**Manual Git**

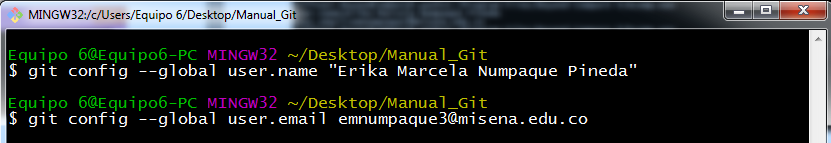
En el siguiente manual se muestra paso a paso los comandos realizado en git, estos comandos son unos de lo mas importantes al momento de trabajar con el entorno de git

* **Git configuración de Nombres y Email**

Git trae una herramienta llamada **git config** que permite obtener y establecer variables de configuración, controlan el aspecto y funcionamiento de Git.

Solo se necesita hacer esto una vez, si especificas la opción **--global** , Git siempre usará esta información para todo lo que haga en el sistema.

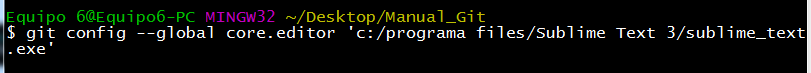
Se utiliza la siguiente línea de código para configurar el nombre y para el email cambia **user.name**  a **user.email @**



* **Editor de git establecer sublime text**

Ahora que la identidad está configurada, establecer el editor que por defecto se utilizará cuando Git necesita que introduzca un mensaje. Si no se indica nada, Git usa el editor por defecto del sistema, que generalmente es Vi o Vim.

Ingrese la siguiente línea de código para establecer el editor. Utilice el ajuste 'core.editor' para cambiar el que se da por defecto’-



* **Herramienta de mergeBeyon Compare.**

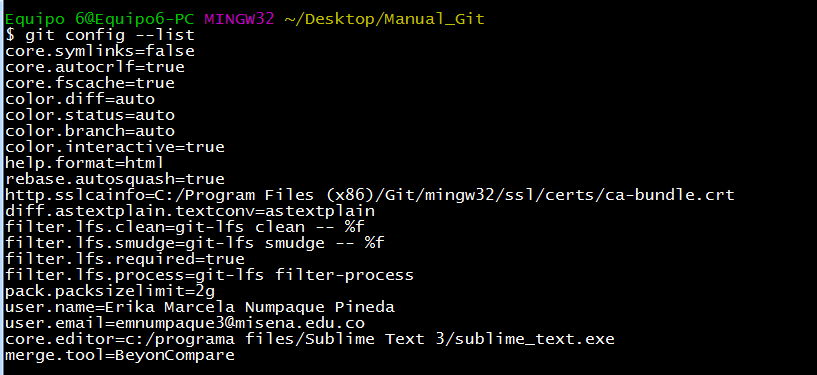
Configurar la herramienta, usada para resolver conflictos de unión (merge). Git acepta kdiff3, tkdiff, meld, xxdiff, emerge, vimdiff, gvimdiff, ecmerge, y opendiff como herramientas válidas.

Ingrese la siguiente línea de código para establecer la herramienta merge. Y entre comillas dobles (“”) para establecer el nombre.

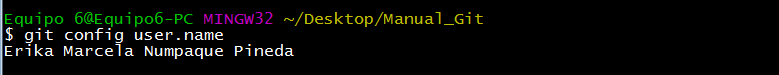


**Ejecute el comando gitconfig --list.**

Para comprobar la configuración, utilice el comando **git config --list** para listar todas las propiedades que Git ha configurado.

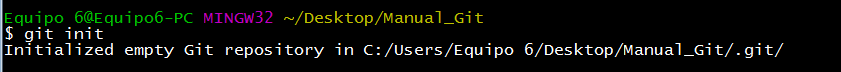


También puede comprobar qué valor quiere que Git le muestre, ejecute una clave específica por ejemplo git config {clave}

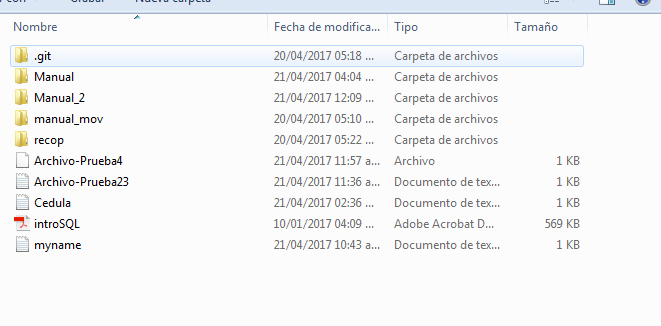


* **Crear un repositorio vacío.**

Para crear un repositorio digite el siguiente comando



Esto crea un nuevo subdirectorio llamado .git que contiene todos los archivos necesarios del repositorio —un esqueleto de un repositorio Git



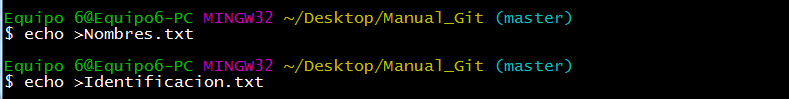
Todavía no hay nada en tu proyecto que esté bajo seguimiento

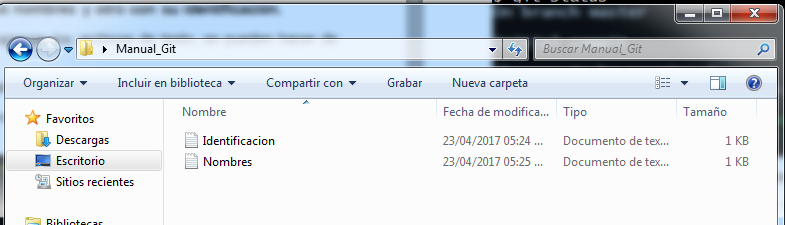
* **Añadir 2 Archivos de Texto uno con sus nombres y otro con su identificación.**

Introduzca la siguiente línea de código para agregar los archivos de texto, se pueden hacer de dos maneras.

La primera es utilizando el comando en Git **echo >**{nombre\_archivo}

Y la otra opción es ir a la carpeta donde se está trabajando dar Click derecho, opción nuevo y elegir Documento de Texto.





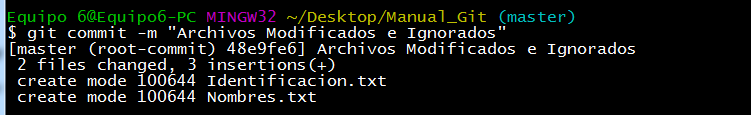
* **Configure el git ignore para ocultar los archivos .pdf y coloque uno dentro del repositorio**

El archivo .gitignore es un simple archivo de texto que indica que tipo archivos quiere que se ignoren



* **Confirmar sus cambios (commit)**

Ahora vamos a confirmar los cambios escribiendo **git commit** seguido de la instrucción **–m** para indicar un mensaje de confirmación.

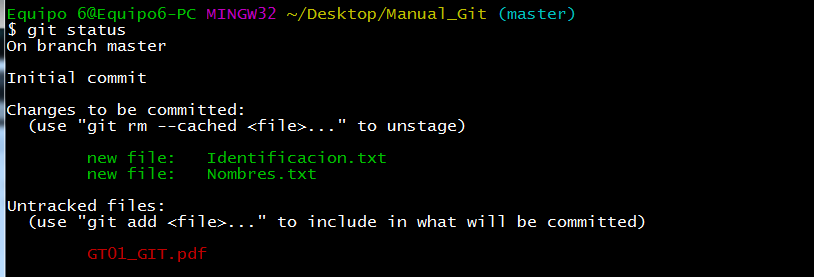


Acabas de crear una confirmación. Puede ver que el comando **commit** ha dado cierta información sobre la confirmación: a qué rama has confirmado (master), cuántos archivos se modificaron, y estadísticas acerca de cuántas líneas se han añadido y cuántas se han eliminado.

* **Mostrar el estado del repositorio**

Con el comando **git status** muestra el estado del árbol de trabajo. Simplemente arroja lo que está pasando con git addy git commit.

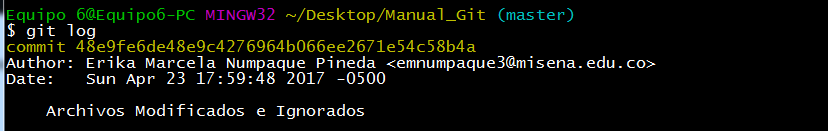
Lista de los archivos que se llevaron a cabo, sin preparar, y sin seguimiento.



Como se puede observar el archivo pdf ha sido ignorado

* **Mostrar un listado de las confirmaciones realizadas en el repositorio.**

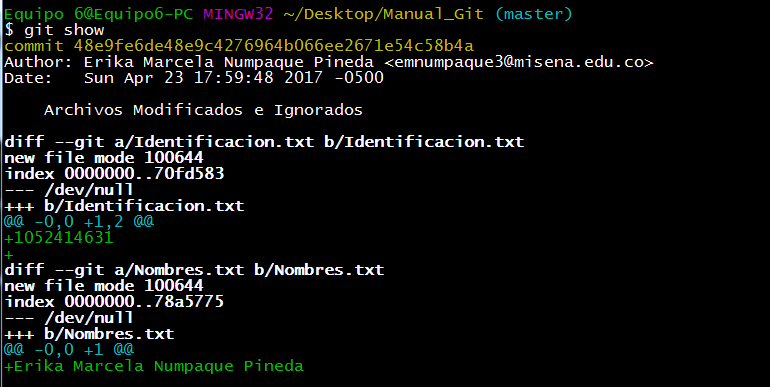
Permite listar la historia del proyecto, se filtran, las confirmaciones (commit) y búsquedas de cambios específicos,



Mientras que git status le permite inspeccionar el directorio de trabajo y el área de ensayo, git log solamente opera en la ‘history’ comprometida.

* **Modificar los archivos y mostrar que cambios se realizaron.**

El comando git show muestra los diversos tipos de objetos.



Para ‘commits’ muestra el mensaje de registro

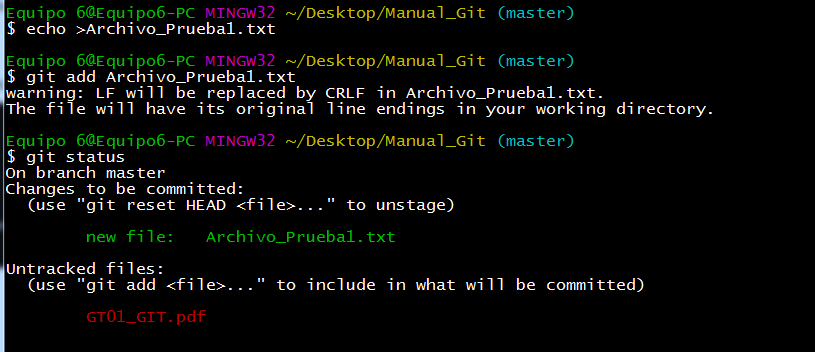
Para las etiquetas, se muestra el mensaje de la etiqueta y los objetos referenciados.

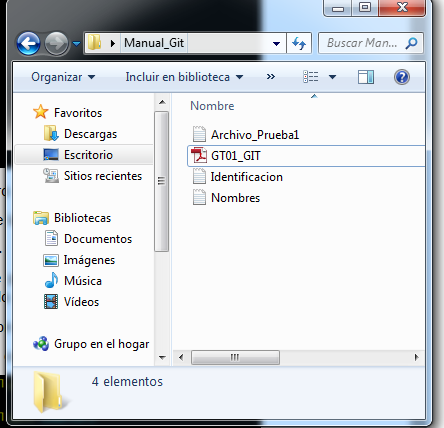
* **Agregue un archivo añádalo, elimínelo.**

Recordando que para agregar un archivo se puede hacer de la manera manual o por Git, en este caso mediante Git se escribe el comando.

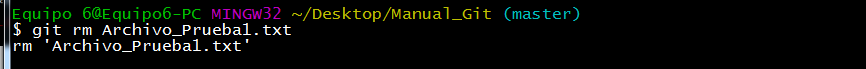
Seguido de esto vamos agregar el archivo con el comando add

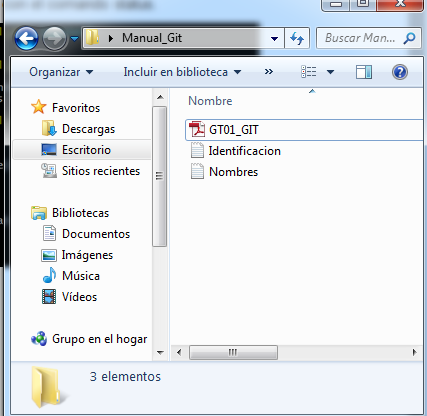
Revisamos que el archivo se haya agregado con el comando status.



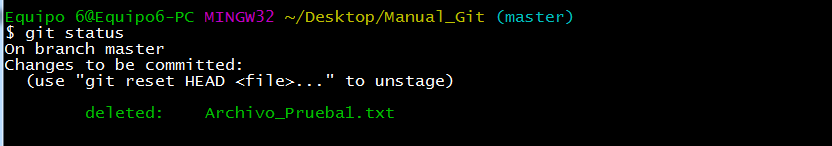


Una vez revisado, ingresamos el comando **git rm** {nombre del archivo} este se encarga deeliminar el archivo del directorio de trabajo, y para que no se pueda visualizar entre los archivos sin seguimiento.



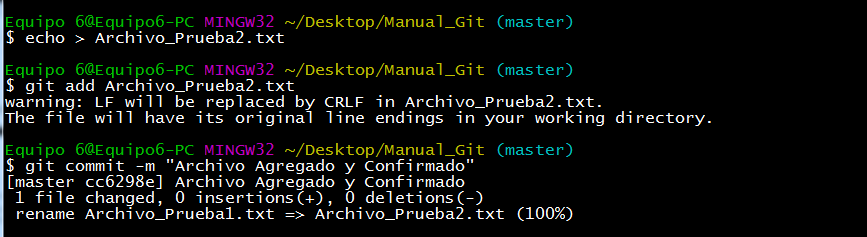


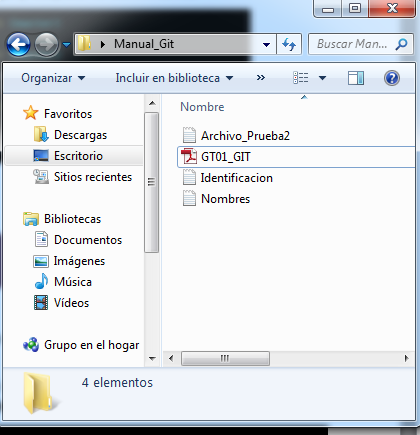
Después verificamos de nuevo con el comando **git status**



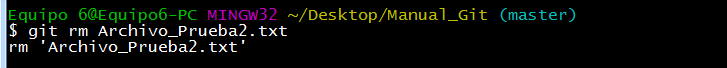
* **Agregue un archivo añádalo, confírmelo y elimínelo.**

Se procede a crear un nuevo archivo, se agrega y después lo confirmamos





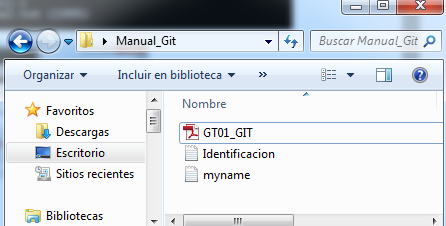
Ingresamos la instrucción rm, para eliminarlo



* **Renombre el archivo con sus nombres a myname.txt y muévalo dentro una carpeta en el repositorio.**

Con la instrucción **mv** se puede modificar el nombre de un archivo, en este caso ingresamos **git mv {nombre del archivo original} {nombre del archivo para modificar}**





Como se puede observar el nombre del archivo quedo modificado lo puede saber mirando la carpeta del repositorio o con un simple ‘status’.

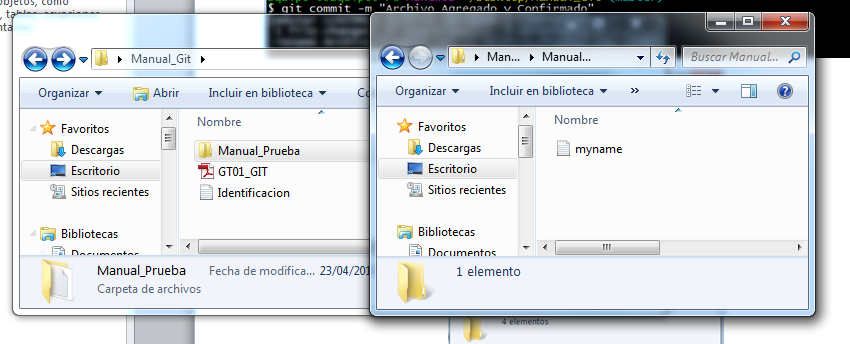
Ahora se procede a mover el archivo de una carpeta a otra. Cree una nueva carpeta en el repositorio, que pueda diferenciarla de la otra.

Se utiliza el mismo comando pero diferente de la siguiente manera:

**git mv {nombre del archivo modificado} {nombre de la nueva carpeta} / {nombre del archivo modificado}**

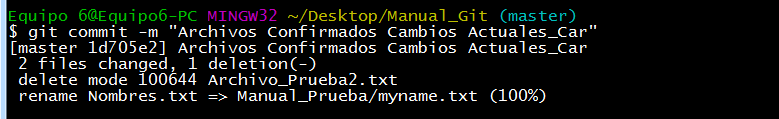


El archivo se pudo mover de una carpeta a otra.

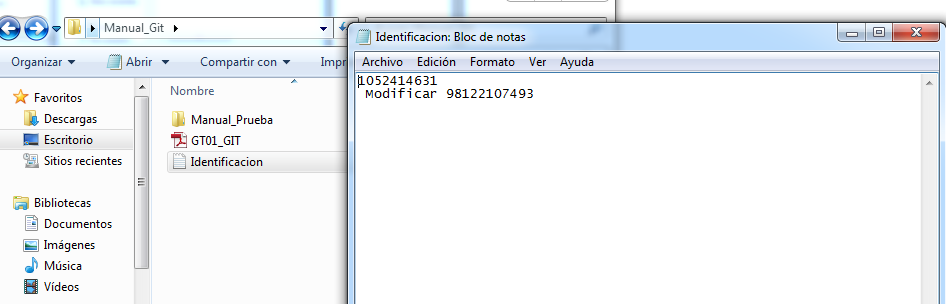


* **Realice un commit con los cambios actuales. Modifique un archivo guárdelo y restáurelo como estaba al inicio.**

Realizamos un commit.



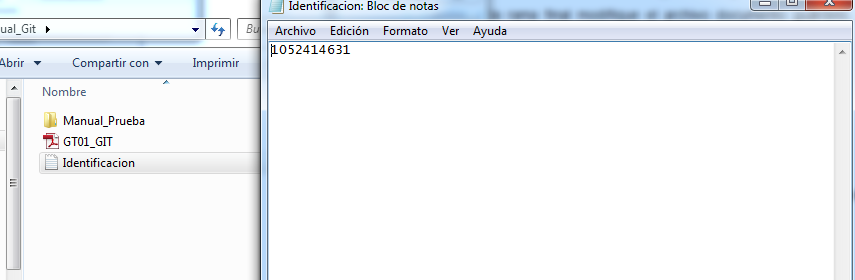
Modificar un archivo y guardarlo



Restaurar como estaba al inicio

Con el comando checkout se puede restablecer desde el principio





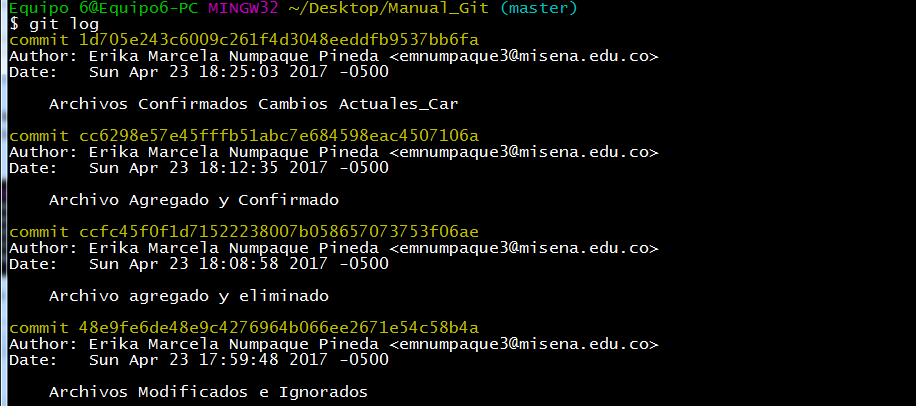
Como se pudo observar los cambios se modificaron.

* **Restaure el archivo al inicio con RESET.**

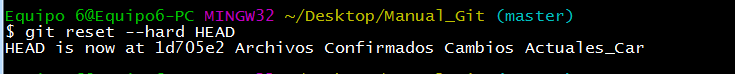


* **Restaure el espacio al anterior commit HARD RESET.**

Digite la instrucción log para saber que confirmaciones tiene

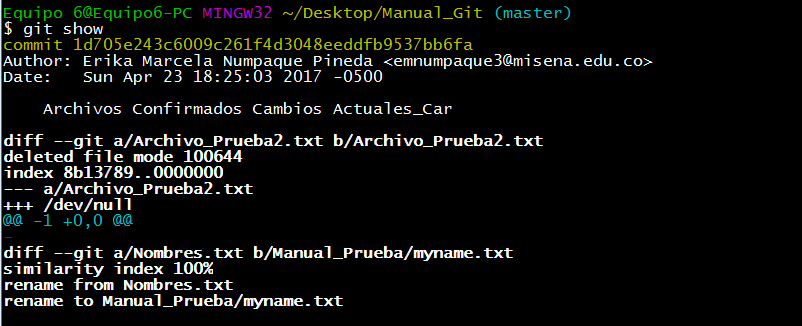


Digite el hard reset para eliminar lo que se hizo en la confirmación



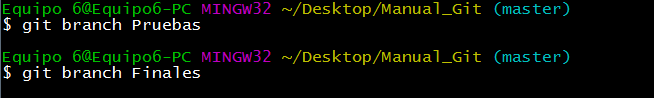
* **Obtenga el HEAD del repositorio actual.**

Digite el comando HEAD para saber en qué repositorio actual esta.

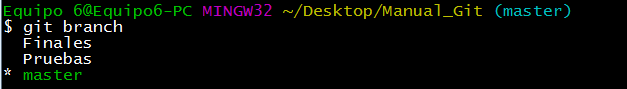


* **Cree dos ramas nuevas pruebas y finales**

Digite el comando branch(rama), es para crear una nueva rama

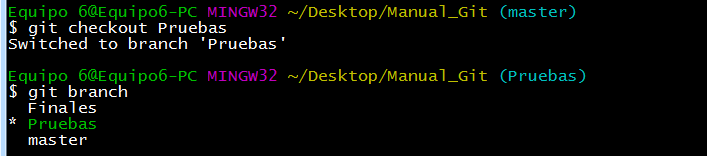


Y para saber que ramas ha creado genere la siguiente instrucción

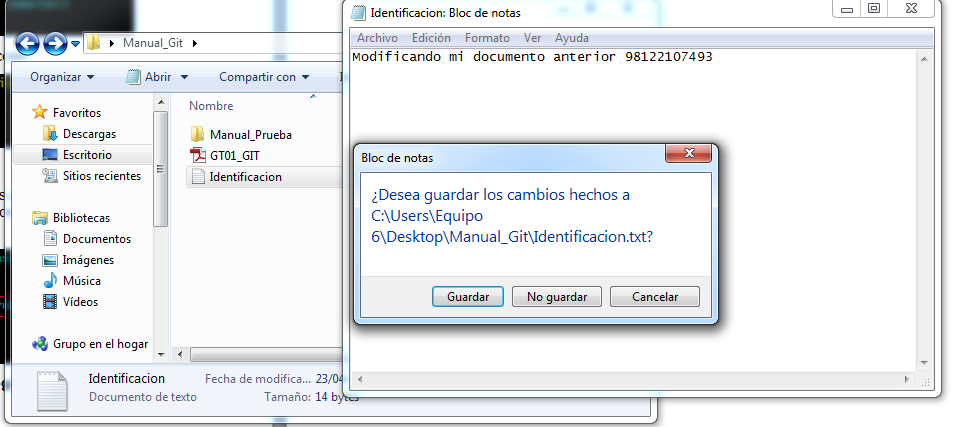


* **Cambie a la rama pruebas**

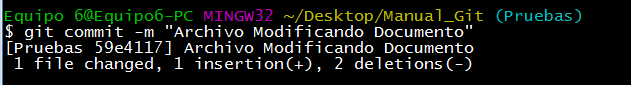
Digite la instrucción **checkout** para cambiar de rama después para visualizar observe que con el comando **git branch** la rama se ha cambiado, esto se reconoce con el \*, además después de la ruta aparece en paréntesis la rama en el que esta.



* **Modifique el archivo de su documento guárdelo, agréguelo y haga un commit.**

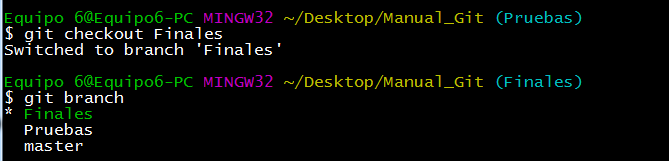




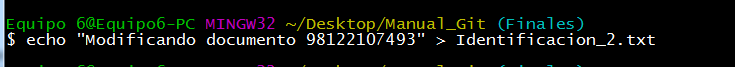


* **Cambie a la rama final modifique el archivo documento guárdelo, agréguelo y haga un commit.**

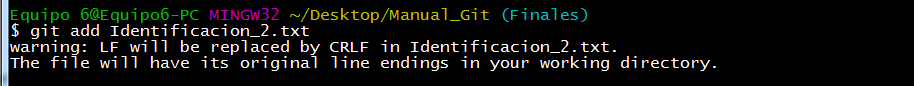
Modificar la rama con la misma instrucción checkout y se comprueba con la instrucción branch



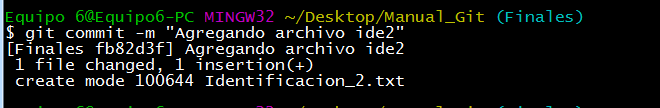
Modificando el archivo



Agregar el archivo

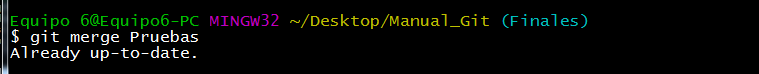


Realizando commit



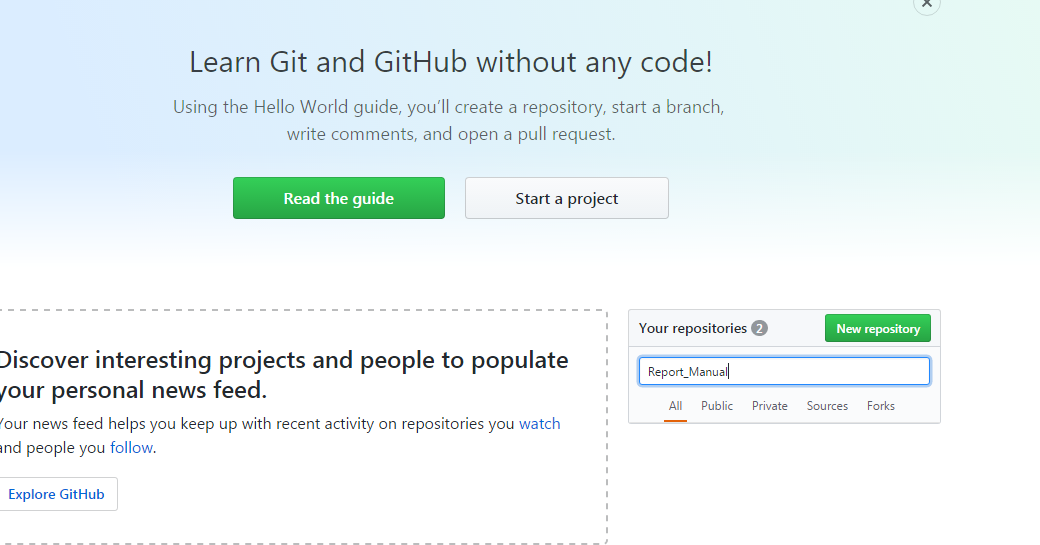
* **Haga un merge entre las dos ramas pruebas y final.**

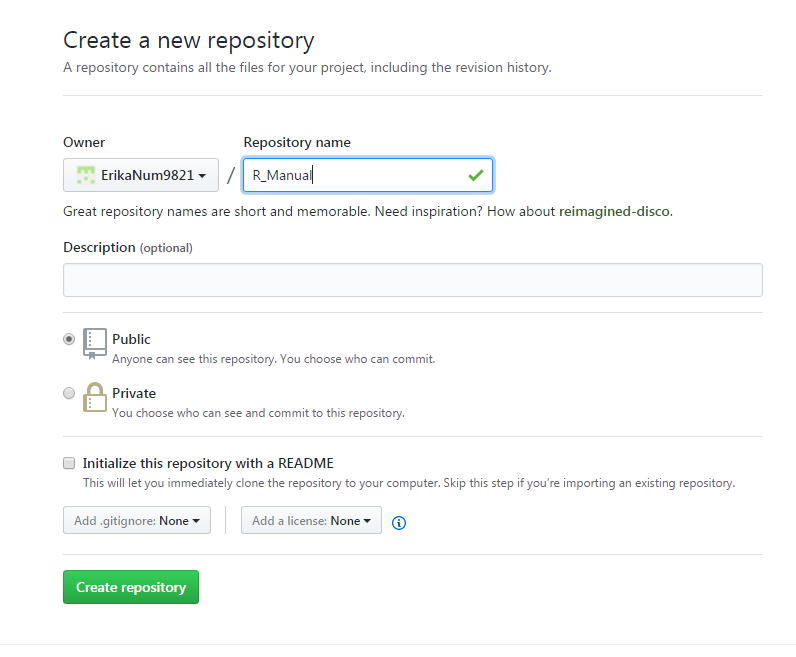
Como se esta ubicado en la rama Finales se hace el merge que significa unir ramas en Pruebas



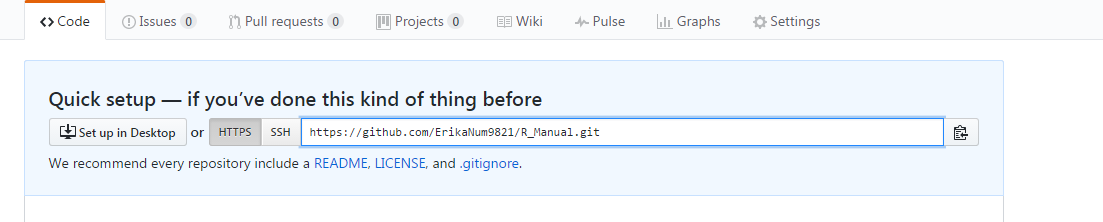
* **Cree un repositorio remoto (github) clónelo.**

Lo primero que tiene que hacer es crear una cuenta en GitHub, crea un nuevo repositorio





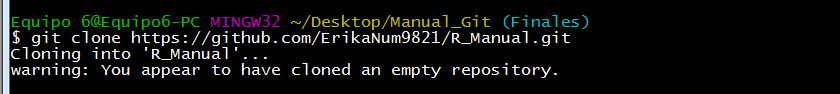
Copiar el link del repositorio creado

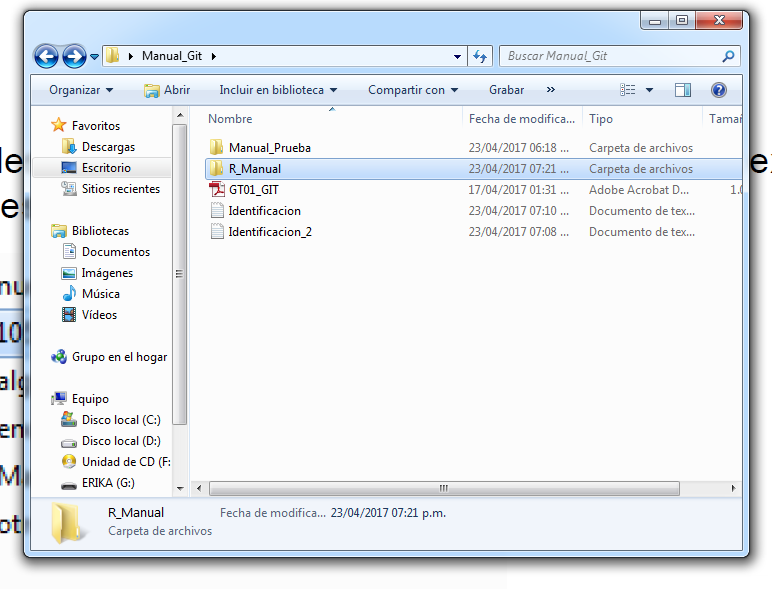


Digitar en Git el comando **git coln**  y copiar la ruta



Y como se pudo observar el repositorio ha sido clonado



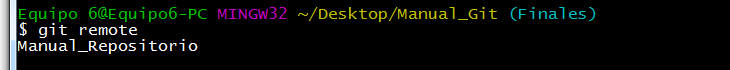


* **Muestre los repositorios remotos**

Para poder visualizar los repositorios primero creamos el repositorio

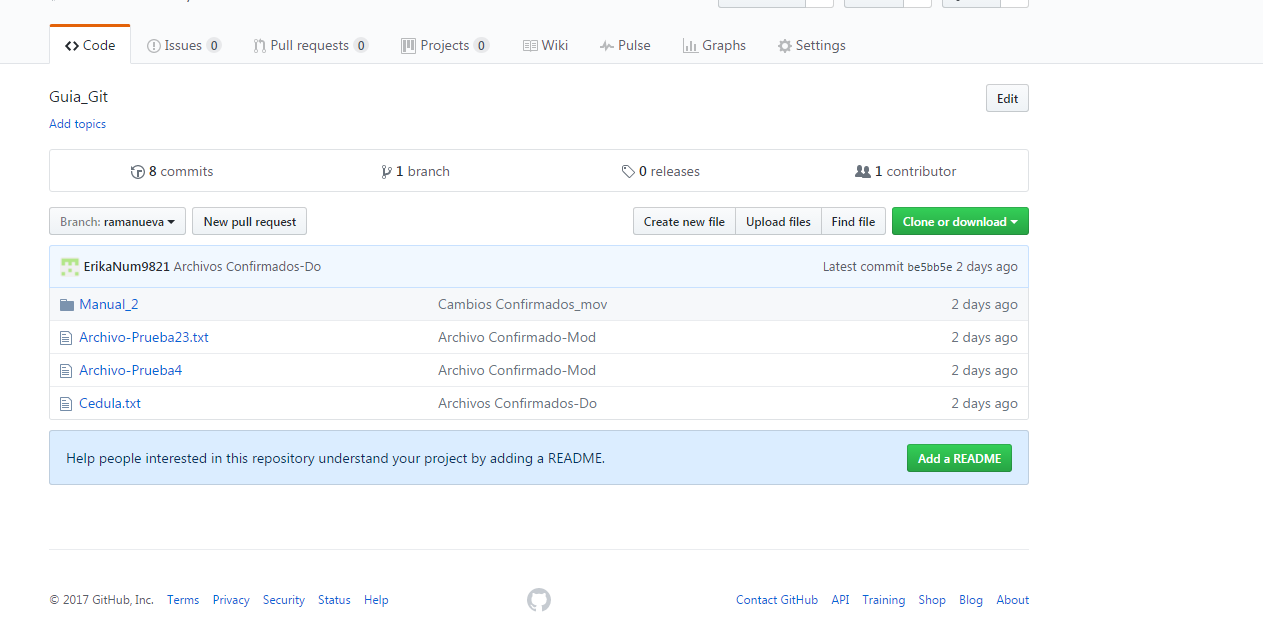


Verificando el remoto creando con la instrucción git remote



* **Agregue un archivo modifíquelo y súbalo al repositorio remoto**





* **Cree una nueva rama y súbala al su repositorio remoto.**

