**Identificando-se através do GIT BASH:**

git config --global user.name "Nathan Cirillo"

git config --global user.email "cirillo.nathan@gmail.com"

**Criando um repositório:**

Se estiver dentro da pasta: git init

Se a pasta não existir: git init <nome-repositório>

**Verificar se os arquivos estão sendo rastreados:**

git status

**Passando a rastrear novos arquivos/alterações:**

**Sobre:** Ao rastrear um arquivo, ele é adicionado a uma área especial chamada Stage. Toda mudança também deve ir para essa área. Sempre antes do commit, faça o rastreamento.

**Rastreia todos de uma vez:** git add .

**Permite rastrear arquivos específicos:** git add <nome-arquivo>

**Gravando (COMMIT) alterações no repositório:**

git commit -m “mensagem explicativa do commit”

**Checando os COMMITs de um repositório:**

Todos os commits: git log

Últimos N commits: git log -n 2

Resumo dos commits: git log –oneline

Resumo com nº de linhas adicionadas ou removidas: git log –stat

Combinando tudo: git log -n 2 –oneline –stat

**Apontando o repositório local para o GITHUB:**

git remote add origin <url-repositorio>

**Enviando os dados do repositório para o GITHUB:**

git push origin <nome-branch-a-ser-enviada>

**Clonando um repositório localmente:**

git clone <url-repositorio-a-ser-clonado>

**Ignorando arquivo no GIT (não versionar):**

Criar um arquivo chamado **.gitignore** na pasta do projeto com as *extensões, arquivos e pastas* que não queremos controlar.

**Rastrear (git add) e comitar (git commit) ao mesmo tempo:**

Em arquivos já existentes e que foram modificados é possível executar ambos os comandos de uma única vez. Se for um arquivo novo é necessário fazer separadamente.

git commit -a -m “mensagem do commit”

git commit -am “mensagem do commit”

**Desfazendo modificações NÃO RASTREADAS:**

Em vários arquivos: git restore .

Em um arquivo específico: git restore <nome-arquivo>

Também é possível usar: git checkout --<nome-do-arquivo>

**Tirando arquivos e modificações de STAGE:**

Em vários arquivos: git restore –staged .

Em um arquivo específico: git restore –staged <nome-arquivo>

Também é possível usar: git reset <nome-arquivo>

**Desfazendo todas as modificações de uma vez:**

Esse comando faz com que o repositório volte ao estado do último COMMIT, ou seja, reseta todos os dados modificados.

git reset --hard

Diferença entre dois COMMITs específicos:

1. Obtenha o identificador do commit desejado: git log -n 2 –oneline
2. Compare-os com o comando: git diff 58ea898..deb063f

Excluir um arquivo já gravado (comitado):

git rm <nome-do-arquivo>

Excluir um arquivo em (stage) use:

git rm <nome-do-arquivo> -f

Recuperar um arquivo após a exclusão:

Obs: sempre que quiser referenciar ao último COMMIT use HEAD

git checkout <id-commit-que-tem-o-arquivo> <nome arquivo>

Renomear um arquivo já gravado:

git mv nome\_antigo.txt nome\_novo.txt

Movendo um arquivo para outro local do repositório:

Obs: no exemplo o diretório deve estar criado

git mv arquivo.txt exemplo/arquivo.txt

Desfazendo (revertendo) mudanças já comitadas:

git revert --no-edit <nº commit>

Obs: se for reverter para o último commit podemos usar no lugar do nº a palavra HEAD.

Atenção: se passar um nº de commit antigo somente as alterações naquele commit serão desfeitas (algo poderoso para correções pontuais).