**EJEMPLO DE USO DE GIT**

**CLONAR REPOSITORIO**

1. ABRIR GIT BASH EN EL EXPLORADOR DONDE SE VA A CLONAR EL REPOSITORIO (C:\Users\Usuario\Documents\Visual Studio 2019\GIT\InventariosPJ)
2. CLONAR REPOSITORIO

$ git clone https://github.com/MirnaGonzalez/InventariosPJ.git

$ cd InventariosPJ

$ git status

$ git checkout Develop

$ git checkout –b DevelopMirna (Crear rama local)

1. GUARDAR CAMBIOS EN REPOSITORIO LOCAL

$ git add .

$ git commit –m “....”

1. PARA SUBIR CAMBIOS A DEVELOP

VERIFICAR SI HAY CAMBIOS EN DEVELOP

$ git checkout Develop

$ git pull

TRAER CAMBIOS DE DEVELOP A MI RAMA LOCAL

$ git checkout DevelopMirna

$ git merge Develop

shift: wq

1. SUBIR CAMBIOS DE LA RAMA LOCAL A LA RAMA DEVELOP

$ git checkout Develop

$ git merge DevelopMirna

$ git push

Nota: El push siempre es en rama Develop sino crea otra rama en la nube.

1. REGRESAR A RAMA LOCAL

$ git checkout DevelopMirna

------------------------------------

Ver ramas, status y log de una rama

$ git branch

$ git status

$ git log

**Configuración Básica**

Configurar Nombre que salen en los commits

git config --global user.name "dasdo"

Configurar Email

git config --global user.email dasdo1@gmail.com

**Iniciando repositorio**

Iniciamos GIT en la carpeta donde esta el proyecto

git init

Clonamos el repositorio de github o bitbucket

git clone <url>

Añadimos todos los archivos para el commit

git add .

Hacemos el primer commit

git commit -m "Texto que identifique por que se hizo el commit"

subimos al repositorio

git push origin master

**GIT CLONE**

Clonamos el repositorio de github

git clone <url>

**GIT ADD**

Añadimos todos los archivos para el commit

git add .

Añadimos el archivo para el commit

git add <archivo>

Añadimos todos los archivos para el commit omitiendo los nuevos

git add --all

Añadimos todos los archivos con la extensión especificada

git add \*.txt

Añadimos todos los archivos dentro de un directorio y de una extensión especifica

git add docs/\*.txt

Añadimos todos los archivos dentro de un directorios

git add docs/

**GIT COMMIT**

Cargar en el HEAD los cambios realizados

git commit -m "Texto que identifique por que se hizo el commit"

Agregar y Cargar en el HEAD los cambios realizados

git commit -a -m "Texto que identifique por que se hizo el commit"

De haber conflictos los muestra

git commit -a

Agregar al ultimo commit, este no se muestra como un nuevo commit en los logs. Se puede especificar un nuevo mensaje

git commit --amend -m "Texto que identifique por que se hizo el commit"

**GIT PUSH**

Subimos al repositorio

git push <origien> <branch>

Subimos un tag

git push --tags

**GIT LOG**

Muestra los logs de los commits

git log

Muestras los cambios en los commits

git log --oneline --stat

Muestra graficos de los commits

git log --oneline --graph

**GIT DIFF**

Muestra los cambios realizados a un archivo

git diff

git diff --staged

**GIT HEAD**

Saca un archivo del commit

git reset HEAD <archivo>

Devuelve el ultimo commit que se hizo y pone los cambios en staging

git reset --soft HEAD^

Devuelve el ultimo commit y todos los cambios

git reset --hard HEAD^

Devuelve los 2 ultimo commit y todos los cambios

git reset --hard HEAD^^

Rollback merge/commit

git log

git reset --hard <commit\_sha>

**GIT REMOTE**

Agregar repositorio remoto

git remote add origin <url>

Cambiar de remote

git remote set-url origin <url>

Remover repositorio

git remote rm <name/origin>

Muestra lista repositorios

git remote -v

Muestra los branches remotos

git remote show origin

Limpiar todos los branches eliminados

git remote prune origin

**GIT BRANCH**

Crea un branch

git branch <nameBranch>

Lista los branches

git branch

Comando -d elimina el branch y lo une al master

git branch -d <nameBranch>

Elimina sin preguntar

git branch -D <nameBranch>

**GIT TAG**

Muestra una lista de todos los tags

git tag

Crea un nuevo tags

git tag -a <verison> - m "esta es la versión x"

**OTROS COMANDOS**

Lista un estado actual del repositorio con lista de archivos modificados o agregados

git status

Quita del HEAD un archivo y le pone el estado de no trabajado

git checkout -- <file>

Crea un branch en base a uno online

git checkout -b newlocalbranchname origin/branch-name

Busca los cambios nuevos y actualiza el repositorio

git pull origin <nameBranch>

Cambiar de branch

git checkout <nameBranch/tagname>

Une el branch actual con el especificado

git merge <nameBranch>

Verifica cambios en el repositorio online con el local

git fetch

Borrar un archivo del repositorio

git rm <archivo>