**Comandos en Git**

Escala de tiempo

Descripción generada automáticamente

**Git init**

Creará un nuevo repositorio local GIT en el directorio actual.

Git init (nombre del Proyecto) puedes crear un repositorio dentro de un nuevo directorio especificando el nombre del proyecto.

**Git clone**

Se usa para copiar un repositorio. Si el repositorio está en un servidor remoto, usa:

Git clone (enlace del repositorio)

**Git status**

Muestra las modificaciones realizadas en un archivo.

**Git add**

Nos permite pasar los archivos del primer estado al segundo estado (Staging Área).

git add nombredelarchivo

Agrega un solo archivo.

git add.

Agrega todos los archivos que están dentro del repositorio.

**Git commit**

Creará una instantánea de los cambios y la guardará en el directorio git.

git commit –m “El mensaje que acompaña al commit va aquí”

**Git push**

Se usa para enviar confirmaciones locales a la rama maestra del repositorio remoto.

git push origin <master>

Reemplaza <master> con la rama en la que quieres enviar los cambios cuando no quieras enviarlos a la rama maestra.

**Git pull**

fusiona todos los cambios que se han hecho en el repositorio remoto con el directorio de trabajo local.

**Git reset**

Sirve para reiniciar el git add y poder hacer un commit en especifico

“git reset nombredelarchivo”

Otra manera en la que podemos resolver nuestro problema cuando sean más de un archivo, podemos hacer “git reset .”

Con este comando podemos deshacer cosas/cambios sobre los commits que hemos hecho anteriormente. Este comando tiene 3 variaciones: soft, mixed, hard

Soft: Lo que hace es regresar los archivos al área de staging.

Mixed: Con el podemos regresar los archivos al directorio del trabajo, nuestros cambios no se tocan, y nos aparecerán como modificaciones.

Hard: Debemos tener cuidado, con hard lo que hace es eliminar todos los cambios del commit que especificamos.

Por defecto el que está configurado es “mixed”.

git reset --(soft, mixed o hard) (hash del commit) sirve para borrar un commit

git reset HEAD~2(el numero representa la cantidad de commits a regresar)

**Git rm**

Lo que podemos hacer es eliminar nuestros archivos de nuestro repositorio y no solo eso, se encarga de agregar ese cambio (Eliminar un archivo) al staging area.

git rm (nombredelarchivo)

-r: esto significa a “recursivo” eliminando todos los archivos que están dentro de alguna carpeta.

–cached: Esta opción es para quitar etapas y eliminar rutas solo del índice. Los archivos de árbol de trabajo, ya sean modificados o no, se dejarán solos.

-f: Forced. se utiliza para eliminar un archivo si el archivo no está actualizado con su última confirmación de salida. Es para evitar que elimine un archivo en el que ha realizado cambios, pero aún no los ha registrado.

-q:Quiet. git rm normalmente genera una línea por cada archivo eliminado. Esta opción suprime esa salida

**Git log**

Podemos ver todos los commits que hemos hecho y a su vez la información sobre ellos, el usuario, email, la fecha e incluso el mensaje que agregamos en él.

git log --oneline Te muestra los commit en una línea junto con su hash (clave del commit) más corto

**Git checkout**

Ahora para poder volver en el tiempo solo tenemos que hacer lo siguiente: “git checkout hashdelcommit”

Para volver a nuestro último commit basta con hacer “git checkout master” o la otra forma es con “git reflog” y buscar nuestro último commit por su mensaje y utilizando su hash.