Resumo de tudo que se encontra nos Rd

// // // // // //// O INICIANDO // //// // //// // //

Projeto que ensina a usar o Github

==> cd \_Diretorio\_ (seleciona o diretorio e a pasta)

==> mkdir \_Diretorio\_ (Em caso de conflito com a pasta)

==> git init (inicializa o repositorio vazio)

(Master. Branch principal criada automáticamente)

.git é uma pasta criada automáticamente para suas proprias usualidades.

// // // // // //// // GIT ADD // //// // //// // //// // //

==> git add \_nomeFile\_ (Manda os arquivos para areas de stading ("chama dos fundos para ser empurrado para o palco (commit)")).

==> git add . (adiciona todos os arquivos para stading)

Ex: git add Readme.md (adicionar o arquivo)

==> git status

// // // // // //// NOMECLATURA MASTER PARA MAIN // //// // //// // //

==> git branch -M "main" (Muda para main, apenas questões visuais.)

// // // // // //// GIT COMMIT // //// // //// // //

==> git commit -m "\_nome do Commit\_" (Aqui já criamos o repositorio local e damos o commit para o github)

// // // // // //// INDO AO GITHUB // //// // //// // //

Crie um repositorio com o mesmo nome da pasta

==> git remote add origin \_linkrepositorio\_ (linkado uma única vez)

(remote, coneção / add para adicionar / origin para o nome no github / link)

==> git push -u origin main

// // // // // //// PUSH APÓS PRIMEIRO PUSH // //// // //// // //

==> git push origin main (Não precisa mais do -u)

// // // // // //// // Criando a BRANCH // //// // //// // //// // //

git checkout -b "\_Nome\_" (criamos uma nova branch)

com isso saímos da principal que estavamos (main/master) para a nova branch criada (\_Nome\_)

////// CLONANDO UM REPOSITORIO ///////

Vá no repositorio do git hub

clique no botão verde \_code\_

Copie o html

==> git clone \_linkDoRepositorio\_

////// CLONANDO UM REPOSITORIO ///////

Vá no repositorio do git hub

clique no botão verde \_Code\_\_

Copie o html

==> git clone \_linkDoRepositorio\_

// O comando git clone é usado para baixar um repositório Git completo para o seu computador. Isso cria uma cópia local do repositório, permitindo que você trabalhe nos arquivos, faça alterações e envie suas modificações de volta para o repositório original.

//Começar a trabalhar em um projeto Git: Baixe o código do repositório para o seu computador.

//Criar uma cópia de um repositório: Faça uma cópia independente do repositório para fins de backup ou experimentação.

// Contribuir para um projeto de código aberto: Baixe o código do projeto para poder fazer alterações e enviar pull requests.

////// ATUALIZANDO O CLONE ///////

git pull

////// FORK ///////

No GitHUb ao clicar em um repositorio de terceiro, é possivel clicar em um botão chamado Fork (ao lado de favoritos)

Esse botão clona o repositorio do terceiro para seu repositorio pessoal.

(Caso alguém faça a alteração do Fork, e o terceiro querer adicionar basta usar o : )

==> git pull request

// // // // // //// // AULAS DE GIT BRANCH // //// // //// // //// // //

Conceito: Uma branch é como um "caminho paralelo". Imagine o código como uma estrada principal. Uma branch permite que você crie uma nova estrada a partir da principal, para testar novas ideias ou corrigir erros sem afetar o código original.

/// VANTAGENS ///

// Permite testar e desenvolver novas ideias sem afetar o código principal.

// Facilita o trabalho em equipe, pois cada desenvolvedor pode trabalhar em sua própria branch.

// Ajuda a manter o código organizado e fácil de entender.

// // // // // //// // Criando a BRANCH // //// // //// // //// // //

==> git checkout -b "\_Nome\_" (criamos uma nova branch)

com isso saímos da principal que estavamos (main/master) para a nova branch criada (\_Nome\_)

// // // // // //// // VOLTANDO PARA A BRANCH MAIN // //// // //// // //// // //

==> git checkout main

// // // // // //// // JUNTANDO BRANCHS (MERGE) // //// // //// // //// // //

PRECISA SER NA MAIN

git checkout main

==> git merge \_nomeDaBranch\_ (isso junta a princial (que você já se encontra, com a Branch que você quer da merge.))