Após instalar, fazer configurações de usuário e e-mail vejamos alguns comandos.

$ git status (verificar o status atual do repositório)

$ git add Readme.md (adiciona arquivo para ser "rastreado")

$ git commit -m "comentário" (envia arquivo ao repositório -m é de mensagem)

$ git commit -am "comentário" (tendo sido add anteriormente, este adiciona e faz commit de itens que estão em unmodified e modified)

**$ git log (apresenta informações hash, autor, data, msg)**

$ git log --decorate

$ git log --author="Paulo" (inf commit do autor)

$ git shortlog (resumo de commits ordem alfabetica autor)

$ git log --graph (formato grafos, quando há outras branch)

**Obs: com a hash identificamos o que foi feito no commit.**

$ git show + hash (mostra o que foi feito APÓS commit)

**$ git diff (apresenta mudanças ANTES de fazer commit)**

$ git diff --name-only (diz somente nome do arquivo modificado)

$ git checkout + nome.extensão (retorna para antes da edição, checkout usa antes de add e commit)

$ git reset HEAD nome.extensão (Arquivo foi add, porém está no staged, ele voltará para modified podendo usar o git checkout)

$ git reset ALTERA histórico do commit, evite git HARD depois que estiver no repositório remoto sendo compartilhado com outros.

$ git reset --soft + hash anterior (mata o commit e volta para staged)

$ git reset --mixed + hash anterior (mata o commit e volta para ANTES staged, ou seja, modified)

$ git reset --hard + hash anterior (ignora a existência do commit, mata tudo que foi feito neste commit)

$ git revert + hash (reverte o estado do commit, mantem histórico, diferentemente do reset)

**$ BRANCH (ponteiro móvel segue o commit, modifica sem alterar o local principal, evita conflitos)**

$ git checkout -b nomebranch (cria um novo branch e já entra nele)

$ git branch (mostra os branch existente)

$ git checkout -D nomebranch (remove a branch)

**$ git merge nomebranch (unir branch criada com a master/main )**

**$ git rebase nomebranch (unir branch rebasetest com master/main)**

**$ touch .gitignore (arquivo que escrevemos os padrões que não serão rastreados)**

**$ git stash (guarda modificações NÃO comitadas)**

$ git stash apply (vai aplicar as modificações)

$ git stash clear (limpa, ou seja, retira todas modificações colocada no stash)

$ git config --global alias.s status (alias = atalho, após o ponto, atalho para o comando, neste o S atalho para status)

$ git tag -a 1.0.0 -m "Readme finalizado" (criando uma nova tag)

$ git push origin master --tags (subir para repositório remoto)

$ git tag -d 1.0.0 (deleta no repositório local a tag)

AGORA INTEGRANDO GIT & GITHUB

$ git clone + chave SSH (permite clonar um repositório existente no repositório remoto)

$ FORK (faz cópia de projeto que não é seu, para o seu repositório remoto, fork é feito em projetos de outros, clone é nos meus)

$ git push origin main/master (origin é onde estamos enviando, no caso repositório remoto é o origin. Main/master é branch onde estamos)

$ git push origin: 1.0.0 (deleta do repositório remoto a tag)

$ git push origin: nomebranch (deleta do repositório remoto a branch)

$ git merge nomebranch (unir branch criada com a master/main )

$ git rebase nomebranch (unir branch rebasetest com master/main)