**Git comandos generales**

pwd = Ver la ruta en donde estoy actualmente

cd / = Va al escritorio – Home- En la raíz del disco

git init = crea un área en memoria RAM llamada staging y un repositorio donde están todos los cambios.

git add historia.txt = agrega el archivo al repositorio

rm historia.txt = remover

git commit -m "primer cambio"

git status = ver el estado del commit

show archivo.txt = Todos los cambios hechos.

add . = Agrega los cambios del todo el proyecto

log = Se muestran todos los commit ,historia del proyecto.

git diff = Es para comparar versiones. Se copia el código único de cada versión y se pegan seguidos y el lo compara.

cd /c

cd users

cd ..

mkdir = Crear carpetas desde GIT

touch = crear archivos vacíos

cat = muestra el contenido del archivo

history = historia de comandos inscritos hasta ahora

**Configurar Git para que salgan los datos de quien hizo el cambio de forma correcta**

git config – global user.name “Erika Valencia”

git config – global user.email “Erika.Valencia@gmail.com”

git config - -list

**Iniciar el proyecto desde el GIT**

git init

git status

git add filename

git commit -m “comentario con el cambio realizado”

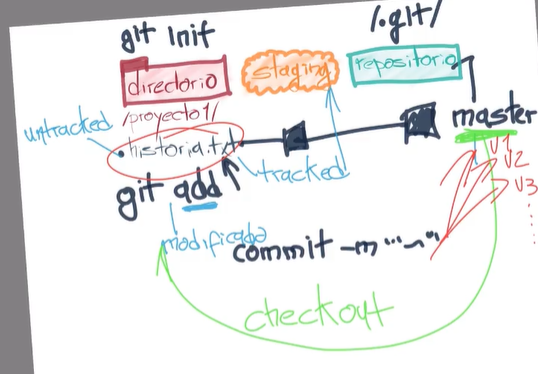
Si no se le pone comentario se le da ESC SHIF ZZ

git add . = agrega toda la carpeta para no tener que cargar archivo por archivo

**Como traer la última versión de un archivo, esto pasa cuando hay varias personas que trabajan en el mismo archivo.**

git checkout master “historia.txt” = esto es para traer la rama del master de ultimo commit

git checkout 19777aac6320630ebc68074b453732b8d69fc7a1 historia.txt



**Como regresar a una versión anterior**

git reset “poner el numero largo de la versión a regresar” - - hard

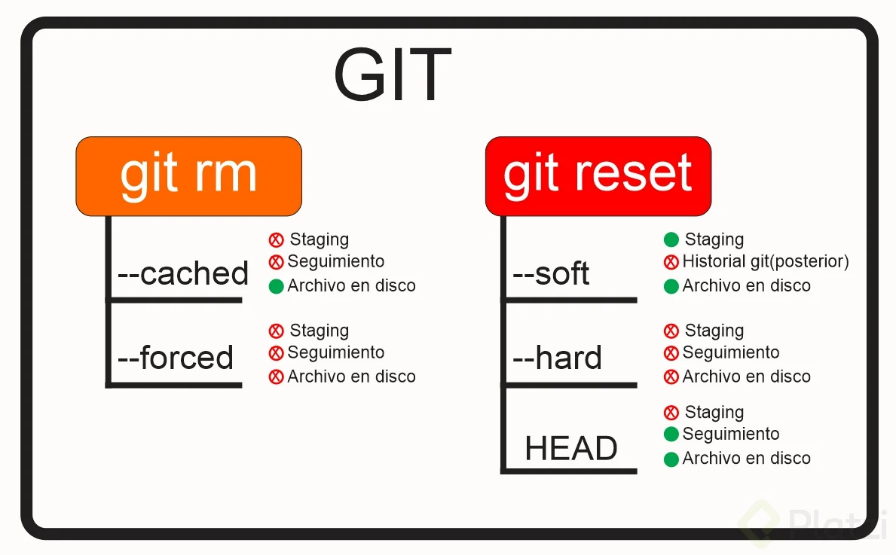
git reset “poner el numero largo de la versión a regresar” - - soft lo que este en el staging no lo borra eso lo mantiene para el proximo commit

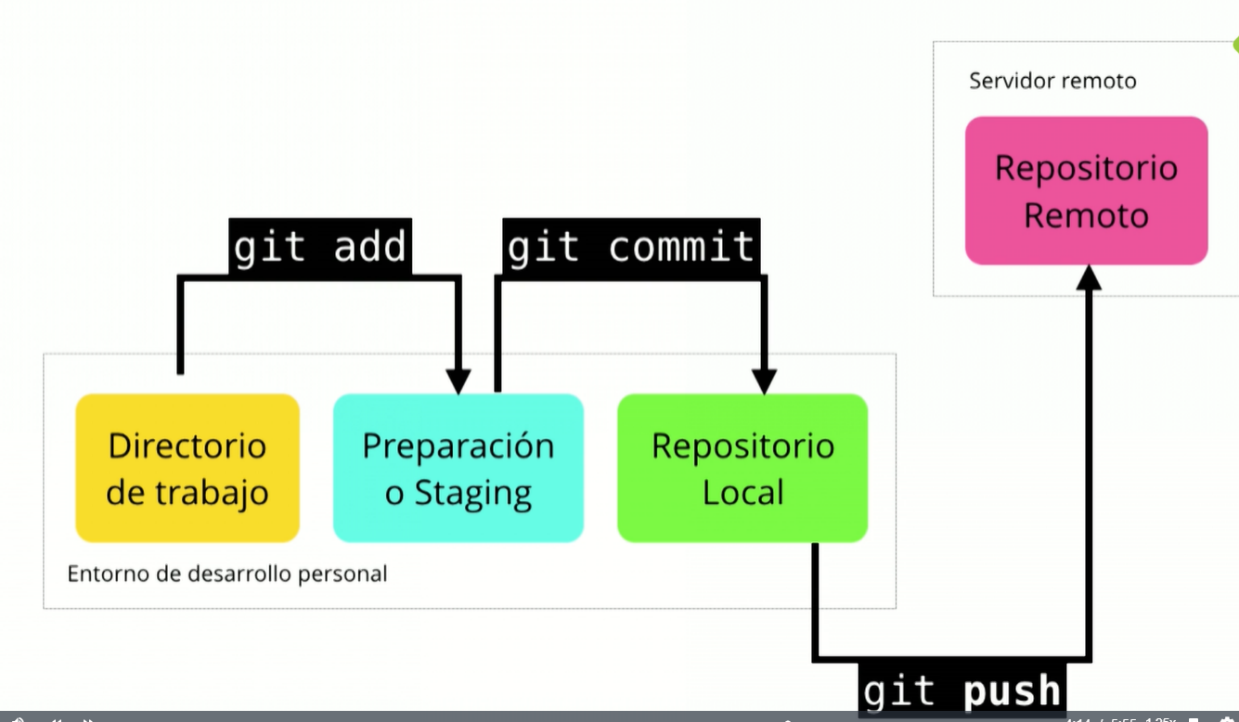
**Git reset vs. Git rm**

git rm --cached: Elimina los archivos de nuestro repositorio local y del área de staging, pero los mantiene en nuestro disco duro. Básicamente le dice a Git que deje de trackear el historial de cambios de estos archivos, por lo que pasaran a un estado untracked.

git rm --force: Elimina los archivos de Git y del disco duro. Git siempre guarda todo, por lo que podemos acceder al registro de la existencia de los archivos, de modo que podremos recuperarlos si es necesario (pero debemos usar comandos más avanzados).

git reset --soft: Borramos todo el historial y los registros de Git pero guardamos los cambios que tengamos en Staging, así podemos aplicar las últimas actualizaciones a un nuevo commit.

git reset --hard: Borra todo. Todo todito, absolutamente todo. Toda la información de los commits y del área de staging se borra del historial.



**Como usar las branch.**