ficheiro **README.md** escrito na linguagem **Markdown**.

8. Na página **Git Hub** do repositório do **Visual Studio Code** é possível identificar:

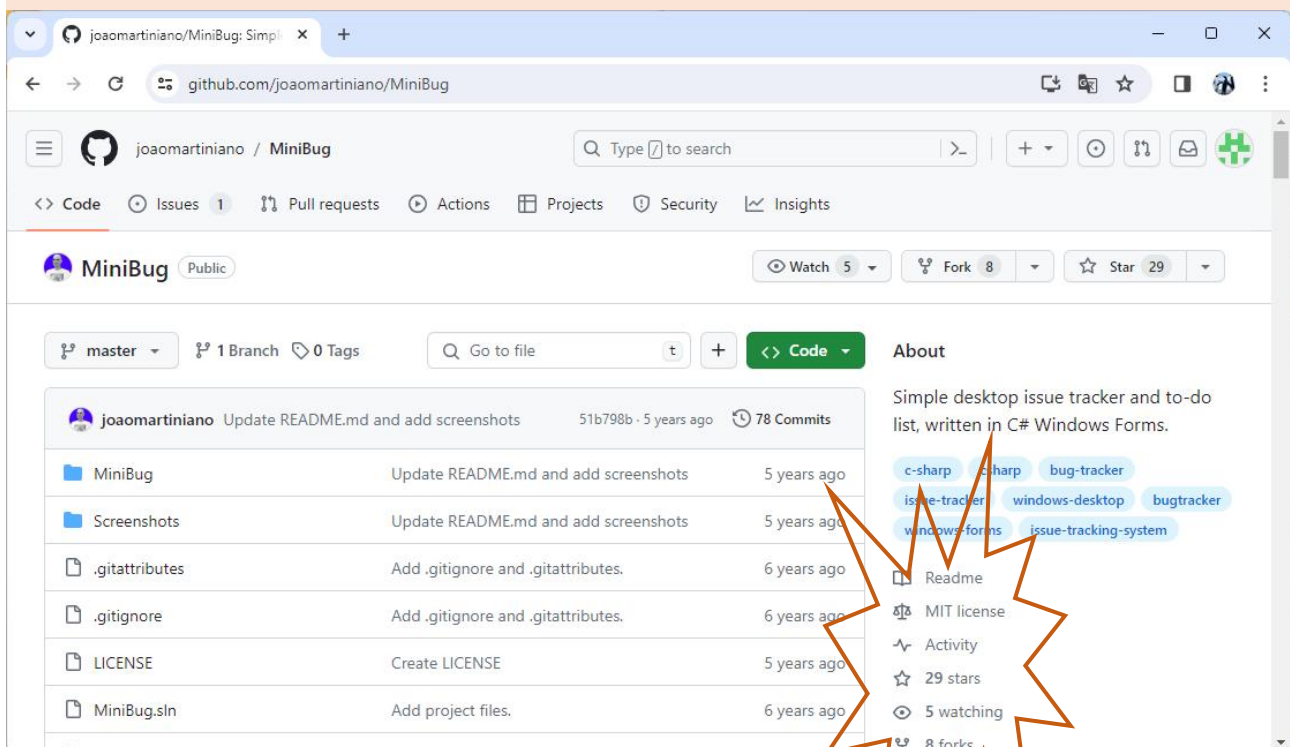
- a) nome
- b) site oficial
- c) tags associadas
- d) licença



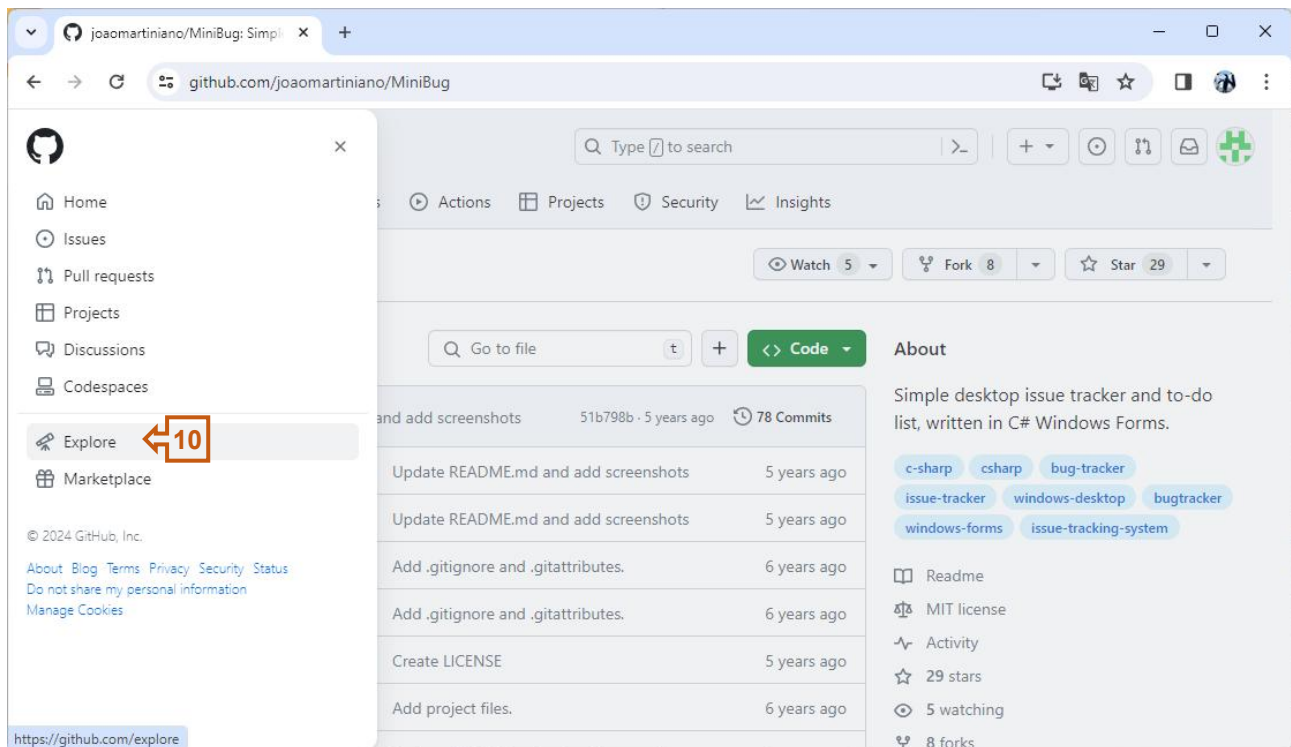
9. Analisando um outro repositório

- a) Licença - MIT license
- b) Stars - 29
- c) Forks - 8

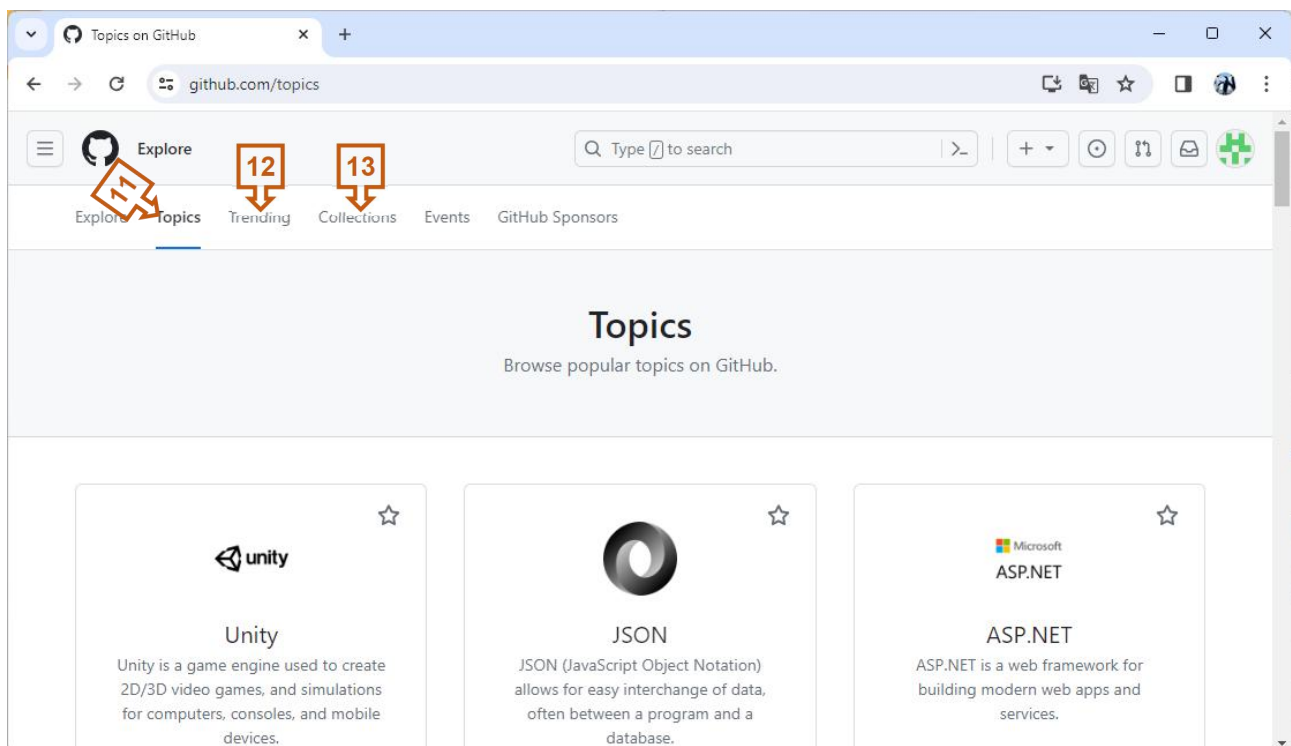
Fork - é um novo repositório que partilha configurações de código e visibilidade com o repositório “upstream” original.



10. Continuando a **Explorar** o site do **Git hub**, através do botão de menu, clicar em **Explore**



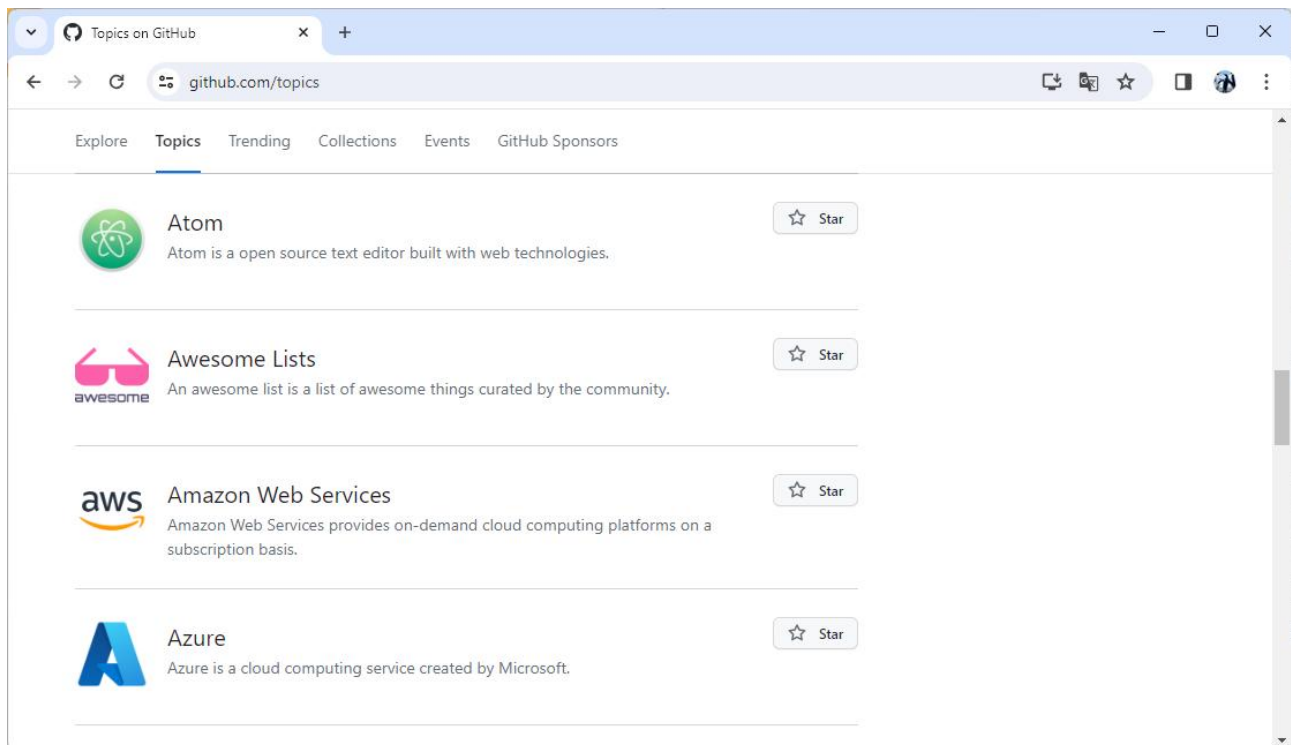
11. E na página aberta clicar em **Topics**



12. Ainda na página aberta através do **Explore** clicar em **Trending** [repositórios com popularidade crescente].

13. Ainda na página aberta através do **Explore** clicar em **Collections** [repositórios organizados por tema].

14. Nem só de programação vive o **Git Hub** também é possível encontrar repositórios de outros temas, por exemplo, através de **Topics**, localizar o repositório **Awesome Lists**.



vinta/awesome-python é um repositório de lista de recursos relacionados com python

