



SÃO  
PAULO  
TECH  
SCHOOL

# **Pesquisa e Inovação**

## **Git e GitHub**

## Antes de começarmos:

Quem aqui nunca "ouviu falar" em git e github?

O que faz um desenvolvedor/a competente?



**Trabalhar em equipe**

# Termos técnicos importantes

- **Front-end + Back-end**
- **URL** (e não "link")
- **Terminal de comando**
- **Diretório** (e não "pasta")
- **Servidor da aplicação**
- **Refatorar e versionar código**

# Termos técnicos importantes

## ➡ **Front-end + Back-end**

- **URL** (e não "link")
- **Terminal de comando**
- **Diretório** (e não "pasta")
- **Servidor da aplicação**
- **Refatorar e versionar código**



# Termos técnicos importantes

- **Front-end + Back-end**



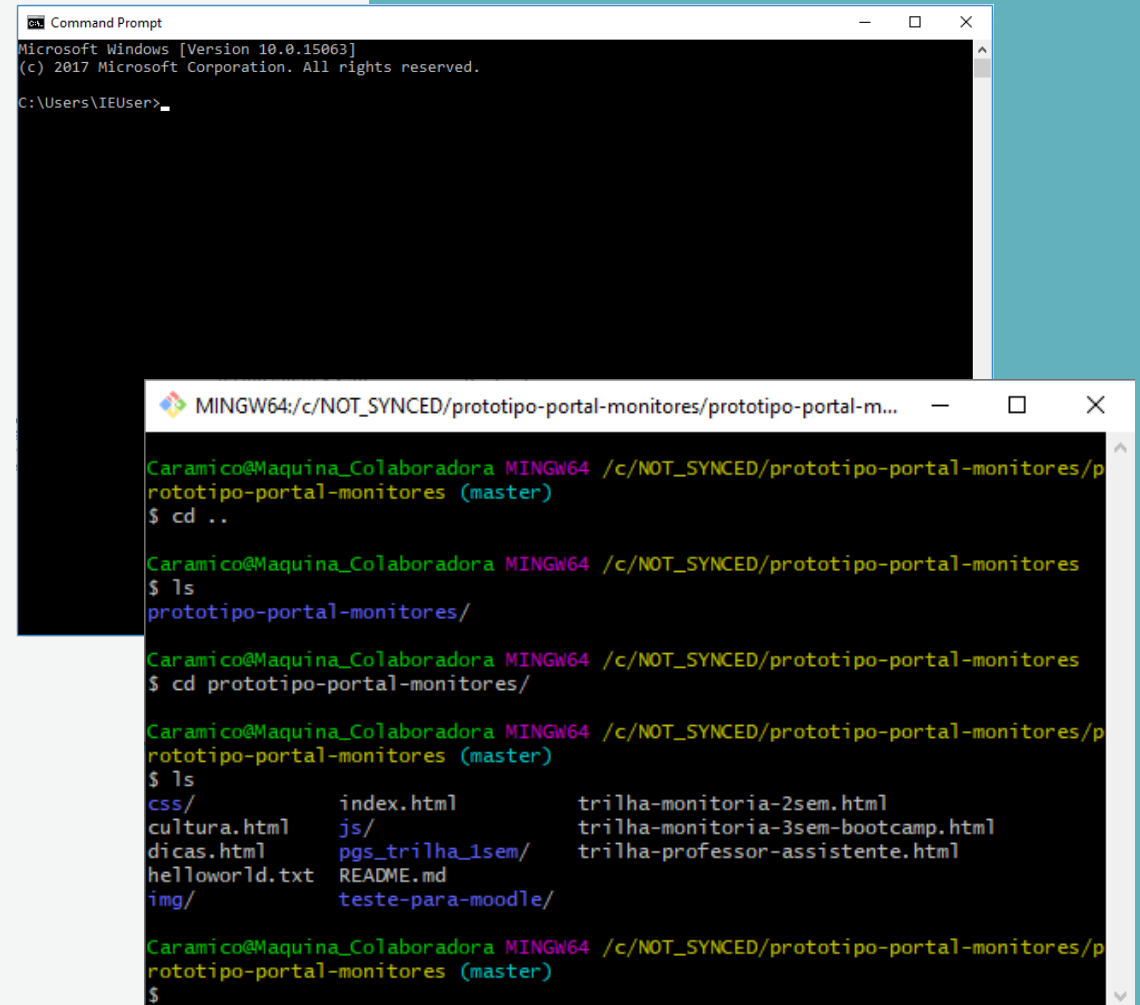
**URL** (e não "link")

*Uniform Resource Locator*  
*[localizador padrão de recursos]*

- **Terminal de comando**
- **Diretório** (e não "pasta")
- **Servidor da aplicação**
- **Refatorar e versionar código**

# Termos técnicos importantes

- **Front-end + Back-end**
- **URL** (e não "link")
- ➔ **Terminal de comando**
- **Diretório** (e não "pasta")
- **Servidor da aplicação**
- **Refatorar e versionar código**



```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\IEUser>

MINGW64:/c/NOT_SYNCED/prototipo-portal-monitores/prototipo-portal-m...
Caramico@Maquina_Colaboradora MINGW64 /c/NOT_SYNCED/prototipo-portal-monitores/p
rototipo-portal-monitores (master)
$ cd ..

Caramico@Maquina_Colaboradora MINGW64 /c/NOT_SYNCED/prototipo-portal-monitores
$ ls
prototipo-portal-monitores/

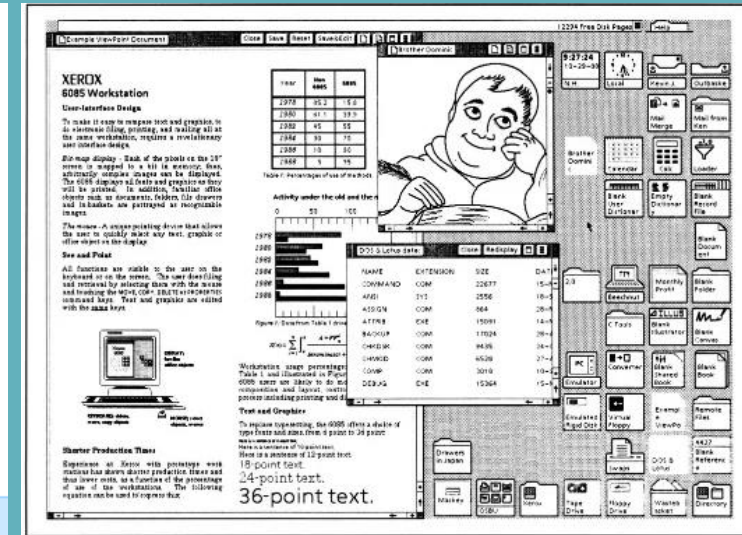
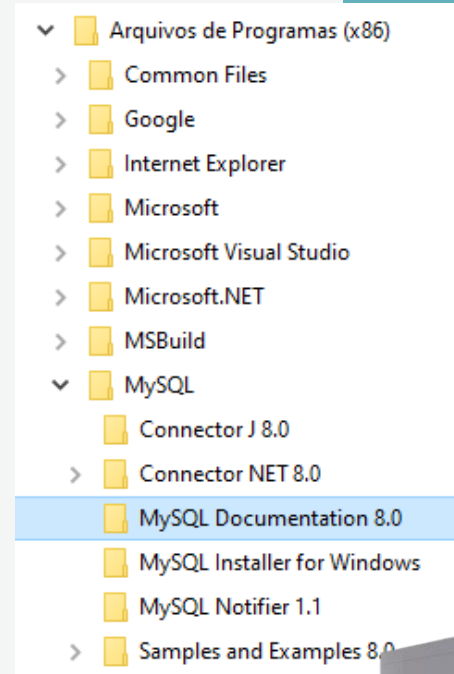
Caramico@Maquina_Colaboradora MINGW64 /c/NOT_SYNCED/prototipo-portal-monitores
$ cd prototipo-portal-monitores/

Caramico@Maquina_Colaboradora MINGW64 /c/NOT_SYNCED/prototipo-portal-monitores/p
rototipo-portal-monitores (master)
$ ls
css/          index.html      trilha-monitoria-2sem.html
cultura.html  js/             trilha-monitoria-3sem-bootcamp.html
dicas.html    pgs_trilha_1sem/ trilha-professor-assistente.html
helloworld.txt README.md
img/          teste-para-moodle/

Caramico@Maquina_Colaboradora MINGW64 /c/NOT_SYNCED/prototipo-portal-monitores/p
rototipo-portal-monitores (master)
$
```

# Termos técnicos importantes

- **Front-end + Back-end**
- **URL** (e não "link")
- **Terminal de comando**
- ➡ **Diretório** (e não "pasta")
- **Servidor da aplicação**
- **Refatorar e versionar código**



## Interface Gráfica do Xerox Alto, 1973





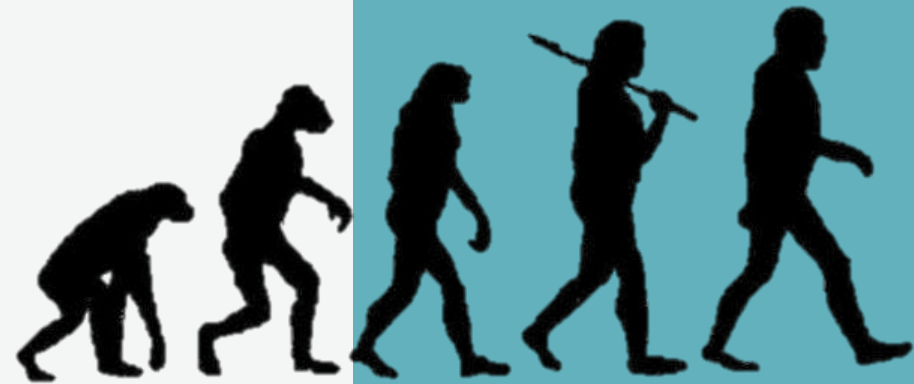
# Termos técnicos importantes

- **Front-end + Back-end**
- **URL** (e não "link")
- **Terminal de comando**
- **Diretório** (e não "pasta")
- ➡ **Servidor da aplicação**
- **Refatorar e versionar código**



# Termos técnicos importantes















- **Front-end + Back-end**
  - **URL** (e não "link")
  - **Terminal de comando**
  - **Diretório** (e não "pasta")
  - **Servidor da aplicação**
- ➡ **Refatorar e versionar código**



Refatoração de código é **melhorar** o programa **sem alterar o comportamento**, com o objetivo de deixá-lo mais eficiente.

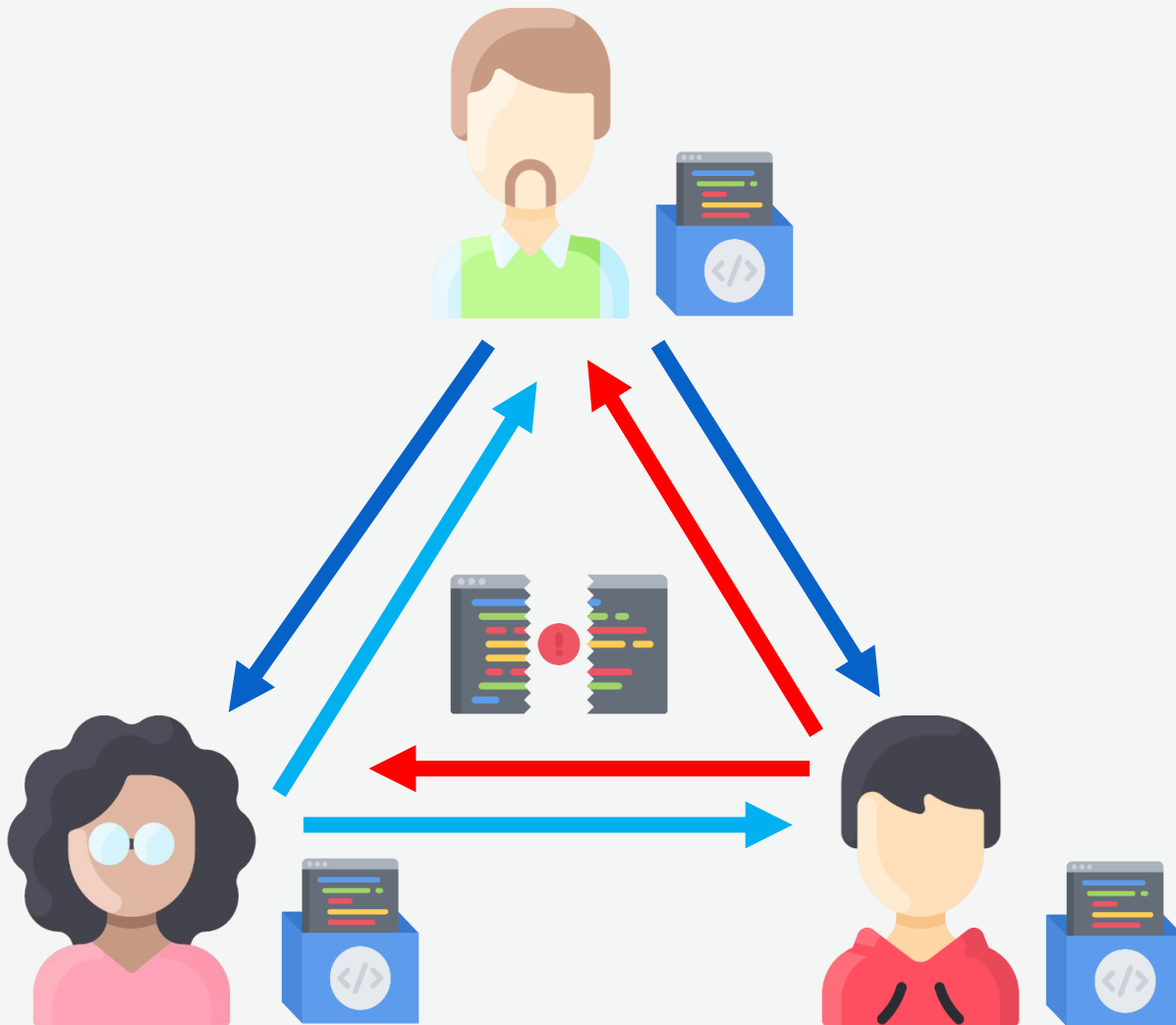
O versionamento é o **gerenciamento de versões** diferentes de algo produzido – pode ser um código, uma ilustração, um tcc...

## Essa imagem... É familiar?

Nome	Tipo
 TCC do grupo - Copia - enviado pelo whats dia 30.docx	Documento do Micros
 TCC do grupo - Copia.docx	Documento do Micros
 TCC do grupo - versão 1 - Copia.docx	Documento do Micros
 TCC do grupo - versão 1.docx	Documento do Micros
 TCC do grupo - versão final - com as imagens.docx	Documento do Micros
 TCC do grupo - versão final 2.docx	Documento do Micros
 TCC do grupo - versão final aaaaaa finalmente.docx	Documento do Micros
 TCC do grupo - versão final agora vai mesmo.docx	Documento do Micros
 TCC do grupo - versão final agora vai.docx	Documento do Micros
 TCC do grupo - versão final FINAL.docx	Documento do Micros
 TCC do grupo - versão FINAL MESMO.docx	Documento do Micros
 TCC do grupo - versão final pronto para enviar.docx	Documento do Micros
 TCC do grupo - versão FIN	Documento do Micros
 TCC do grupo .docx	Documento do Micros

**Não queremos isso!!**

## Em um cenário sem versionamento:



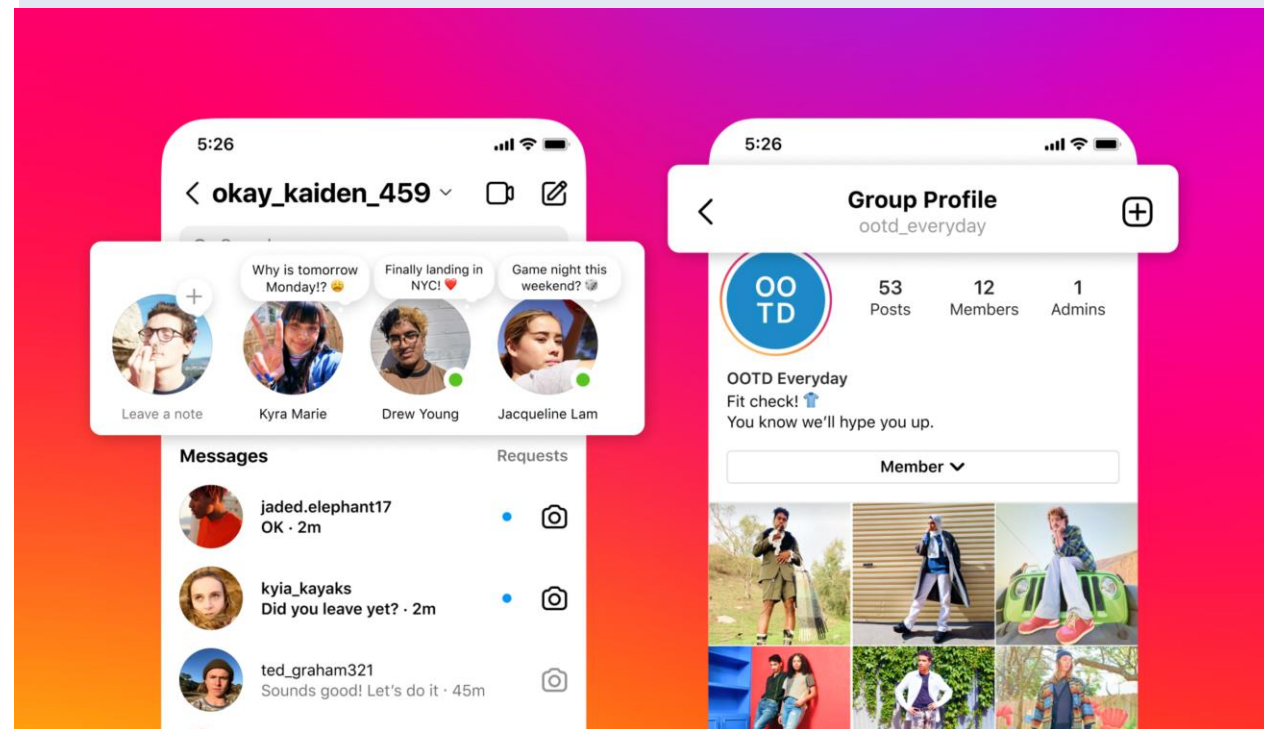
- **Sem rastreabilidade:**
  - Quem/quando/por que fez?
  - E se precisar voltar?
- **Suscetível a erros:**
  - Sobrescrever alterações
  - Código "quebrado"

# Imagine em projetos grandes?!

## Instagram - 2010



## Instagram - 2023



Quantas versões será que tiveram que ser geradas? 🤔





# Linus Torvalds

Criou o Git em 2005 para **auxiliar no desenvolvimento** do kernel (*núcleo, cérebro*) do Linux (*sistema operacional que ele criou*).



# Git x Git Hub



!=



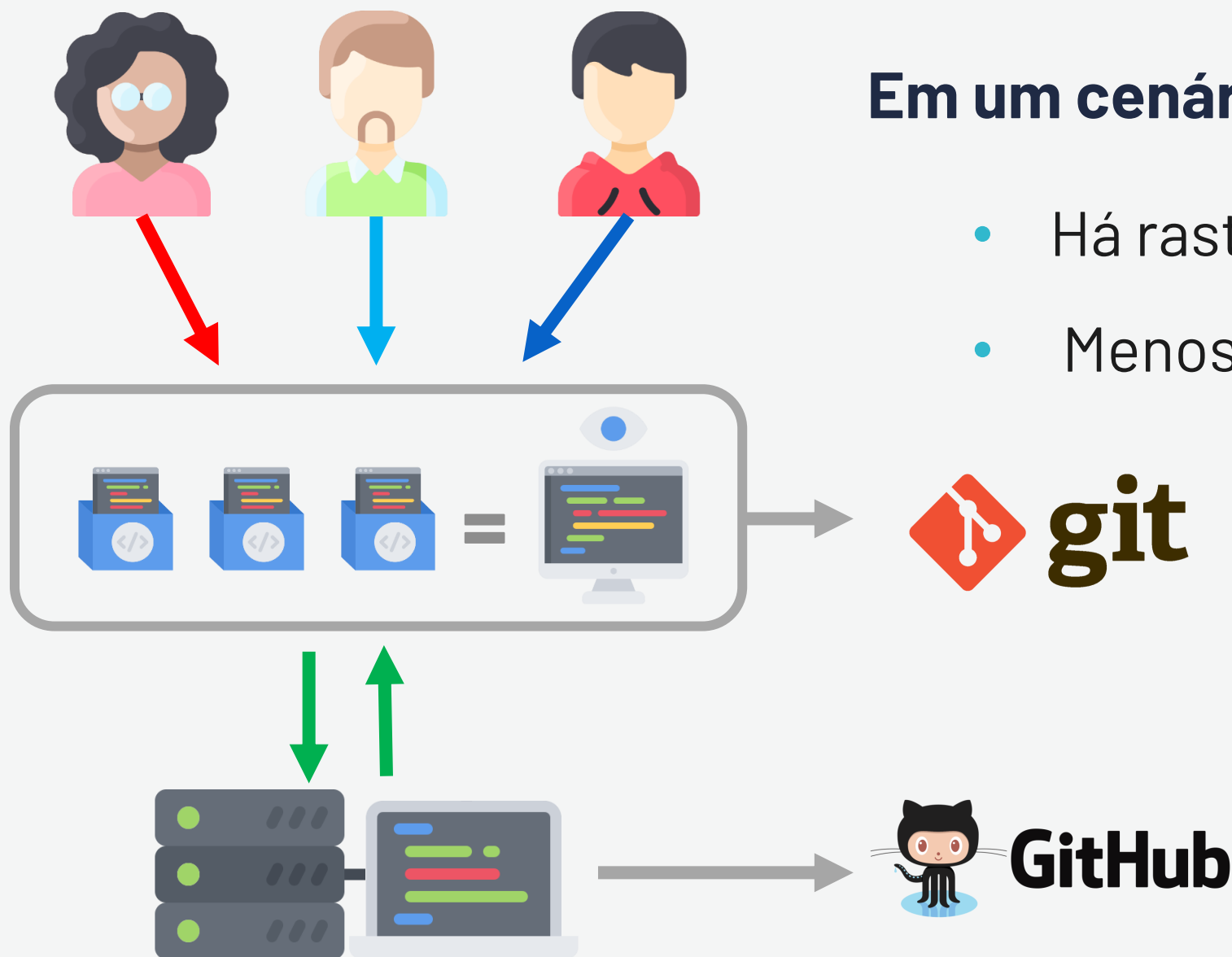
Ferramenta para  
**versionamento de  
arquivos.**

Uma plataforma usada  
para **compartilhar**  
projetos **usando git.**  
Há outras.

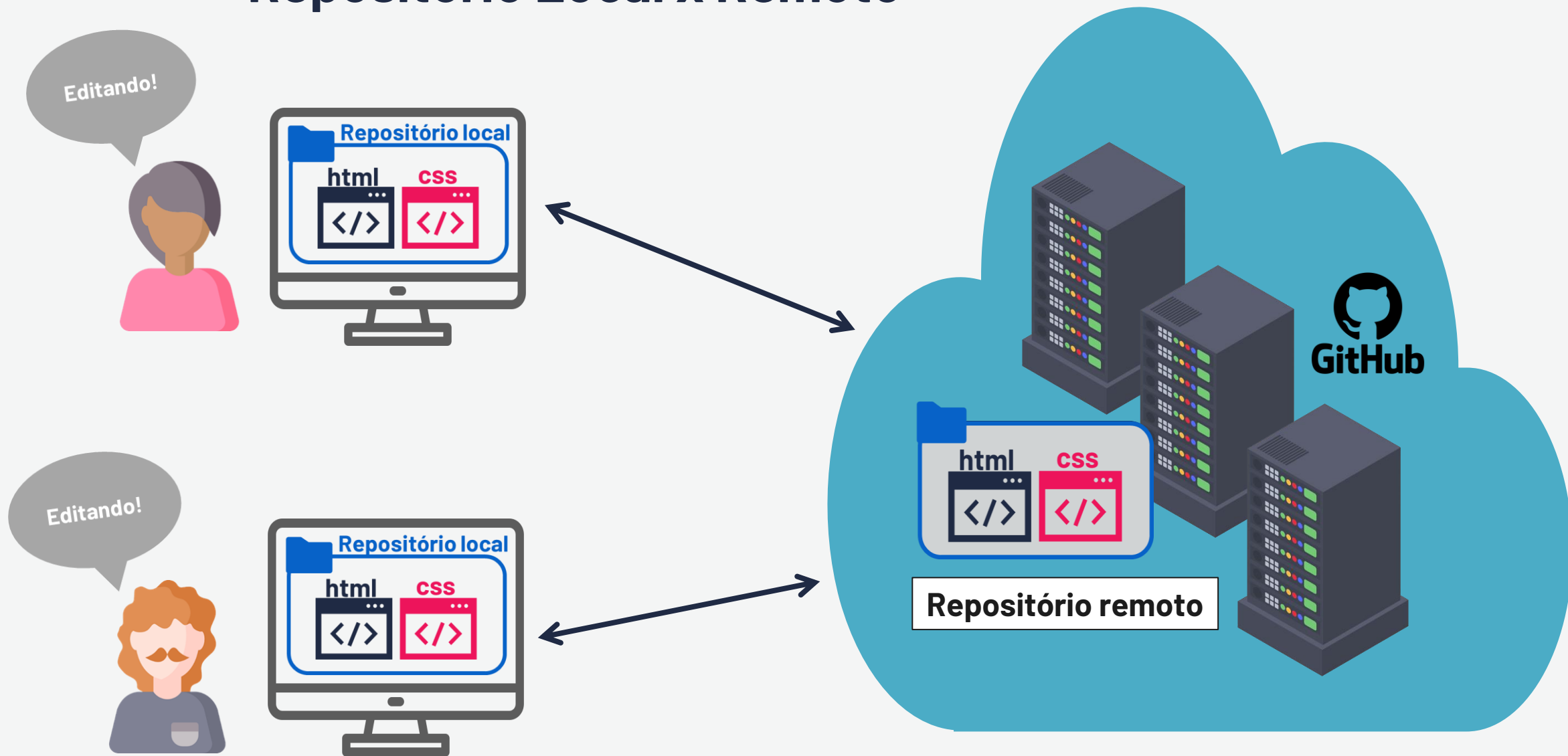


## Em um cenário com versionamento

- Há rastreabilidade
- Menos suscetível a erros



# Repositório Local x Remoto



## Próximos passos

Criar conta no GitHub

✓ Use o e-mail da faculdade

Apesar de ser uma “rede social”, **o GitHub é um ambiente profissional** e deve ser tratado como tal.



# Vazamento de senhas do Ministério da Saúde expõe informações de pacientes do Covid-19, diz jornal

A brecha foi revelada pelo jornal 'O Estado de S. Paulo', que publicou acesso a dados de ao menos 16 milhões de pessoas, incluindo membros do governo.

Por G1

26/11/2020 10h43 · Atualizado há um ano

todo o país, segundo o jornal (veja detalhes mais abaixo).

As senhas estavam em uma planilha, de acordo com o "O Estado de S. Paulo", que foi publicada por um funcionário do Hospital Albert Einstein, em São Paulo, em um site de compartilhamento de códigos de programação e arquivos chamado "GitHub". Segundo o jornal, o Einstein disse que tinha acesso aos dados porque está trabalhando em um projeto com o Ministério da Saúde.

O GitHub é como se fosse uma **rede social** usada por programadores e cientistas de dados para compartilhar códigos de programação e contribuir com projetos da área. O serviço também é utilizado pelos



## Próximos passos

Criar conta no GitHub

- ✓ Use o e-mail da faculdade

Conhecendo a plataforma

- ✓ Primeiro repositório
  - ✓ Adicione o README.md!

## Próximos passos

Assistir ao video:

<https://www.youtube.com/watch?v=KHkA1xFtEAU&>

2:14 -> login

6:38 -> tokens de acesso

Instalar o GitBash em sua máquina pessoal

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

Exemplos de Repositórios

<https://github.com/torvalds/linux>

<https://github.com/FortAwesome/Font-Awesome>

<https://github.com/EbookFoundation/free-programming-books>

Vamos ver como funciona o  
versionamento?





# Como funciona?

Quando você pede uma pizza, há na pizzeria o preparo e a entrega



Pizza é preparada...

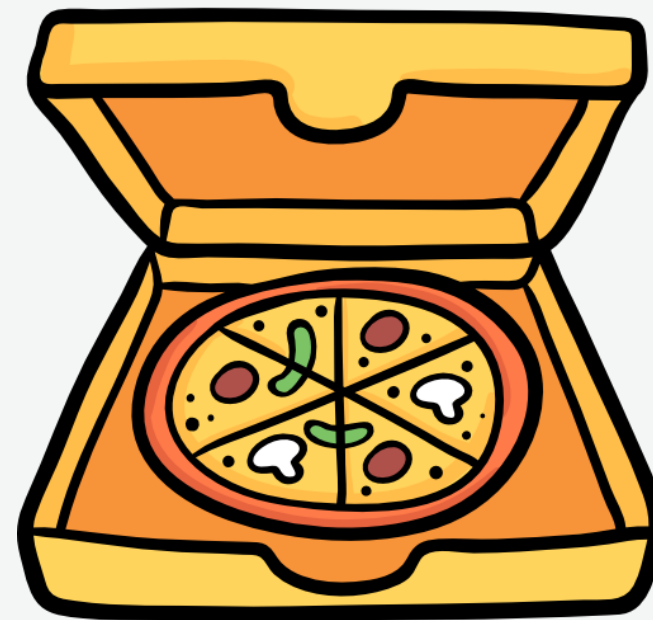
# Como funciona?



... e, quando pronta, é colocada na caixa.

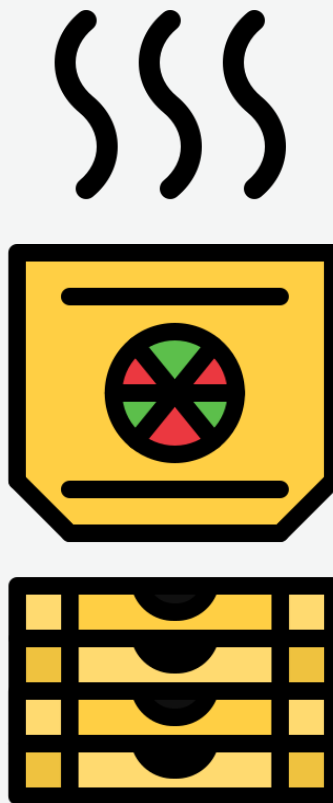


# Como funciona?



Caixa pronta?  
Feche-a!

# Como funciona?



Caixas fechadas?  
Hora de entregar!

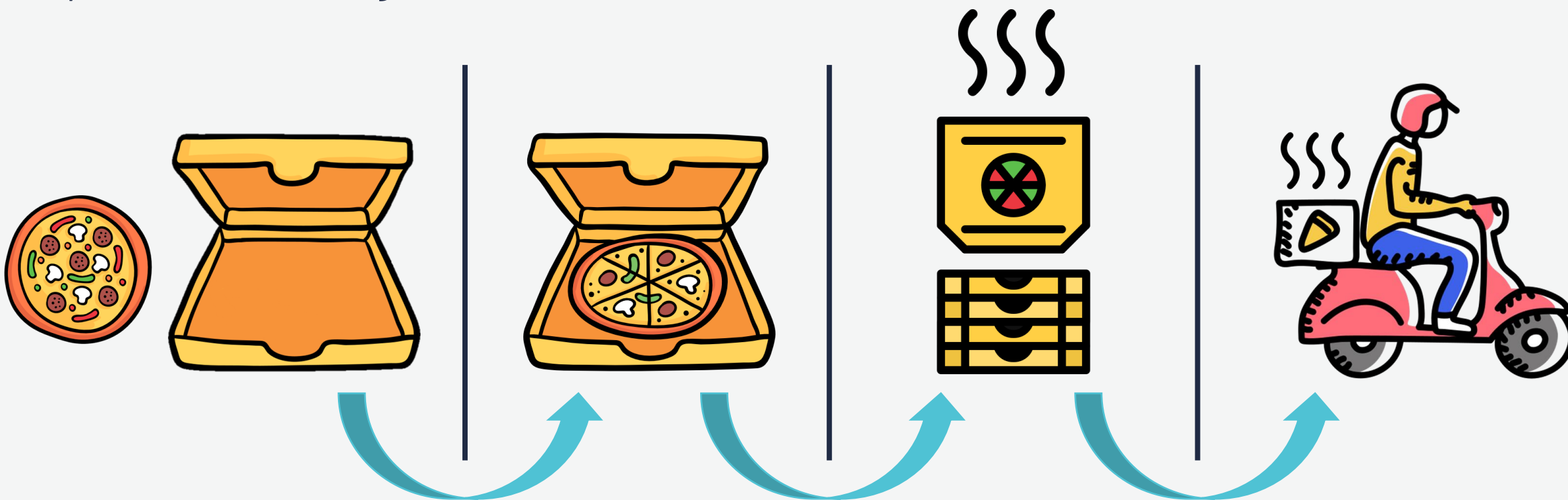
# Como funciona?



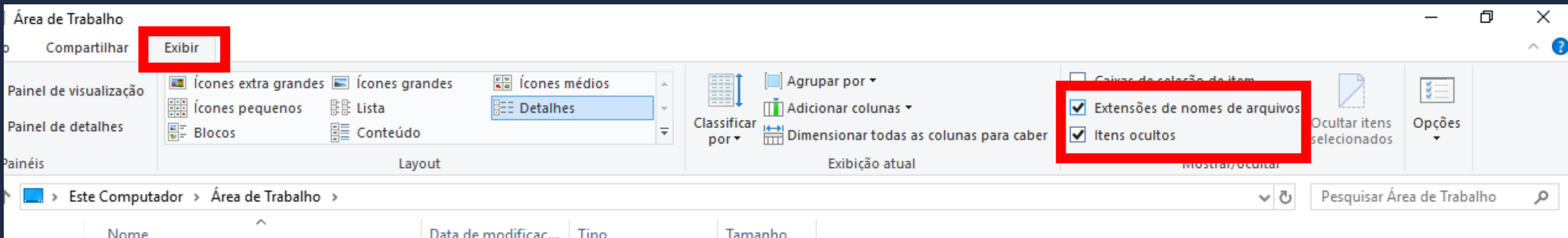
Entregando!

# Como funciona?

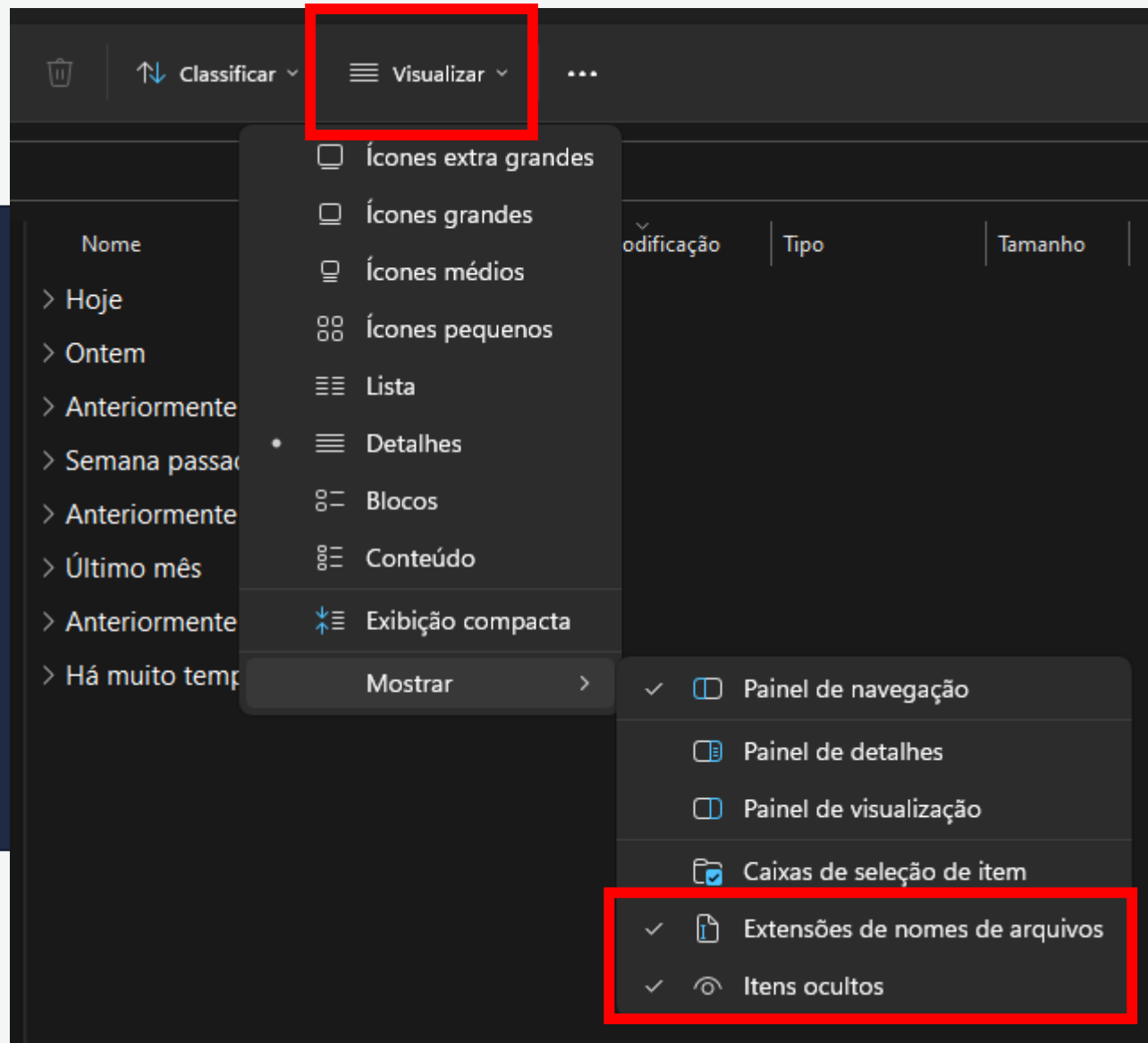
*A pizza é o seu código!*



# Não se esqueça!



**E dica... Desative as extensões de  
tradução do Google Chrome! 😊**





# Qual ferramenta usar de acordo com seu sistema operacional:



## Git bash

Para executar comandos git no **Windows**, baixe o **GitBash**:

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



## Terminal

O atalho **Linux** para abrir o terminal é: **Ctrl+Alt+T**.



## Terminal



No **MacOS**, abra o Spotlight clicando em **Command + Espaço**, (ou clique no ícone do Launchpad no Dock) **digite Terminal** no campo de busca e clique em Terminal.

## Navegando pelo terminal – Comandos básicos

Primeiros comandos para navegar no terminal:

**ls** (ou **dir**) → listar itens no diretório

**cd** *meu-diretorio* → mudar de diretório (**c**hange **d**irectory)

**cd** .. → ir para pasta “acima”, diretório pai

[up arrow] = [seta p/ cima] → comandos anteriores

[tab] → completar o texto iniciado

*(ex. Diretório com nome muito longo, digite as primeiras letras e o tab completará)*

## Autenticando-se

Abra o terminal em qualquer diretório em sua máquina.

Vamos verificar qual é o usuário que está configurado:

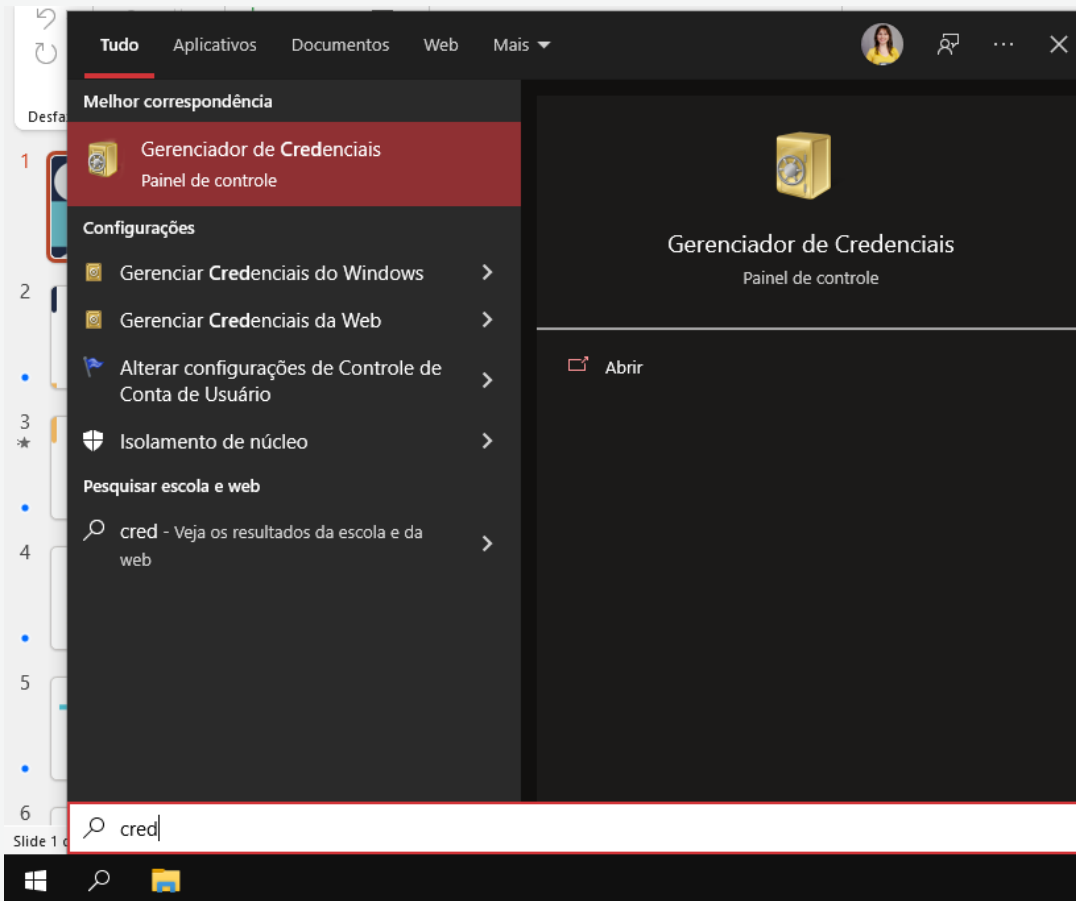
```
git config --global user.name  
git config --global user.email
```

# Autenticando-se

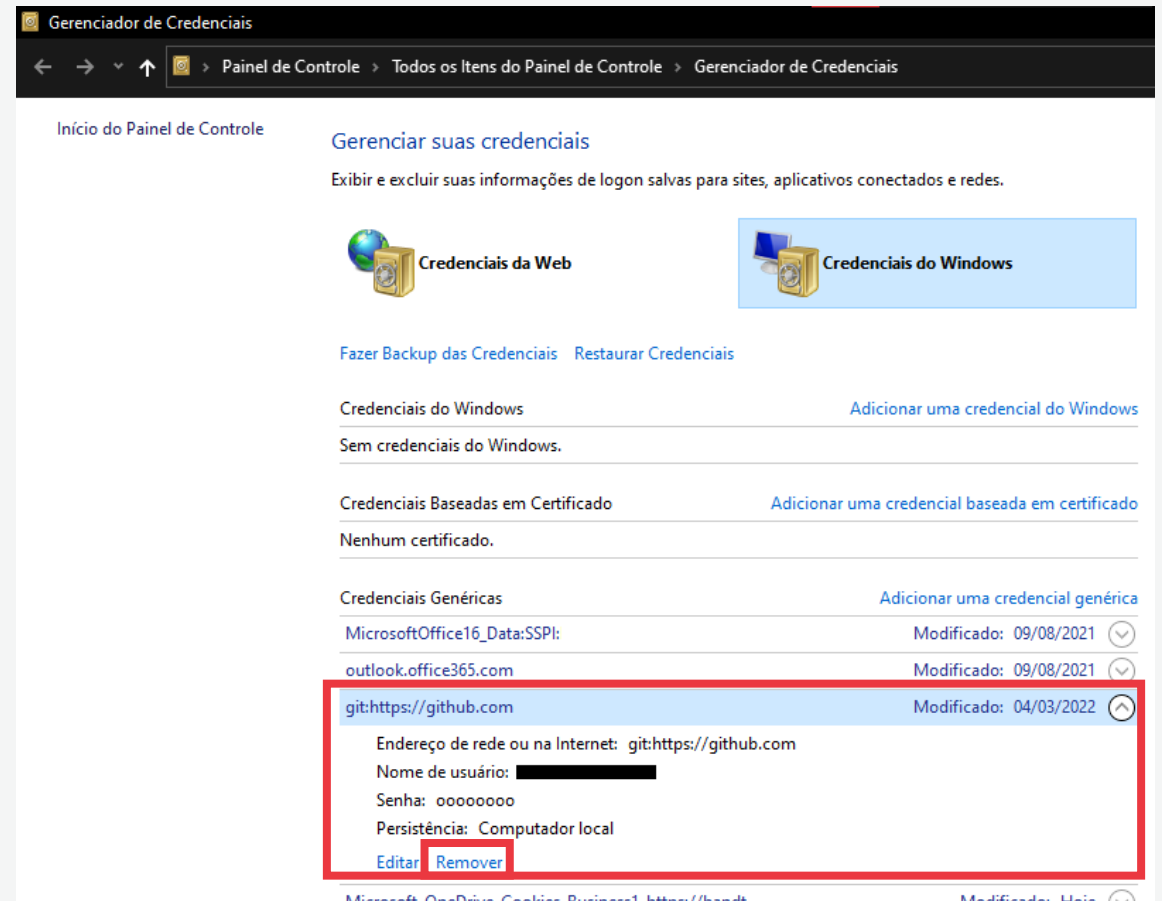
Se for identificado que há um usuário cadastrado e você estiver usando **Windows**, verifique se há alguma informação no Gerenciador de Credenciais:



## 1. Busque por "Gerenciador de Credenciais"



## 2. Identifique a credencial "github" e clique em "Remover"



## Autenticando-se

Configure então o seu usuário...

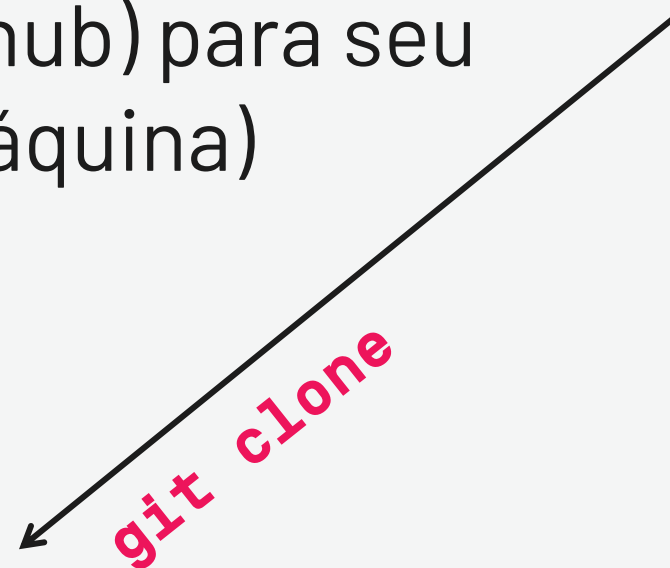
```
git config --global user.name SeuUsuario  
git config --global user.email email.com
```

... E confirme se a alteração foi efetuada...

```
git config --global user.name  
git config --global user.email
```

# GIT CLONE

Cria uma 'cópia' do projeto do repositório **remoto** (git hub) para seu repositório **local** (sua máquina)



# GIT CLONE

*“Como sei se estou “dentro” do meu repositório clonado?”*

No GitBash:

```
co/primeiro-repo-20211-1adsa /C/NOTES/ACADEMIA/primeiro-repo-20211-1adsa  
co/meu-repo (main)
```

No explorador de arquivos do Windows:



## GIT PULL

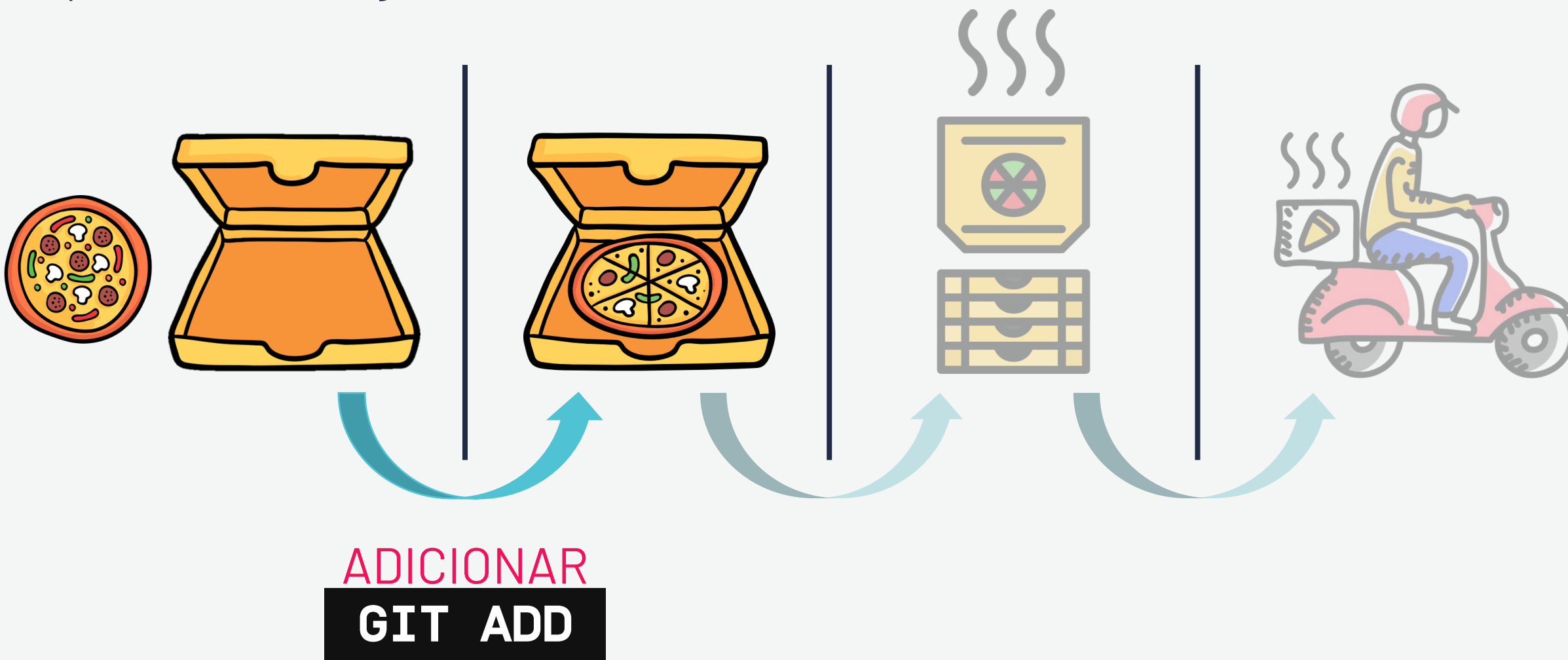
Traz as alterações feitas no repositório remoto para o local



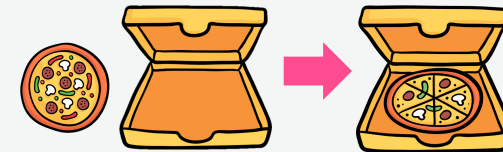


# Como funciona?

*A pizza é o seu código!*

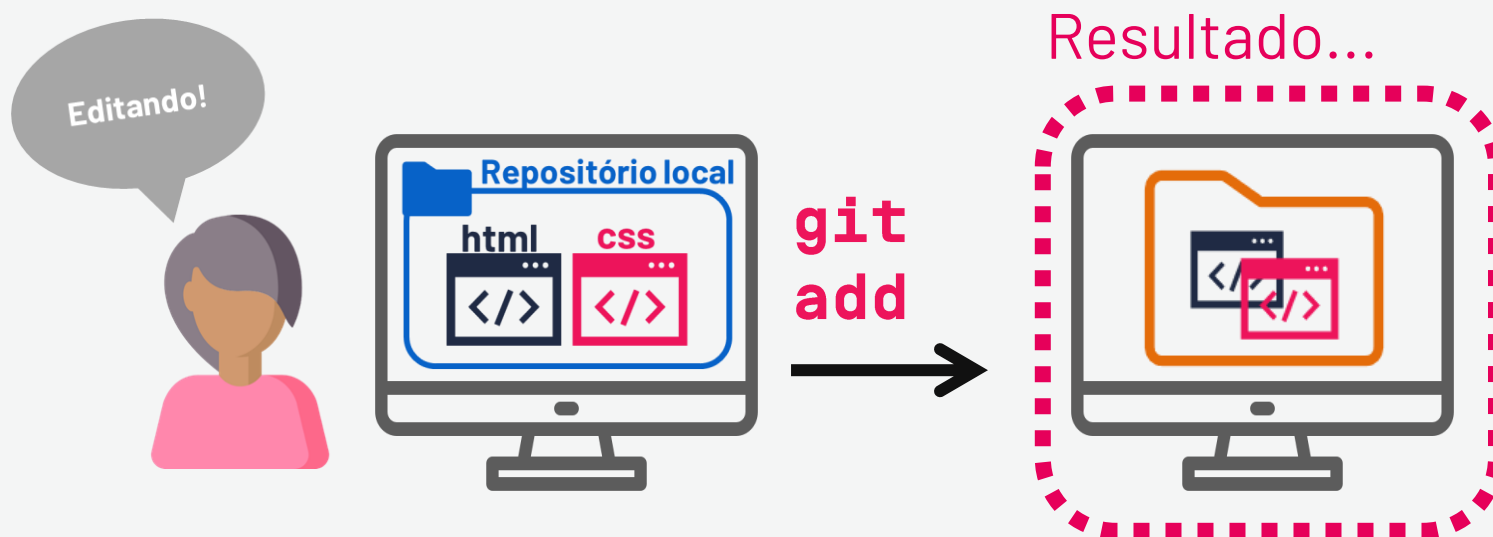


## GIT ADD



**Adiciona arquivo a um pacote aberto**, que será posteriormente enviado (não envia!)

*Você pode executar esse comando várias vezes para o mesmo pacote*

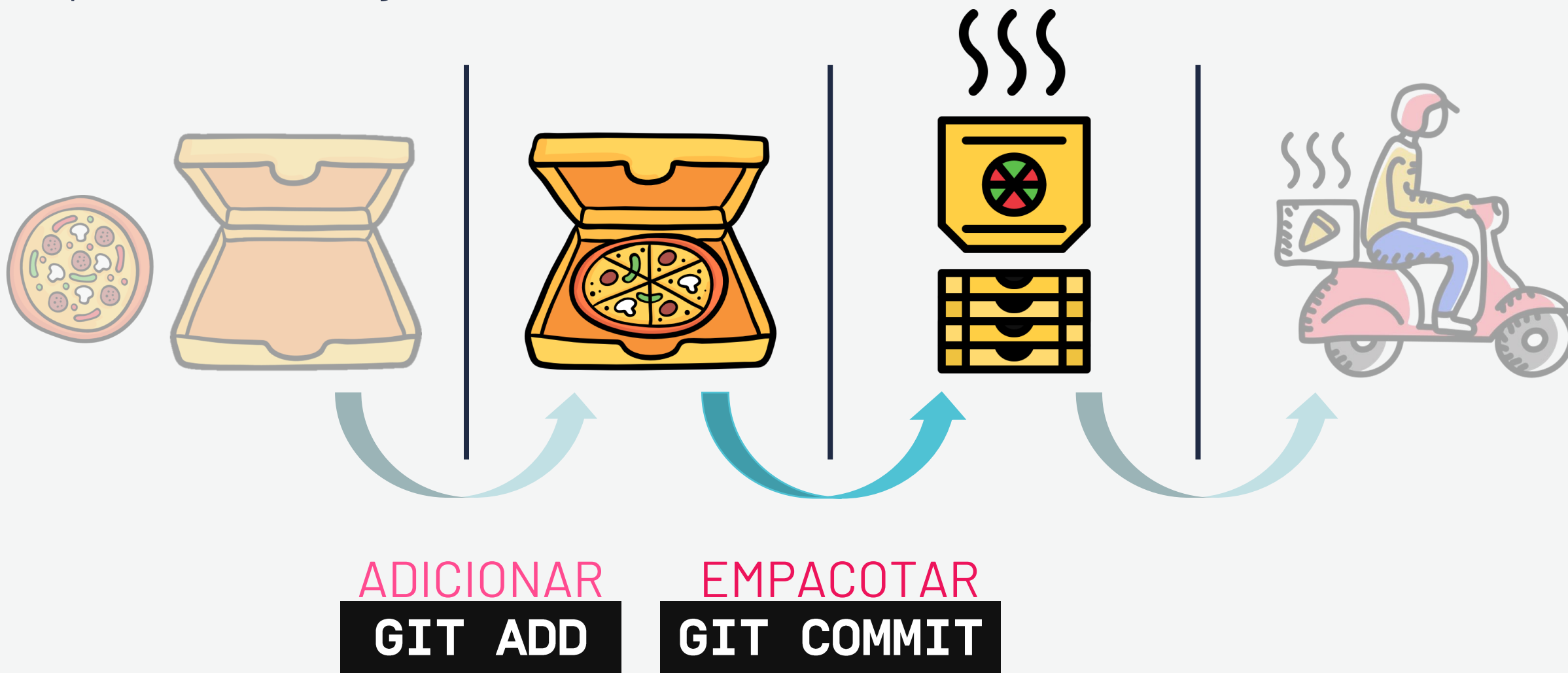


`git add .` → adiciona tudo

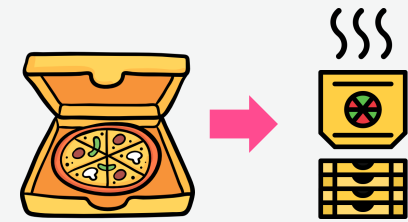
`git add nomeDoArquivoSemEspacos.txt` → adiciona só o arquivo

# Como funciona?

*A pizza é o seu código!*



# GIT COMMIT



```
git commit -m 'mensagem que descreva o  
que foi alterado'
```

Envia alterações para o controle de versão na sua máquina, **separadas por 'pacotes' fechados**



git  
add



git  
commit

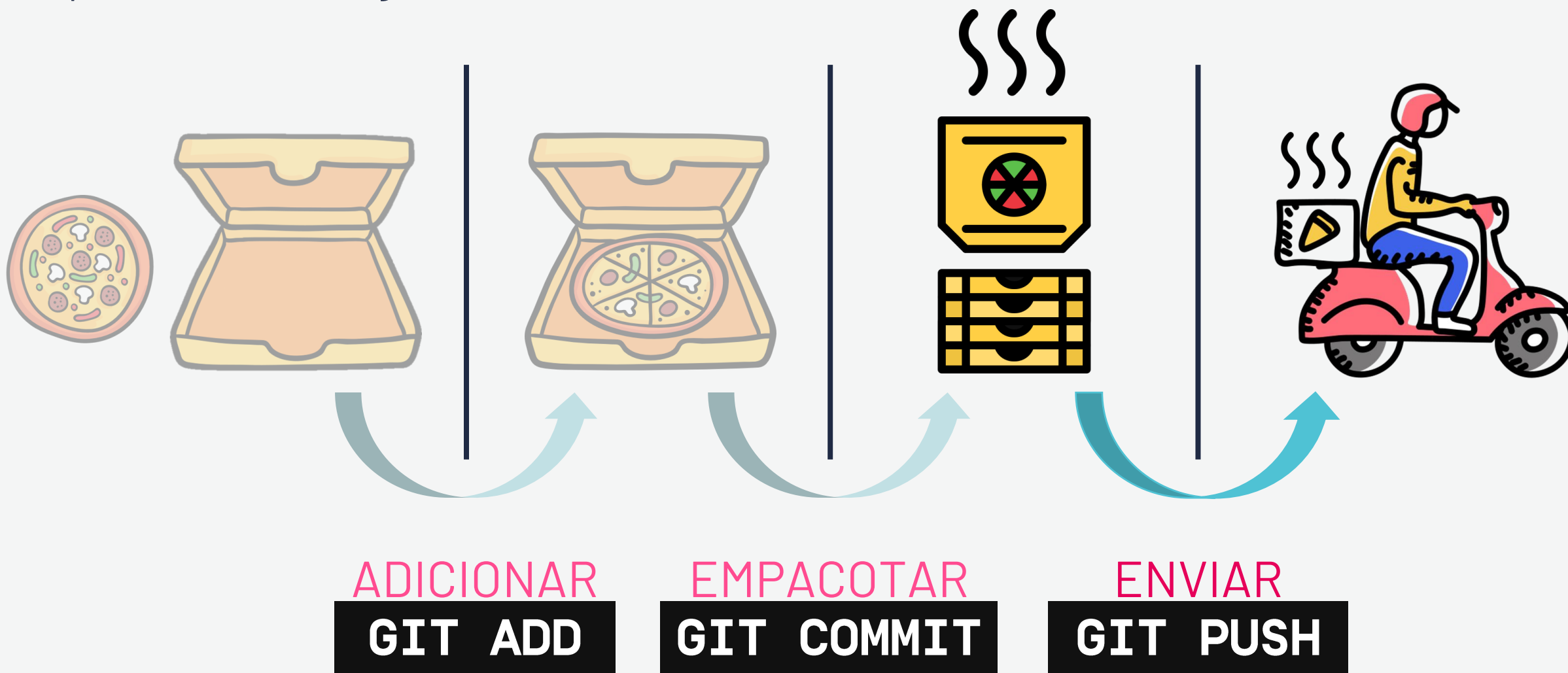


Resultado...



# Como funciona?

*A pizza é o seu código!*

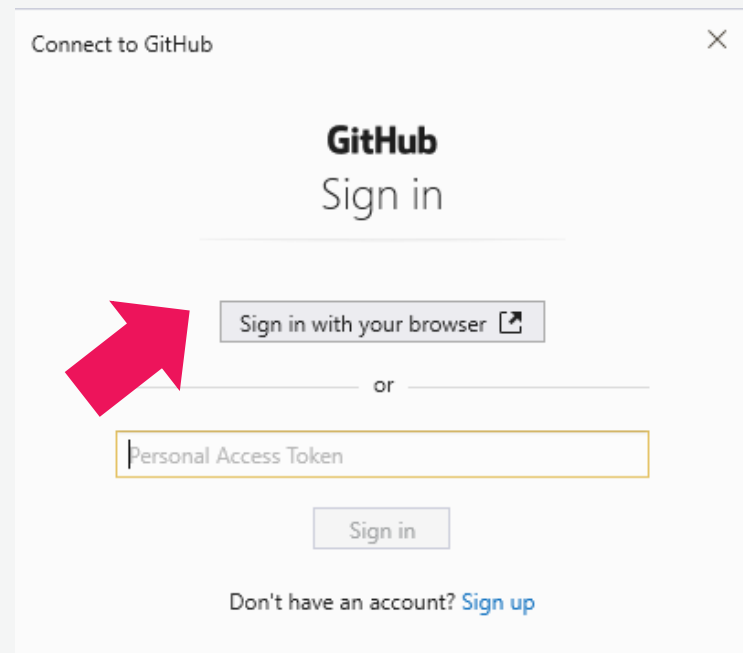


## **Hora de colocar a senha**

O GitHub permite que você efetue suas alterações nos repositórios utilizando Token de acesso em vez da sua senha ou por um código de acesso único.


# Efetuando os comandos que precisam de autenticação

Ao efetuar estes comandos, você verá uma janela parecida com esta:



Connect to GitHub

**GitHub**  
Sign in

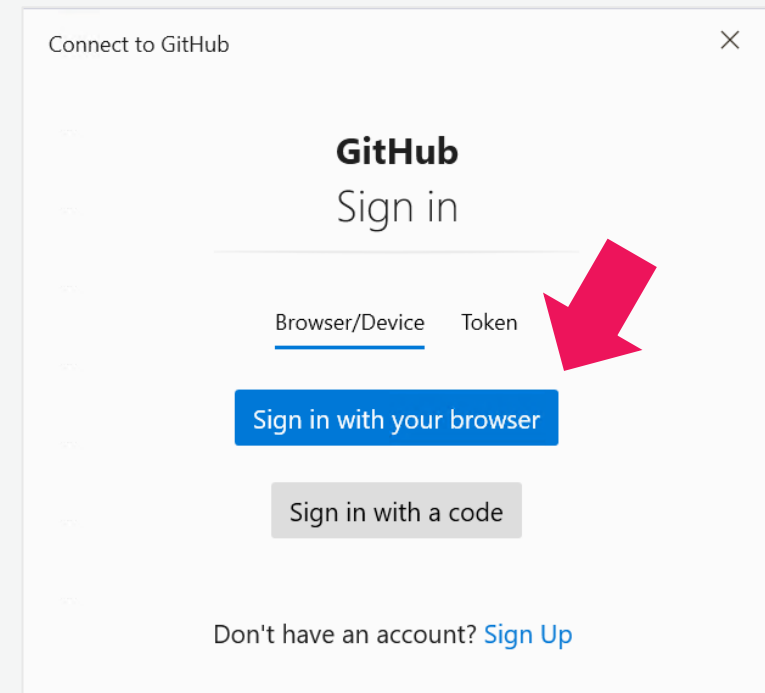
Sign in with your browser 

or

Personal Access Token

Sign in

Don't have an account? [Sign up](#)



Connect to GitHub

**GitHub**  
Sign in

Browser/Device Token

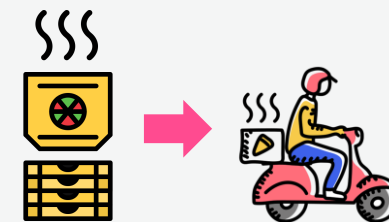
Sign in with your browser

Sign in with a code

Don't have an account? [Sign Up](#)

# GIT PUSH

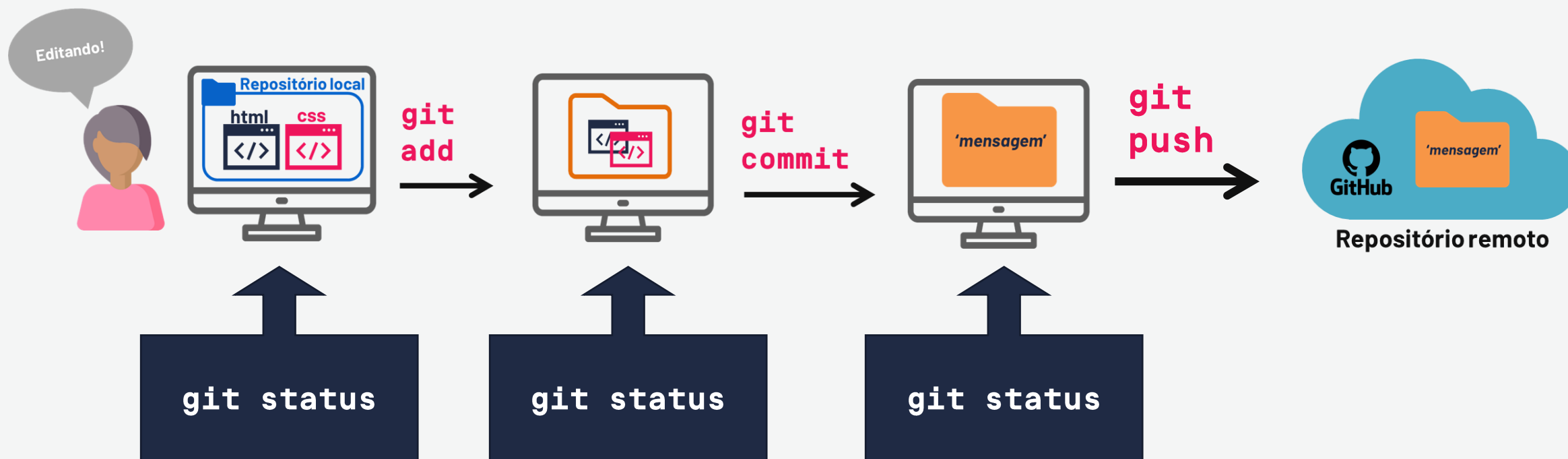
Finalmente **envia** os commits (pacotes) para o repositório remoto (github)





# GIT STATUS

A qualquer momento, pode ser usado para **chechar o estado atual** do repositório.



# GIT STATUS

Após fazer alterações

```
Fernanda Caramico@CARAMICO-47NGDVK MINGW64 ~/teste/meu-primeiro-repo-2022-2 (main)
$ git status
On branch main
Your branch is up to date with 'origin/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:   README.md

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

Depois do **git add**,  
antes do **git commit**

```
Fernanda Caramico@CARAMICO-47NGDVK MINGW64 ~/teste/meu-primeiro-repo-2022-2 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md
```

Depois do **git commit**,  
antes do **git push**

```
Fernanda Caramico@CARAMICO-47NGDVK MINGW64 ~/teste/meu-primeiro-repo-2022-2 (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
```

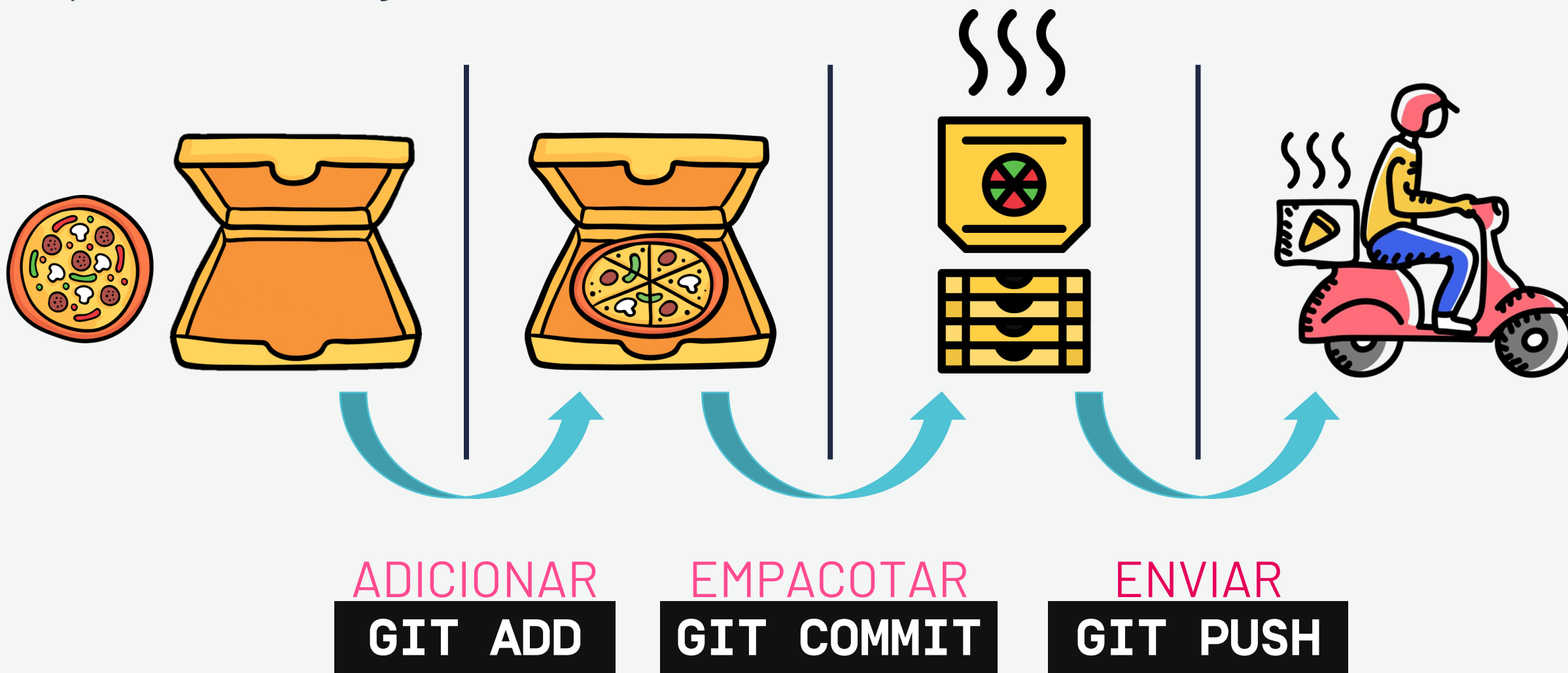
Quando não há alterações  
para serem enviadas

```
Fernanda Caramico@CARAMICO-47NGDVK MINGW64 ~/teste/meu-primeiro-repo-2022-2 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

# Como funciona?

*A pizza é o seu código!*



## Autenticando-se



**Muito importante!** Limpar seu usuário:

```
git config --global --unset user.name  
git config --global --unset user.email
```

... E confirme se a alteração foi efetuada.

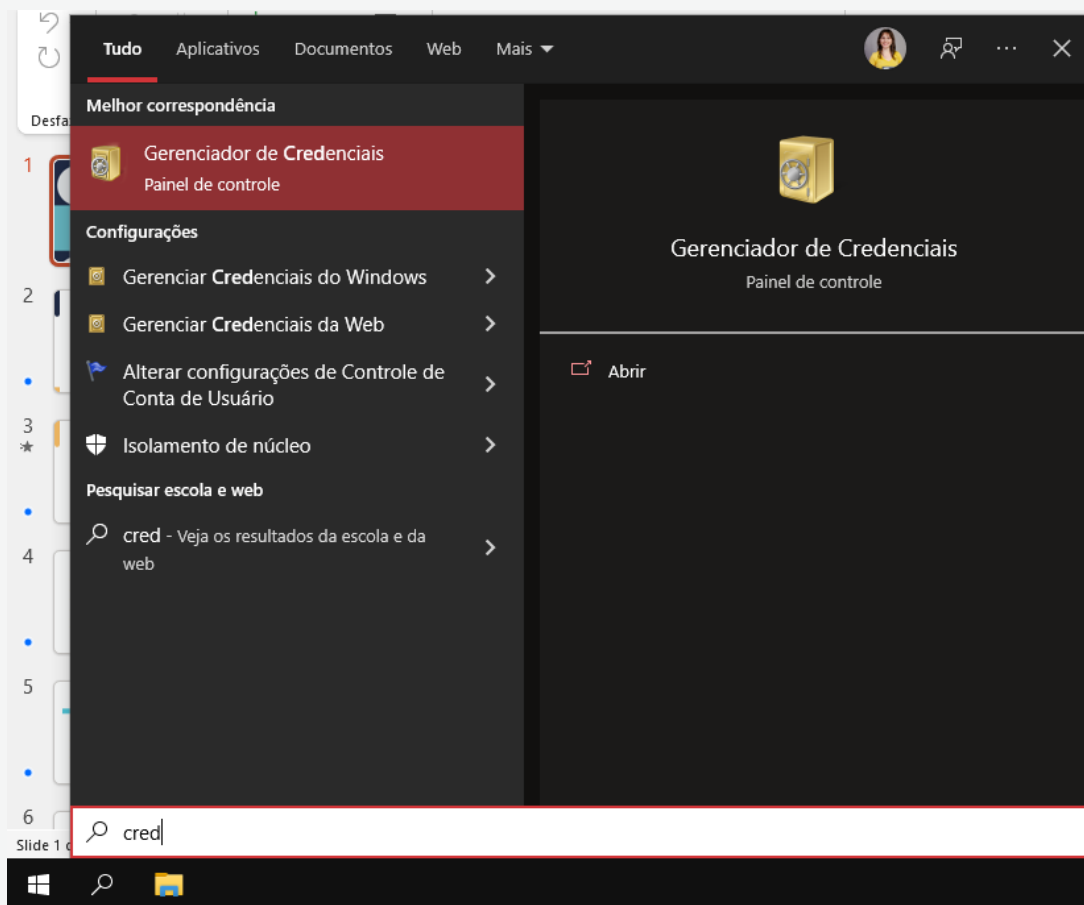
```
git config --global user.name  
git config --global user.email
```

# Autenticando-se

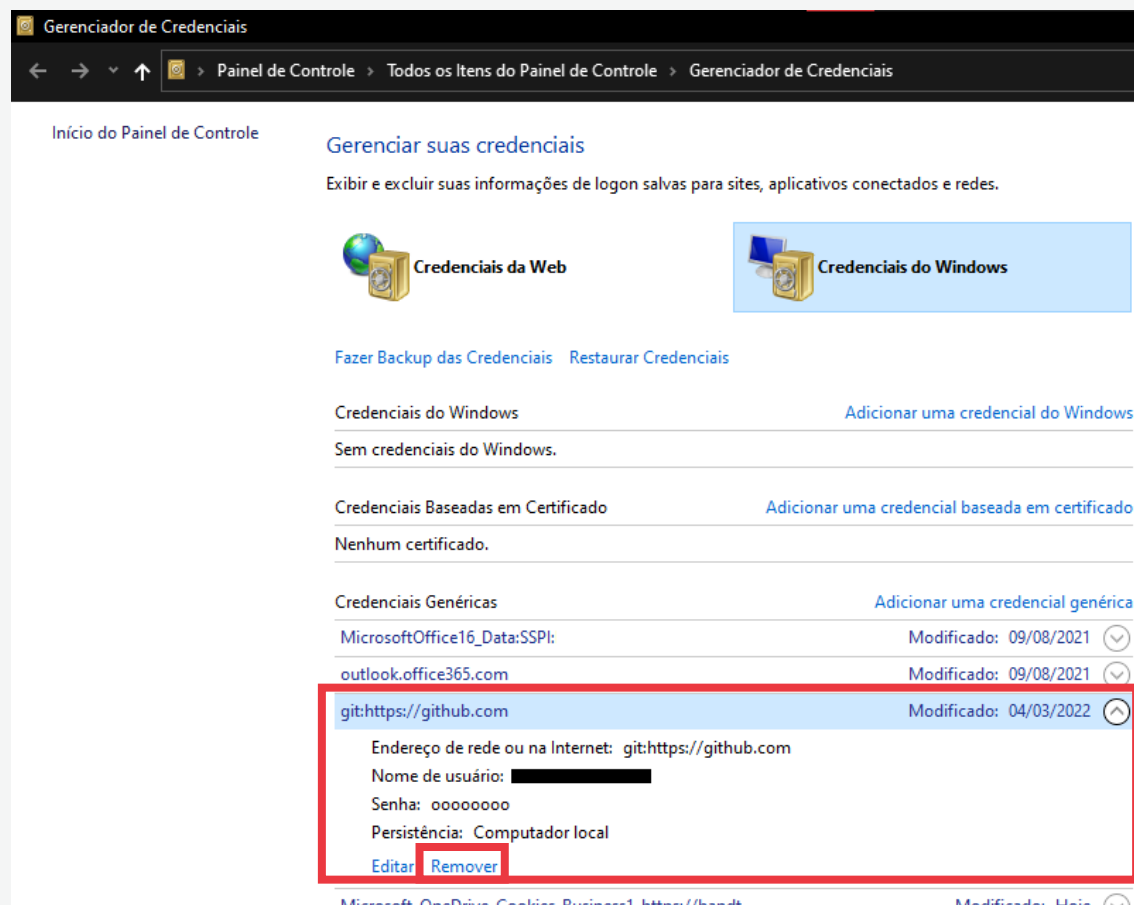
Se for identificado que há um usuário cadastrado e você estiver usando Windows, verifique se há alguma informação no Gerenciador de Credenciais:



## 1. Busque por "Gerenciador de Credenciais"



## 2. Identifique a credencial "github" e clique em "Remover"



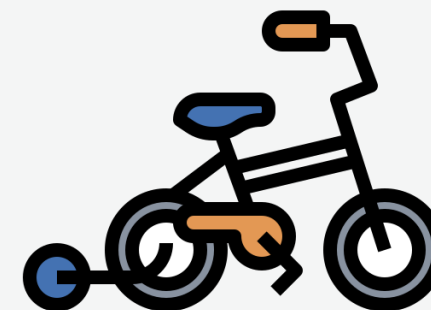
# Training wheels

1ª vez apenas! → **git clone**

Próximas vezes →

- **git pull**  
~~~ *faz as alterações* ~~~
- **git add .**
- **git commit -m 'mensagem'**
- **git push**

A qualquer momento → **git status**



## Links e referências

- Comandos Básicos:

<https://www.hostinger.com.br/tutoriais/comandos-basicos-de-git/>

- Download git: <https://git-scm.com/downloads>
- Git reference: <https://git-scm.com/book/pt-br/v2/>
- Fonte das imagens: Arquivo pessoal + <http://flaticon.com>

**Agradeço**  
a sua atenção!



SÃO  
PAULO  
TECH  
SCHOOL