Comandos Fundamentais do GIt

Versão do software:

Git –version

Validade se possui algum repositório inicializado na pasta

Git status

Criar um repositório:

Git init

Adicionar um arquivo no repositório:

Git add “nome do arquivo”

Adicionar vários arquivos no repositório:

Git add .

Adicionar comentário em um arquivo adicionado ao repositório:

Git commit “nome do arquivo” -m “Estou enviando somente nome do arquivo”

Adicionar comentário único em todos arquivos que serão adicionados ao repositório:

Git commit -a -m “Estou enviando todos os arquivos”

Enviar arquivos para servidor do Git:

Git Push -u origin master para primeira vez que for enviar

Git Push para demais vezes que for enviar

Receber mudanças que foram feitas no repositório:

Git pull

Clonar o Repositório:

Git clone <https://github.com/lbsilva44/Curso_Github.git>

Remover arquivos do repositório:

Git rm “nome do arquivo”

Verificar as alterações por meio de log

Git log

Altera nome do arquivo

Git mv “nome do arquivo” “pasta/nome arquivo”

inserir dentro de uma pasta já no repositório:

Git mv “nome do arquivo” “pasta/nome arquivo”

Defazer alterações no arquivo:

Git checkout “nome do arquivo”

Ignorando Arquivos no repositório

Primeiramente precisa criar um arquivo .gitignore para incluir quais arquivos você quer que seja ignorado.

Após criação do arquivos, digite o nome do arquivo que será ignorado dentro do gitignore

Resetar mudanças que já foram feitas no branch

Git reset --hard origin/main

O que é uma Branch?

Serve para separar código fonte (master) de uma versão que pode ser alterada, Branch conhecido como uma ramificação de um projejo, separando as versões dos projetos a partir de um branch master. Quando o projeto é iniciado e criado a branch master. Após a finalização das alterações feita em cada branch, são unidos para ter um código-fonte final.

Visualizar os branchs disponíveis

Git branch

Para criar um branch

Git branch “nome”

Toda vez que for criada uma branch nova, tem que ser necessário ser criada da master pois criar uma segunda branch da branch primaria não e o correto pelas pratica de um dev.

Deletando Branch

Git branch -d “nome da branch” ou –delete

Não e comum deletar um branch, normalmente é guardado o histórico do projeto, geralmente usa o delete quando o branch foi criado errado.

Como mudar de branch

Git checkout -b “nome”

Alterando o branch podemos levar alterações que não foram commitadas junto, TOME CUIDADO!

Como unir as branches

Git merge “nome da branch”

Salva modificações atuais para seguir com uma outra abordagem de solução e não perde o código.

Git stash

Listando código que foi excluído com stash

Git stash list

Recuperando o código com stash

Git stash apply “numero da stash”

Verificar as alteração da stash

Git stash show -p “numero da stash”

Removendo todas stashs

Git stash clear

Removendo stash selecionada

Git stash drop “nome da stash”

Utitlizando Tags

Criar tag

Git tag -a “nome da tag” -m “msg da tag”

A tag e diferente do stash, serve como um checkpoint de um branch. E utilizada para demarcar estágios do desenvolvimento de algum recurso.

Listasr Tags criada

Git tag

Verificar a tag

Git show “nome da tag”

Trocar de tag

Git checkout “nome da tag”

Desta maneira podemos retroceder ou avanacar em checkpoint de uma branch

Compartilhas a Tags

Git push origin –tags

Compartilhando uma tag em especifico

Git push origin “nome da tag”