VS Code

"terminal.integrated.profiles.windows": {"GitBash": {"path": "A:\\Program Files\\Git\\\bin\\bash.exe"}, },

"terminal.integrated.defaultProfile.windows": "GitBash", (Fazer GitBash Aparecer no Terminal do

VsCode )

VS Code

sfdx force:source:retrieve -p force-app (retrieve de td)

sfdx force:apex:log:list (Show all logs)

sfdx force:apex:log:get --logid <id> (get specific log)

Git-bash

-----FUNCTIONAL-----

$ Ctrl + l ou clear (Sobe a tela)

$ ls ou ls -l (Mostra f/d do path atual)

-----BASIC-----

$ git init  (Inicializa git repo)

$ git add .  (Stage changes)

$ git commit -m <”commit”>(Commit changes)

$ git push origin main (Sobe para GitHub)

-----BASIC ENVIRONMENT-----

$ git config --global user.name <”username”> (Configura usuário)

$ git config --global user.email <”email”> (Configura email)

$ git config --global user.password <”password”> (Configura senha)

-----BRANCH-----

$ git checkout -b <”dev”> (Cria branch local – cópia main)

$ git rm -rf . (I - Apaga tds arq da branch nova criada)

$ git clean -fdx (II - Apaga tds arq da branch nova criada)

$ git branch (Show all local branchs)

$ git branch -d <”name”> (Deleta branch local)

$ git checkout <”name”> (Muda de branch)

-----MERGE-----

$ git merge dev (Estando na master eu mergiei a dev na master)

$ Nano <“Filename”> (Fix conflicts)

$ git commit (Conclude merge)

$ git merge --abort (Abort the merge)

-----MANAGEMENT-----

$ git log (--oneline) (--all) (Show all commits)

-----UNDO LOCALLY -----

$ git reset --hard (<"HEAD”>) (Aponta p commit / Traz tudo do commit atualizando se tiver alguma mudança nos arquivos existentes / Unstage tudo quando commited ou staged, mas modifica estrutura de pastas e transforma em arquivos / Apaga tudo quando eles já são arquivos somente em staged)

$ git reset (<"HEAD”> )  (Aponta p commit / Traz tudo do commit não atualiza se tiver alguma mudança nos arquivos existentes / Unstage tudo independente do estado das mudanças, mas não modifica estrutura de pastas)

$ git reset --soft (<"HEAD”>) (Aponta p commit / Traz tudo do commit não atualiza se tiver alguma mudança nos arquivos existentes / Stage tudo independente do estado das mudanças, mas não modifica estrutura de pastas)

& git restore . --staged (Undo all staged changes to unstaged changes, no structure modifications)

$ git clean -fd (Remove all unstaged files and folders / Remove all empty staged folders / Remove all empty commited folders)

$ git clean -f <> (Remove spec unstaged file)

$ git clean -f <filename> (Remove all unstaged files)

-----UNDO GLOBALLY -----

$ git revert <HEAD> (-–no-edit)

(Undo locally pushed commit)

$ git push origin branch (Apply to GitHub)

-----STASH-----

$ git stash (<-m "Name stash”>)

(Create a stashed files and folders from staged changes)

$ git stash list (Show all stashed files and folders)

$ git stash pop <“name”> or <number> (Applies the changes and removes the files from the stash)

$ git stash apply <“name”> or <number>  (Applies the changes and leaves a copy in the stash)

-----QUESTIONS-----

$ git remote add origin <REMOTE\_URL>

$ git remote -v

$ git config --global credential.helper store

(configurar token (?))

-----Git LFS – Pre-Config-----

$ git config --global http.version HTTP/1.1

$git config --global http.postBuffer 157286400

-----Git LFS - Basic-----

$ git lfs install

$ git lfs track “\*.<extension>” (Adiciona as extensoes que precisam ser carregadas)

$ git add .attributes (Adiciona Arquivo para conseguir utilizer o LFS)

$ git add . (Normal)

$ git commit -m “msg” (Normal)

$ git push origin main (Normal)

PYTHON

Windows key + R 🡪 sysdm.cpl 🡪 (PATH SCRIPTS)C:\...\Scripts

pip install nome-de-algum-pacote

pip uninstall nome-de-algum-pacote

pip install -r requirements.txt

pip${version} install nome-de-algum-pacote

CMD

$ Mkdir <name> (create a directory)

$ Echo <filename> (não sei)

$ touch <filename> (Create a file)

$ rm <filename> (Remove a file)

$ rm -d <directory> (Remove a directory ONLY empty)

$ rm -r <directory> (Remove a directory ALL)

$ Cat <filename> (Print file)

$ Nano <filename> (Edit a file)

$ Vim <filename> + insert (Edit a file)

(Cont) esc -> :wq Ou (Salvar e sair)

(Cont) :q! (Sair)

$ cls (Sobe a tela)

$ Dir (Show files and folders)

$Code . (Open spec folder in VS Code)

$ cd /d <Disk letter> + :(Mudar de disco)

$ tasklist | findstr "Teams.exe"

$ taskkill /IM Process Name /IM Process Name /F

$ taskkill /PID PID  /PID PID /F