

# **GIT, GITHUB & VS CODE**



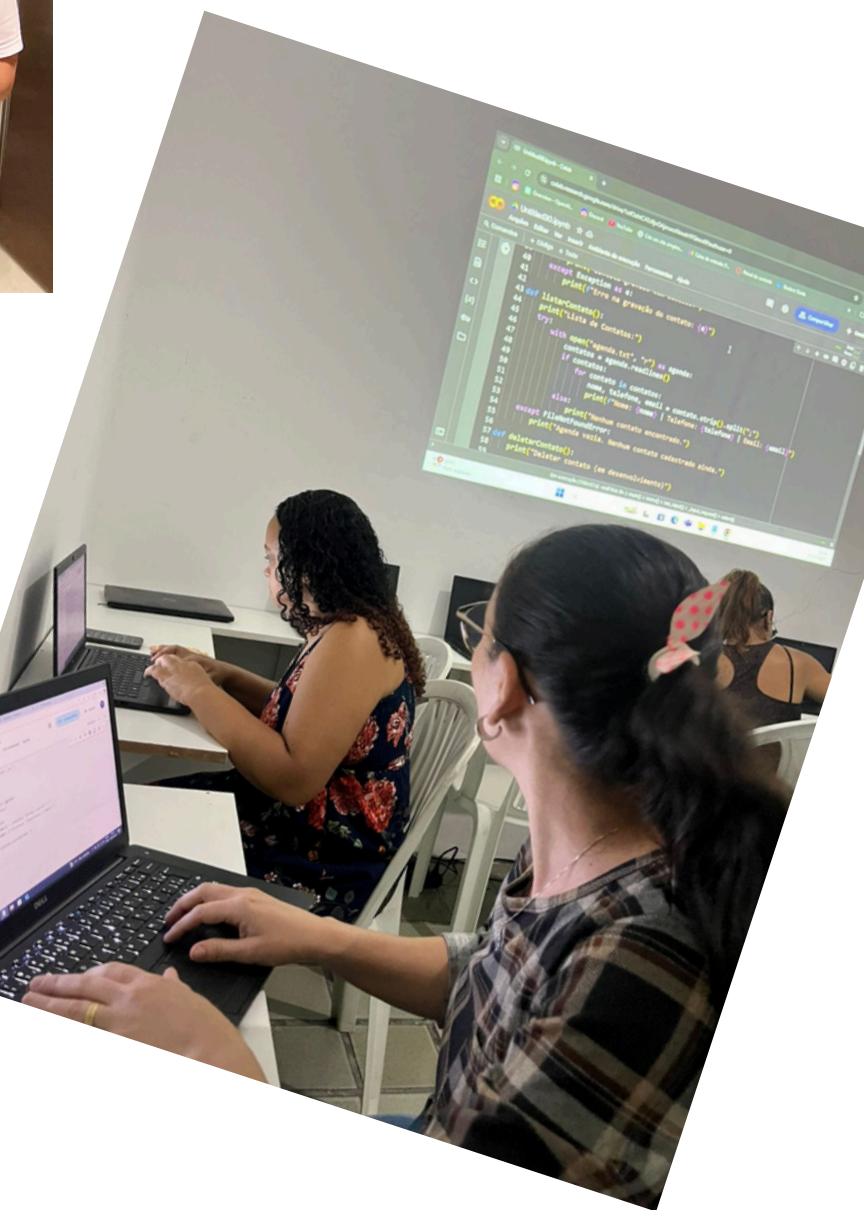
**Professora: Tereza Oliveira**



# QUEM É A PROFa?

Especialista em Tecnologias Web, mentora e professora. Fundadora da Virada no Café, impulsiona pessoas na transição para a tecnologia. Palestrante e consultora em diversidade, inclusão e ESG.

Tereza Oliveira



# **GIT, GITHUB & VS CODE**

**DURANTE O  
DESENVOLVIMENTO DE PROJETOS WEB  
EM EQUIPE, É PRECISO SEGUIR AS BOAS  
PRÁTICAS DE DOCUMENTAÇÃO E  
VERSIONAMENTO.**

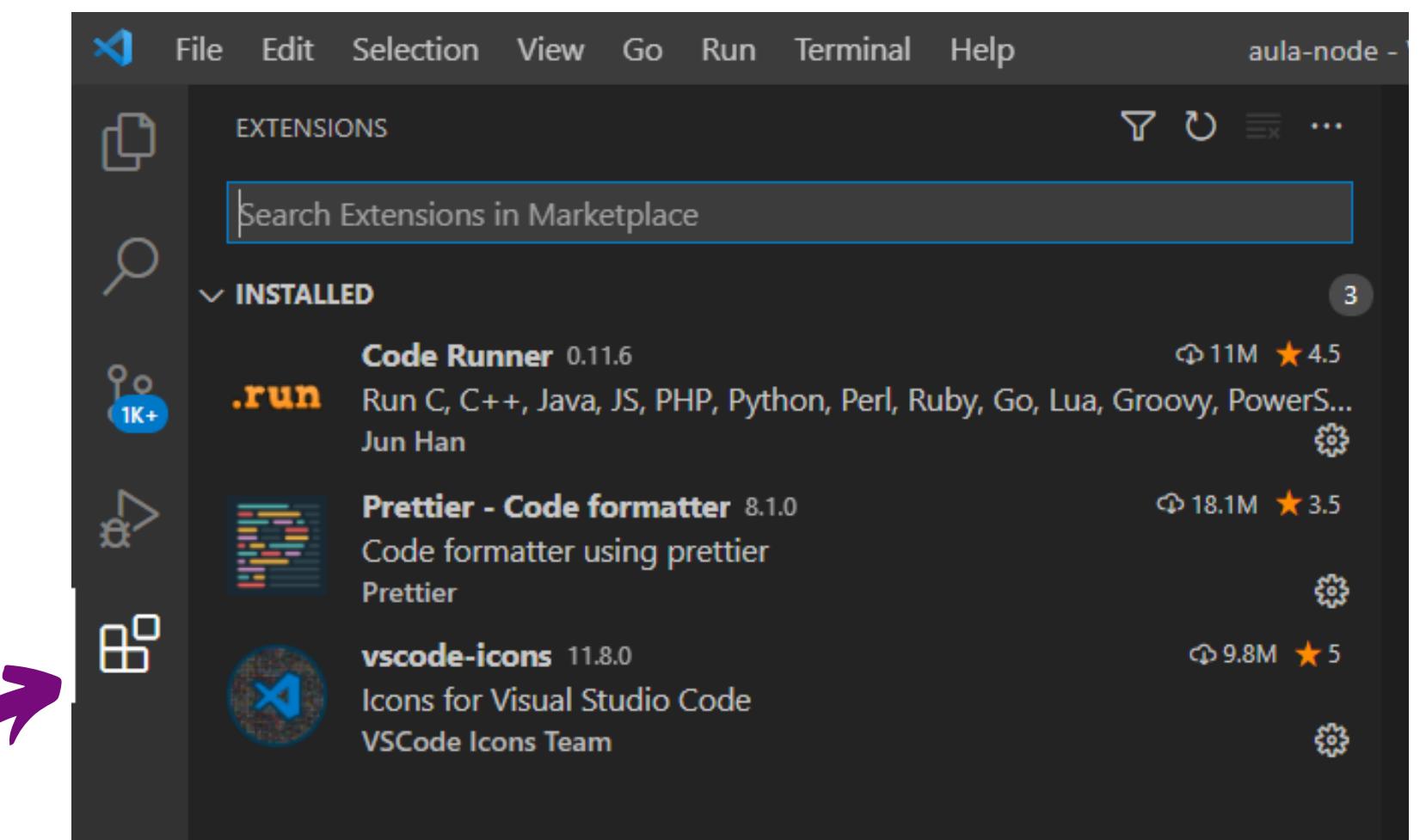
**O VERSIONAMENTO É CONTROLADO  
PELO QUE CHAMAMOS DE SISTEMA DE  
CONTROLE DE VERSÕES.  
NORMALMENTE, ESSES SISTEMAS SÃO  
UTILIZADOS NO DESENVOLVIMENTO DE  
SOFTWARE PARA CONTROLAR AS  
DIFERENTES VERSÕES E HISTÓRICO DE  
DESENVOLVIMENTO DO CÓDIGO.**

**UM SISTEMA DE CONTROLE MUITO  
CONHECIDO É O GIT. TAMBÉM VAMOS  
USAR O GITHUB E O VS CODE PARA NOS  
AUXILIAR NO VERSIONAMENTO.**

# Visual Studio Code

O VSCode é um editor de texto, criado em 2015 pela Microsoft. Ele é gratuito e open-source, isso significa que várias pessoas podem contribuir para melhorá-lo na criação de bugs e extensões.

A extensões facilitam nossa vida no processo de desenvolvimento dos códigos, agregando tecnologias e melhores estruturas e layout. Você pode acessar as extensões clicando no ícone dos bloquinhos no lado esquerdo.



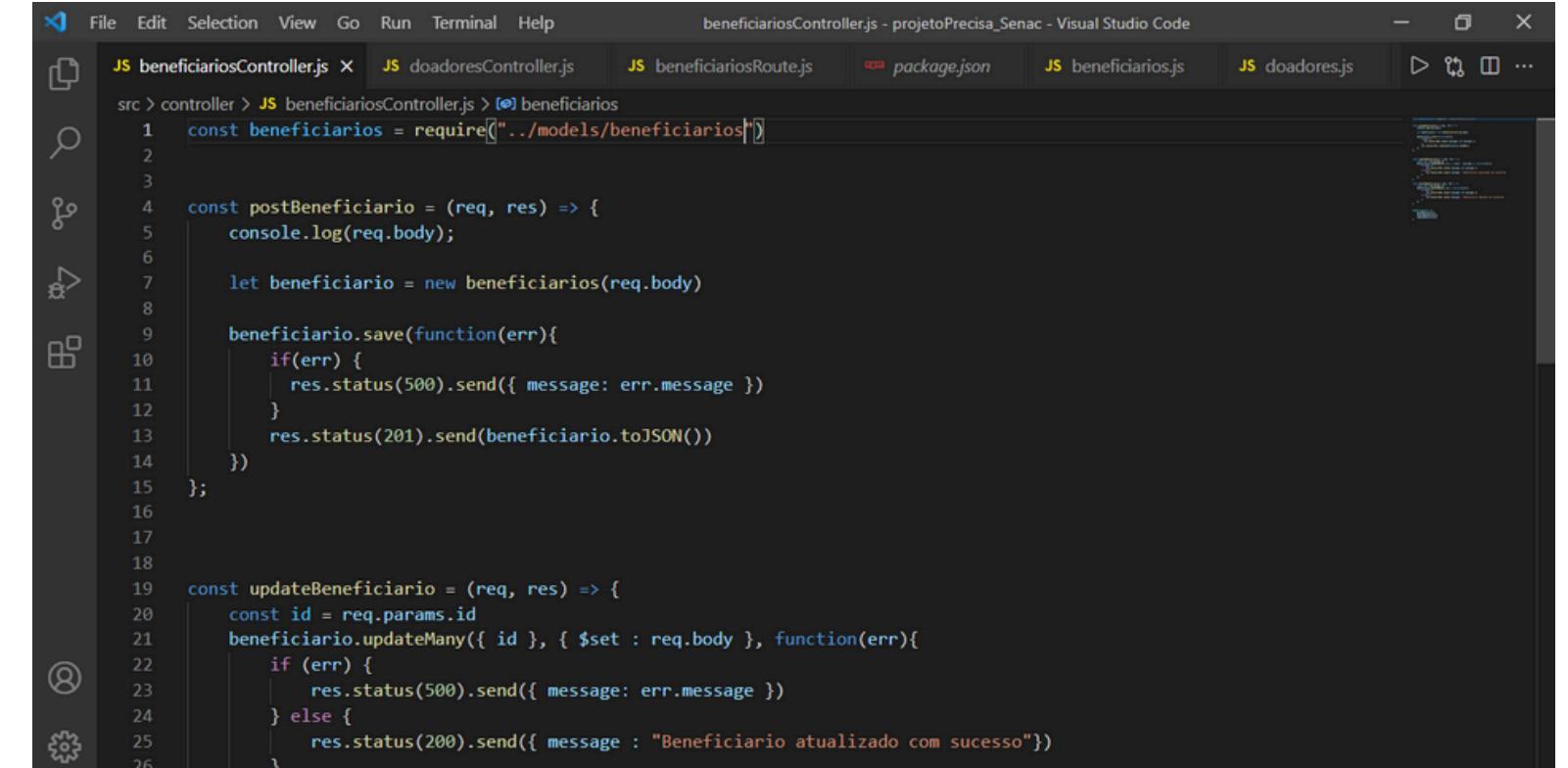
DICA: Instale as extensões Code Runner, Prettier e vscode-icons

# Visual Studio Code

## Como instalar no Windows?

Após fazer o download, execute o instalador e siga as instruções, vá clicando em “Next”. Lembre-se de marcar a opção “Add to path” para que o VS Code fique disponível nas suas variáveis de ambiente. Após a instalação, se tudo ocorrer bem, basta abrir o editor.

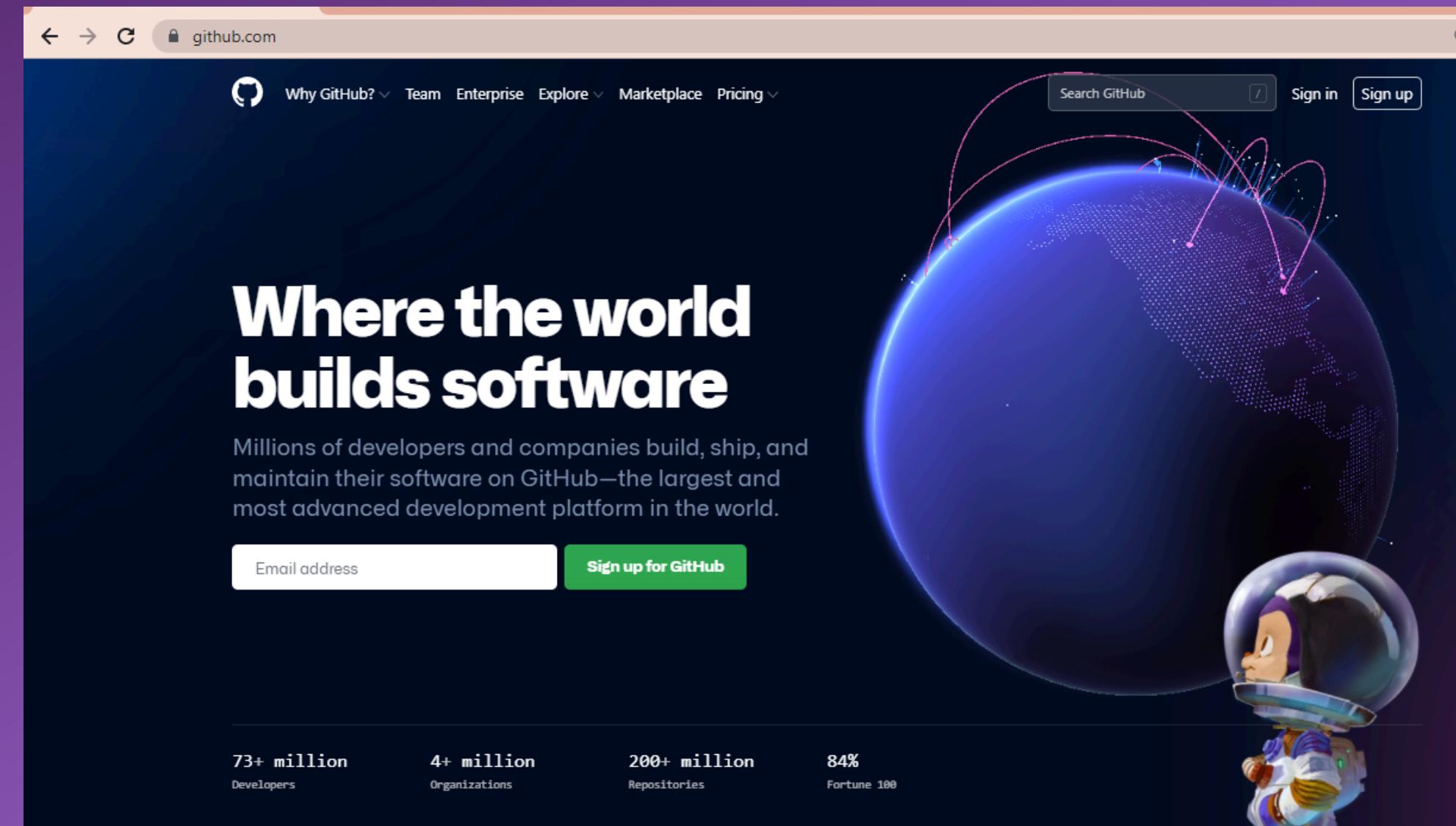
- <https://code.visualstudio.com/Download>
- <https://blog.betrybe.com/ferramentas/vs-code-guia-completo/>



```
JS beneficiariosController.js x JS doadoresController.js JS beneficiariosRoute.js package.json JS beneficiarios.js JS doadores.js
src > controller > JS beneficiariosController.js > beneficiarios
1 const beneficiarios = require("../models/beneficiarios")
2
3
4 const postBeneficiario = (req, res) => {
5   console.log(req.body)
6
7   let beneficiario = new beneficiarios(req.body)
8
9   beneficiario.save(function(err){
10     if(err) {
11       res.status(500).send({ message: err.message })
12     }
13     res.status(201).send(beneficiario.toJSON())
14   });
15 };
16
17
18
19 const updateBeneficiario = (req, res) => {
20   const id = req.params.id
21   beneficiario.updateMany({ id }, { $set : req.body }, function(err){
22     if (err) {
23       res.status(500).send({ message: err.message })
24     } else {
25       res.status(200).send({ message : "Beneficiario atualizado com sucesso"})
26     }
27   });
28 }
```

# O que é o GitHub?

GitHub é uma plataforma de hospedagem de código-fonte com controle de versão usando o Git. Ele permite que programadores ou qualquer usuário cadastrado na plataforma contribuam em projetos privados e/ou Open Source de qualquer lugar do mundo.



<https://github.com/>

# Mas antes, você precisa saber...

## REPOSITÓRIO, DIRETÓRIO OU PASTA

São sinônimos, afinal de contas irão guardar arquivos. Porém, quando se está no controle de versão (git ou outros) um repositório é um local onde os arquivos são armazenados e monitorados. Daí o termo "repositório" ser comum no desenvolvimento, guardando a aplicação dos códigos.

## BRANCH

No repositório poderá trabalhar mais de uma pessoa, por esse motivo existirão ramificações(branch). Imagina que será uma área exclusiva pra trabalhar naquele mesmo projeto, sem alterar na aplicação principal. A branch principal do git se chama "Master/Main", outras branches existirão conforme as pessoas forem criando.

## PULL REQUEST

O pull request, é o pedido para que o repositório original, ou uma branch do repositório original, faça a ação de pull (puxar) as atualizações do repositório fork ou de um branch do próprio repositório

## MERGE

O Git merge permite que você pegue as linhas criadas a partir do Git branch e faça uma integração para a ramificação principal. É importante notar que o comando git branch cria uma nova ramificação a partir da branch que o desenvolvedor está situado.

# o que é README.md ?

README é uma palavra em *inglês* que traduzida fica LEIAME. É um arquivo com extensão .md (Markdown).

Contém informações necessárias para entender o objetivo do projeto.

github.com/Duduxs/Awesome-README-Templates/blob/main/Profile-README/duduxs-descriptive.md

41 lines (27 sloc) | 1.85 KB

Oi gente, eu sou o Seu nome aqui.

👤💻 Quem sou eu:

- 🌐 Seus interesses
- 🎓 Sua faculdade
- 💻 O que você está estudando no momento

⌚ Competências Técnicas:

- 💻 Suas linguagens de programação
- 🌐 Suas áreas de atuação (Android, Desktop, Web)
- 📅 Seus bancos de dados
- 🔧 Suas IDEs

Gmail LinkedIn WhatsApp Facebook Instagram

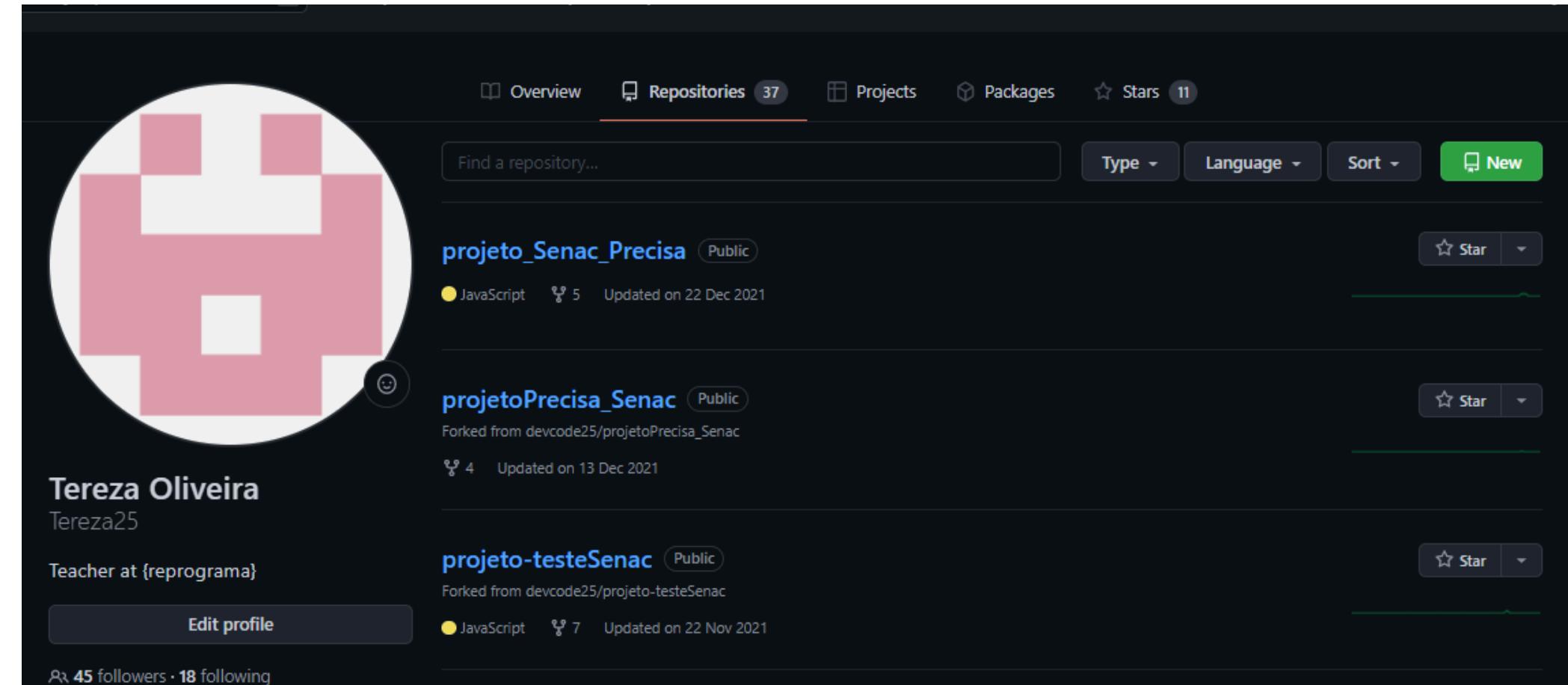
Eduardo José's GitHub Stats

⭐ Total Stars:	58
⌚ Total Commits:	1.2k
⚠ Total PRs:	17
⌚ Total Issues:	13
💻 Contributed to:	5

- <https://github.com/Duduxs/Awesome-README-Templates>
- <https://raullesteves.medium.com/github-como-fazer-um-readme-md-bonito%C3%A3o-c85c8f154f8>
- <https://blog.rocketseat.com.br/como-fazer-um-bom-readme/>

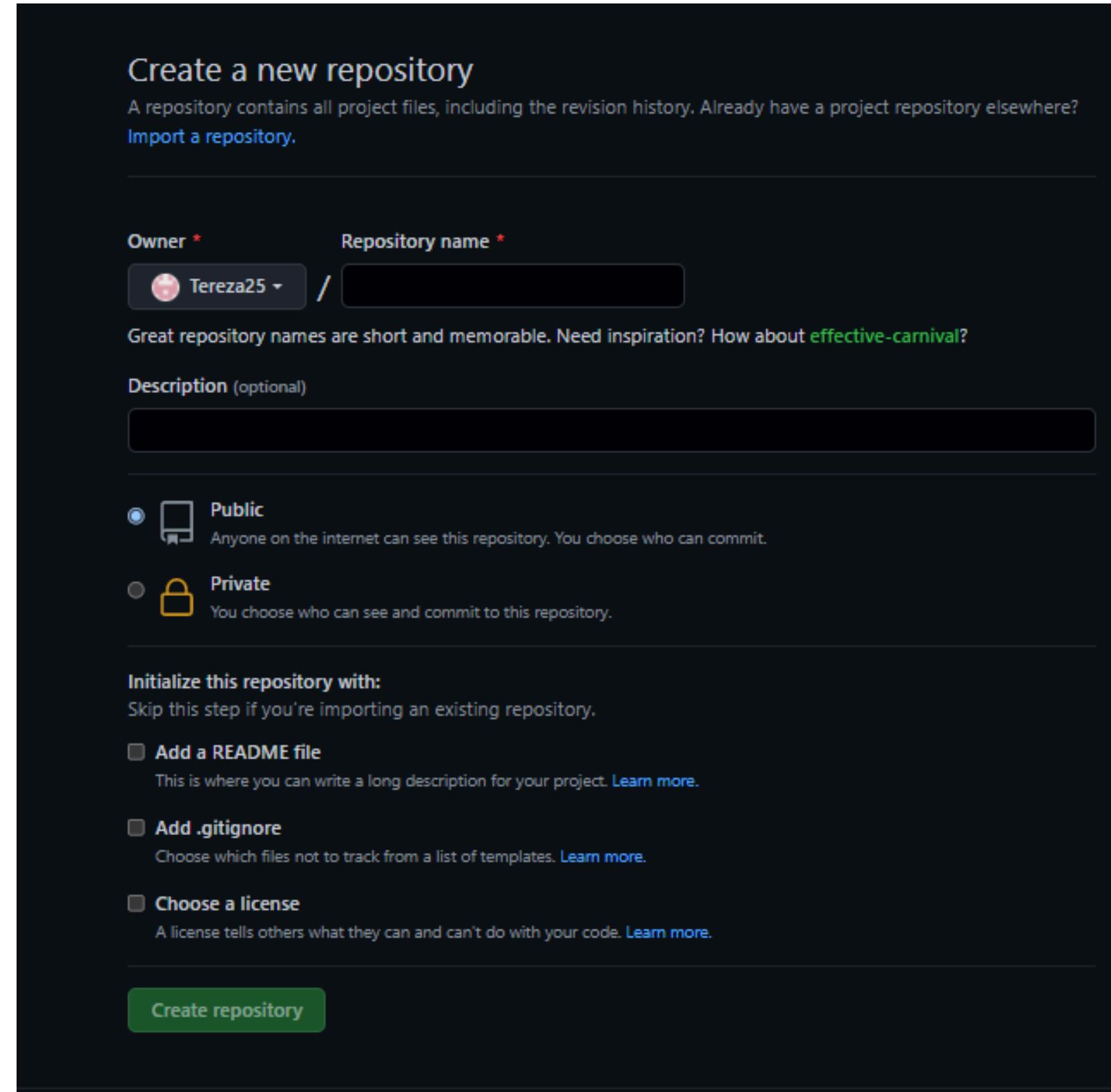
# Criando repositório no Github

**1 - No seu perfil do Github clique no campo "Your repositories" em seguida no botão "New".**

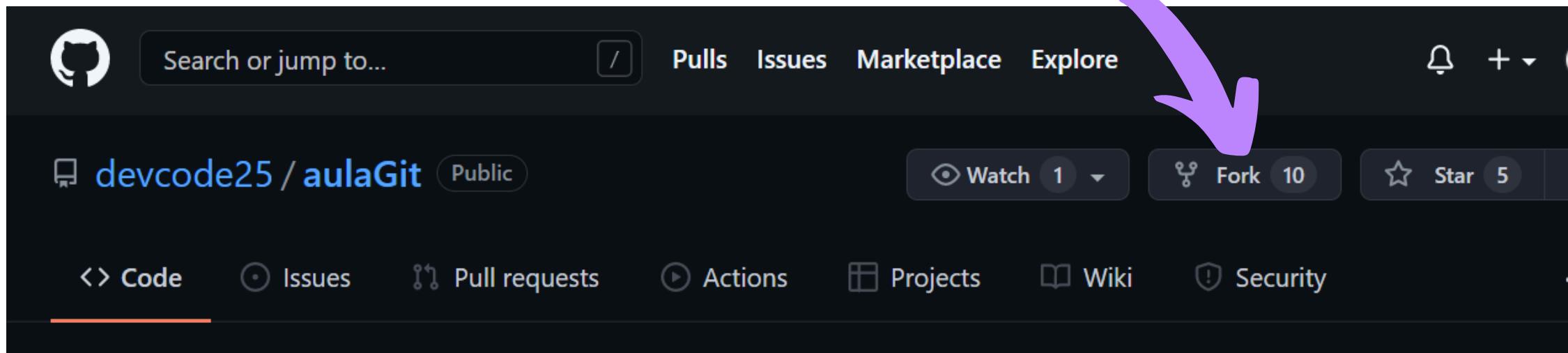


# Criando repositório no Github

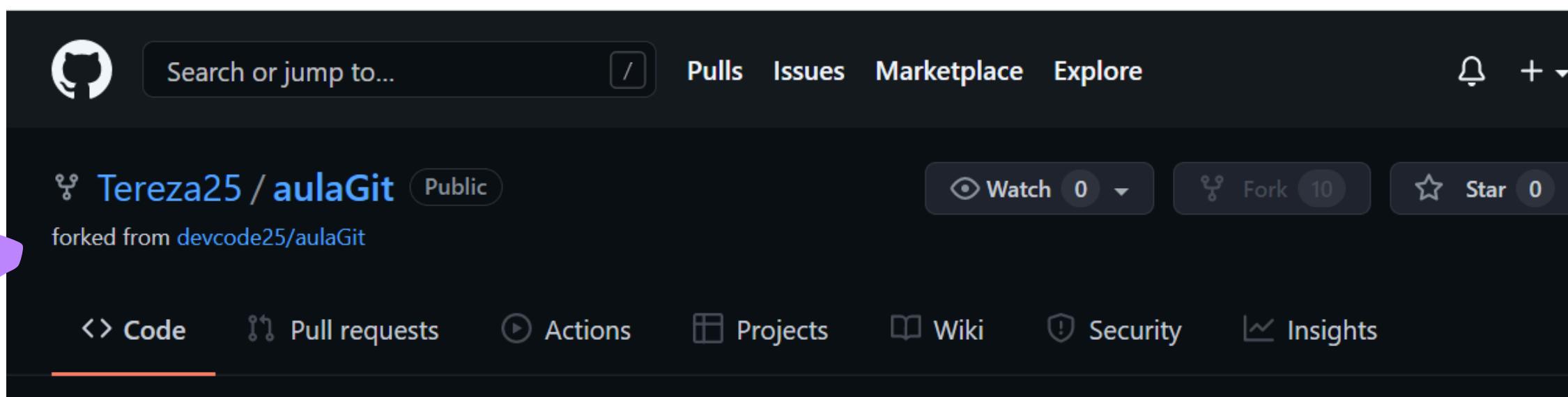
**2 - Na página de criação, você escolherá um nome para o repositório, o nível de privacidade (público ou privado), e adicionar README file. Clica em "Create repository".**



# O que é um "Fork"? Como faz pra "forkar" um repositório?

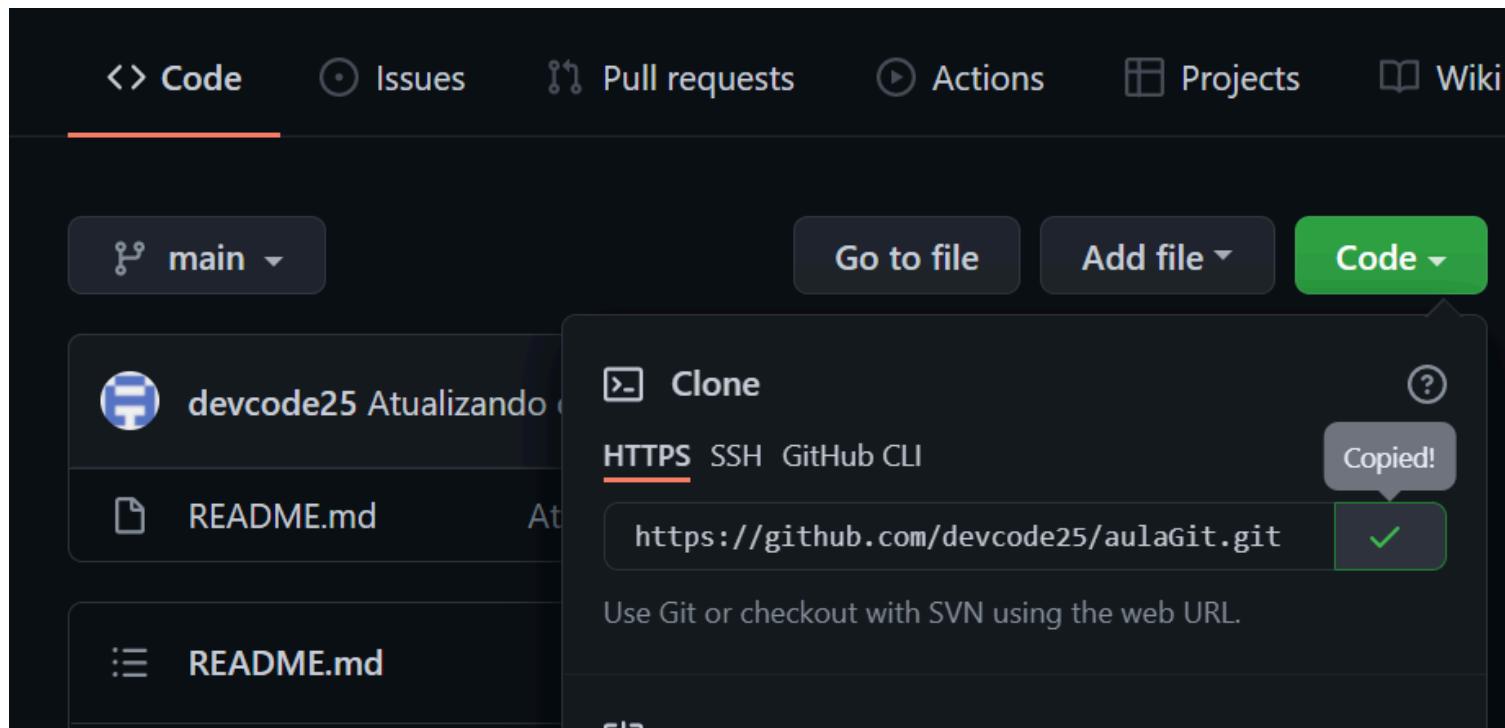


Na rede social do Github, você irá encontrar muitos repositórios legais. Então é comum as pessoas usarem o "fork" para arrastar aquele repositório para sua conta pessoal. Isso não quer dizer que o repositório será seu, mas poderá consultar quando quiser. Basta clicar no ícone **Fork**.



Caso queira trabalhar nesse repositório, poderá clona-lo para sua máquina e fazer contribuições criando uma branch.

# Aprendendo a clonar repositório pelo Git e Github



O comando **git clone** serve para baixar, na sua máquina, um projeto que está hospedado no github.

- Acesse o repositório do projeto que quer baixar
- clicar no botão **clone or download**
- copiar url
- Na sua máquina abra a linha de comando e vá até a pasta onde deseja colocar o projeto
- rode o comando: **git clone url-do-repositorio**

```
MINGW64:/c/Users/Usuario/Desktop
Usuario@DESKTOP-5KVL510 MINGW64 ~/Desktop (main)
$ git clone https://github.com/devcode25/aulaGit.git
```

A screenshot of a terminal window titled 'MINGW64:/c/Users/Usuario/Desktop'. It shows the command '\$ git clone https://github.com/devcode25/aulaGit.git' being typed. A purple arrow points from the text above to this terminal window.

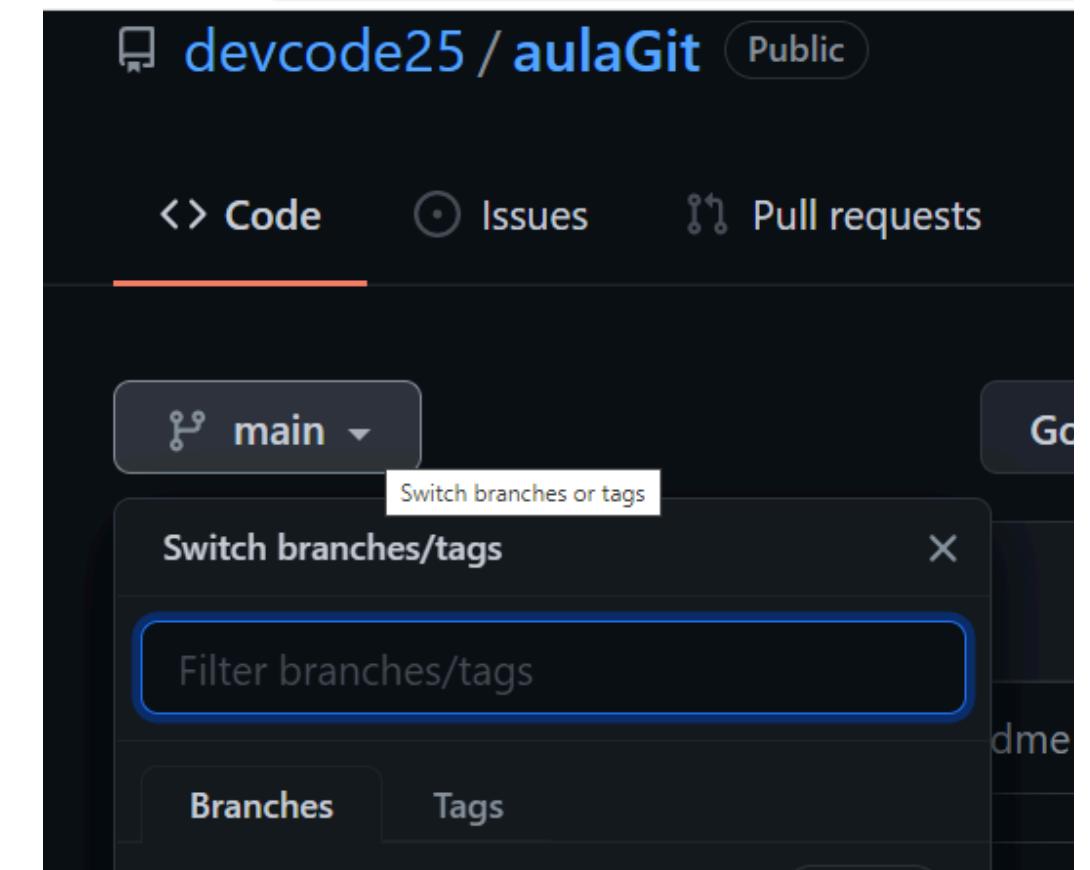
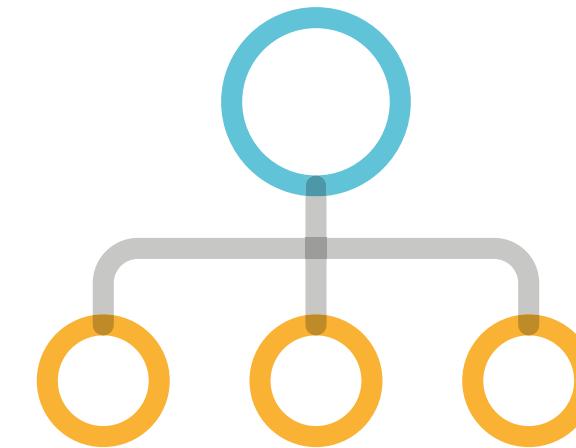
# Criando uma "branch"

Branch significa, galho, ramificação. Usando branches podemos trabalhar paralelamente nos projetos sem alterar a versão principal(master).

- **git checkout -b nome-da-branch**, cria uma nova branch e entra nela.
- **git branch** mostra as branches que existem no projeto/repositório, destacando a branch atual.
- **git checkout nome-da-branch**, para mudar de branch.

## git pull

como diz o nome, para **puxar** atualizações de uma branch remota para a branch atual(local). git pull origin master atualiza a branch que você está trabalhando com a master.



## O QUE É?

Git é um sistema de controle de versões , usado principalmente no desenvolvimento de software. O Git é um software livre, foi inicialmente projetado e desenvolvido por Linus Torvalds para o desenvolvimento do kernel Linux. Ele já vem instalado na maioria dos computadores Mac e Linux, se for o seu, apenas digite na linha de comando:

```
> git --version
```

Caso esteja instalado, esse comando mostrará a versão do git.

SITE OFICIAL > <https://git-scm.com/downloads>

A instalação é simples, basta seguir a sugestão e clicar em "Next". Depois de instalado. Clica com o botão direito do mouse na área de trabalho e clica em "Git Bash Here"



# CONFIGURAÇÕES

Agora que conseguimos instalar e abrir, você precisa configurar seu Git com o usuário e e-mail, os mesmos usados no Github.

## Para adicionar usuário:

```
> git config --global user.name "nome_do_Github"  
> git config --global user.email "seuEmailDoGithub@hotmail.com"
```

## Para remover usuário:

```
> git config --global --unset user.name "nome_do_Github"  
> git config --global --unset user.email  
"seuEmailDoGithub@hotmail.com"
```



# Comandos do Git

- **mkdir nome-da-pasta** - *cria uma pasta/diretório, que servirá como repositório*
- **cd ~** - *volta para a raiz*
- **cd ..** - *volta uma pasta*
- **cd nome-da-pasta** - *para entrar em uma pasta*
- **touch nome-do-arquivo** - *para criar um arquivo*
- **rm nome-do-arquivo** - *para remover um arquivo*
- **git init** - *inicia um repositório git oculto na pasta criada*
- **git status** - *Ajuda a ver os estágios dos arquivos, para ver quais arquivos foram criados e ou modificados*
- **code .** - *abre o visual studio code na pasta local*
- **git add .** - *para adicionar todos os arquivos que estão na sua pasta de uma vez, caso queria adicionar apenas um arquivo digite git add nome-do-arquivo*
- **git commit -m “coloque sua mensagem aqui”** - *é usado para confirmar as alterações na cabeça da aplicação*
- **git push origin master** - *envia as alterações feitas para a branch mestre do repositório*

# Já tenho uma aplicação numa pasta do meu PC e agora quero subir o repositório pro Github, como faz?

...or push an existing repository from the command line

```
git remote add origin https://github.com/Tereza25/Test.git  
git branch -M main  
git push -u origin main
```

- Criar novo repositório no seu github e dar nome a ele.
- no sua máquina, pela linha de comando, vá até a pasta do projeto.
- dentro da pasta do projeto rode o comando: **git init** (pasta git oculta é criada)
- **git add .** para adicionar todos os arquivos que estão na sua pasta de uma vez, caso queria adicionar apenas um digite `git add nome-do-arquivo`
- **git commit -m "mensagem para explicar o que voce fez no código"**
- **git remote add origin url-que-voces-copiram-do-github**
- **git remote -v** mostra as url que o git está apontando
- **git push origin master**

# Contribuindo num projeto remoto

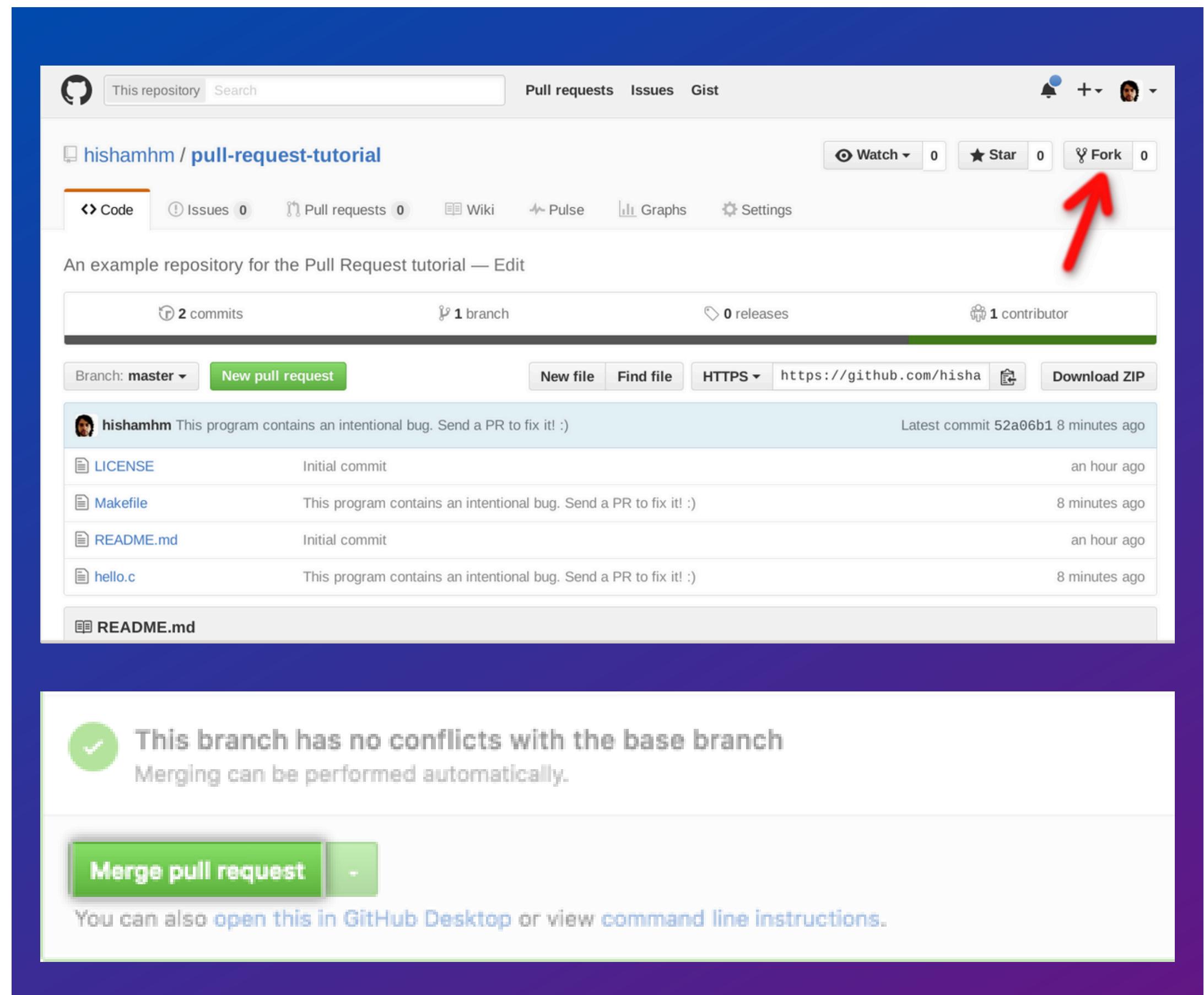
- clonar repositório (**git clone url-do-projeto**)
- caso já tenha o projeto na sua máquina, precisamos atualizar a branch master local (**git pull origin master**)
- criar sua branch para trabalhar (**git checkout -b nome-da-sua-branch**)
- finalizado o trabalho vamos subir nossas mudanças:
  - **git add .**
  - **git commit -m "o que eu fiz"**
  - **git push origin nome-da-sua-branch**
- agora vamos no github abrir a **PR**

## Pull request (a famosa PR)

Na Pull request, você está simplesmente solicitando ao responsável(ou responsáveis) do repositório aprovação para que as alterações que você fez sejam integradas a branch principal (normalmente a branch master). Na pull request, precisamos escrever o que fizemos. Cabe ao responsável aceitar, negar, pedir correções, fazer um code review da sua solicitação e etc

# Merge

- A 'mesclagem' é o jeito do Git de unificar históricos de galhos diferentes. Ou seja, unificar duas branches em uma.
- É o que acontece quando mergeamos a **Pull Request**. Estamos pegando a nossa branch, com as alterações que fizemos, e unificando com a branch master



# Fim do módulo I

**Chegamos ao final do primeiro módulo,  
e agora você já está pronta para dar  
novos passos na programação.**

**Bons estudos e até  
os próximos módulos. :)**



@terezolaiveira



@terezolaiveira.oficial

# Sites para ajudar nos estudos

- **Gerador de README.md**
- **Guia Markdown**
- **Como fazer um Readme bonitão**
- **Como fazer Readme**
- **Guia sobre repositório Github**