* ***Para remover a pasta / diretório apenas do repositório git e não do local:***

**- Etapas para remover o diretório do repositório remoto no gitHub**

git rm -r --cached FolderName

git commit -m "Removed folder from repository"

git push origin master

**(\*) Para remover apenas do repositório local git (criado no pc) retirar o –r**

git rm --cached FolderName

git commit –m “Removido do repositório local”

**- Etapas para ignorar essa pasta nas próximas confirmações**

Para ignorar essa pasta das próximas confirmações, crie um arquivo na raiz chamado **.gitignore** e coloque o nome dessa pasta nela. Você pode colocar quantas quiser

**O** arquivo **.gitignore** ficará assim

/FolderName

**(\*) Não esquecer de dar um git pull para alinhar o repositório git com o local.**

**Para excluir pasta do repositório remoto e do pc ao mesmo tempo:**

git rm -rf Nome\_Pasta

git commit -m "pasta deletada"

git push

**(\*) Cuidado: pasta será deletada do repositório no gitHub e TAMBÉM no computador**

* ***Gerenciando branchs/ramos***

**- Criar branch**

git branch nova-branch

git checkout nova-branch

git checkout –b nova-branch #cria e já acessa o novo ramo nova-branch

**- Deletar branch do repositório remoto (que pode ser um servidor local ou do GitHub)**

git push nome-remote :nome-branch

**(\*) Obs: A branch a ser deletada do remote não pode estar determinada como principal (default)**

**- Deletar branch do repositório local (sua pasta transformado num repositório git)**

git branch –d nome-branch

* ***Copiar arquivos ou pastas de uma branch para outra:***

**(\*) Primeiro: lembrar que branchs armazenam commits (tanto faz se arquivo ou pasta).**

Exº: copiar o arquivo Teste.txt que esta na branch branch\_de\_origem para a branch branch\_de\_destino

**- entre na branch de destino:**

git checkout branch-de-destino

**- dentro da branch de destino faça uma cópia do commit (arquivo/pasta):**

git checkout branch-de-origin arquivo/pasta

git commit –m “arquivo commitado da branch branch-de-origem”

git push repositori-remoto branch-de-destino

## *Como renomear uma branch remota*

Primeiramente, devemos alterar o nome da nossa branch local como explicitado anteriormente:

git branch -m "novo-nome"

**(\*) Atenção: O método abaixo exclui a branch remota. Tenha certeza de que possui a última versão da branch remota em seu repositório local para poder fazer o push dela logo em seguida.**

Obs: Substitua em "nome-antigo" pelo nome da sua branch antes de renomear.

git push origin :nome-antigo

O comando abaixo envia a branch renomeada para o repositório remoto.

git push --set-upstream origin novo-nome

## *Como criar repositório local “espelhando” remoto e incluindo novas branchs*

**- Crie o repositório remoto na pasta/diretório do pc:**

git init

git remote add origin https://caminho\_repositorio\_compartilhado

git pull

**- Crie e acesse a nova branch:**

git checkout -b nova-branch

**- Crie um commit vazio para poder dar push na nova branch:**

git commit --allow-empty

git push origin nova-branch

* ***Criando um repositório puro (bare) para ser nosso servidor (na própria máquina/pc)***

**- Crie e acesse uma pasta que será nosso servidor**

mkdir nome-servidor

cd caminho/NomeServidor

git init --bare

**- No diretório (pasta) que contém nosso repositório (arquivos e pastas) anteriormente criado (através do git init), adicione o repositório servidor nomeando o servidor remoto**

git remote add nome-servidor-remoto caminho/NomeServidor

**Obs: nome-servidor-remoto pode ser local, origin (padrão quando clonamos servidores remotos), ou qualquer coisa**

**(\*) Para editar o endereço/caminho de um servidor remoto**

git remote set-url nome-servidor-remoto novo/caminho/NomeServidor

**(\*) Listando todos os controles remotes em que nosso repositório local esta associado**

git remote –v

Exemplo de saída:

O repositório do exemplo esta associado a 3 controles/servidores remotes: local, local2 e origin.

local C:/ServidorLocal (fetch)

local C:/ServidorLocal (push)

local2 C:/ServidorLocal (fetch)

local2 C:/ServidorLocal (push)

origin https://github.com/LealJP/alura-git.git (fetch)

origin https://github.com/LealJP/alura-git.git (push)

**(\*) Para excluir um controle remote**

Para o exemplo acima poderíamos excluir o local2 que possui mesmo caminho do local: git remote rm local2

git remote rm nome-servidor-remoto

**(\*) Para adicionar arquivos/pastas no servidor remoto**

git push nome-sevidor-remoto nome-branch

Obs: O comando acima criará dentro da pasta do sevidor NomeServidor um arquivo nomeado nome-branch (dentro das subpastas refs e heads)

* ***Clonando (por um outro usuário) nosso servidor remoto***

Imagine que há outro usuário desenvolvendo um projeto em conjunto com você, esse usuário, Ana, por exemplo, precisará de acesso ao seu servidor remoto que constará todas as alterações feitas no projeto tanto por você quanto por Ana.

Ana deve criar um diretório próprio em sua máquina, onde constará tudo o que for desenvolvido por Ana, e deverá criar um repositório git clone do repositório servidor para ter acesso as alterações feitas por você e também pela própria Ana.

**- Na pasta em que estará seu projeto, Ana clona o servidor repositório e cria em sua máquina a pasta servidorClonado onde puxará e enviará todas as alterações para o servidor**

git clone caminho/NomeServidor servidorClonado

git remote –v #a saída será origin pois por padrão o clone cria um controle remote nomeado origin

git rename remote origin novo-nome-remote #renomeando o controle remote

git remote –v #saída novo-nome-remote caminho/NomeServidor (fetch/push)

**- Ana deve acessar a pasta clone e puxar os arquivos do repositório servidor**

cd caminho/servidorClonado

ls #retorna vazio porque ainda não puxou os arquivos e pastas

git pull novo-nome-remoto main #puxa para a pasta clone todos os arquivos/pastas do repositório servidor colocando na branch main (que acabou de criar)