

# Manual Prático do GitHub Prof. Marcio - Passo a Passo

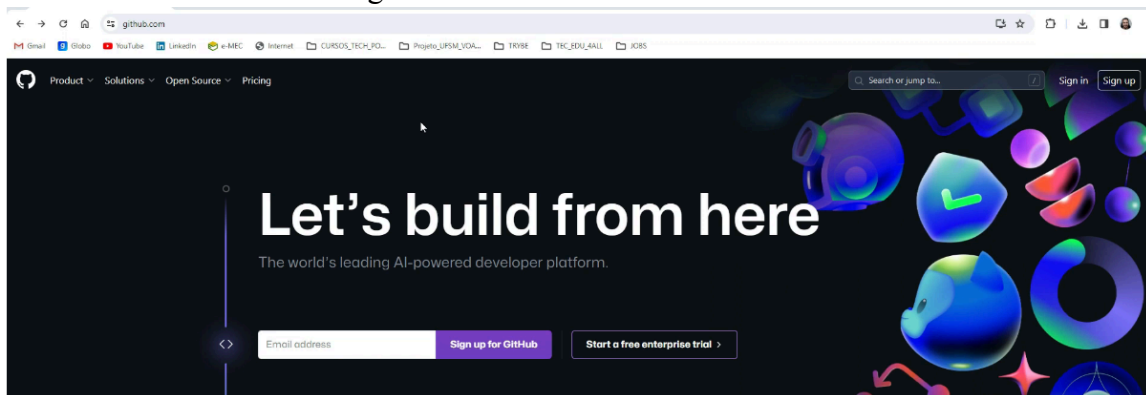
## O que é o GitHub?

O **GitHub** é uma plataforma para hospedagem de código que usa o **Git**, um sistema de controle de versão. Ele permite que você acompanhe alterações no seu código e colabore com outras pessoas. Passos:

### 1. Criar uma Conta no GitHub

1. Acesse: <https://github.com>
2. Clique em **Sign up**.
3. Preencha:
  - Nome de usuário
  - E-mail
  - Senha
4. Confirme o e-mail enviado pela GitHub.

Criar o cadastro na conta do github



### 2. Instalar o Git no seu computador

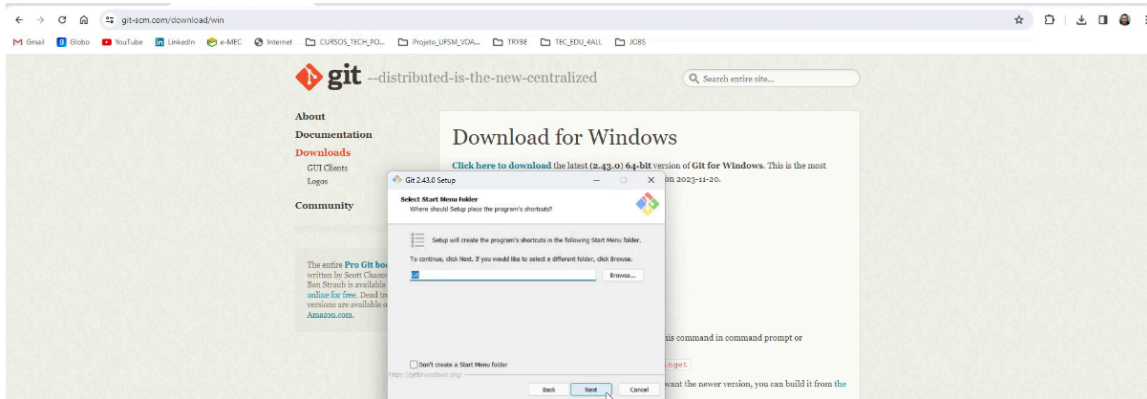
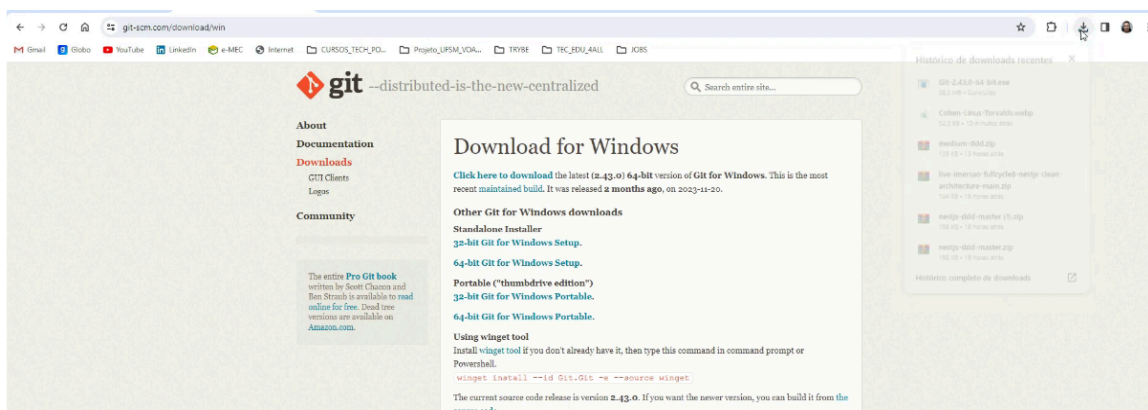
**No Linux (ex: Manjaro):**

```
sudo pacman -S git
```

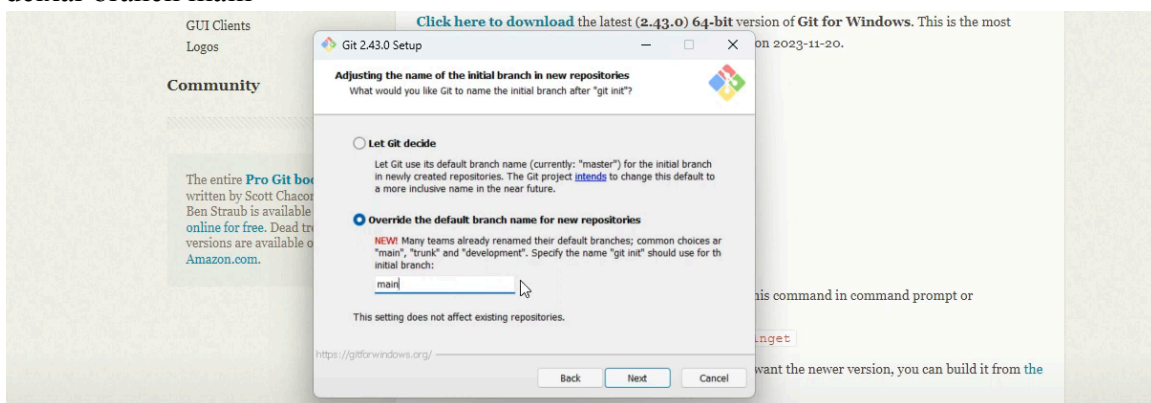
**No Windows:**

- Baixe e instale: <https://git-scm.com>

Buscar git download.



deixar branch main



Deixar tudo next.

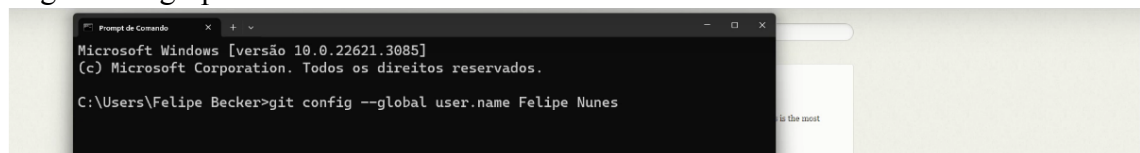
### ✓ 3. Configurar o Git no Terminal

bash

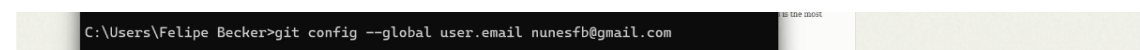
```
git config --global user.name "Seu Nome"
```

```
git config --global user.email "seuemail@exemplo.com"
```

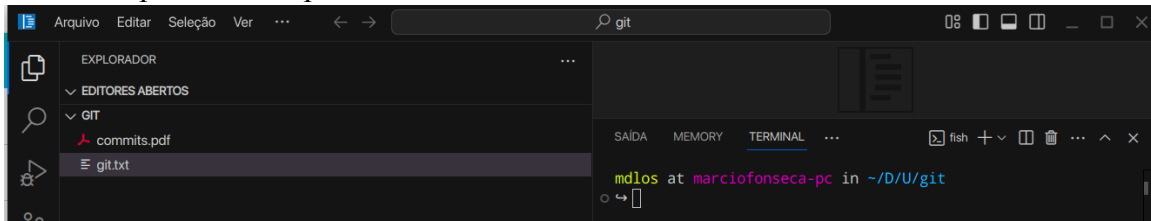
Registra no git por meio do terminal *nome*



email



Criar uma pasta e abrir pelo vscode ou terminal



Criar um arquivo, aqui está sendo criado o git.txt

## ✓ 4. Criar um Repositório no GitHub

1. Acesse o site do GitHub e faça login.
2. Clique em + no canto superior direito → **New repository**.
3. Preencha:
  - Nome do repositório
  - Descrição (opcional)
  - Público ou Privado
4. Clique em **Create repository**.

## ✓ 5. Subir Arquivos Locais para o GitHub pela Primeira Vez (sem clonar)

Se você já tem um projeto local e quer enviar para um repositório novo no GitHub:

**a) Crie o repositório no GitHub (como mostrado no passo 4)**

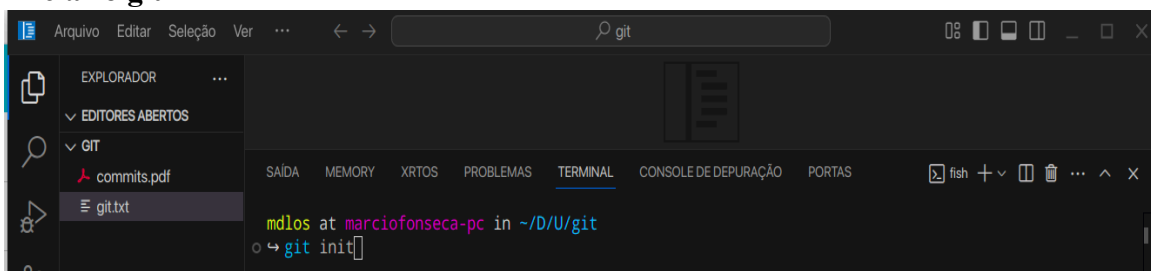
**b) No terminal, entre na pasta do seu projeto:**

```
bash
cd /caminho/do/seu/projeto
```

**c) Inicialize o repositório Git:**

```
bash
git init
```

**Iniciar o git**



```
bash
git status //git status Mostra arquivos modificados
```

```
mdlos at marciofonseca-pc in ~/D/U/git
↳ 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:   git.txt

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

## d) Adicione os arquivos:

bash

git add .

```
mdlos at marciofonseca-pc in ~/D/U/git
↳ git add .
↳ git commit -m "manual git"

mdlos at marciofonseca-pc in ~/D/U/git
↳ git add git.txt
```

ou

```
mdlos at marciofonseca-pc in ~/D/U/git
↳ git add .
↳ git commit -m "manual git"
```

```
mdlos at marciofonseca-pc in ~/D/U/git
↳ git add .
mdlos at marciofonseca-pc in ~/D/U/git
↳ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   git.txt
```

## e) Faça o commit:

bash

git commit -m "Primeiro commit"

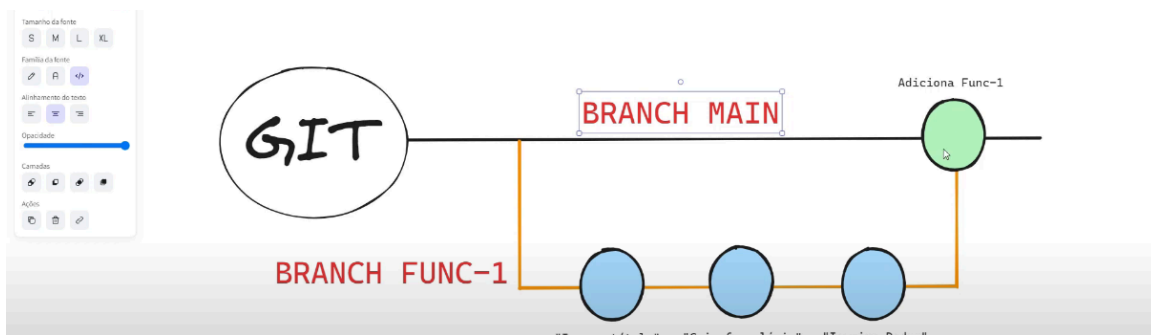
```
mdlos at marciofonseca-pc in ~/D/U/git
↳ git commit -m "Primeiro commit - criando arquivo git.txt"
[main 3dbc767] Primeiro commit - criando arquivo git.txt
1 file changed, 6 insertions(+), 1 deletion(-)
```

bash

git log // verificar se os commits foram salvos por meio de um hash

```
git commit -m 'mensagem git' -m 'mensagem git'
mdlos at marciofonseca-pc in ~/D/U/git
git log
commit 3dbc767f9b2b3645a5cf1bbd4e6ea3721fae3a5a (HEAD -> main)
```

neste primeiro momento estamos no branch main



## f) Conecte ao repositório remoto:

bash

```
git remote add origin https://github.com/seuusuario/seurepositorio.git
```

## g) Envie os arquivos para o GitHub:

bash

```
git push -u origin main
```

⚠ Se o GitHub estiver usando a branch master, troque main por master. O comando:

bash

```
git checkout -b func_1
```

significa:

🟡 Criar e mudar para uma nova branch chamada func\_1.

## 🔍 Explicação:

- git checkout: comuta (troca) entre branches.
- -b func\_1: cria uma nova branch chamada func\_1 e já muda para ela.

## ✅ Quando usar?

Esse comando é útil quando você quer começar a trabalhar em uma nova funcionalidade, correção ou experimento, mantendo o código separado da branch principal (main ou master).

## 🧠 Exemplo de fluxo completo:

bash

```
git checkout -b func_1 # cria e entra na nova branch
```

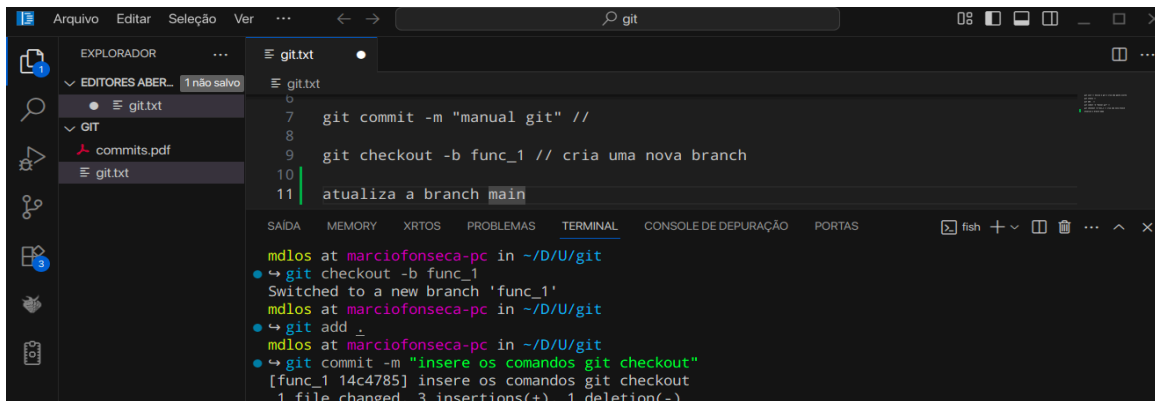
```
# faz suas alterações...
```

```
git add .
```

```
git commit -m "Adiciona funcionalidade 1"
```

```
git push origin func_1          # envia a nova branch para o GitHub
```

Depois, no GitHub, você pode abrir um **Pull Request** para juntar essa branch (func\_1) com a branch principal.



The screenshot shows a code editor with a file named `git.txt` containing the following commands:

```
6  
7 git commit -m "manual git" //  
8  
9 git checkout -b func_1 // cria uma nova branch  
10  
11 atualiza a branch main
```

The terminal window at the bottom shows the execution of these commands:

```
mdlos at marciofonseca-pc in ~/D/U/git  
↳ git checkout -b func_1  
Switched to a new branch 'func_1'  
mdlos at marciofonseca-pc in ~/D/U/git  
↳ git add .  
mdlos at marciofonseca-pc in ~/D/U/git  
↳ git commit -m "insere os comandos git checkout"  
[func_1 14c4785] insere os comandos git checkout  
1 file changed, 3 insertions(+), 1 deletion(-)
```

Para sair da branch func\_1 e voltar para a branch main, use:

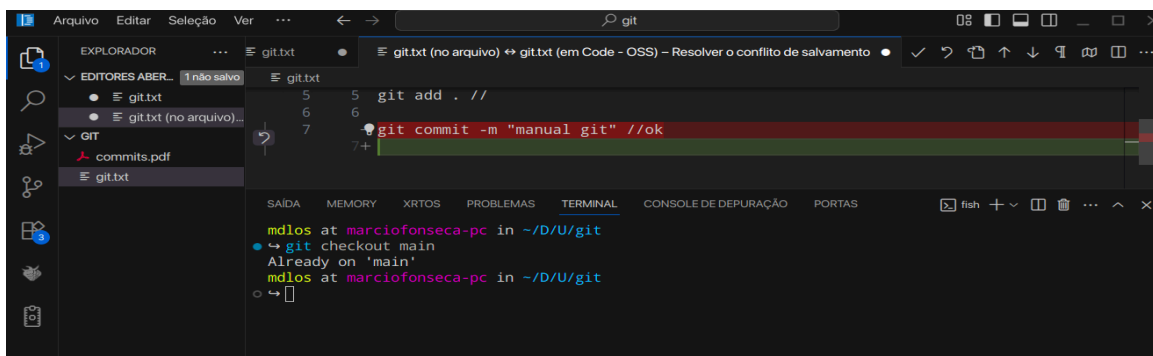
## ✓ Passo a passo:

### 1. Verifique a branch atual (opcional):

```
bash  
git branch
```

### 2. Mude para a branch main:

```
bash  
git checkout main
```



The screenshot shows a code editor with a file named `git.txt` containing the following commands:

```
5 git add . //  
6  
7 git commit -m "manual git" //ok
```

The terminal window at the bottom shows the execution of these commands:

```
mdlos at marciofonseca-pc in ~/D/U/git  
↳ git checkout main  
Already on 'main'  
mdlos at marciofonseca-pc in ~/D/U/git  
↳
```

Se você estiver usando uma versão mais recente do Git, também pode usar:

```
bash  
git switch main
```

O comando:

```
bash  
git merge func_1
```

significa:

↩ Mesclar o conteúdo da branch `func_1` na branch atual (normalmente `main`).\*\*\*

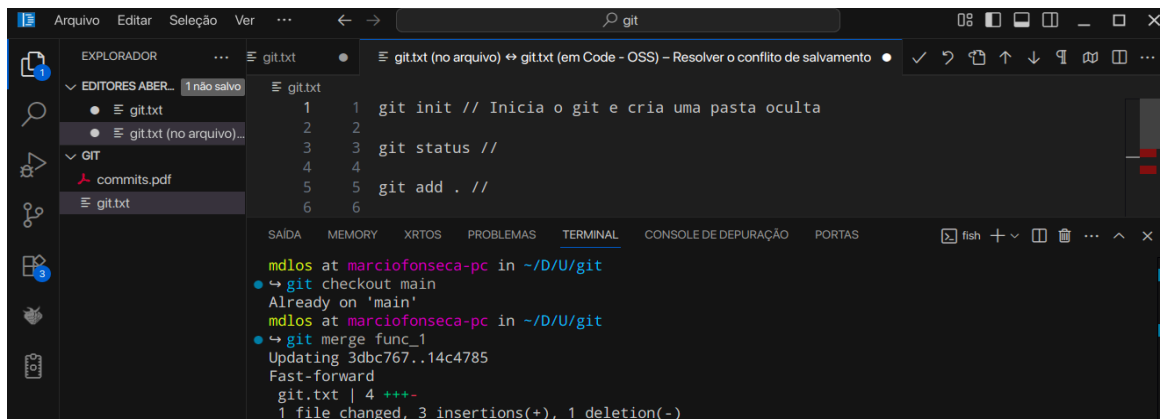
## ✓ Passo a Passo Completo para fazer o merge:

1. Mude para a branch que receberá as mudanças (ex: `main`):

```
bash
git checkout main
```

2. Execute o merge da branch `func_1`:

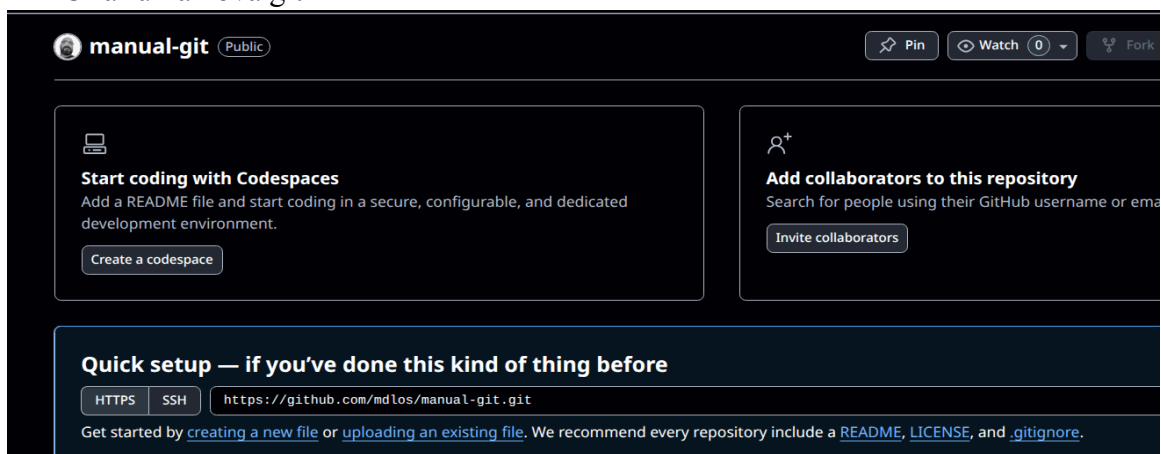
```
bash
git merge func_1
```



The screenshot shows a VS Code interface with a terminal window open. The terminal output is as follows:

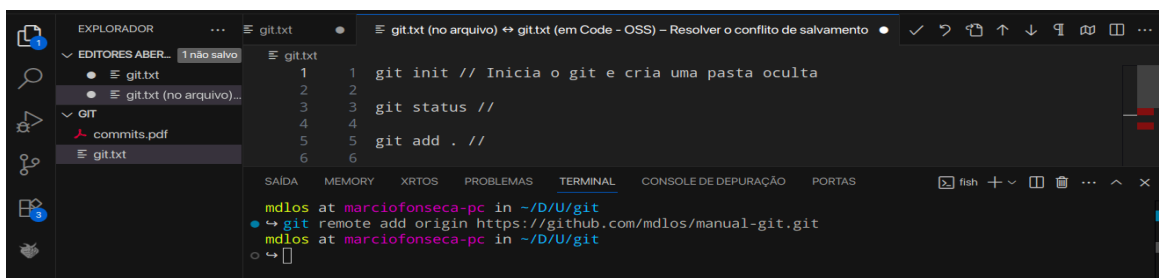
```
mdlos at marciofonseca-pc in ~/D/U/git
↳ git checkout main
Already on 'main'
mdlos at marciofonseca-pc in ~/D/U/git
↳ git merge func_1
Updating 3dbc767..14c4785
Fast-forward
 git.txt | 4 +++-
 1 file changed, 3 insertions(+), 1 deletion(-)
```

\*\*\* Criar uma nova git



### Esse comando:

```
bash
git remote add origin https://github.com/mdlos/manual-git.git
```



The screenshot shows a VS Code interface with a terminal window open. The terminal output is as follows:

```
mdlos at marciofonseca-pc in ~/D/U/git
↳ git remote add origin https://github.com/mdlos/manual-git.git
mdlos at marciofonseca-pc in ~/D/U/git
↳
```

significa:

 Conectar seu repositório local ao repositório remoto no GitHub. ### 🧠

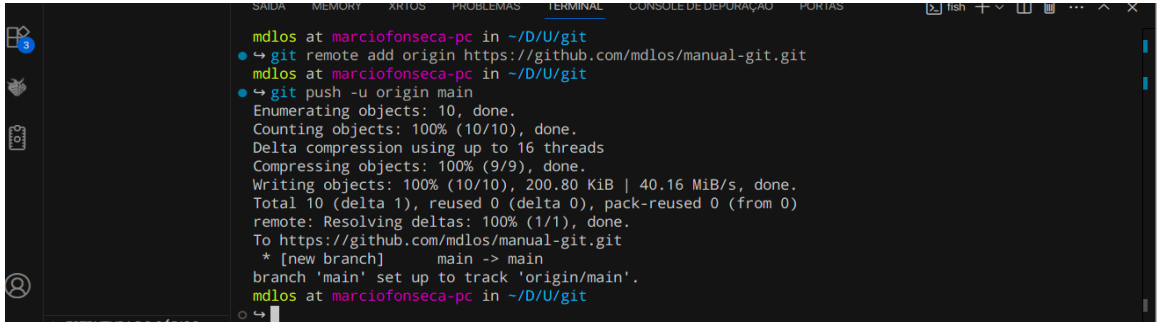
**Explicação:** `git remote add`: adiciona um repositório remoto. \* `origin`: é o apelido\*\* usado para o repositório remoto (padrão do Git). \*

`https://github.com/mdlos/manual-git.git`: é a **URL do repositório** no GitHub.

\*\*\* ### ✅ Depois de adicionar o remoto, o que fazer?

### 1. Enviar o projeto para o GitHub (primeira vez):

```
bash
git push -u origin main
```



```
mdlos at marciofonseca-pc in ~/D/U/git
• git remote add origin https://github.com/mdlos/manual-git.git
mdlos at marciofonseca-pc in ~/D/U/git
• git push -u origin main
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 16 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (10/10), 200.80 KiB | 40.16 MiB/s, done.
Total 10 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/mdlos/manual-git.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
mdlos at marciofonseca-pc in ~/D/U/git
```

> Substitua `main` por `master` se for o nome da sua branch principal.

### 2. Nas próximas vezes, apenas:

```
bash
git push
```

⚠ **Se já havia um `origin` adicionado:**

Se você já adicionou antes e quiser substituir:

```
bash
git remote set-url origin https://github.com/mdlos/manual-git.git
```

⚠ **Dica:**

Antes de mudar de branch, certifique-se de que **não há alterações não salvas**. Se houver, você pode: \* Fazer commit: `bash git add . git commit -m "Salvando alterações"` \* Ou guardar temporariamente com stash: `bash git stash` E depois, ao voltar para a `func_1`, recuperar com:

```
bash
git stash pop
```