# PrimeiroRepositorio

Teste do Git Hub para desafio da DIO bootcamp da TQI

**Olá Colegas! Estou muito feliz por estar entrando neste mundo, espero aprender e ajudar aqui nessa magnifica plataforma!**

**Gostaria de deixar neste repositório meu aprendizado, o passo a passo para conseguir realizar.**

1º Realizei o curso "Introdução ao Git e ao GitHub"com Otávio Reis no bootcamp;

2º Realizei o curso "Criando seu repositório no GitHub para Compartilhar seu progresso" com Venilton Falvo Jr no bootcamp da TQI;

3º Realizei os passos com o video do "Curso de Git e Github COMPLETO 2021 [Iniciantes] + Desafios + Muita Prática"com Jhonatan da DevAprender.

1. - Instalei o Git no computador Windows 64 bits

link: <http://git-scm.com/download/win>

Segui confirmando todas telas para instalação;

1. - Instalei o Typora, aplicativo indicado pelo Otávio Reis para salvar arquivo em tipo md

link: <https://typora.io/>

1. - Criei minha conta no GitHub = wirlamaia

link: <https://github.com/signup?source=login>

1. - \*\*\*Importante\*\*\* Para conseguir realizar conexões da máquina com o GitHub precisei registrar um token (Senha)

[Seguir este Tutorial. Link](https://www.alura.com.br/artigos/nova-exigencia-do-git-de-autenticacao-por-token-o-que-e-o-que-devo-fazer)

1. - Para excluir um repositório do seu GitHub, seleciona-o, clica no botão Settings, ao final da página botão “**Delete this repositor”, digita o nome de usuario e / (barra) e o nome do repositório, botão “I Understand”, Senha de usuário, botão “Confirm password”; Pronto, excluído.**
2. Principais comandos utilizados, importantes ao iniciar:

**COMANDOS** GIT UTILIZADOS

$ **git clone** <https://github.com/wirlamaia/PrimeiroRepositorio.git> - Clonar o repositório do GitHub online para o Git local;

$ **cd** “Nome do repositório” – Trabalhar no repositório Git local;

$ **touch .gitignore** – Criar arquivo “.gitignore” que se informa os arquivos e pastas que não precisam ser enviados da pasta para a versão Git repositório;

$ **CTRL+L** – Limpar a tela;

$ **ls** – visualizar arquivos da pasta do repositório local;

$ **ls -a** – visualizar arquivos da pasta do repositório local, inclusive os ocultos;

$ **git status**  – Verificar situação do Git repositório local e arquivos conectados (sim ou não);

$ **git branch**  – Demonstra as branchs existentes, sendo a que acompanha o \* símbolo é a atual utilizada;

$ **git branch NovoNome**  – Criar uma nova branch (particição de versão apoio a original);

$ **git checkout NomeBranch**  – Alternar para branch descrita;

$ **git branch -M “main”** – Alternar para branch descrita, tornando a branch como principal;

$ **git checkout -b NomeDaNovaBranch master** – Criar nova branch copia da branch master;

$ **git add Nome do arquivo** -  Adicionar arquivo especifico no repositório da versão git local;

$ **git add .** -  Adicionar todos arquivos e pastas no repositório da versão git local;

$ **git add restore .** - Atualiza os arquivos que foram alterados da pasta na versão git local;

$ **git commit -m “Commit Inicial Nome dado a versão”** - Importante para identificar as fases do repositório;

$ **git init** – Criar arquivo “git.” na pasta no computador, um dos primeiros passos, iniciando o repositório local;

$ **git config --global user.email** **“**[**email@gmail.com**](mailto:email@gmail.com)**”** – Informar ao Git local o email de usuário utilizado;

$ **git config --global user.name** **“NomeUsuario”** – Informar ao Git local o Nome de usuário utilizado;

$ **git config --global – unset user.name** **“NomeUsuario”** – Limpar ao Git local o Nome de usuário utilizado;

$ **git config --list** – Demonstrar lista de configurações;

$ **git remote add origin** [**https://github.com/NomeUsuario/NomeProjetoRepositorio**](https://github.com/NomeUsuario/NomeProjetoRepositorio) - Informar ao Git repositório local para onde será enviado, endereço do GitHub;

$ **git push** – Enviar a versão Git local para GitHub online;

$ **git push –set-upstream origin main** – Ao rodar o comando git push, o Git informa para dar este comando exemplo relativo, que realizamos para enviar corretamente a versão;

$ **git push -u origin main** – Enviar a versão Git local para GitHub online pela primeira vez, utilizando a branch main;

$ **git pull** – Trazer a versão repositório do GitHub online para Git local;

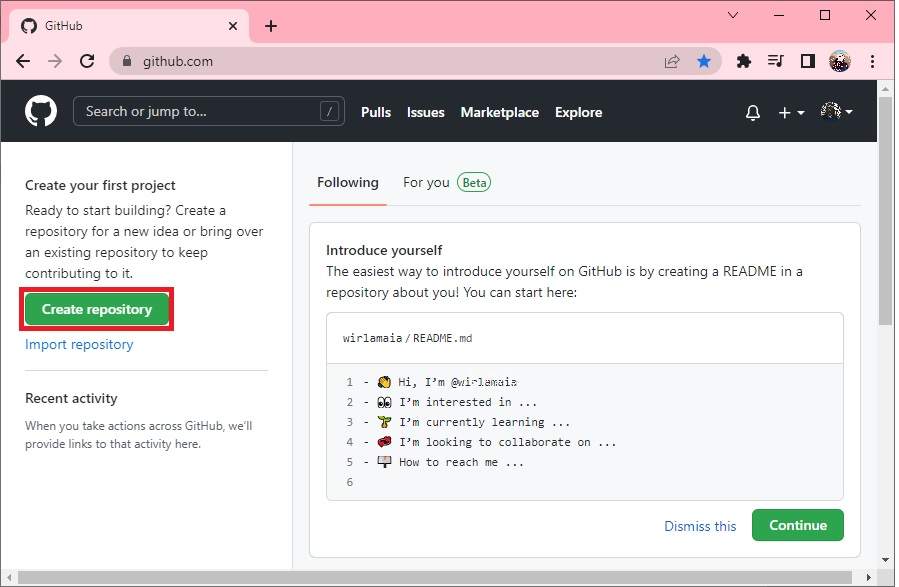
$ **git merge NomeBranch** – Passa os arquivos do repositório da NomeBranch para o repositório master;

$ **git reflog** – Demonstrar as versões commitadas;

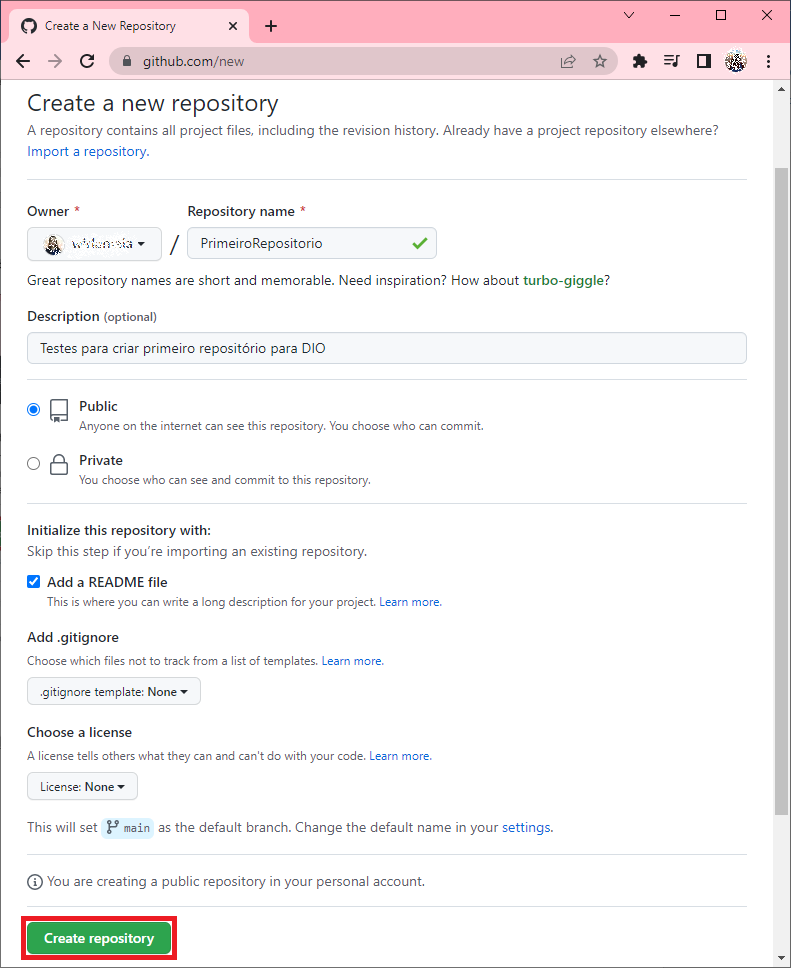
$ **git reset** **–hard** **Numero123Nu** – Volta o repositório para a versão informada, Código da versão demonstrada no comando git reflog, que foi inserida no comando git commit -m “Nome”;

$ **git remote -v** - Visualizar arquivos que não foram enviados para o GitHub repositório online;

1. Pratica de realizar um Repositório no GitHub:
   1. Criar um novo repositório – Botão ‘Create repository’

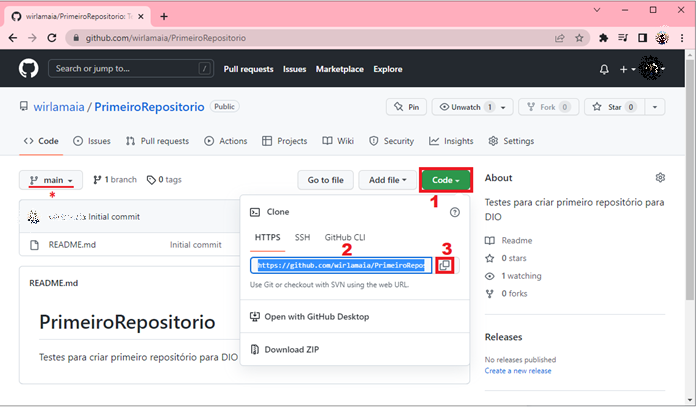


* 1. Digitar nome do repositório, no caso deixei público e a criação do arquivo README.md (Leia-me), Botão ‘Create repository’

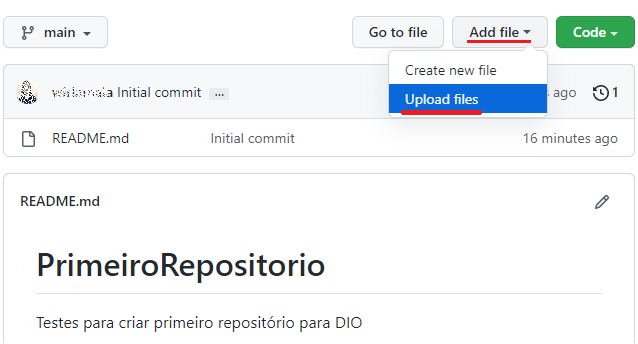


* 1. Observo que a branch principal tem o nome ‘main’;

No botão ‘Code” 1, aba HTTPS 2, copiei o link do repositório 3.

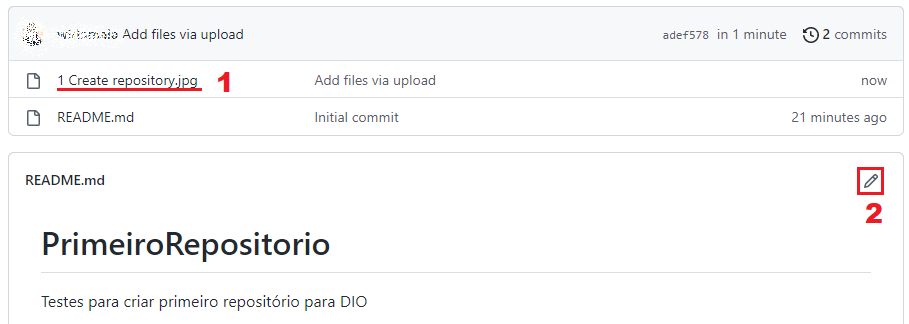


* 1. Adicionei uma imagem, no botão ‘Add file’, sub-botão ‘Upload files’, de imagem JPG, pelo campo ‘Choose your files’ e botão ‘Commit changes’



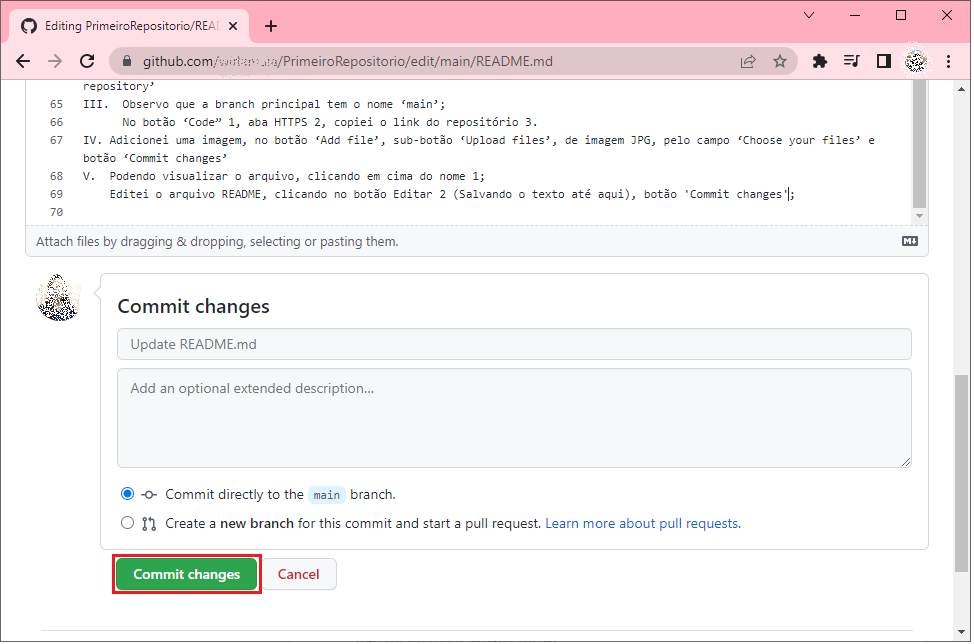
* 1. Podendo visualizar o arquivo, clicando em cima do nome 1;

Editei o arquivo README, clicando no botão Editar 2;

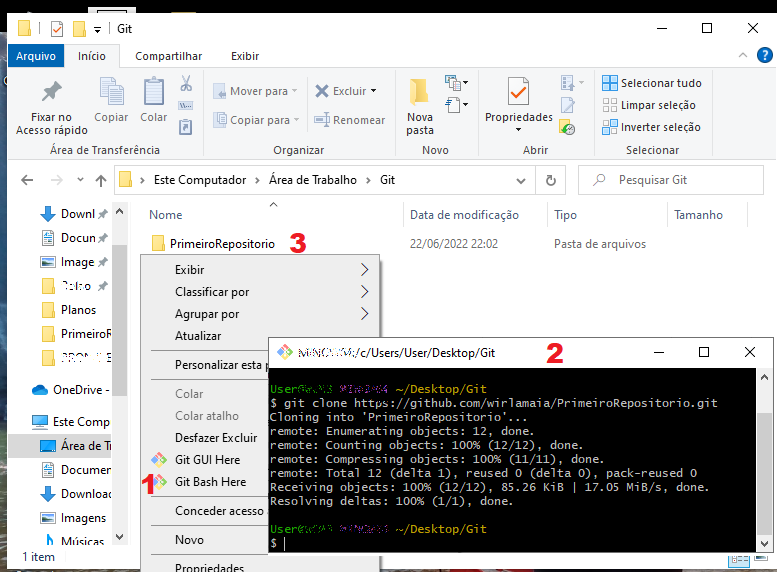


* 1. Incluso texto, botão ‘Commit changes’;

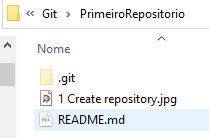
Algumas frases se juntaram, editei novamente visualizando pelo ‘Preview’ antes de Commitar finalmente.



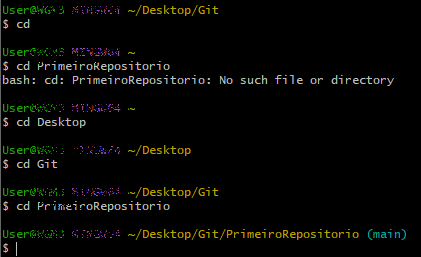
1. Criei no meu computador uma pasta na área de trabalho, nome ‘Git’;
   1. Dentro da pasta criada para o repositório, cliquei com botão direito do mouse e selecionei o ‘Git Bash here’;
   2. Comando ‘$ **git clone** <https://github.com/wirlamaia/PrimeiroRepositorio.git>’, link do repositório;
   3. Comando ‘$ CTRL+L’ para limpar tela;



* + 1. Na pasta há os arquivos corretamente:

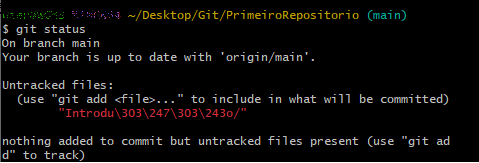


* 1. Comando $ **cd**, testei voltando e indo para a pasta do repositório local, ao entrar nela é demonstrado a branch (main)

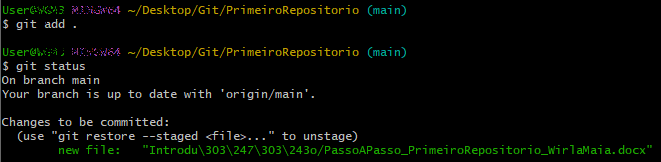


* 1. Adicionei uma pasta Introdução com o arquivo docx (este);

Comando $ **git status**;

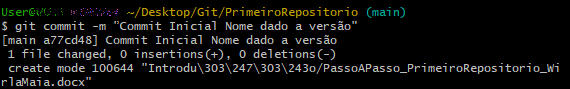


* 1. Comando $ **git add .**, adicionando o arquivo na versão local;

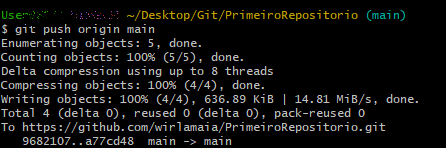


* 1. Comando $ **git commit -m “Commit Inicial Nome dado a versão”**, Informando as mudanças ocorridas para organizar os commits;

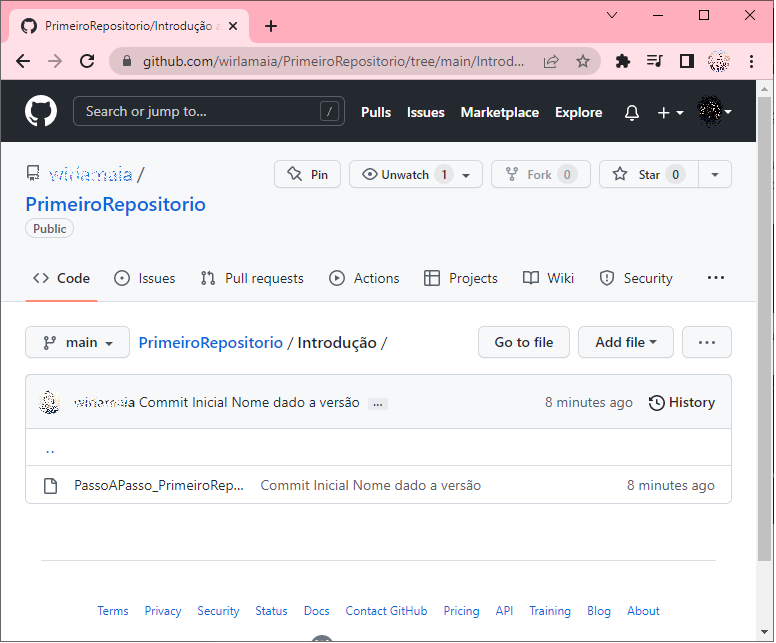
Obs: para colar o texto na tela de comandos, preciso clicar com o botão direito do mouse e clicar em ‘paste’, devido o atalho informado SHIFT+Ins não funcionar; e não consigo copiar do word quando o comando possui aspas, pois desconfigura na tela de comando, então copio do bloco de notas;



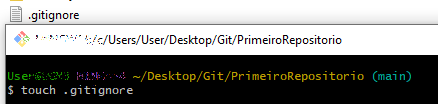
* 1. Comando $ **git push origin main**, Enviando as alterações local da máquina para o servidor do repositório no GitHub online;



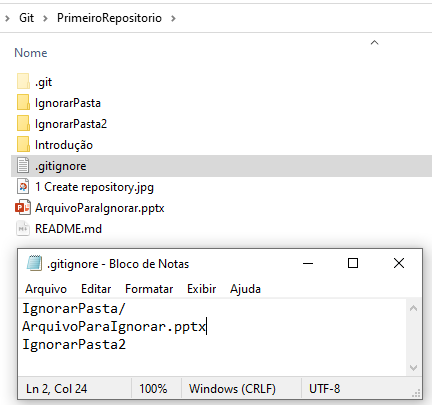
GitHub, F5 para atualizar, já está com as alterações:



* 1. Comando $ **touch .gitignore** – Criar um arquivo na pasta para ignorar o que não desejar que seja enviado a versão do repositório;

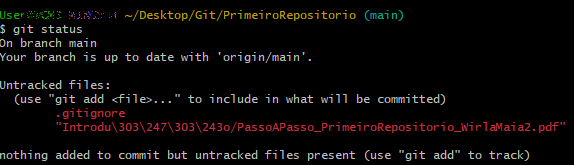


Nesse arquivo coloquei uma pasta que não precisa ir para a versão:



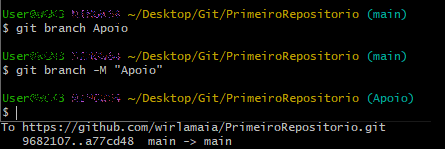
Assim estes arquivos não são demonstrados para atualização, apenas os inseridos que podem realizar o upload aparecem. Vou salvar este arquivo em PDF, dentro da pasta Introdução para exemplo e o arquivo .gitignore é obrigatório ser enviado.

* 1. Comando $ **git status**;



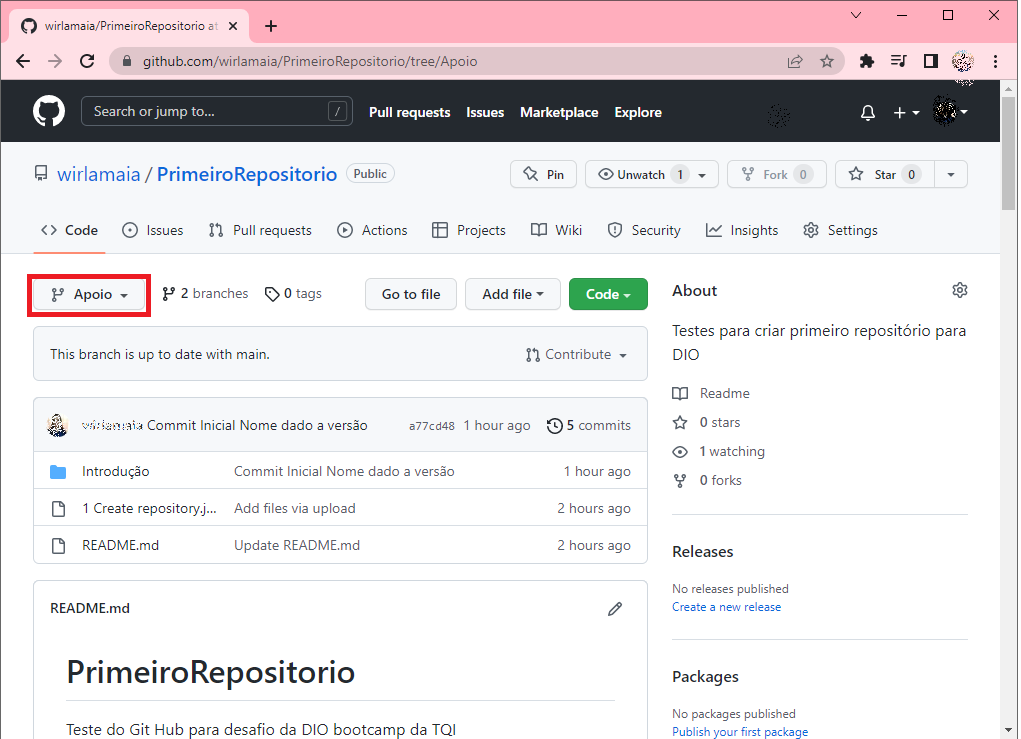
* 1. Comando $ **git branch Apoio**  – Criar uma nova branch (particição de versão apoio a original);

Comando $ **git branch -M** "Apoio" – Alternar o repositório para branch descrita. Perceba que muda a branch na tela escrita em azul.



* 1. Comando $ **git push origin** **Apoio**, Enviar as alterações para a branch Apoio no Git Hub

Essa alteração na branch apoio pode ser levada para a principal Main, pelo comando $ **git merge main** – Porém não irei realizar para ficar arquivos diferentes em cada branch.



* 1. Finalizando Arquivo e repositório.

Comando $ **git status**;

Comando $ **git add .** ;

Comando $ **git commit -m “Commit Segunda versão atualização, branch apoio”**;