

PETPOURRI

FERRAMENTAS E CONCEITOS
ESSENCIAIS PARA O ESTUDANTE DE
COMPUTAÇÃO

O QUE SERÁ ABORDADO?

Aula 2:

Versionamento
Conhecendo Git e Github



VERSIONAMENTO

O que é?

Processo de gerenciar e rastrear diferentes versões e alterações feitas em um projeto de código ao longo do tempo.



VERSIONAMENTO

Pra quê?

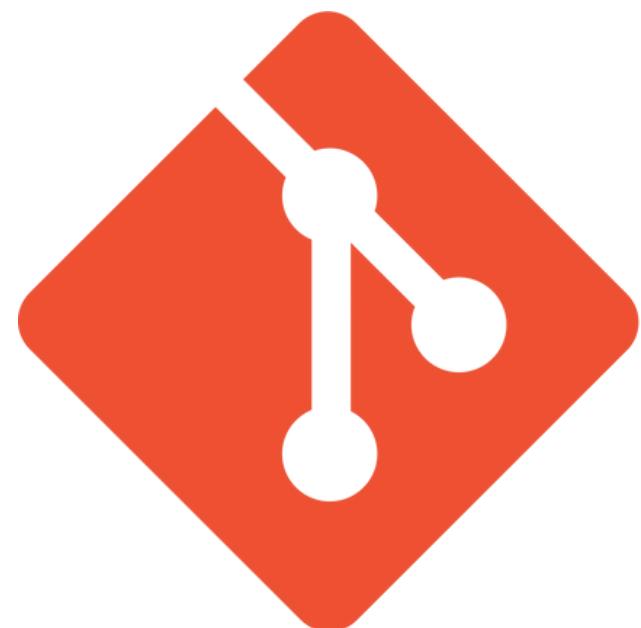
O versionamento de um projeto facilita:

- O desenvolvimento colaborativo
- Criação de um histórico de alterações
- O controle de versões do código
- Correção de bugs
- Gerenciamento de backups

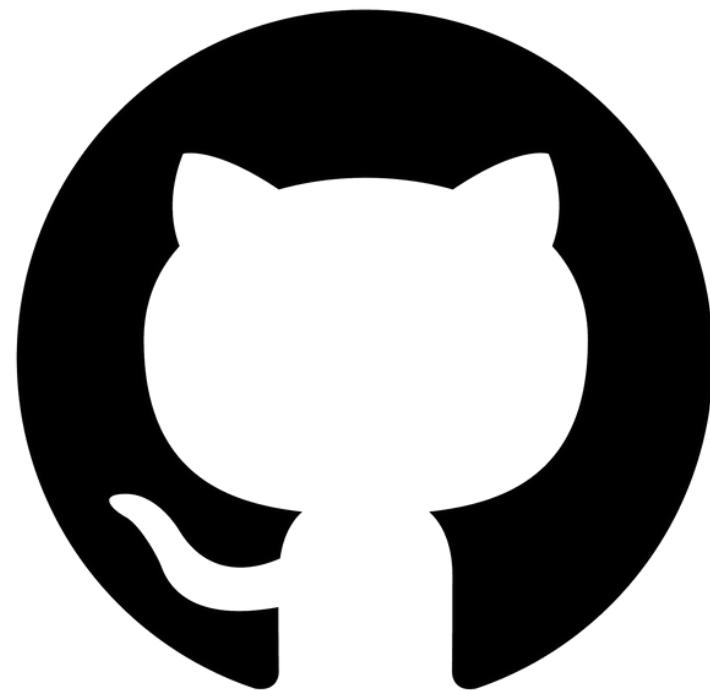


VERSIONAMENTO

Como?



git



GitHub

GITHUB



O que é?

Plataforma de hospedagem de repositórios de código-fonte baseada em nuvem.

Permite que pessoas desenvolvedoras armazenem, compartilhem e colaborem em projetos de software.



GITHUB

Perfil

Toda pessoa é capaz de criar uma conta no Github e construir um perfil. Universitários ainda contam com benefícios extras na plataforma ao se cadastrarem com seu e-mail institucional

Seu perfil será uma "vitrine" online para mostrar suas habilidades, experiências e projetos.



USER/ README.md

É um arquivo especial que funciona como uma introdução ou guia para um perfil de usuário.

Aqui, é interessante acrescentar:

👉 **Uma apresentação**

🎓 **O que você faz ou estuda**

🔧 **Suas principais ferramentas**

🎯 **Seus objetivos**

✉️ **Seus meios de contato**

RayssaBuarque / README.md

Hello World 🙌

Eu sou a Rayssal

Sou uma estudante de tecnologia apaixonada por hackathons e maratonas de desenvolvimento de projetos 🎉

- 🌐 Front-End Developer
- 💻 Cursando Sistemas de Informação
- 🎓 Formada em Desenvolvimento de Sistemas
- ☁️ Atualmente estudando AWS
- 帼 Girls Who Code Alumni 2022

 [LINKEDIN](#)

 [BEECROWD](#)

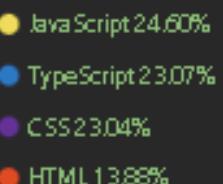
 [LEETCODE](#)

 [My Stats](#)

Rayssa Buarque's GitHub Stats

⭐ Total Stars Earned:	12
⌚ Total Commits:	725
👉 Total PRs:	11
⚠ Total Issues:	0
📅 Contributed to (last year):	3

Most Used Langu



🛠 Linguagens e Ferramentas:

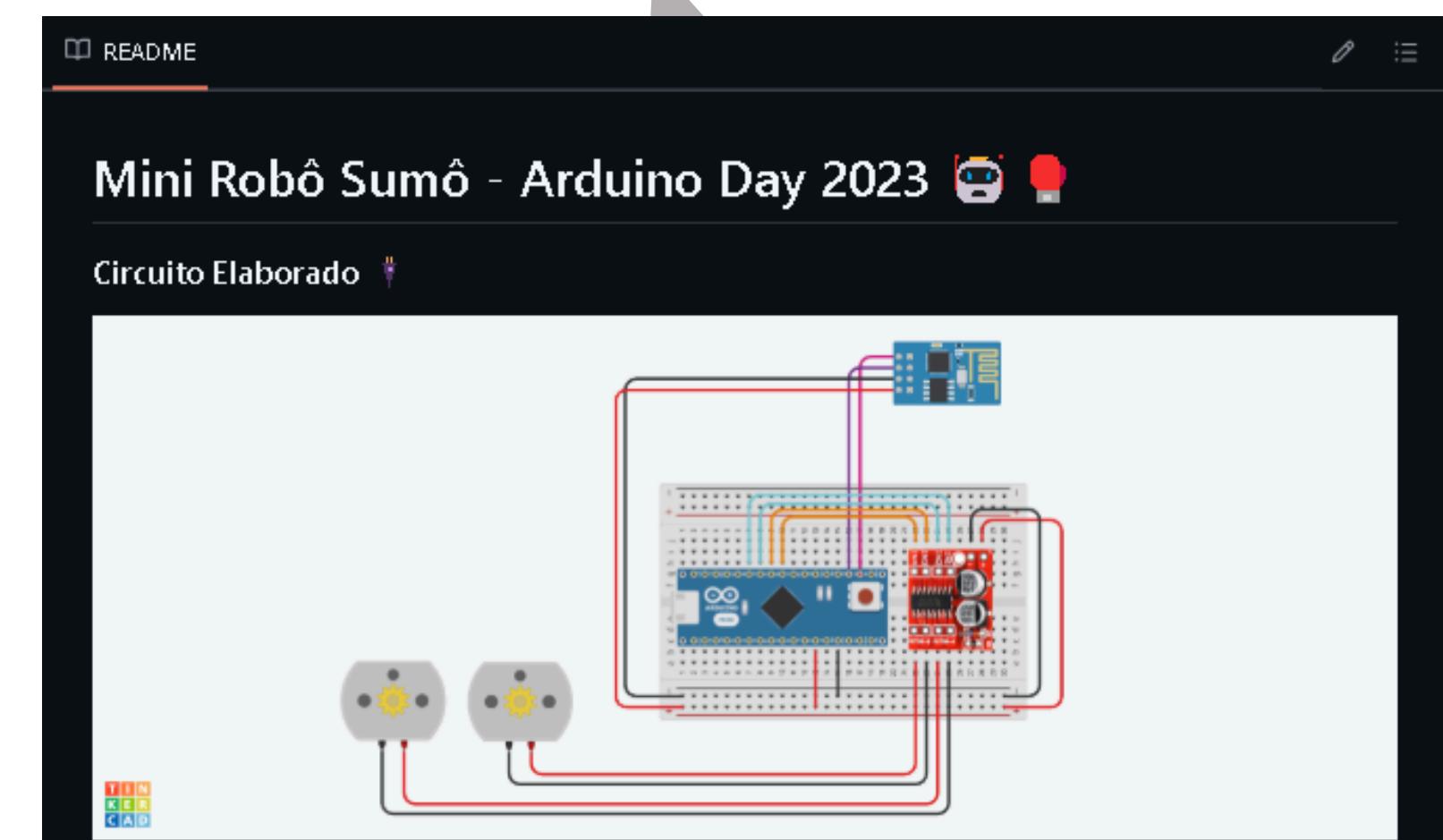


MARKDOWN

o que é?

É uma linguagem de marcação de texto simples, usada para personalizar documentos de forma fácil utilizando elementos de formatação.

Em repositórios do github, arquivos .md (markdown) são usados para documentar repositórios.



Componentes utilizados

- 1 Arduino Nano
- 2 Motores DC com caixa de Redução
- 1 Driver Motor Ponte H
- 1 Módulo Bluetooth HC-05

Modelos 3d

- [Printable Sumo Bot Jr Kit by makenai](#)
- [Ikeda Kogeki Minisumo Robot Sumo Body by JoeCarnine](#)

```
7  ## Circuito Elaborado
8  ![[Circuito dos Robôs](./assets/sumostrike.png)]
9
10 ## Componentes utilizados
11 - 1 Arduino Nano
12 - 2 Motores DC com caixa de Redução
13 - 1 Driver Motor Ponte H
14 - 1 Módulo Bluetooth HC-05
15
16 ## Modelos 3d
17 - [Printable Sumo Bot Jr Kit by makenai](https://www.thingiverse.com/thing:357369)
18 - [Ikeda Kogeki Minisumo Robot Sumo Body by JoeCarnine](https://www.thingiverse.com/thing:3911221)
```

Sintaxe básica

```
# A first-level heading  
## A second-level heading  
### A third-level heading
```

** **negrito** **

* *itálico* *

> citação

- item de lista

``

citando código

``

[link relativo](<https://seu.link.com>)

[imagem - descrição](<https://link..com/foto.jpg>)

[Github docs](#)



ATIVIDADE! !

Proposta:

Criar o seu próprio documento user/README.md para destacar seu perfil!

O que será praticado?:

Utilização de componentes Markdown.

REPOSITÓRIOS

Como funcionam?

É um diretório que armazena arquivos, códigos e o histórico de revisão de um projeto por meio de **Branches** e **Commits**.

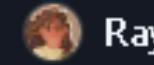
Eles podem ter mais de um colaborador, visibilidade pública ou privada, além de também serem documentados com arquivos README.md



REPOSITÓRIOS

 BookShire-Website Public

 main  1 Branch  0 Tags  Go to file  Add file  Code

 RayssaBuarque update gallery 62e633c · last year 50 Commits

 .vscode initial commit 2 years ago

 src update gallery

 .browserslistrc initial commit

 .editorconfig initial commit

 .gitignore initial commit

 README.md README

 angular.json first commit

 karma.conf.js initial commit

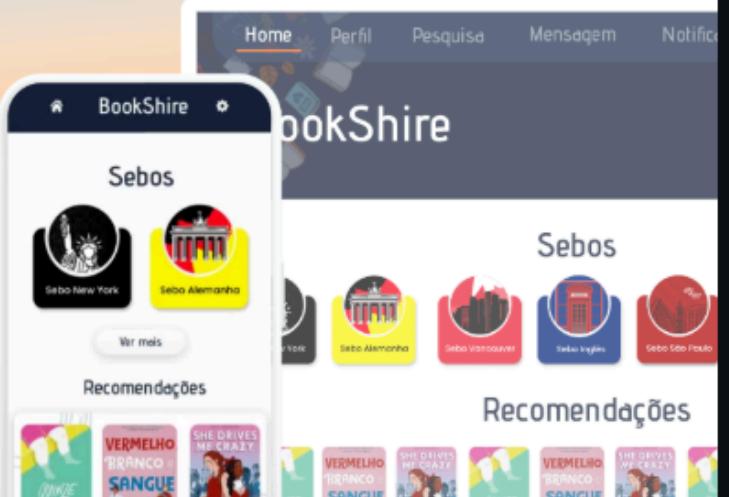
 package-lock.json initial commit

 package.json initial commit

 BookShire Sobre nós Galeria Desenvolvedoras Contate-nos

Porque todos merecemos o direito à leitura.

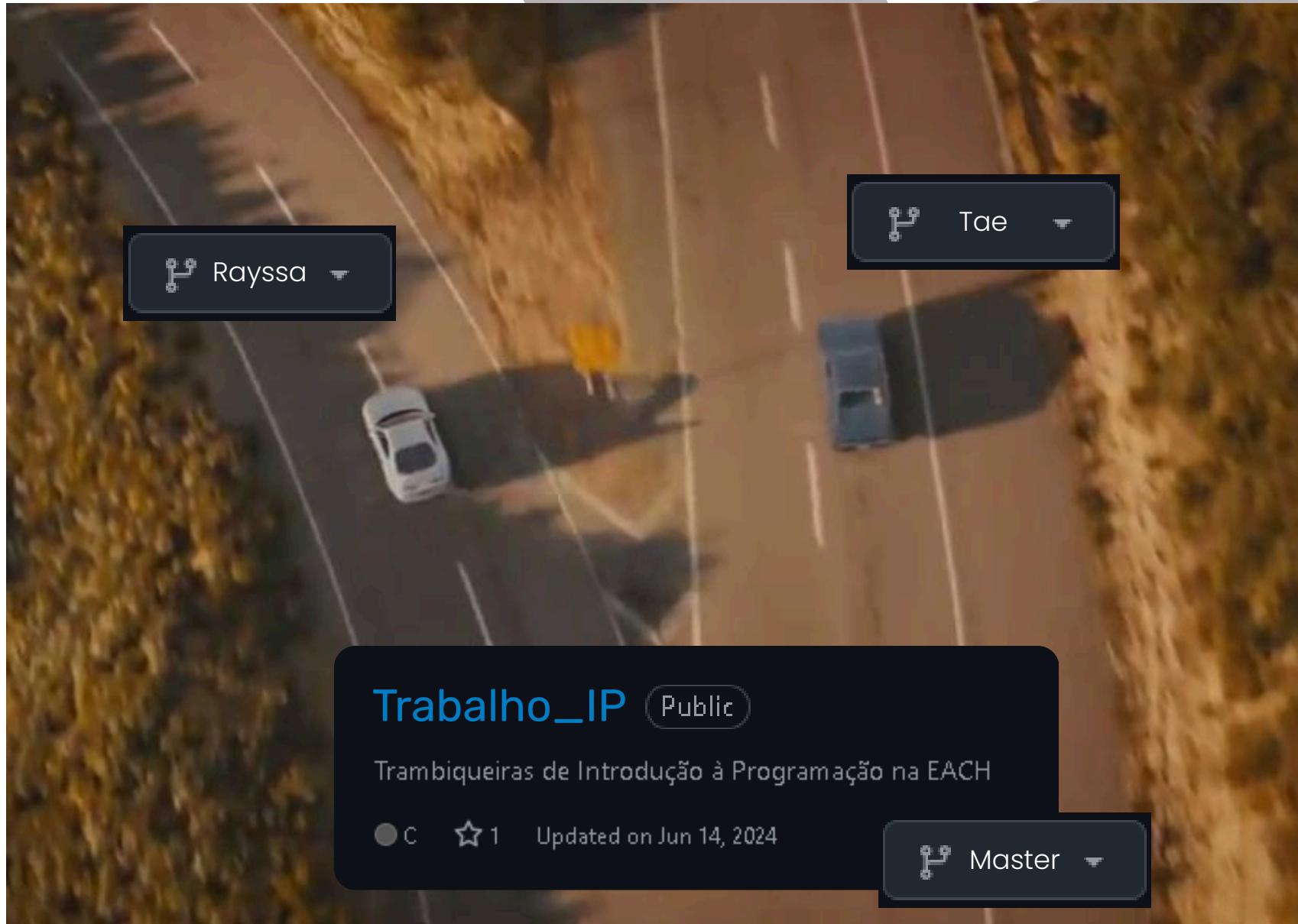
O BookShire é uma plataforma de vendas, doações e trocas de livros usados que visa facilitar o acesso à literatura.



BRANCHES

Ramificações de um repositório. São como linhas de desenvolvimento independentes, que permitem que múltiplos desenvolvedores trabalhem em diferentes partes do código simultaneamente, sem interferir uns nos outros.

Novas branches podem ser criadas ou unidas (**merge**) a qualquer instante de um projeto.



FORK

Cópia de projeto de um repositório público originalmente pertencente a outro usuário.

main 2 Branches 0 Tags Go to file

IsadoraFerrao Merge pull request #16 from JorgeSantosxp/main

README.md Update README.md

index.html linkden

README

Curriculum Vitae Online

Sejam todos bem vindos ao projeto de Curriculum Vitae Online oferecidos pela Digital In...

O projeto tem como objetivo montar nossa primeira página web que será a replica de um conceito de html e Github Pages. Ao fim teremos nosso currículo pronto e disponível de...

RayssaCV_DIO Public

forked from [digitalinnovationone/cv](#)

main 1 Branch 0 Tags Go to file Add file Code

This branch is 4 commits ahead of, 5 commits behind [digitalinnovationone/cv:main](#).

Contribute Sync fork

RayssaBuarque update color palette

d245133 · 2 years ago 18 Commits

FORK

petsi-each/workshop-github

Você pode fazer um fork do repositório “workshop-github”, público no perfil do PETSI-EACH para acompanhar a oficina a partir de agora.

<https://github.com/petsi-each/workshop-github>



COMMITs

São registros individuais de alterações feitas em um ou mais arquivos de um projeto. Que é guardado em um histórico de versões de uma Branch.

- o Commits on Mar 20, 2025
- chore: add hashtag in redirected messages #5
 committed on Mar 20 e028f47  
- o Commits on Mar 7, 2025
- Merge pull request #8 from . /refactor/onboarding  d062cc2  
 authored on Mar 7
- docs: add contributing file
 9be861c  
 authored on Mar 7
- refactor: fix circular dependency
 1ea210b  
 authored on Mar 7

COMMITTS

O que NÃO fazer?

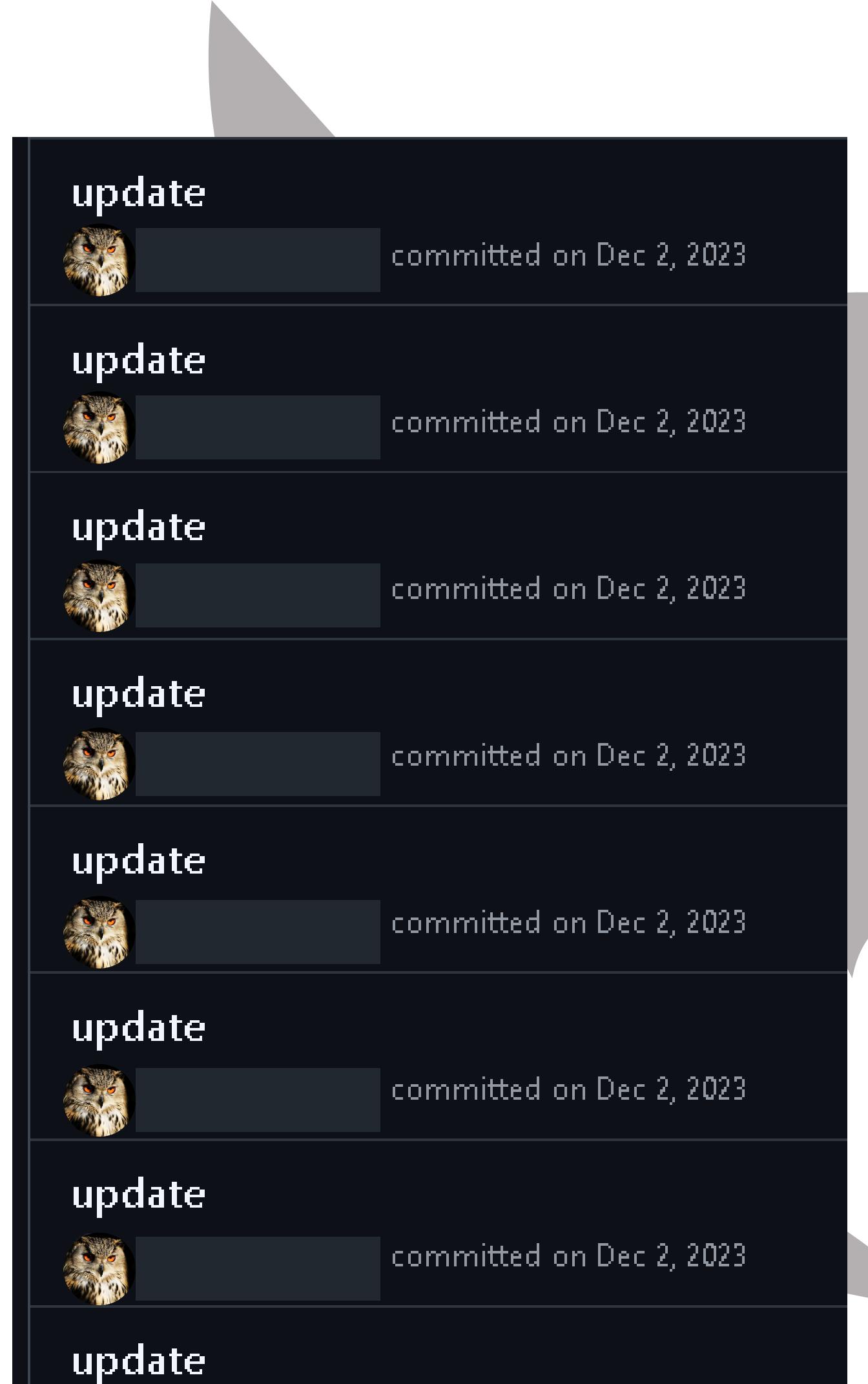
- Registros pouco informativos, com títulos vagos ou não padronizadas.
 - Registro de múltiplas alterações em um único *Commit*.

AAAAAAAAAAAAAAA login/cadastro

 committed on Nov 27, 2023

arrumando bug de login/cadastro no celular

 committed on Nov 27, 2023



COMMITS

Boas Práticas

Utilize **Commits Atômicos** – cada commit é responsável por uma única alteração e, idealmente, representa a alteração completa.

Padronize suas mensagens com **Conventional Commits**, regras que facilitam a compreensão do histórico de versões de seu projeto.



Commit Type	Title	Description	Emoji
feat	Features	A new feature	✨
fix	Bug Fixes	A bug Fix	🐛
docs	Documentation	Documentation only changes	📄
style	Styles	Changes that do not affect the meaning of the code (white-space, formatting, missing semi-colons, etc)	💎
refactor	Code Refactoring	A code change that neither fixes a bug nor adds a feature	📦
perf	Performance Improvements	A code change that improves performance	🚀
test	Tests	Adding missing tests or correcting existing tests	🎯
build	Builds	Changes that affect the build system or external dependencies (example scopes: gulp, broccoli, npm)	🛠️
ci	Continuous Integrations	Changes to our CI configuration files and scripts (example scopes: Travis, Circle, BrowserStack, SauceLabs)	⚙️
chore	Chores	Other changes that don't modify src or test files	♻️
revert	Reverts	Reverts a previous commit	⏪



O que é?

O Git é um sistema de controle de versão de arquivos comandado por terminal; responsável por registrar e guardar o histórico de alterações sempre que alguém modifica arquivos monitorados por ele.

Não confunda Git com Github, enquanto o Git é um sistema de versionamento, o Github é uma plataforma projetada para hospedar e compartilhar repositórios git.

COMO USAR O GIT?

- 1. Instale** o sistema do Git em seu computador;
- 2. Configure seu perfil** de usuário que irá assinar os commits;

```
> git config --global user.name "Seu Nome"  
      git config --global user.email "seu@email.com"
```

- 3. Inicie ou clone um repositório git.**

```
> git init
```

```
> git clone https://github.com/seunome/seurepositorio.git
```

- Adicionando arquivos ao controle de versão.

>— git add nome-do-arquivo

- Criando um **commit** dos arquivos no controle de versão

>— git commit -m "Sua mensagem de commit aqui"

- Criando uma **nova Branch**

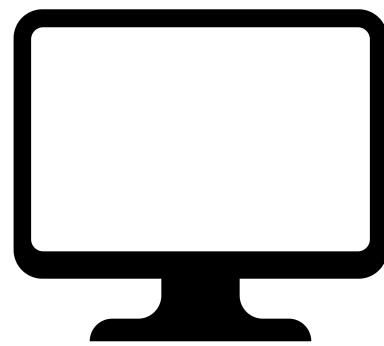
>— git branch nome-da-branch

- **Navegando** entre *branches* ou versões *commitadas*

>— git checkout nome-da-branch-ou-hash-do-commit

- **Puxe a versão de uma branch** para o seu repositório local

>— git pull origin nome-da-branch



FLUXO DE USO

LOCAL

DIRETÓRIO

Os arquivos que você mexe ficam aqui

STAGING AREA

Suas mudanças ficam guardadas aqui por um tempo, antes do commit

REPOSITÓRIO

O git guarda as informações sobre o controle da versão do seu computador aqui

REMOTO

Esta é a raiz do Repositório, armazenada em uma plataforma como o GitHub!

Todas as suas alterações ficam salvas remotamente aqui.

FLUXO DE USO

LOCAL

DIRETÓRIO

Os arquivos que você mexe ficam aqui

STAGING AREA

ficam guardadas aqui por um tempo, antes do commit

REPOSITÓRIO

git guarda informações sobre o controle da versão do seu computador

REMOTO

GIT CLONE

a raiz do diretório, armazenada em uma plataforma como o GitHub!

GIT PULL

Todas as suas alterações ficam armazenadas remotamente aqui.

GIT CHECKOUT

git guarda

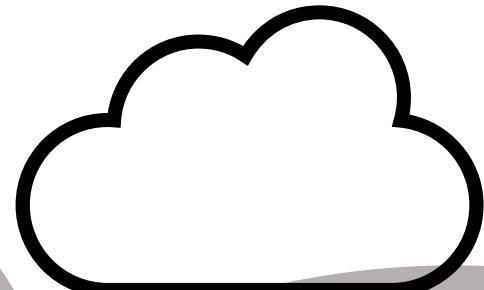
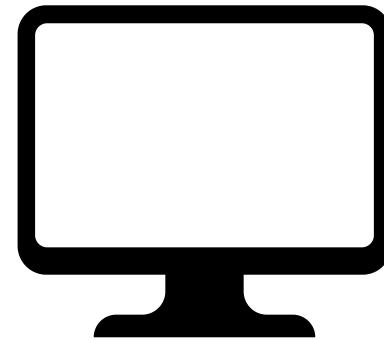
informações

sobre o

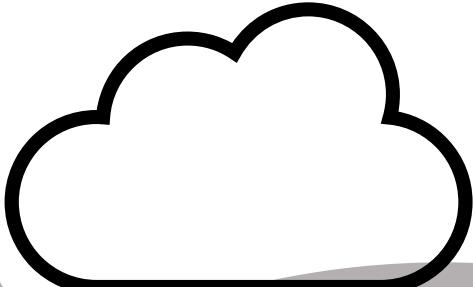
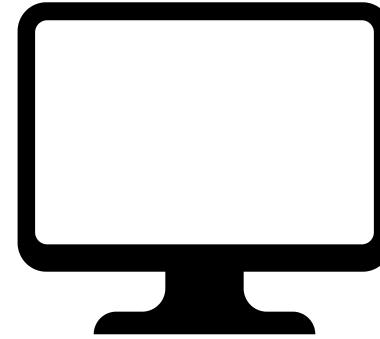
controle da

versão do seu

computador



FLUXO DE USO



PULL REQUEST

Conforme desenvolvedores vão trabalhando em *branches* individuais, eventualmente será necessário combinar seu código com a *branch master* de novo.

Para isso, devs solicitam um **pull request** antes de enviar suas alterações de volta.

Dessa forma, caso algum arquivo tenha recebido mudanças conflitantes com códigos de outra pessoa, o Git retornará um **merge conflict**, que é um pedido de ajuda para análise dessa sobreposição.

Merge Conflict apresentado na interface do VS Code

```
16 */  
17 <<<<< HEAD (Current Change)  
18 function printMessage(showUsage, message) {  
19   console.log(message);  
20  
21 ====  
22 function printMessage(showUsage, showVersion) {  
23   console.log("Welcome To Line Counter");  
24   if (showVersion) {  
25     console.log("Version: 1.0.0");  
26   }  
27 >>>>> theirs (Incoming Change)  
28 }
```

I SHOULD PUT THIS



IN A GIT REPO

**OBRIGADA POR
COMPARECEREM,
VOCÊS SÃO UNS
QUERIDES**