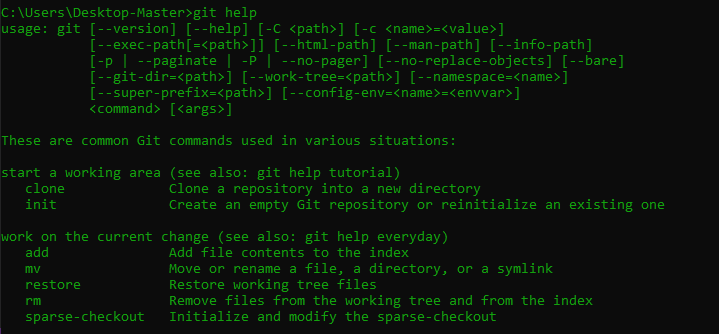
**Comandos Git**

Git es un sistema de control de versiones

**git --version**: muestra la versión de git que tenemos instalado

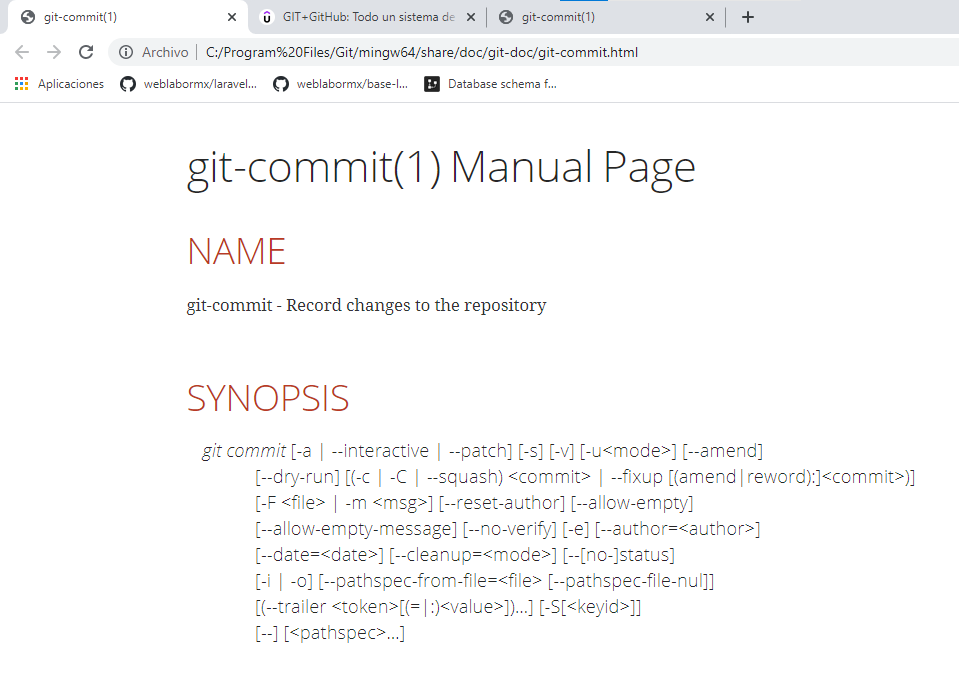


**git help**: nos da una ayuda, de los comandos de git



**git help commit**: en las versiones más recientes de git, abre en el navegador el manual de comando que hemos solicitado la ayuda





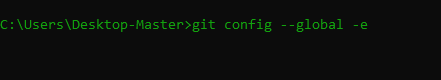
**git config --global user.name “Nombre de usuario”**: es para decirle a git quienes somos

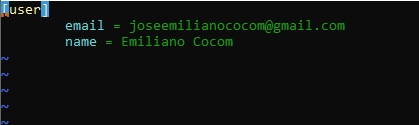


**git config --global user.email “Email de la cuenta de github”**: debemos tener una cuenta de github u otro servicio de la nube, para almacenar repositorios, ese es el correo que debemos ingresar



**git config --global -e**: muestra la configuración del usuario y alias si existe, para salir de la vista se utiliza :qa o para escribir y salir :wq!

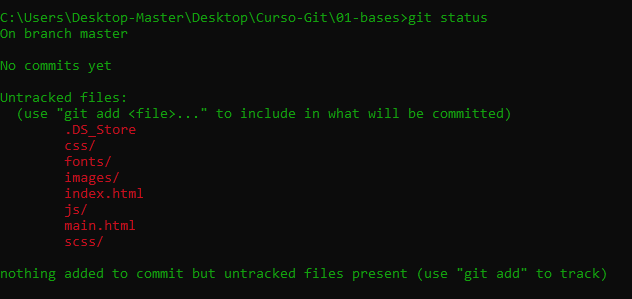




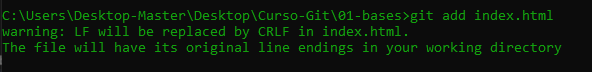
**git init**: inicializa el repositorio



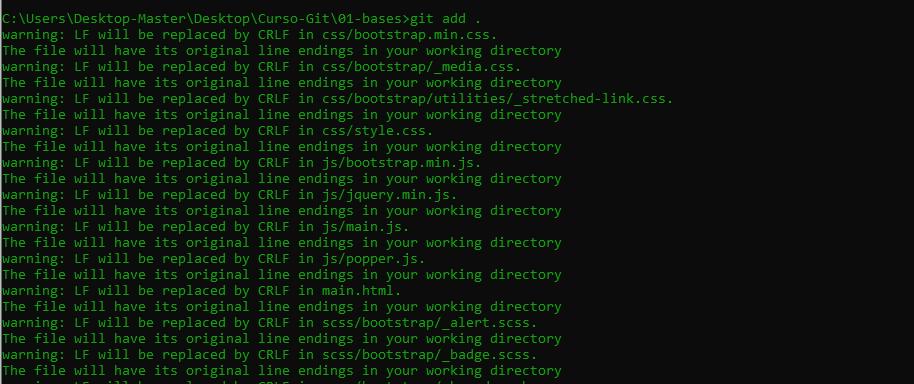
**git status**: muestra el status de los archivos que tienen o no seguimiento y cambios



**git add nombre\_del\_archivo**: agregar un archivo, para ser versionado



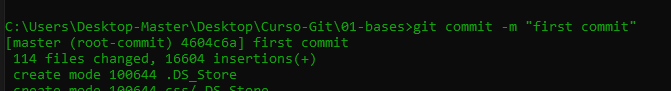
**git add .** : agrega todo los archivos para que sean versionados



**git reset nombre\_del\_archivo** : sirve para quitar un archivo que no queramos versionar

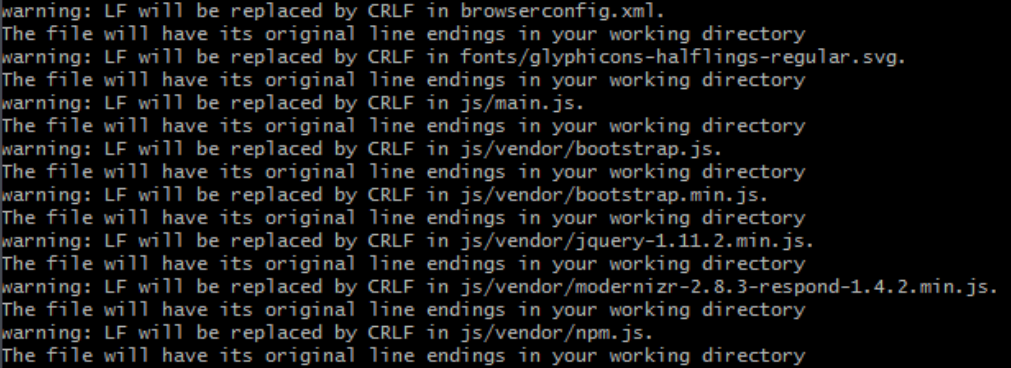


**git commit -m “mensaje para identificar cambios”** : se utiliza para confirmar cambios

****

**git config core.autocrlf true** : Soluciona un problema de CRLF, que es la interpretación de carácter

Fotografía del posible error



**git checkout -- .** : reconstruye el proyecto (archivos y carpetas) hasta el último commit, pero solo los archivos que se le esta dando seguimiento



**git branch** : nos muestra la rama en la cual estamos trabajando



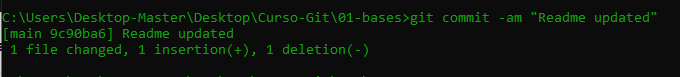
**git branch -m master main** : cambia el nombre de la rama master por main



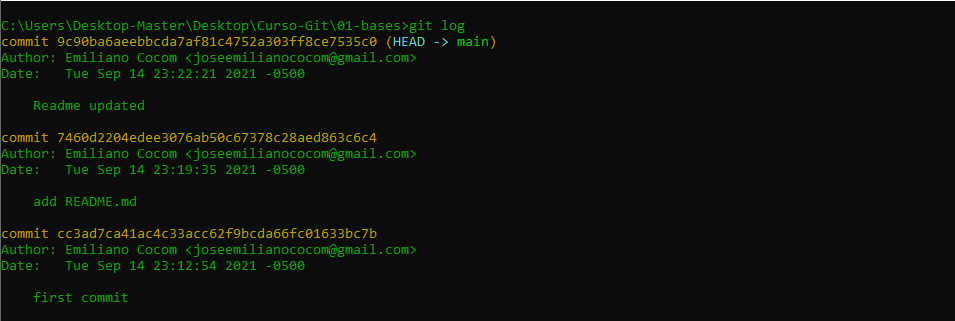
**git config --global init.defaultBranch main** : establece como configuración global que cada vez que se inicialice un repositorio la rama principal sea main



**git commit -am “mensaje del cambio”** : agregar cambios y confirmarlos, en un solo comando a de add y m de message, solo funciona si el archivo ha sido agregado con anterioridad, de lo contrario solo se ejecutar el commit con los cambios en los archivos versionados



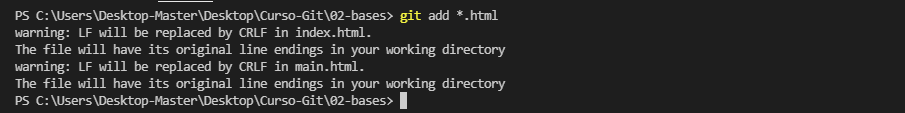
**git log** : muestra todos los log del repositorio, nos muestra todos los commits realizados



**git reset nombre\_del\_archivo\_uno nombre\_del\_archivo\_dos**: una forma de bajar del stage a varios archivos



**git add \*.extensión\_del\_archivo** : agrega al stage o escenario todos los archivos con la extensión indicada

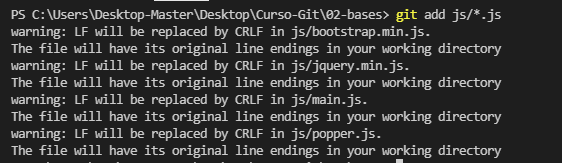


**git reset \*.extension\_del\_archivo** : baja del stage o escenario todos los archivos con la extensión indicada



**Otra forma de agregar varios archivos**

**git add carpeta/\*.extension\_del\_archivo** : agrega todos los archivos que esten dentro de la carpeta con la extensión indicada

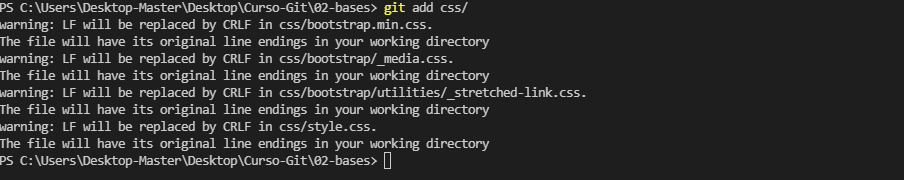
****

**Nota**: si un directorio esta vacío git no lo toma en cuenta, a menos que este contenga algún archivo, en caso de requerir que una carpeta tenga un control de versiones se debe agregar un archivo .gitkeep

**Otra forma de agregar el archivo .gitkeep**

****

**git add nombre\_de\_la\_carpeta/** : agrega todos los archivos que se encuentren dentro de la carpeta



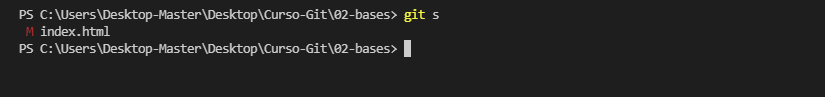
**git status --short** : muestra el estatus de los archivos de manera simplificada



**Alias**

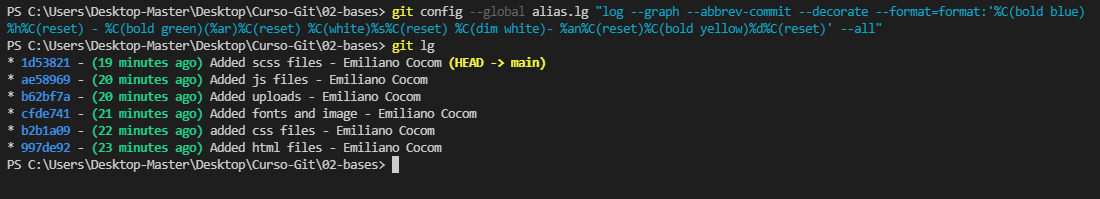
**git config --global alias.s “status –short”** : crea un alias de git status –short a simplemente git s

****

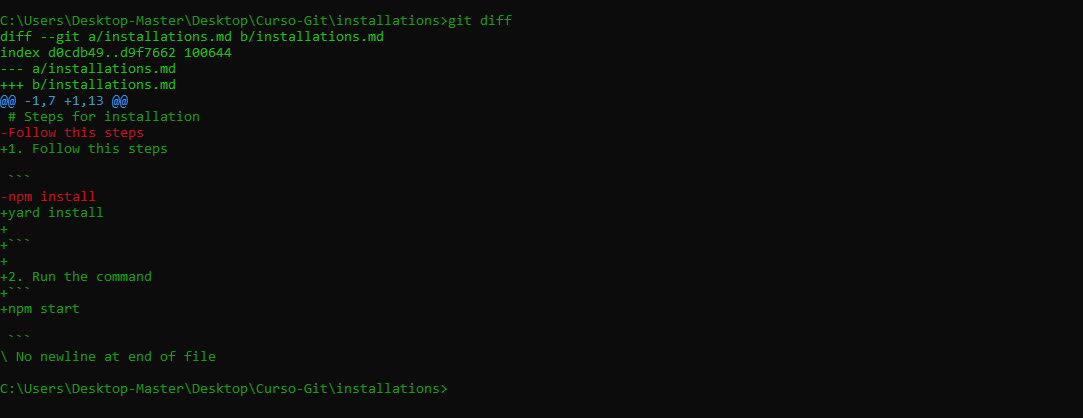
****

**git config --global alias.lg "log --graph --abbrev-commit --decorate --format=format:'%C(bold blue)%h%C(reset) - %C(bold green)(%ar)%C(reset) %C(white)%s%C(reset) %C(dim white)- %an%C(reset)%C(bold yellow)%d%C(reset)' --all"**

crea un alias que nos muestra la información de la siguiente manera

****

**git diff** : nos muestra los cambios que no se han subido al stage

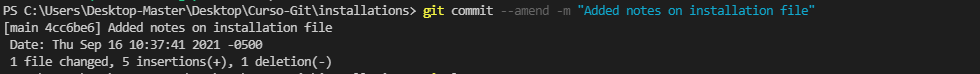


**Nota:** Si ya se han subido los cambios al stage git diff no mostrara estos cambios. Tenemos que ejecutar otro comando git diff –staged

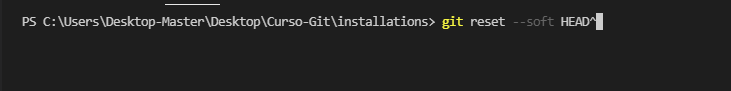
**git diff --staged** : muestra los que se han subido al stage



**git commit --amed -m “mensaje del commit”**: sirve para cambiar el mensaje del último commit

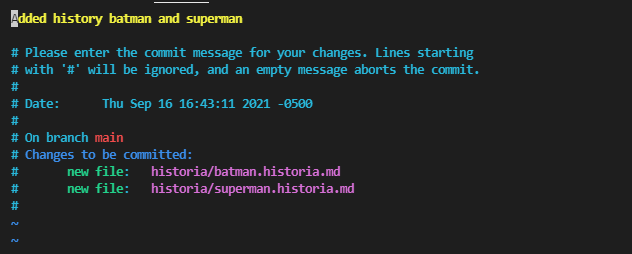


**git reset --soft HEAD^** : Elimina el ultimo commit

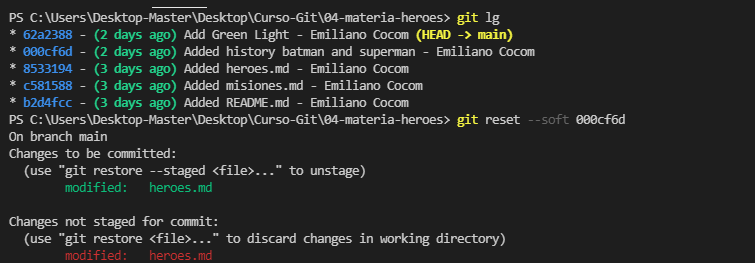


**git commit --amend** : con este comando se puede editar todo la información del último commit

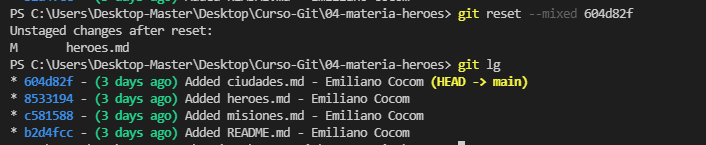




**git reset --soft hash** : Elimina el commit que le pasemos el hash



**git reset --mixed hash** : saca todo del stage y los cambios quedan listos para volver a añadirlos.



**git reset –hard hash** : Elimina todos los cambios que hemos realizado hasta el hash del commit

