pwd: nos muestra la carpeta actual en la que nos encontramos
mkdir: nos permite crear carpetas. Ej. mkdir *NuevaCarpeta*
touch: nos permite crear archivos nuevos. Ej. touch *NuevoArchivo.txt*
cat: nos permite ver el contenido de un archivo. Ej. cat *NuevoArchivo.txt*
cd: nos permite cambiarnos de carpeta. Ej. cd *NuevaCarpet*
ls: nos permite ver los archivos de la carpeta donde estamos actualmente.
rm: Nos permite borrar un archivo o carpeta. Ej: rm NuevoArchivo.txt
cp “nombre del archivo que queremos copiar” “nombre del directorio a donde lo queremos copiar”: nos permite copiar un archivo.
mv “el directorio de donde queremos mover/el nombre del archivo” “el directorio hacia donde lo queremos mover”: nos permite mover un archivo.
clear: nos permite limpiar la pantalla.
history: ver los últimos comandos que ejecutamos y un número especial con el que podemos volver a repetir el comando.
Git Clone: Git clone es un comando para descargarte el código fuente existente desde un repositorio remoto (como Github, por ejemplo). git clone <https://link-con-nombre-del-repositorio>
Git branch: Nos permite trabajar en áreas paralelas al proyecto principal; podemos usar el comando git branch para crearlas, listarlas y eliminarlas. git branch <nombre-de-la-rama>
Git checkout: Se usa principalmente para cambiarte de una rama a otra. También lo podemos usar para chequear archivos y commits.

git checkout <nombre-de-la-rama>
Git status: El comando de git status nos da toda la información necesaria sobre la rama actual. git status
Git add: Necesitamos usar el comando git add para incluir los cambios del o de los archivos en tu siguiente commit. git add <archivo>
Git commit: Git commit es como establecer un punto de control en el proceso de desarrollo al cual puedes volver más tarde si es necesario. git commit -m "mensaje de confirmación"
Git push: Después de haber confirmado tus cambios, el siguiente paso que quieres dar es enviar tus cambios al servidor remoto. Git push envía tus commits al repositorio remoto. git push <nombre-remoto> <nombre-de-tu-rama>
Git pull: se utiliza para recibir actualizaciones del repositorio remoto. Este comando es una combinación del git fetch y del git merge lo cual significa que cundo usemos el git pull recogeremos actualizaciones del repositorio remoto (git fetch) e inmediatamente aplicamos estos últimos cambios en local (git merge). git pull <nombre-remoto>
Git revert: Una manera segura para deshacer nuestras commits es utilizar git revert. Para ver nuestro historial de commits, primero necesitamos utilizar el git log--oneline: git revert 3321844
Git merge: Cuando ya hayas completado el desarrollo de tu proyecto en tu rama y todo funcione correctamente, el último paso es fusionar la rama con su rama padre (dev o master). Git merge básicamente integra las características de tu rama con todos los commits realizados a las ramas dev (o master). Es importante que recuerdes que tienes que estar en esa rama específica que quieres fusionar con tu rama de características. git merge <nombre-de-la-rama>