**LISTA DE COMANDOS GIT**

cd D:/Git/Curso\_de\_Git

git init (criar repositório)

rm -rf .git (remove um repositório já criado)

git status(status do repositório, arquivos que estão e não estão no controle de versão e etc)

git add "nome do arquivo"(adiciona ao controle de versão(arquivo agora está sendo monitorado))

git add \*"nome da extensão"(adiciona todos os arquivos com essa extensão)

git add --all ou .(adiciona todos os arquivos)

git add . -m "legenda exemplo(Reverter esse commit/Adicionar log)"

code nomedoarquivo.extensao(abre o arquivo no VS Code para edição)

git log(ver versões)

git log -p(ver logs mais detalhados)

git log -n(n é a quantidades de logs a serem mostrados)

git log --oneline(mostra um resumo dos logs)

git log --oneline -n(n é a quantidade de logs a serem mostrados)

git commit -a( git add + git commit)

git rm --cached(remover o arquivo do controle de versão)

git rm (remove tanto do controle de versão tanto quanto do S/O)

git diff(mostra o que mudou no arquivo desde a ultima versão até a atual)

git diff versao x versao y(mostra o que foi alterado entre a versão x e a versão y)

git diff versao x (mostra o que foi alterado entre a versão x até a versão atual)

touch(para criar um arquivo novo em branco)

.gitignore(arquivo que não deve ser versionado)

Dentro de .gitignore:

# Ignorando um único arquivo

imagem1.jpg

# Ignorando arquivos de uma extensão predefinida

\*.png

# Ignorando pastas/diretórios(basta adicionar o nome da pasta e / no final)

vendor/

node\_modules/

docs/

git restore(remove o arquivo não commitado do controle de versão)

git reset(reverte o arquivo commitado[exemplo: git reset --soft HEAD~1{volta 1 commit antes do atual HEAD}])

--soft(move o ponteiro do head para onde definir e não descarta nada da área de preparo e da área de trabalho, mantem os arquivos e o conteúdo dos arquivos)

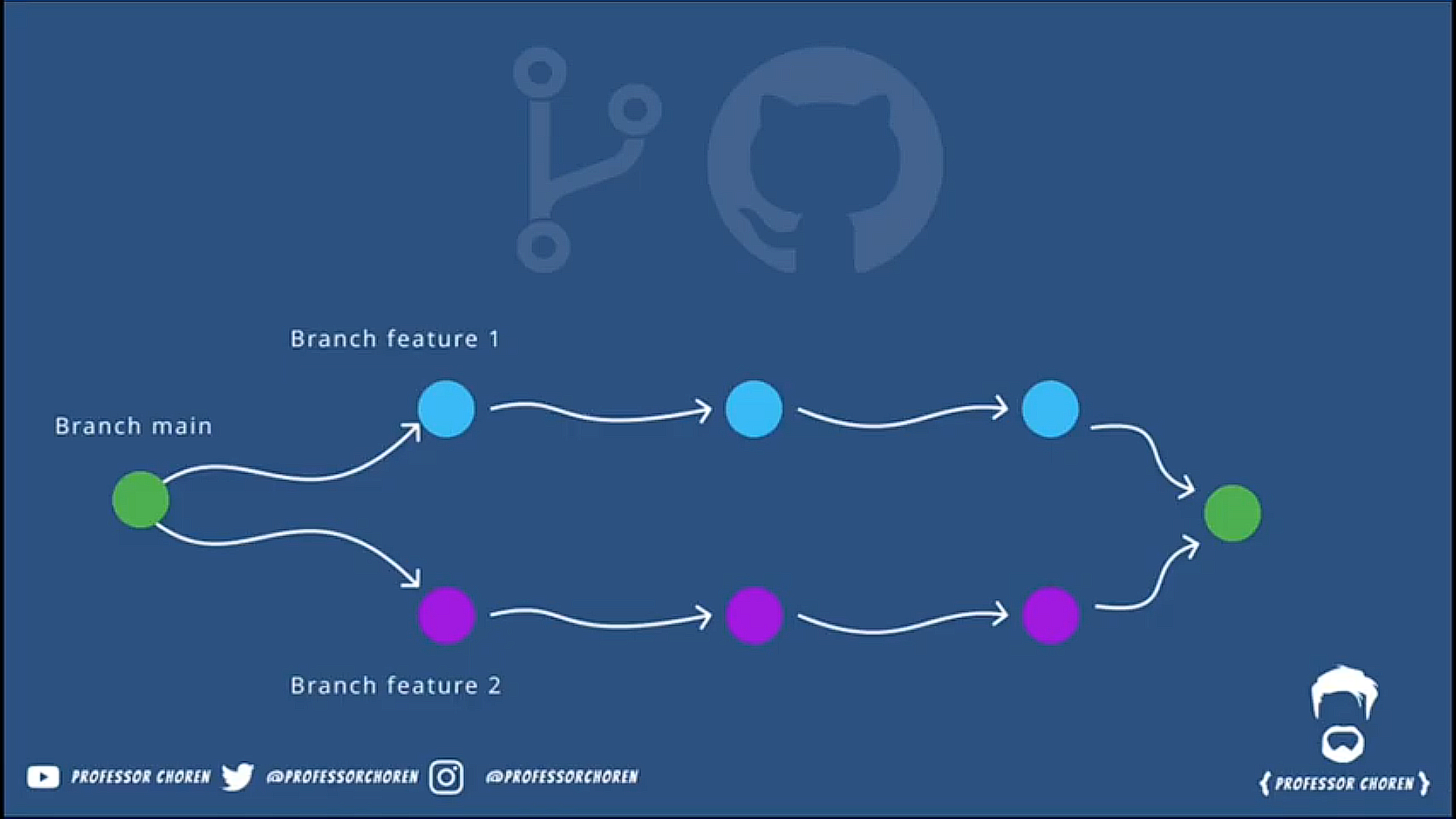
--mixed(além de mover o ponteiro do head para o commit ele faz o espelhamento desse commit para a área de preparo, ele não afeta o diretório de trabalho, ele não desfaz o git add, desfaz o git commit)

--hard(desfaz qualquer trabalho pendente que não tenha sido commitado)

**Branch:**

O que acontece nos ramos secundários não afetam o ramo principal.

Quando as features estiverem prontas para se unirem a branch principal utiliza-se o comando **git checkout**



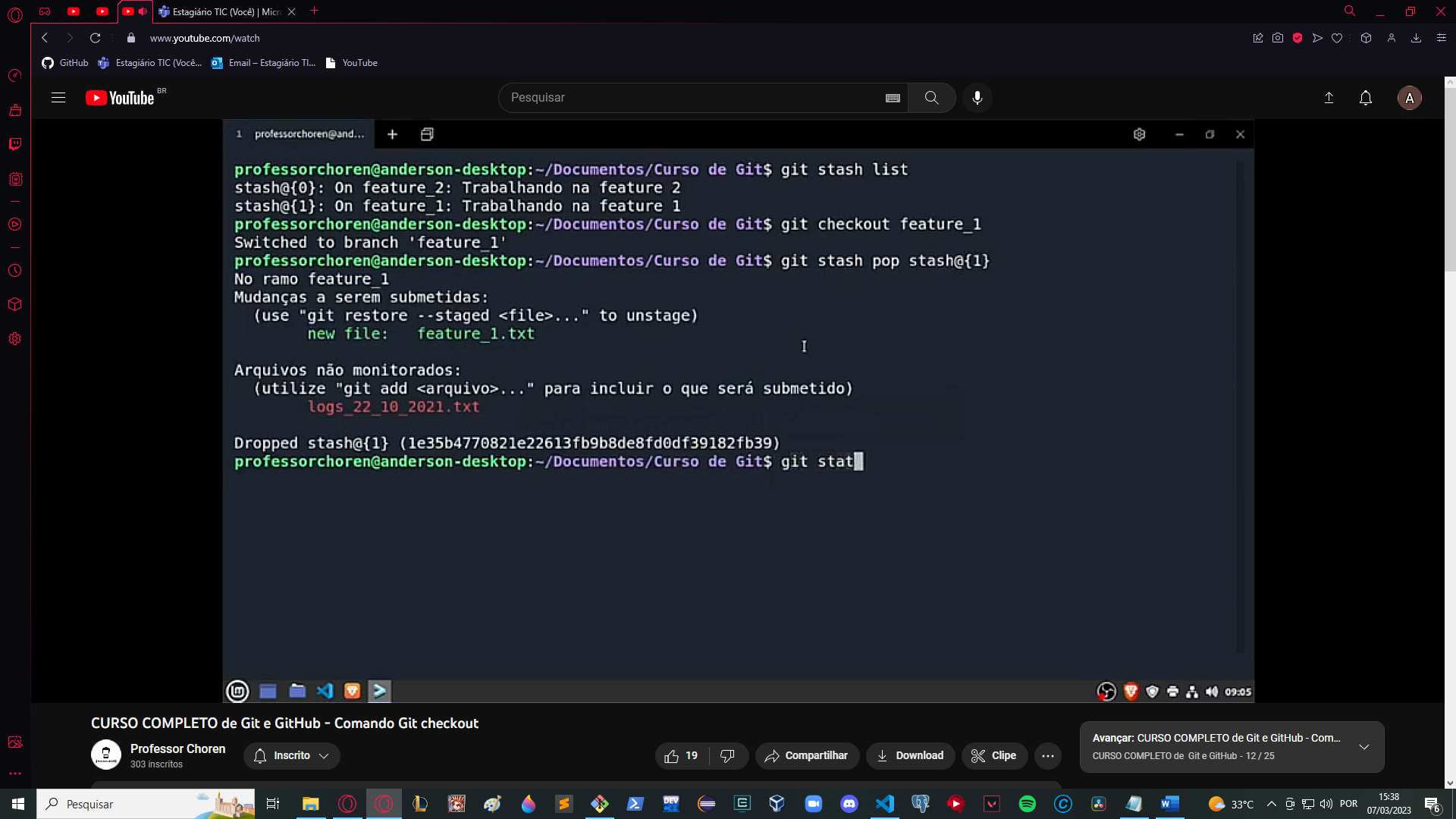
git branch(exibe as branchs existentes)

git checkout ”nome da branch”(swita para outra branch) -b(cria uma nova branch) “nome da nova branch” (ex:.git checkout -b feature\_1))

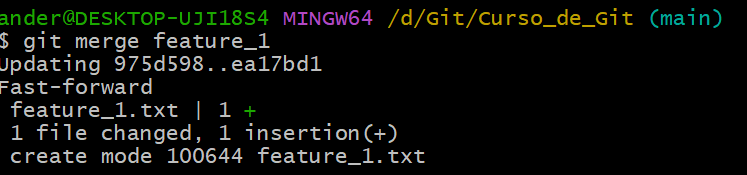
Obs.:se for trocar de branch e ainda estiver trabalhando na feature use **git stath save** “nome do index”(ex.: Trabalhando na feature 1).(Para salvar o arquivo deve estar sendo monitorado)

Para ver as pendencias utilize **git stath list.**

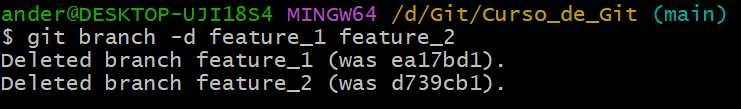
**Desempilhando lista:**



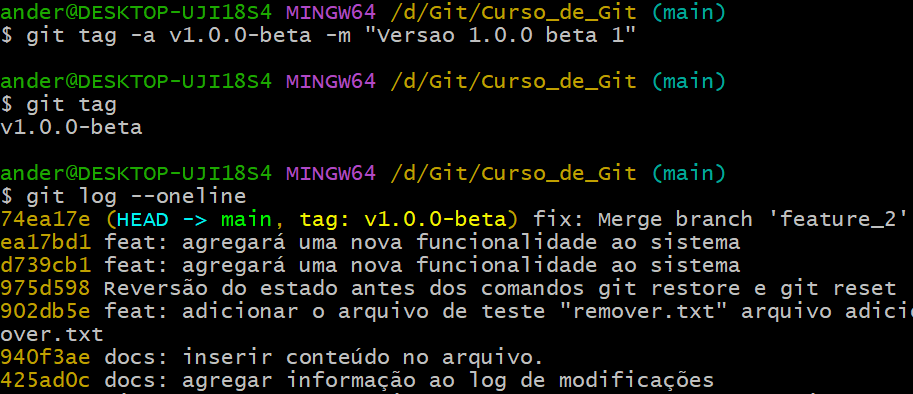
git merge “nome da branch a ser unida”



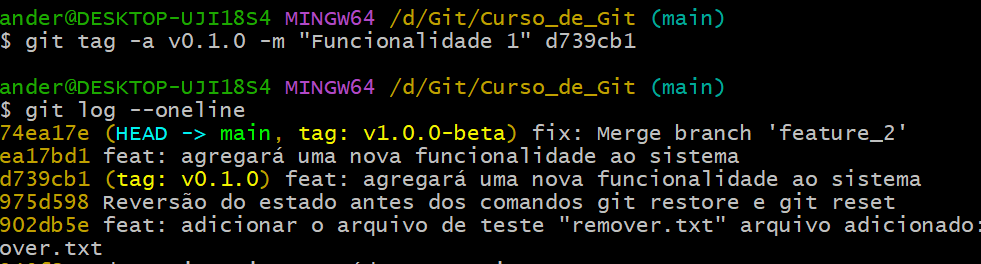
git branch -d “nome da branch a ser deletada”



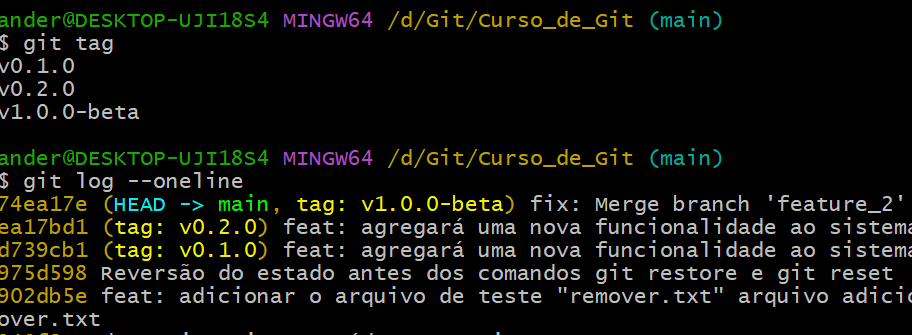
git tag -a v1.0.0-beta(nome da tag) -m “Versao 1.0.0 beta 1”(mensagem da tag)(git tag são formas de definir marcadores dentro do repositório - versão do projeto )

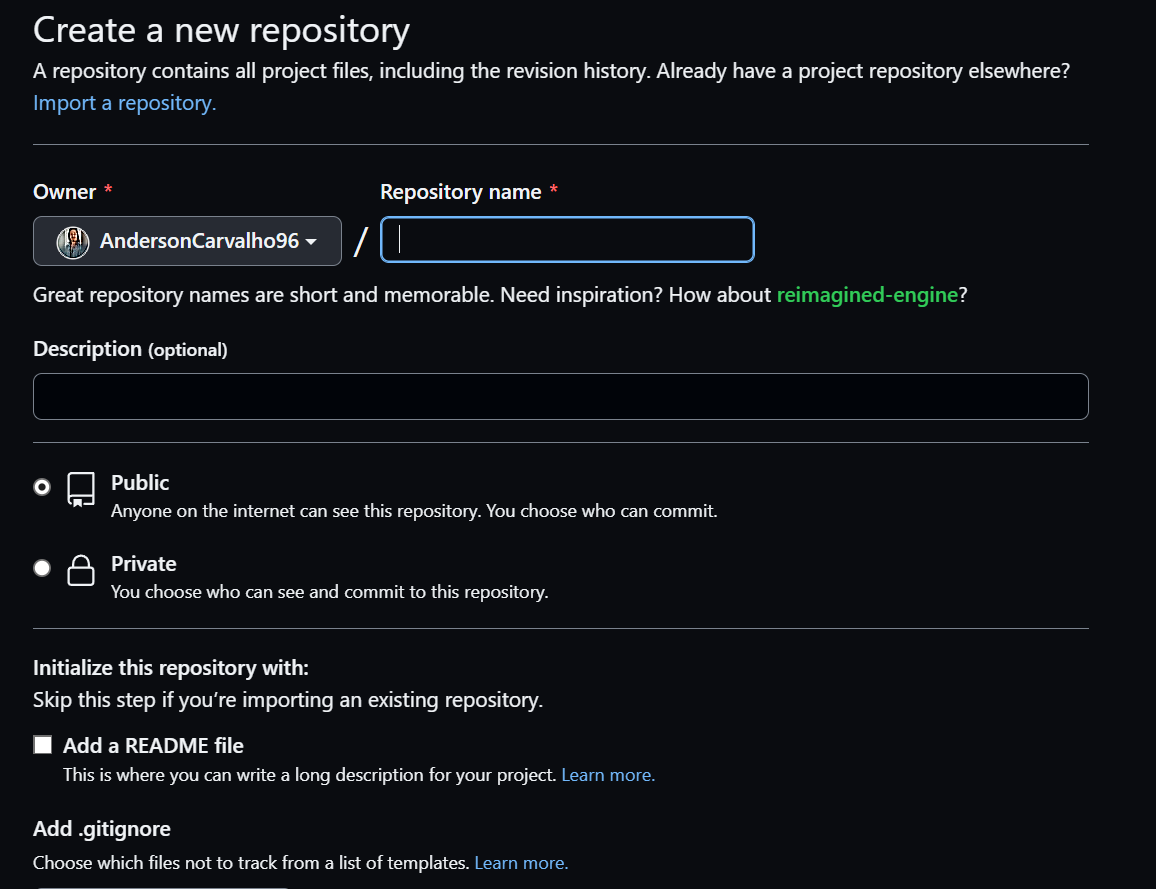


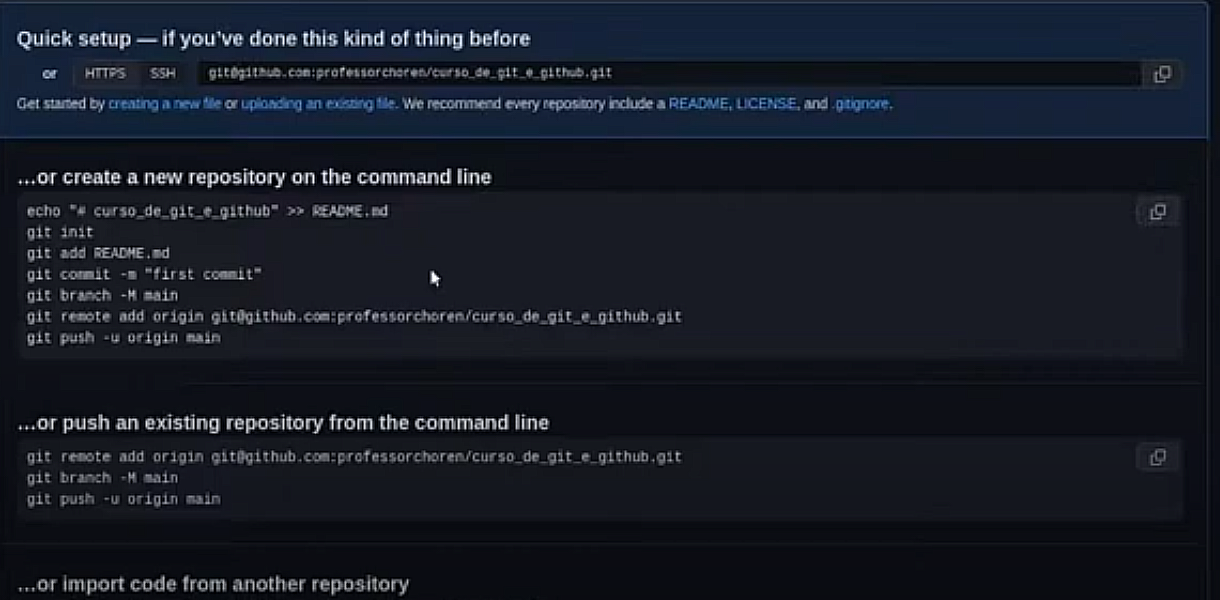
git tag -a v0.1.0 -m “Funcionalidade 1” (código da funcionalidade)



git tag -a v0.2.0 -m “Funcionalidade 2” (código da funcionalidade)



Criando um novo repositorio github e vinculando ao git



A cima temos quatro opções, sendo a primeira o gitclone e

o link do repositório github

Depois as opções:

...ou criar um novo repositório com linha de comando.

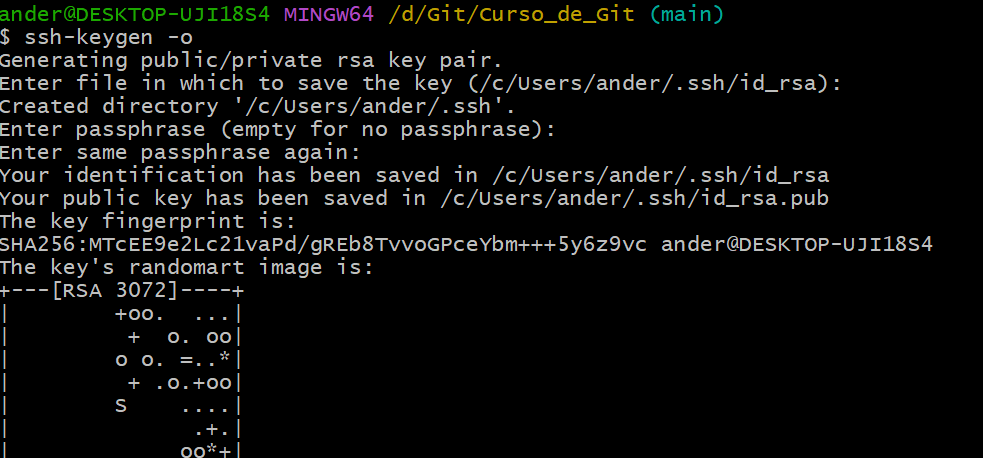
...ou vincular um repositório já existente através de linha de comando.

...ou importar o código de outro repositório.

git remote add “nome do repositório”(origin é um nome padrão) “link do github”

ssh -keygen -o(gerar key ssh)

após isso crie uma senha, e depois será concluído

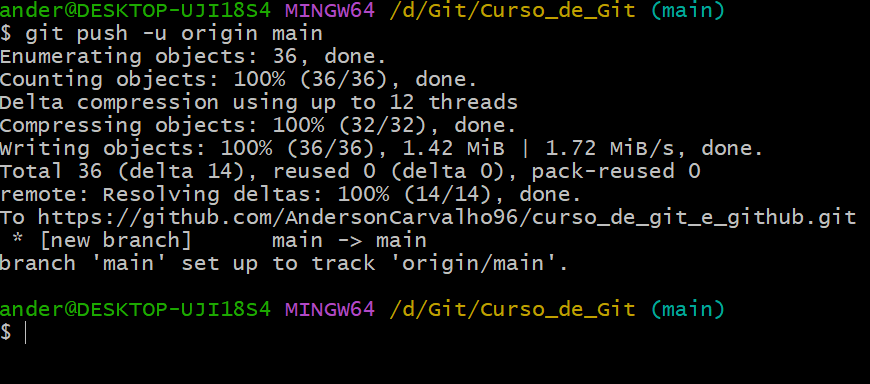


Após navegue até o diretório /c/Users.../.ssh/id\_rsa e copie a chave.  
Vá ao github, conta, configurações(settings) vá a opção SSH and GPG Keys

Crie uma nova chave e cole a chave copiada.

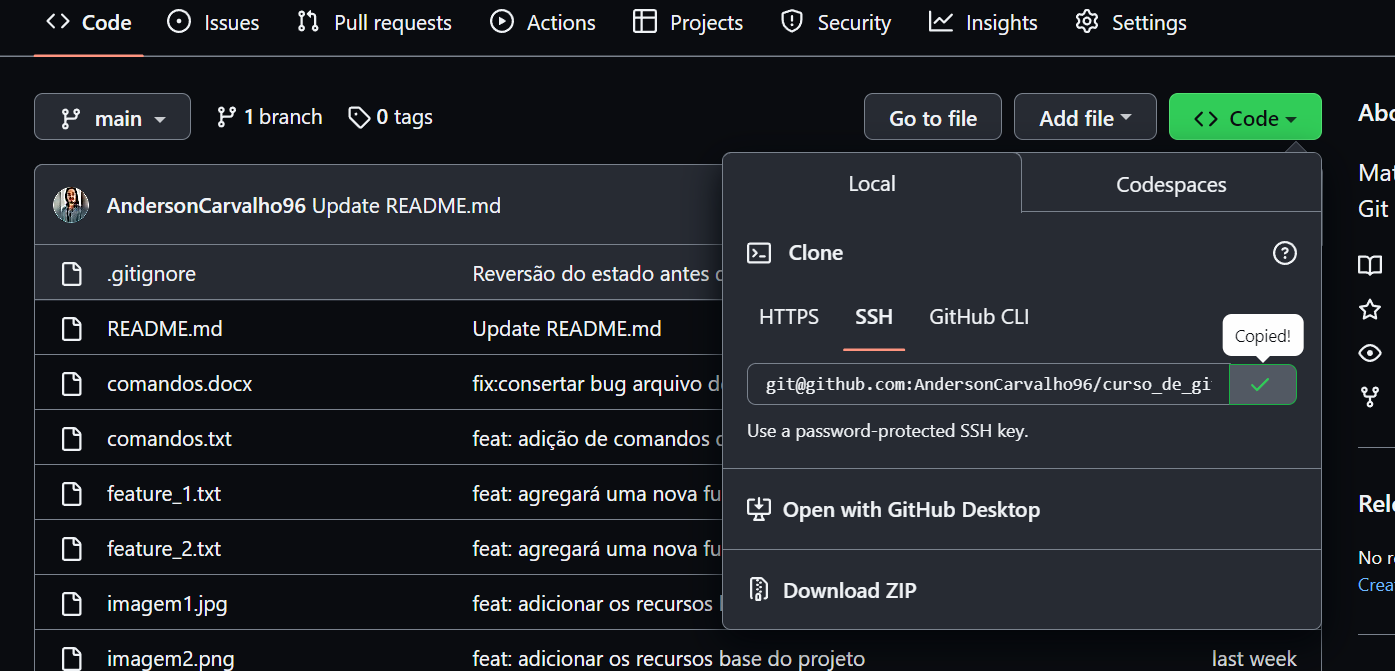
Cada perfil dentro de um computador que queiram utilizar o git terá q criar uma chave nova

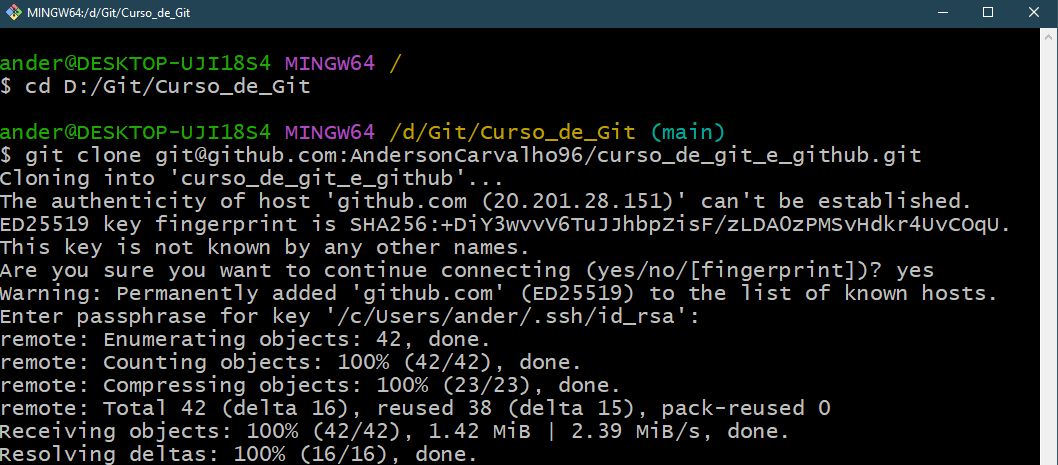
git push -u(na primeira vez use o -u) “nome do repositório online” “nome do repositório local”



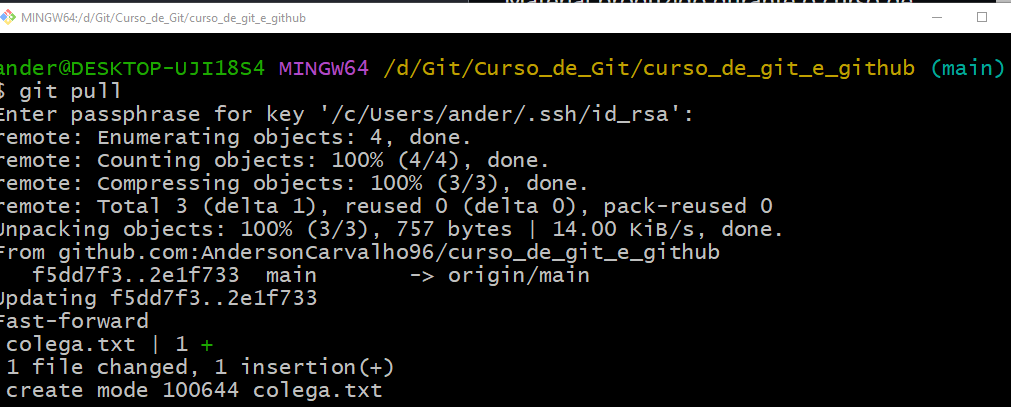
Agora a ‘main’ do repositório local está trackeada a ‘origin/main’ do repositório web, isso quer dizer que no github tem uma main clonando a main local

git clone (puxar do github para maquina local)

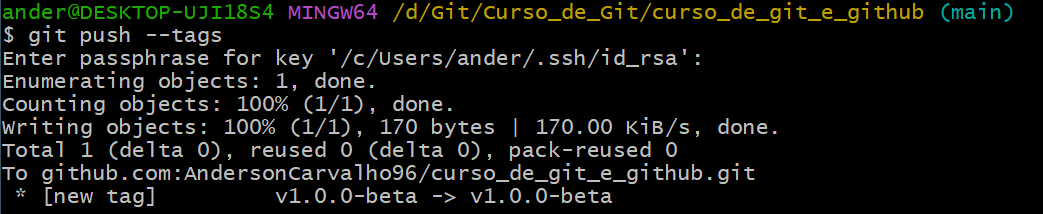




git pull(baixar localmente atualização de algum arquivo do github)



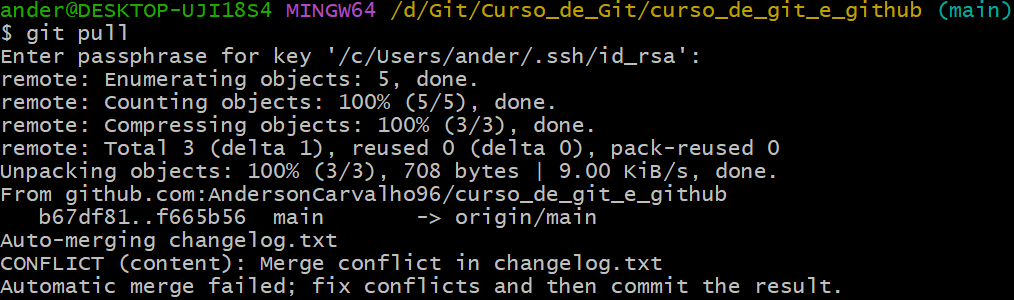
git push --tags(enviar tags para o github)





Gestão de conflitos:

Na imagem abaixo temos um conflito entre a versão local e a versão do github(simulando como se um colega fizesse uma alteração no código e você também fizesse a alteração no mesmo)



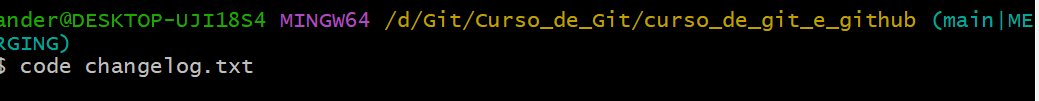
Tentando fazer um merge automático:

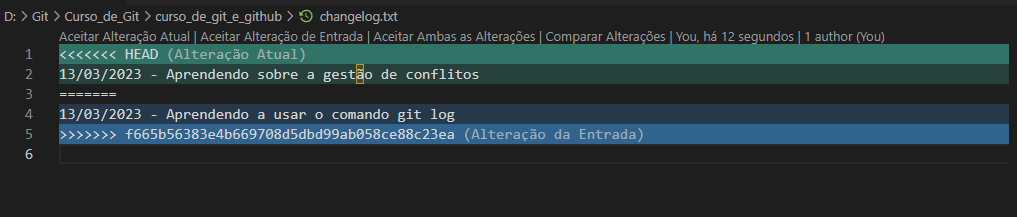


Erro ao fazer o merge automático pois tem divergências de conteúdo:

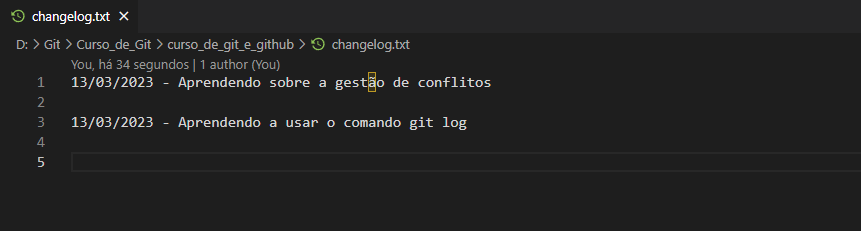


Abrindo changelog.txt

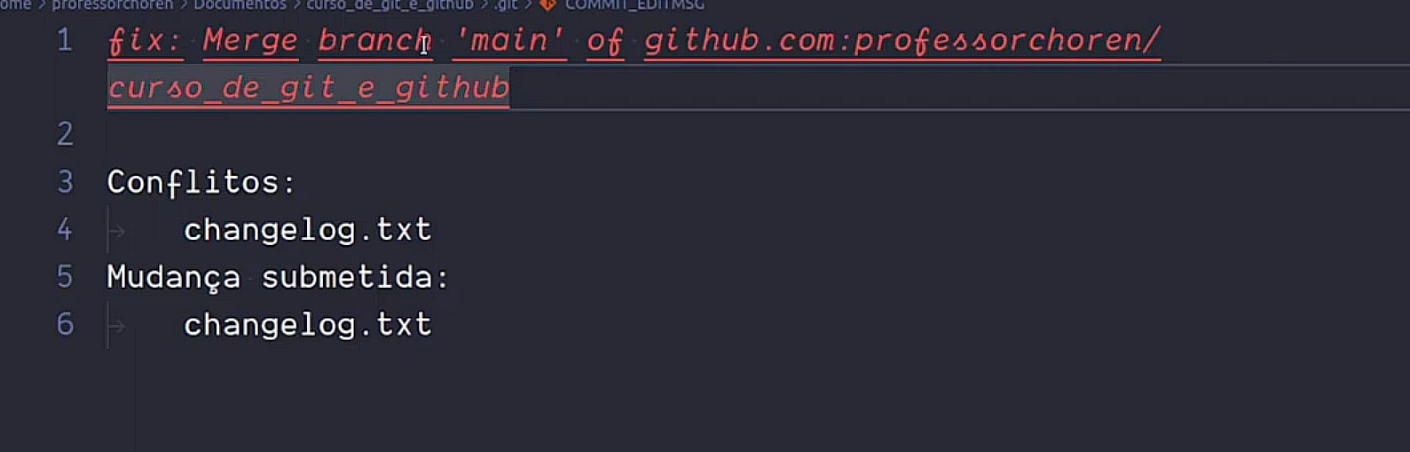


Exibindo divergências( HEAD - repositório local enquanto o outro o web)

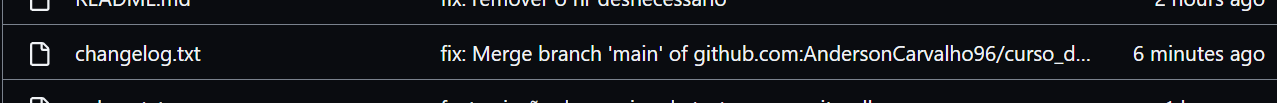
Conserte o arquivo para a forma que será commitada:

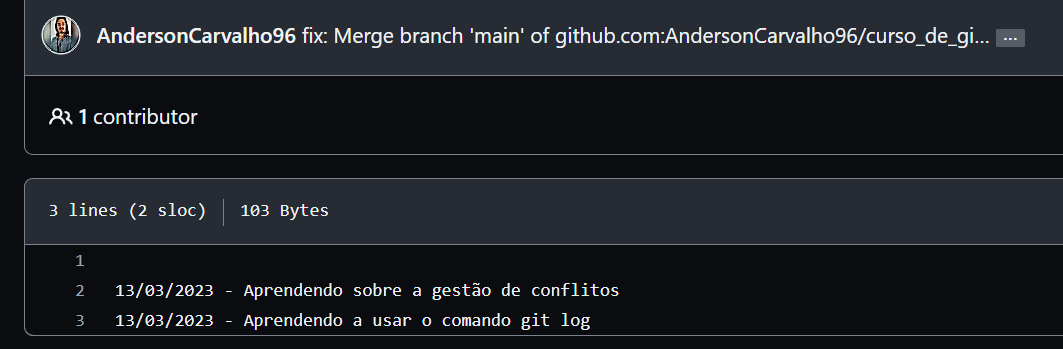


De um commit -a, e ajuste o commit:



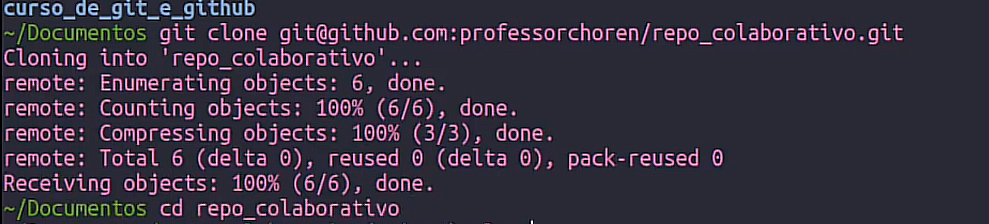
Após de um git push para atualizar o repositório remoto e pronto:



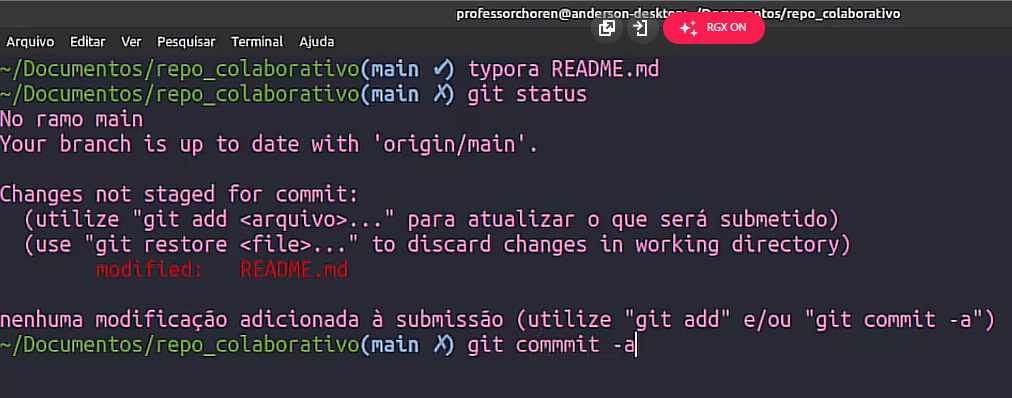


Solicitando um pull request através do fork

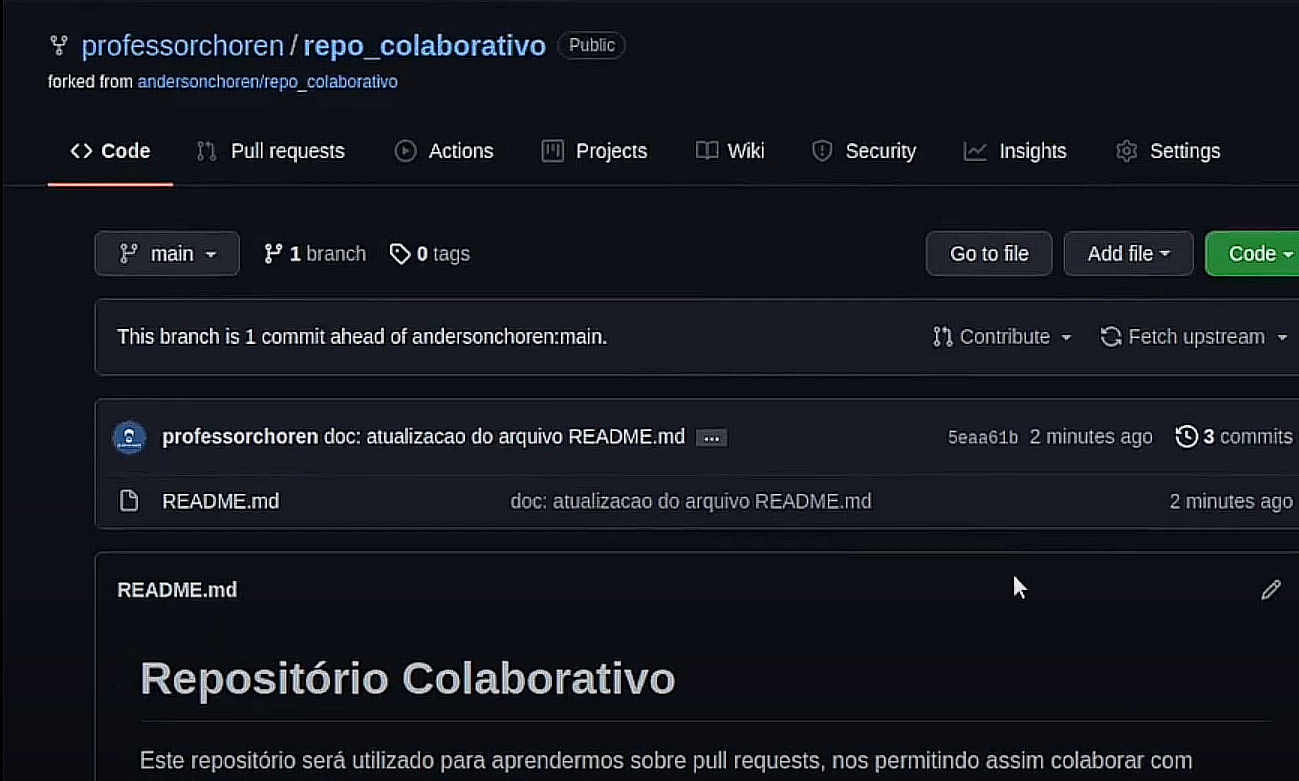
Clonando um repositório usando o link do fork

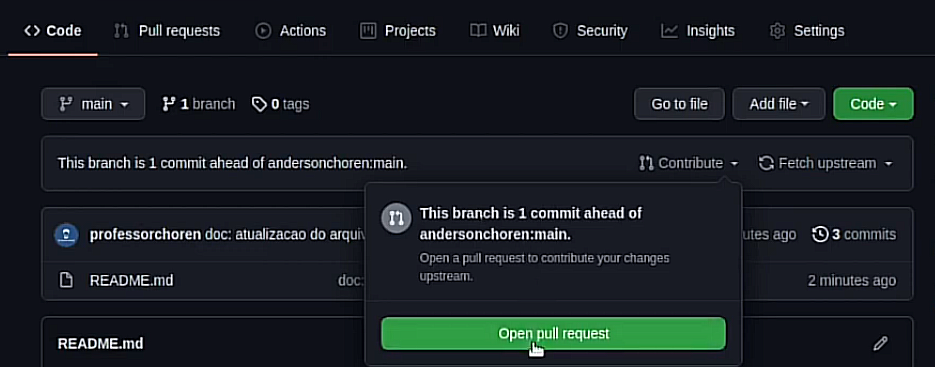


Faça as modificações no arquivo,commite e depois faça um git push

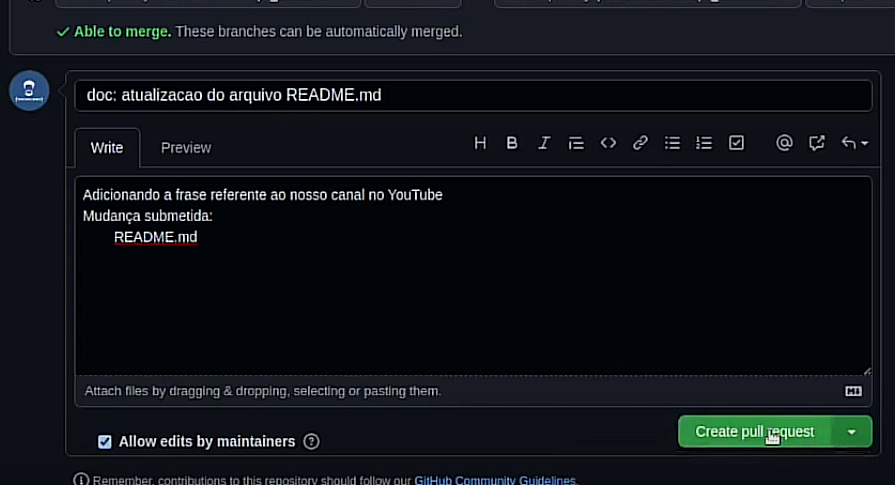


O “meu” repositório agora está 1 commit na frente do repositório original

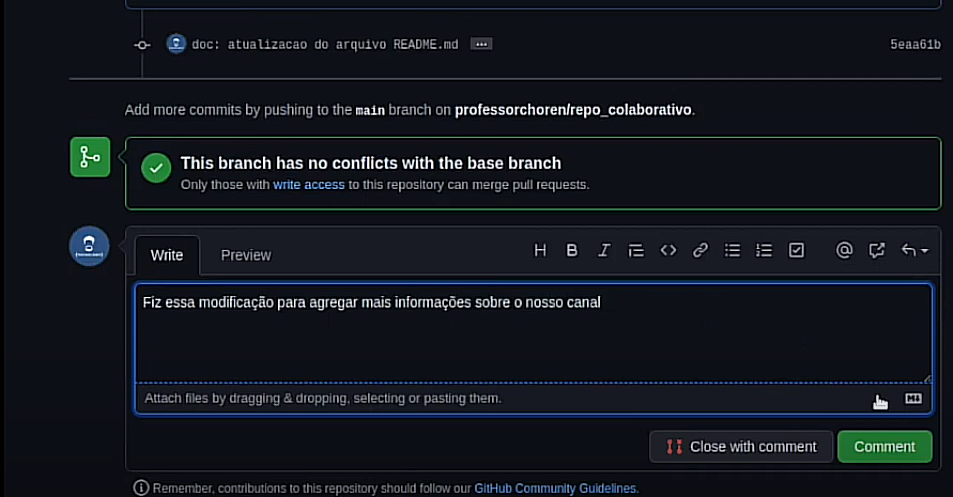


Abra uma pull request

Crie a pull request



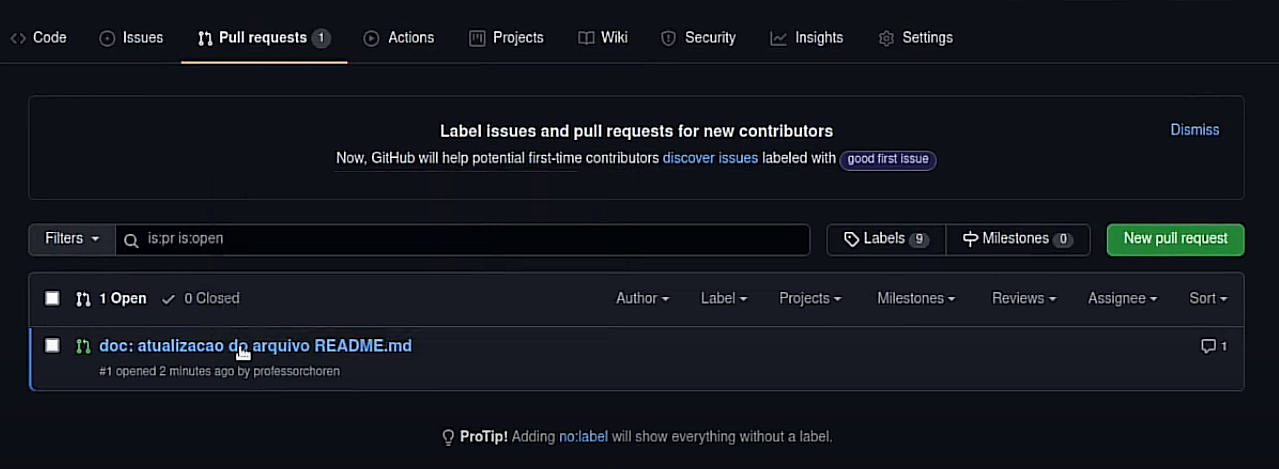
Comente sobre a requisição com o dono do repositório original



Caso queira fechar(cancelar) o pull request clique em “Close pull request” ou “Close with comment” ao lado de Comment

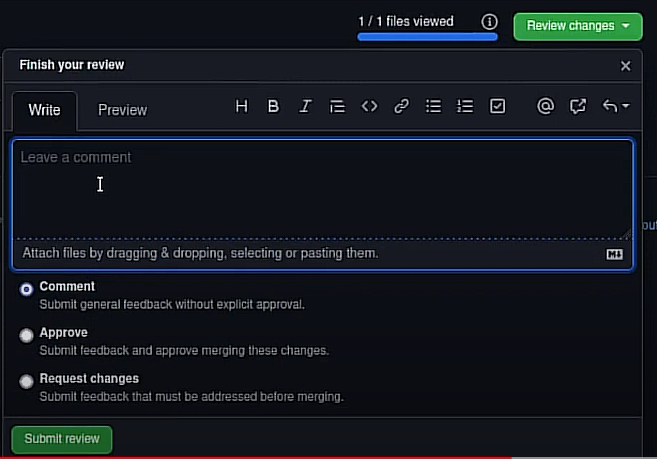
Aceitando pull request:

Na aba “Pull requests” verifique a solicitazção

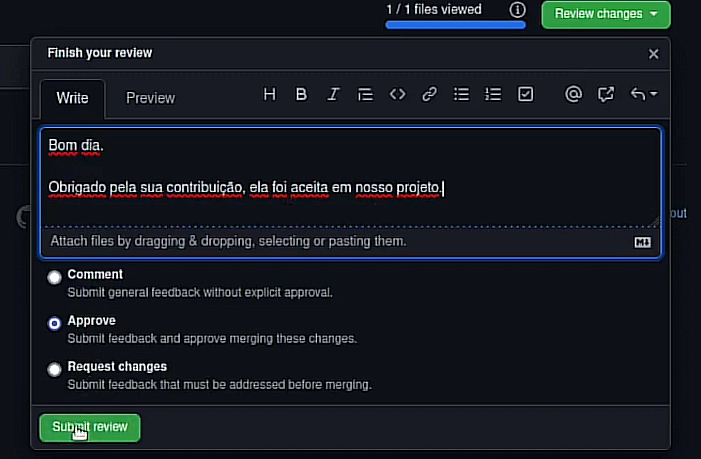


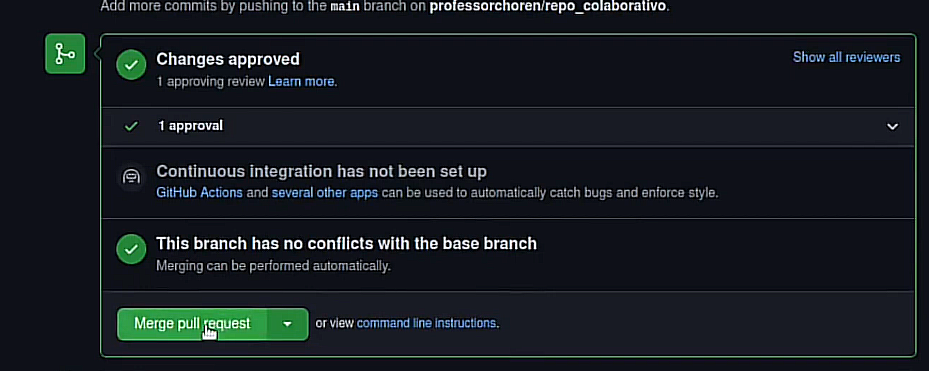
Ao clicar você pode ver a conversa, os commits, checks e arquivos modificados



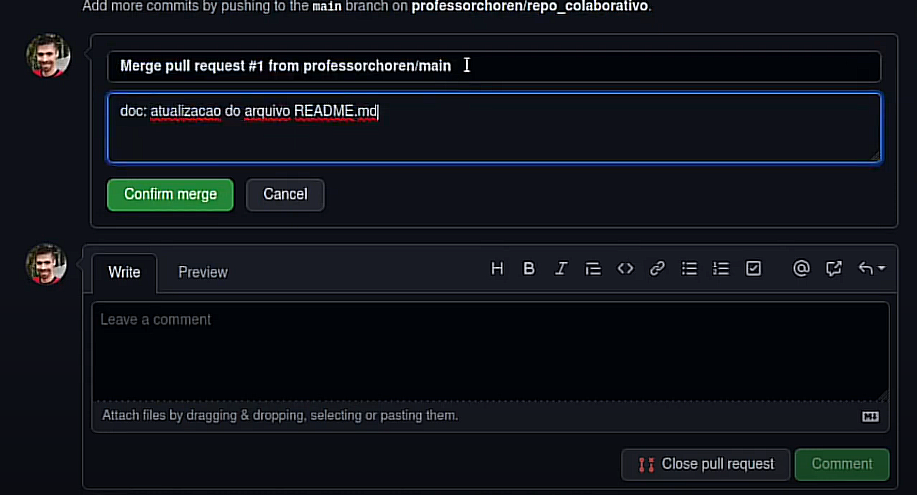
Em Review também é possível comentar por exemplo falando para o solicitante fazer alguma alteração somente como comentario, aprovar caso queria aprovar, ou requisitar alguma modificação como mudança de padrão etc...  


Mudança aprovada

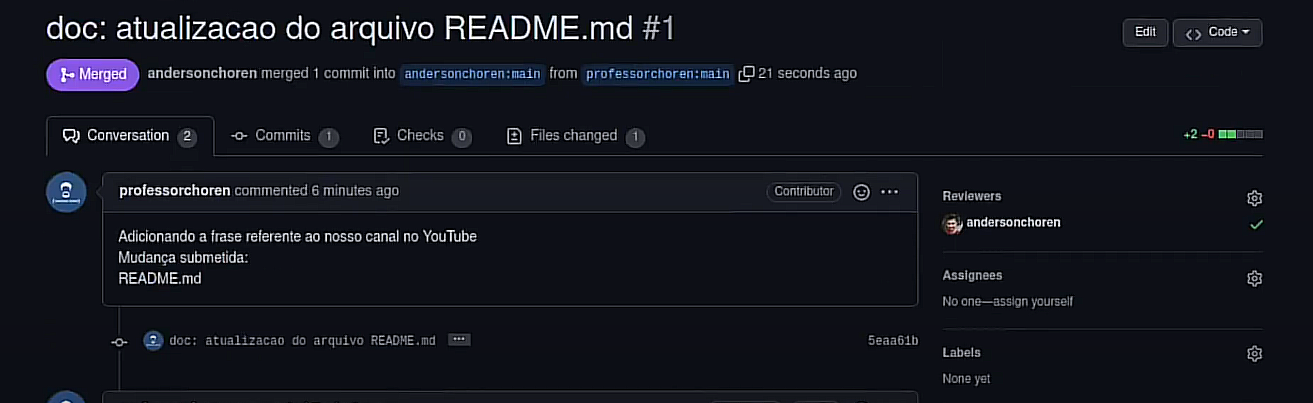


Fazendo o merge da requisição

Fazendo o commit e confirmando o merge



Pronto



**Git Flow**

**Documentação:**

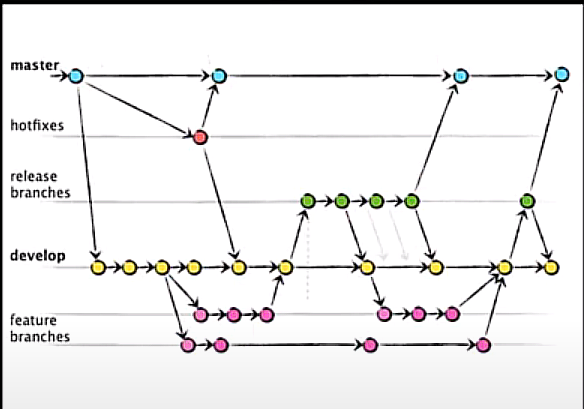
[**https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow**](https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow)

Master – ambiente de produção

Release – ambiente de homologação

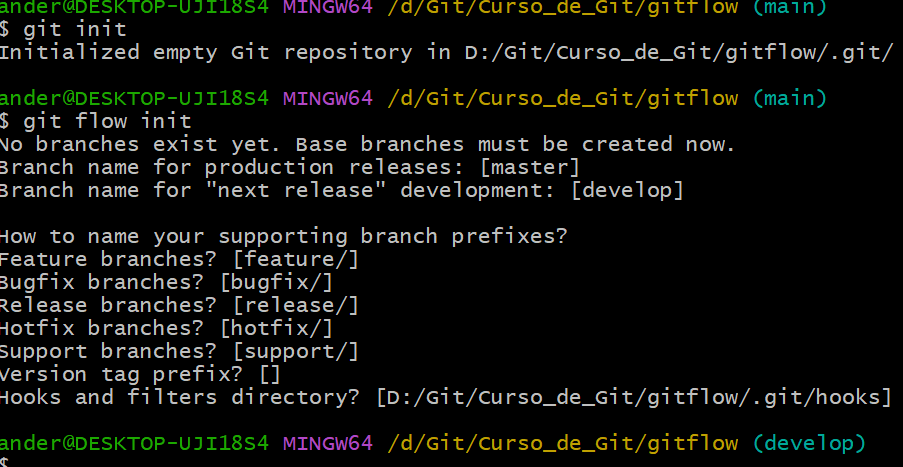
Develop(Dev) – ambiente de desenvolvimento

Feature – ramificações da Develop, quando concluídas são mescladas a Dev

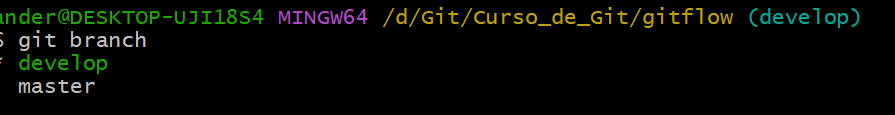


Iniciando git flow:

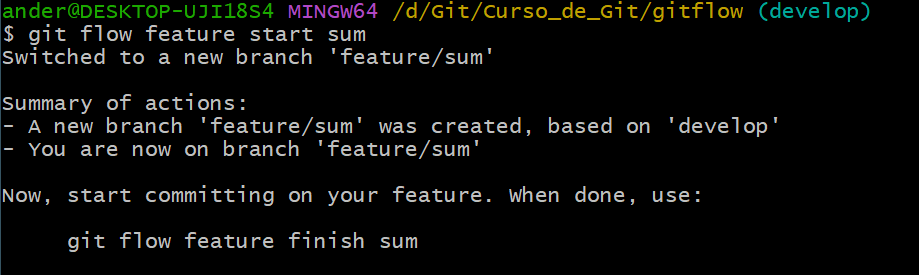
Nomeando branchs(usando os nomes sugeridos)

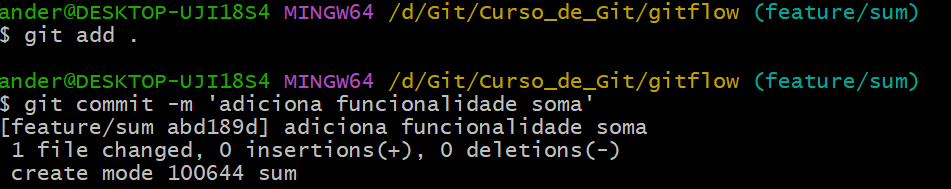
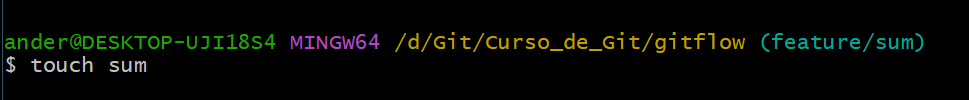


As branchs **develop** e **master** são as únicas branchs fixas, as outras são criadas e excluídas conforme utilizadas e não mais utilizadas.



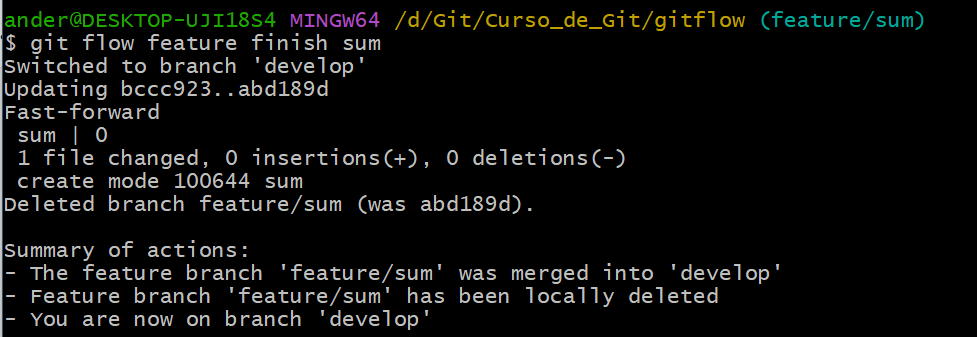
Git flow feature “nome da feature” (criando a feature)



Criando sum e commitando

Terminada as modificações use:

git flow feature finish ‘nome da feature’

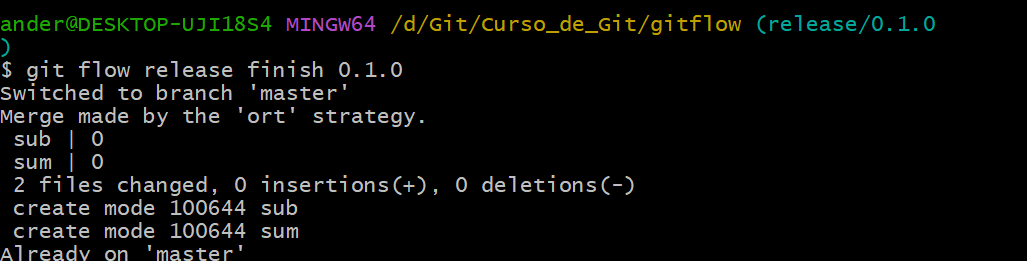


Automaticamente a branch muda para a develop, a feature/sum é mesclada(merged) com a develop e é deletada.

Git flow release start ‘número da versão’(iniciando uma release)



Git flow release finish ‘número da versão’(fechando a release)



**Fonte:**

**Canal Professor Choren:** [**Curso Completo de git e GitHub**](https://www.youtube.com/watch?v=beMnH51P-T4&list=PLmMxPWmzYRGcTabffOwHBORBjtKa2wCXS&index=1)

**Canal Angelo Luz:** [**Git flow na prática**](https://www.youtube.com/watch?v=wzxBR4pOTTs)

**Links Úteis:**

[**Conventional commits**](https://www.conventionalcommits.org/en/v1.0.0/)

[**Git flow atlassian**](https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow)

[**Git flow hotfix**](https://git-flow.readthedocs.io/en/latest/hotfix.html)