



ÍNDICE

ÍNDICE	1
1.SISTEMA DE CONTROL DE VERSIONES	2

COMANDOS UTILIZADOS PARA LA ELABORACIÓN DE LA PRÁCTICA

[\(git init \)](#)

[\(git config --global user.name / user.email \)](#)

[\(git add \)](#)

[\(git rm -cached \(archivos\) \)](#)

[\(git commit \)](#)

[\(git reflog \)](#)

[\(git push \)](#)

[\(git remote add origin \)](#)

[\(git pull \)](#)

[\(git clone \)](#)

[\(git ignore \)](#)

[\(git diff \)](#)

[\(git branch \)](#)

[\(git merge "nombre de la otra branch" \)](#)

SISTEMA DE CONTROL DE VERSIONES

En este apartado haremos uso de GIT, siendo este un sistema utilizado para el control de cambios en el código fuente y otros archivos de un proyecto. Permite a los usuarios trabajar de manera simultánea y coordinar cambios en diferentes partes del proyecto.

Además permite la creación de ramas (branches), lo que facilita la gestión de distintas versiones del software.

También proporciona herramientas para la gestión de conflictos y la resolución de problemas en caso de errores.

A Continuación una lista de los principales comandos:

```
git init: Inicia un repositorio Git vacío en el directorio actual.

git add <archivo>: Agrega un archivo específico al área de preparación
para ser incluido en el próximo commit.

git add .: Agrega todos los archivos del directorio actual y sus
subdirectorios al área de preparación.

git commit: Registra los cambios realizados en el área de preparación en
el repositorio Git con un mensaje de commit.

git status: Muestra el estado actual del repositorio Git, incluyendo los
archivos modificados, eliminados o agregados.

git log: Muestra una lista de los commits realizados en el repositorio
Git.

git checkout <rama>: Cambia a la rama especificada.

git branch: Muestra una lista de las ramas del repositorio Git.

git merge <rama>: Fusiona la rama especificada con la rama actual.

git push: Envía los cambios locales al repositorio remoto.

git pull: Obtiene los cambios del repositorio remoto y los fusiona con
la rama actual.

git clone <url>: Crea una copia local del repositorio remoto.
```

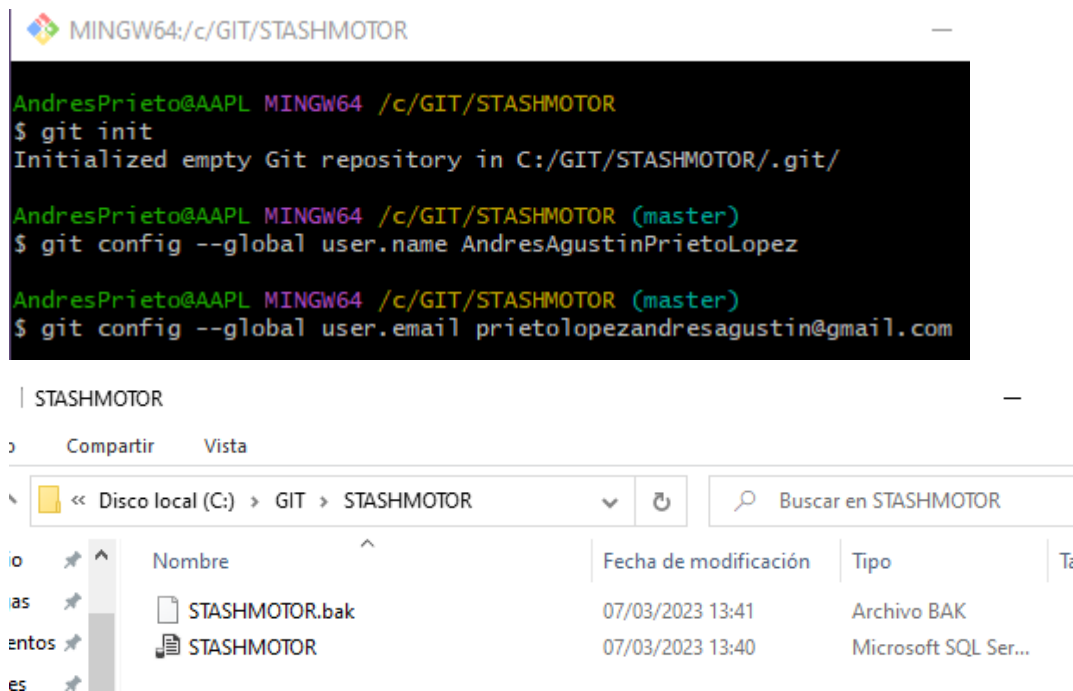
Con fines de organizar los scripts realizados en clases, todo lo relacionado a la elaboración del proyecto. Crearemos 3 repositorios distintos, y almacenaremos dentro de ellos la información correspondiente.

(git init)

A su vez hemos configurado nuestros datos personales en cada repositorio

(git config --global user.name / user.email) [Volver al índice→](#)

STASH MOTOR



The image shows two screenshots related to the 'STASHMOTOR' repository. The top screenshot is a terminal window titled 'MINGW64:/c/GIT/STASHMOTOR' showing the following commands and output:

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR
$ git init
Initialized empty Git repository in C:/GIT/STASHMOTOR/.git/

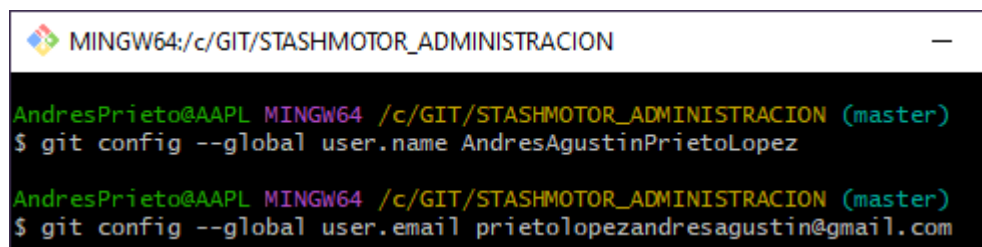
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR (master)
$ git config --global user.name AndresAgustinPrietoLopez

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR (master)
$ git config --global user.email prietolopezandresagustin@gmail.com
```

The bottom screenshot is a Windows File Explorer window titled 'STASHMOTOR' showing the contents of the 'C:\GIT\STASHMOTOR' directory. It contains two files:

Nombre	Fecha de modificación	Tipo
STASHMOTOR.bak	07/03/2023 13:41	Archivo BAK
STASHMOTOR	07/03/2023 13:40	Microsoft SQL Ser...

STASH MOTOR ADMINISTRACION



The image shows a terminal window titled 'MINGW64:/c/GIT/STASHMOTOR_ADMINISTRACION' with the following commands and output:

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git config --global user.name AndresAgustinPrietoLopez

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git config --global user.email prietolopezandresagustin@gmail.com
```

STASHMOTOR_ADMINISTRACION

Compartir Vista

<< GIT > STASHMOTOR_ADMINISTRACION

Buscar en STASHMOTOR_ADMINISTRACION

	Nombre	Fecha de modificación	Tipo	Tamaño
is	FILESTREAM	03/03/2023 0:42	Microsoft SQL Ser...	2 KB
ntos	FILETABLE	03/03/2023 0:42	Microsoft SQL Ser...	2 KB
s	PARTICIONES	03/03/2023 12:37	Microsoft SQL Ser...	6 KB
E DATOS	STASHMOTOR_CONTENIDA_BBDD	16/02/2023 4:47	Microsoft SQL Ser...	2 KB

SCRIPTS DE CLASES

```

MINGW64:/c/GIT/SCRIPTS_CLASES

AndresPrieto@AAPL MINGW64 /c/GIT/SCRIPTS_CLASES
$ git init
Initialized empty Git repository in C:/GIT/SCRIPTS_CLASES/.git/

AndresPrieto@AAPL MINGW64 /c/GIT/SCRIPTS_CLASES (master)
$ git config --global user.name AndresAgustinPrietoLopez

AndresPrieto@AAPL MINGW64 /c/GIT/SCRIPTS_CLASES (master)
$ git config --global user.email prietolopezandresagustin@gmail.com

```

Listamos los archivos de cada directorio. (**ls -la**) [Volver al índice](#)→

STASH MOTOR

```

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR (master)
$ ls -la
total 24840
drwxr-xr-x 1 AndresPrieto 197121      0 Mar  7 13:42 ./
drwxr-xr-x 1 AndresPrieto 197121      0 Mar  7 12:52 ../
drwxr-xr-x 1 AndresPrieto 197121      0 Mar  7 13:51 .git/
-rw-r--r-- 1 AndresPrieto 197121 25353728 Mar  7 13:41 STASHMOTOR.bak
-rw-r--r-- 1 AndresPrieto 197121  66948 Mar  7 13:40 STASHMOTOR.sql

```

STASH ADMINISTRACIÓN

```

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ ls -la
total 68
drwxr-xr-x 1 AndresPrieto 197121      0 Mar  7 13:03 ./
drwxr-xr-x 1 AndresPrieto 197121      0 Mar  7 12:52 ../
drwxr-xr-x 1 AndresPrieto 197121      0 Mar  7 13:51 .git/
-rw-r--r-- 1 AndresPrieto 197121  1903 Mar  3 00:42 FILESTREAM.sql
-rw-r--r-- 1 AndresPrieto 197121  1423 Mar  3 00:42 FILETABLE.sql
-rw-r--r-- 1 AndresPrieto 197121  5699 Mar  3 12:37 PARTICIONES.sql
-rw-r--r-- 1 AndresPrieto 197121  1409 Feb 16 04:47 STASHMOTOR_CONTENIDA_BBDD.sql
-rw-r--r-- 1 AndresPrieto 197121  2192 Mar  6 21:48 STASHMOTOR_RLS.sql
-rw-r--r-- 1 AndresPrieto 197121 21112 Mar  3 00:44 STASHMOTOR_SCRIPT_BBDD.sql
-rw-r--r-- 1 AndresPrieto 197121  1223 Mar  6 15:16 STASHMOTOR_TABLA_INMEMORY.sql
-rw-r--r-- 1 AndresPrieto 197121  4011 Mar  6 13:50 STASHMOTOR_TABLAS_VS.sql

```

SCRIPTS DE CLASES

```

AndresPrieto@AAPL MINGW64 /c/GIT/SCRIPTS_CLASES (master)
$ ls -la
total 384
drwxr-xr-x 1 AndresPrieto 197121      0 Mar  7 13:49 ./
drwxr-xr-x 1 AndresPrieto 197121      0 Mar  7 12:52 ../
drwxr-xr-x 1 AndresPrieto 197121      0 Mar  7 13:50 .git/
-rw-r--r-- 1 AndresPrieto 197121    2225 Dec 19 12:28 '1 Calling a Stored Procedu
re from PowerShell.ps1'
-rw-r--r-- 1 AndresPrieto 197121    5230 Dec 19 12:05 '1 Calling a Stored Procedu
re from PowerShell.sql'
-rw-r--r-- 1 AndresPrieto 197121     898 Dec 12 11:00 '1 Ejemplo-Primero_ADO_Net
(1).ps1'
-rw-r--r-- 1 AndresPrieto 197121     864 Dec 12 12:23 '1 Ejemplo-Primero_ADO_Net
BD Pubs Tabla authors.ps1'
-rw-r--r-- 1 AndresPrieto 197121   20015 Dec  2 07:53 '1 POWERSHELL SQL SERVER 5
diciembre 2022.ps1'
-rw-r--r-- 1 AndresPrieto 197121    1193 Dec  1 09:08 '1 PSDRIVE.ps1'
-rw-r--r-- 1 AndresPrieto 197121   13015 Jan  9 12:08 1-COLEGIO_MEDICO_TemporalT
ables_ENERO.sql

```

Vemos el estado de uno de ellos y visualizamos que se encuentren “**untracked**”, se refiere a los archivos que están presentes en el directorio de trabajo, pero que no están siendo rastreados por Git

```

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    FILESTREAM.sql
    FILETABLE.sql
    PARTICIONES.sql
    STASHMOTOR_CONTENIDA_BBDD.sql
    STASHMOTOR_RLS.sql
    STASHMOTOR_SCRIPT_BBDD.sql
    STASHMOTOR_TABLA_INMEMORY.sql
    STASHMOTOR_TABLAS_VS.sql

nothing added to commit but untracked files present (use "git add" to track)

```

Ahora procedemos a ingresarlos al “**staging area**” y que git pueda realizar el seguimiento de los mismos.

(**git add**) [Volver al índice→](#)

```

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git add FILESTREAM.sql FILETABLE.sql PARTICIONES.sql STASHMOTOR_CONTENIDA_BBDD.sql STASH
MOTOR_RLS.sql STASHMOTOR_SCRIPT_BBDD.sql STASHMOTOR_TABLA_INMEMORY.sql STASHMOTOR_TABLAS_VS
.sql

```

Verificamos nuevamente su estado

(**git status**) [Volver al índice→](#)

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   FILESTREAM.sql
    new file:   FILETABLE.sql
    new file:   PARTICIONES.sql
    new file:   STASHMOTOR_CONTENIDA_BBDD.sql
    new file:   STASHMOTOR_RLS.sql
    new file:   STASHMOTOR_SCRIPT_BBDD.sql
    new file:   STASHMOTOR_TABLA_INMEMORY.sql
    new file:   STASHMOTOR_TABLAS_VS.sql
```

Podríamos revertir la operación anterior con

(**git rm -cached (archivos)**) [Volver al índice→](#)

Realizamos un cambio en el fichero y veamos que sucede. (**git status**)

```
MINGW64:/c/GIT/STASHMOTOR_ADMINISTRACION
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   FILESTREAM.sql
    new file:   FILETABLE.sql
    new file:   PARTICIONES.sql
    new file:   STASHMOTOR_CONTENIDA_BBDD.sql
    new file:   STASHMOTOR_RLS.sql
    new file:   STASHMOTOR_SCRIPT_BBDD.sql
    new file:   STASHMOTOR_TABLA_INMEMORY.sql
    new file:   STASHMOTOR_TABLAS_VS.sql

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   STASHMOTOR_CONTENIDA_BBDD.sql

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$
```

Una vez realizado los cambios, hacemos un commit el cual creará una instantánea del estado de los archivos. También podemos agregar un comentario indicando el cambio realizado.

(**git commit**) [Volver al índice→](#)

También podríamos hacer un **checkout** lo que llevaría a revertir los cambios realizados.

Realizamos el **commit** y añadimos el mensaje.

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git commit -m "Confirmación de cambios realizados en STASH_CONTENIDA_BBDD"
[master (root-commit) 075423b] Confirmación de cambios realizados en STASH_CONTENIDA_BBDD
8 files changed, 1787 insertions(+)
create mode 100644 FILESTREAM.sql
create mode 100644 FILETABLE.sql
create mode 100644 PARTICIONES.sql
create mode 100644 STASHMOTOR_CONTENIDA_BBDD.sql
create mode 100644 STASHMOTOR_RLS.sql
create mode 100644 STASHMOTOR_SCRIPT_BBDD.sql
create mode 100644 STASHMOTOR_TABLA_INMEMORY.sql
create mode 100644 STASHMOTOR_TABLAS_VS.sql
```

Añadimos los cambios y realizamos nuevamente una instantánea.

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git add STASHMOTOR_CONTENIDA_BBDD.sql
```

Realizamos nuevamente un git status y verificamos los cambios.

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   STASHMOTOR_CONTENIDA_BBDD.sql
```

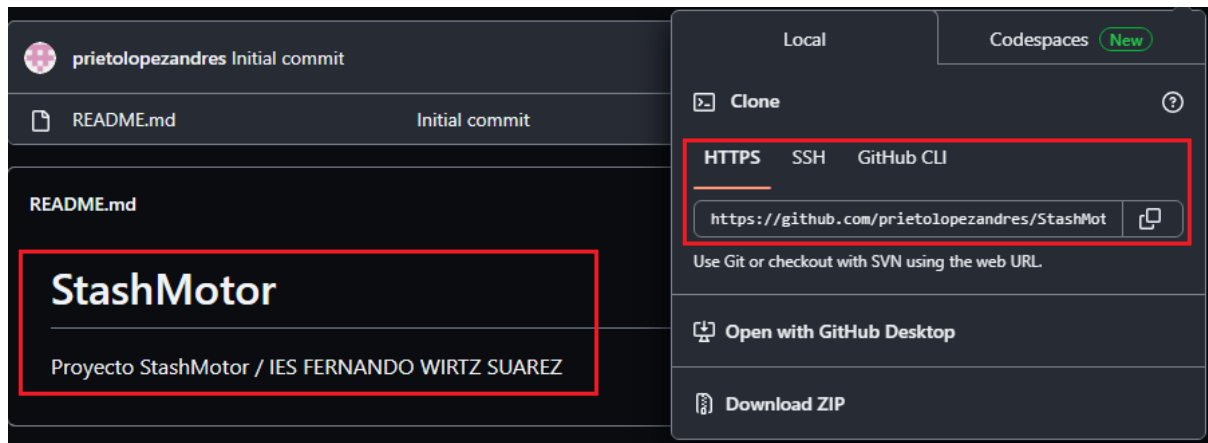
Podemos visualizar las instantáneas que hemos realizado con un **(git reflog)**

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git reflog
075423b (HEAD -> master) HEAD@{0}: commit (initial): Confirmación de cambios realizados en STASH_
DD
```

Todos los directorios con su contenido que visualizamos anteriormente se encuentran en local, pero queremos añadirlos a un repositorio en GitHub

Para ello, crearemos un repositorio en github “STASHMOTOR”, este a su vez generará una URL que será la ruta a donde haremos el

(git push) [Volver al índice→](#)



<https://github.com/prietolopezandres/StashMotor.git>

Añadimos el repositorio a git con

(git remote add origin) [Volver al índice→](#)

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git remote add origin https://github.com/prietolopezandres/StashMotor.git
```

Añadimos el contenido al repositorio haciendo uso del **(git push)** [Volver al índice→](#)

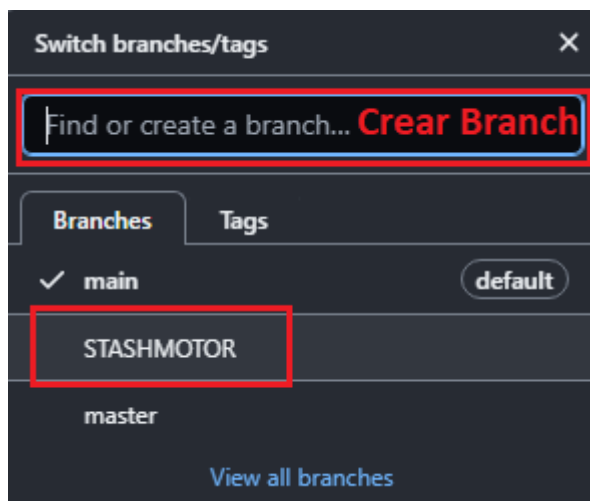
```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git push -u origin master
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Compressing objects: 100% (10/10), done.
Writing objects: 100% (10/10), 9.21 KiB | 3.07 MiB/s, done.
Total 10 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/prietolopezandres/StashMotor/pull/new/master
remote:
To https://github.com/prietolopezandres/StashMotor.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

Desde la interfaz de GitHub nos indica con una notificación la existencia de un push, pudiendo observar los cambios en el código.



Procedemos a realizarlo con cada uno de los repositorios.

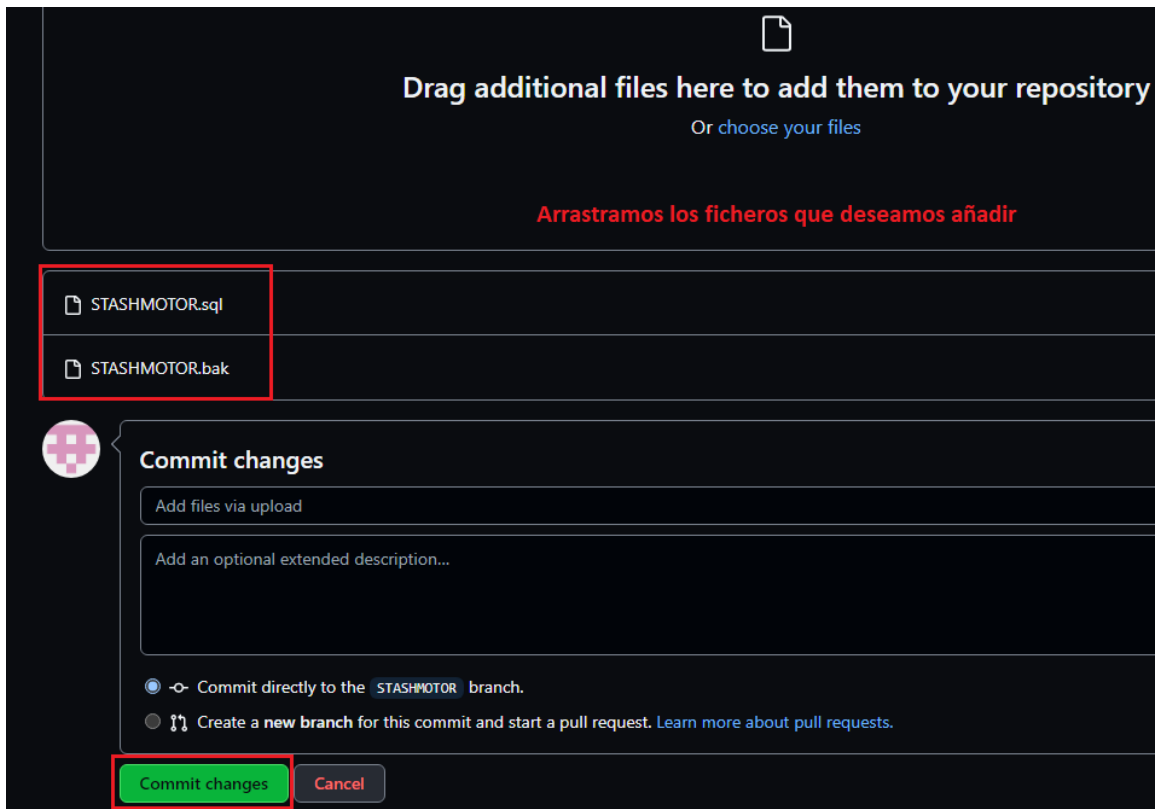
Esta vez lo haremos desde el entorno gráfico, crearemos una nueva rama o branches STASHMOTOR y desde ella agregamos el contenido.



[Volver al índice→](#)



Arrastramos los ficheros que deseamos añadir y aplicamos los cambios, nos permite a su vez hacerlo directamente sobre el branch que estamos trabajando, o realizar un nuevo branch para dicho commit.



STASHMOTOR.bak	Add files via upload	now
STASHMOTOR.sql	Add files via upload	now

[Volver al índice→](#)

Ahora también podríamos trasladar contenido de GitHub a local, para ello podríamos hacerlo a través de

(git pull) [Volver al índice→](#)

Previamente forzamos la modificación de un fichero con la finalidad de demostrar, que dicha acción nos permite actualizar los datos que tenemos en local, contra lo que se encuentra en github

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git pull
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 10 (delta 3), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (10/10), 7.08 MiB | 1.83 MiB/s, done.
From https://github.com/prietolopezandres/StashMotor
* [new branch]      STASHMOTOR -> origin/STASHMOTOR
* [new branch]      main -> origin/main
Already up to date.
```

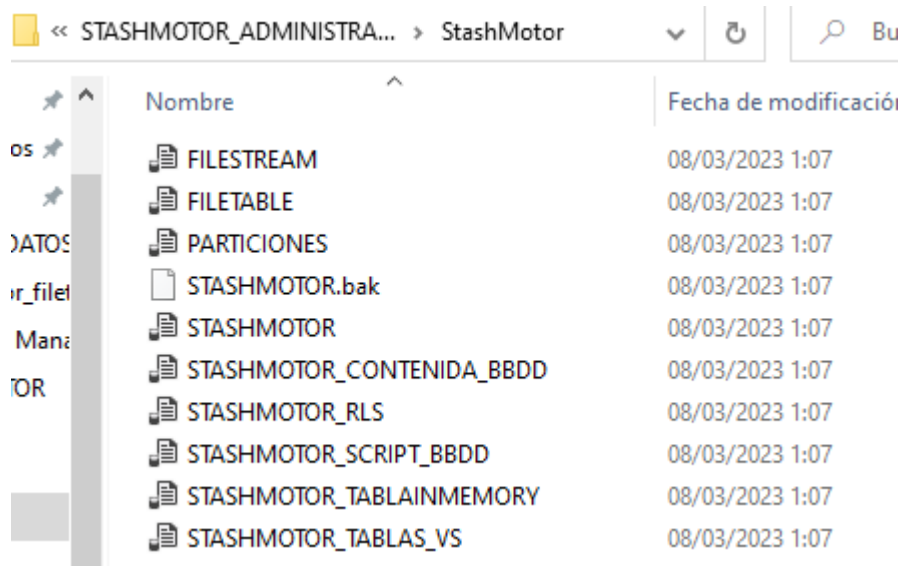
La otra opción disponible, sería hacer un (git clone) con la finalidad de recuperar algún archivo en caso de haberlo perdido, también es aplicable si deseamos testear algún script, etc.

(git clone) [Volver al índice→](#)

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git clone https://github.com/prietolopezandres/StashMotor.git
Cloning into 'StashMotor'...
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 20 (delta 3), reused 8 (delta 0), pack-reused 0
Receiving objects: 100% (20/20), 7.09 MiB | 12.03 MiB/s, done.
Resolving deltas: 100% (3/3), done.
```

Generando el siguiente resultado.

GIT > STASHMOTOR_ADMINISTRACION >		▼	↺	🔍 E
^	Nombre	Fecha de modificaci		
📁	StashMotor	08/03/2023 1:07		



	Nombre	Fecha de modificación
os	FILESTREAM	08/03/2023 1:07
	FILETABLE	08/03/2023 1:07
DATOS	PARTICIONES	08/03/2023 1:07
ir_file	STASHMOTOR.bak	08/03/2023 1:07
Mane	STASHMOTOR	08/03/2023 1:07
TOR	STASHMOTOR_CONTENIDA_BBDD	08/03/2023 1:07
	STASHMOTOR_RLS	08/03/2023 1:07
	STASHMOTOR_SCRIPT_BBDD	08/03/2023 1:07
	STASHMOTOR_TABLAINMEMORY	08/03/2023 1:07
	STASHMOTOR_TABLAS_VS	08/03/2023 1:07

También podríamos hacer uso del (git ignore) en caso de que deseemos ignorar los cambios en ciertos ficheros.

(git ignore) [Volver al índice→](#)

Podemos visualizar los cambios que se han realizado entre ficheros con un (git diff), permitiéndonos visualizar las diferencias entre los mismos.

(git diff) [Volver al índice→](#)

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git diff
diff --git a/STASHMOTOR_TABLAS_VS.sql b/STASHMOTOR_TABLAS_VS.sql
deleted file mode 100644
index 99e9b26..0000000
--- a/STASHMOTOR_TABLAS_VS.sql
+++ /dev/null
@@ -1,182 +0,0 @@
---TABLAS TEMPORALES / VERSION DEL SISTEMA--
-
---CREACION BASE DE DATOS--
-
-DROP DATABASE IF EXISTS STASHMOTOR_TABLATEMPORAL_VS
-GO
```

Branch o rama: es una línea de desarrollo separada que permite a los usuarios trabajar en distintas versiones de un proyecto simultáneamente.

En otras palabras, un "branch" es una copia del código en un estado particular que puede evolucionar de forma independiente del resto del proyecto.

Cada vez que se crea una nueva rama, se crea una bifurcación en el historial del proyecto, lo que permite a los usuarios trabajar en diferentes funcionalidades,

características o correcciones de errores sin interferir con el trabajo de otros colaboradores.

Al final de todo esto, es importante destacar que se puede unificar toda la información en un fichero en común.

Para ello haremos lo siguiente:

(git branch) [Volver al índice→](#)

```
MINGW64:/c/GIT/STASHMOTOR_ADMINISTRACION
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git init
Reinitialized existing Git repository in C:/GIT/STASHMOTOR_ADMINISTRACION/.git/
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git branch STASH_V2
```

Ahora visualizamos las ramas que se han creado. (git show-branch) Que en este caso son la rama MASTER Y STASH_v2

(git show-branch) [Volver al índice→](#)

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git show-branch
! [STASH_V2] Confirmación de cambios realizados en STASH_CONTENIDA_BBDD
* [master] Confirmación de cambios realizados en STASH_CONTENIDA_BBDD
--
+* [STASH_V2] Confirmación de cambios realizados en STASH_CONTENIDA_BBDD
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
```

También podemos movernos entre ramas con el siguiente comando

(git checkout "nombre de la rama") [Volver al índice→](#)

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git checkout STASH_V2
Switched to branch 'STASH_V2'
M      STASHMOTOR_CONTENIDA_BBDD.sql
D      STASHMOTOR_TABLAS_VS.sql
```

Ahora haremos una modificación sobre uno de los ficheros

```
AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (STASH_V2)
$ nano STASHMOTOR_TABLAS_VS.sql
```

```

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (STASH_V2)
$ git add STASHMOTOR_TABLAS_vS.sql

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (STASH_V2)
$ git commit -m "test branch"
[STASH_V2 b1ca15f] test branch
1 file changed, 2 insertions(+)

```

Podemos comprobar con Master que los cambios se encuentran en ramas distintas, por lo que no seremos capaces de apreciar el cambio estando en Master.

```

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (STASH_V2)
$ git checkout master
Switched to branch 'master'
M      STASHMOTOR_TABLAS_VS.sql
Your branch is up to date with 'origin/master'.

```

```

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git show-branch
! [STASH_V2] test branch
* [master] Confirmación de cambios realizados en STASH_CONTENIDA_BBDD
--
+ [STASH_V2] test branch
+* [master] Confirmación de cambios realizados en STASH_CONTENIDA_BBDD

```

Merge: en Git permite combinar cambios de dos o más ramas de un repositorio. Cuando se utiliza la función merge, Git fusiona los cambios realizados en dos ramas diferentes y los combina en una sola.

El proceso de fusión en Git funciona al comparar las diferencias entre dos ramas y aplicarlas a una nueva rama de fusión, que luego se convierte en la rama actual

Para ello utilizamos el comando

(git merge "nombre de la otra branch") [Volver al índice→](#)

```

AndresPrieto@AAPL MINGW64 /c/GIT/STASHMOTOR_ADMINISTRACION (master)
$ git merge STASH_V2
Updating 075423b..b1ca15f
Fast-forward
 STASHMOTOR_CONTENIDA_BBDD.sql | 2 ++
1 file changed, 2 insertions(+)

```

Veamos los logs. **(git log --oneline)** [Volver al índice→](#)