

MANUAL DE GITHUB

Índice

INTRODUCCIÓN	1
DESCARGA.....	1
DESCARGA DE Git	1
DESCARGA DE VISUAL STUDIO CODE.....	3
CREAR CUENTA EN GitHub	4
EMPAREJAMIENTO DE CARPETA LOCAL CON EL REPOSITORIO DE GitHub.....	7
ENVIO DE ARCHIVOS A UN REPOSITORIO GitHub	10
ACTUALIZACIÓN DE ARCHIVOS EN LÍNEA	12
CLONAR REPOSITORIOS.....	13

1. INTRODUCCIÓN

En este manual se va a explicar los principales pasos para trabajar con repositorios en línea alojados en GitHub.

GitHub es una herramienta de código abierto que nos permite trabajar con archivos compartidos en repositorios alojados en la nube y poder acceder a ellos desde cualquier terminal. Para poder establecer una conexión desde una estación local al servidor de GitHub, se van a necesitar la instalación de los siguientes programas en el equipo: Git y Visual Studio Code.

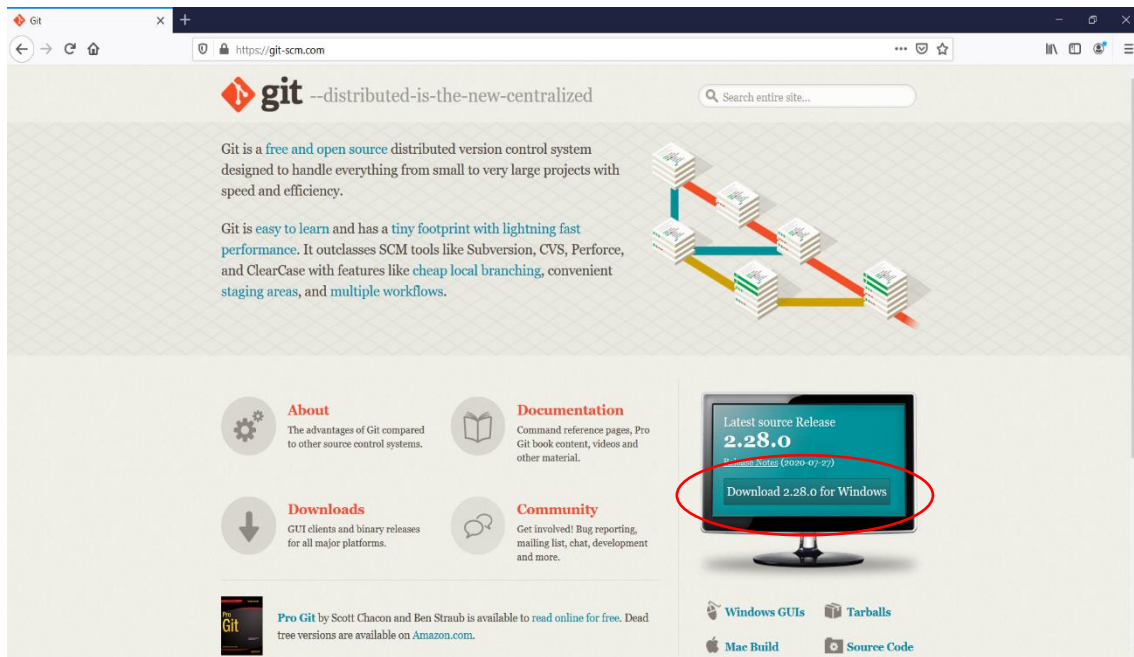
2. DESCARGA

2.1 DESCARGA DE Git

Git es un programa libre de control de sistemas y de código abierto de versión distribuida, diseñado para gestionar todo tipo de proyectos con agilidad y eficiencia. La función de este software es establecer comunicación entre el host y el servidor de GitHub.

Para descargar este programa nos debemos de dirigir a la dirección <https://git-scm.com/> desde el navegador instalado en el equipo, el cual nos redireccionara a la página oficial.

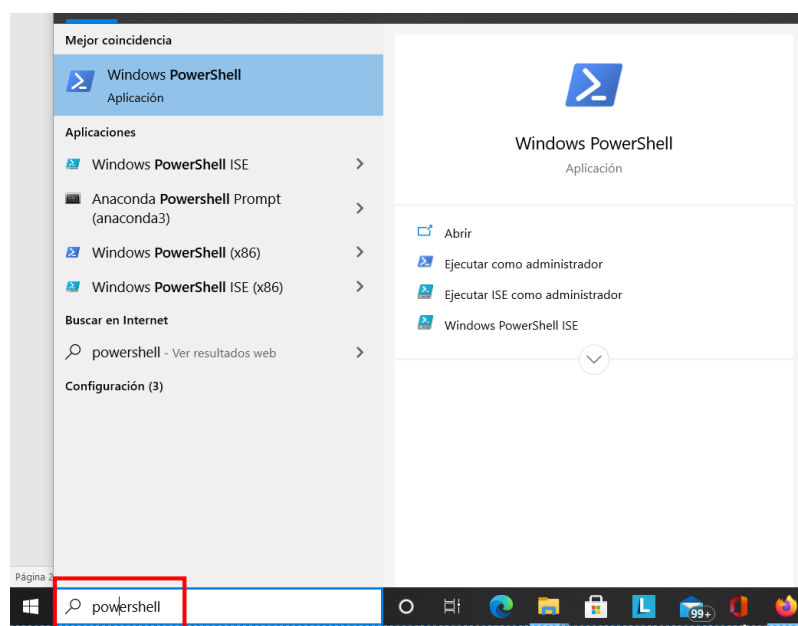
Una vez clicado en el botón señalado, automáticamente se descargará el ejecutable del programa.



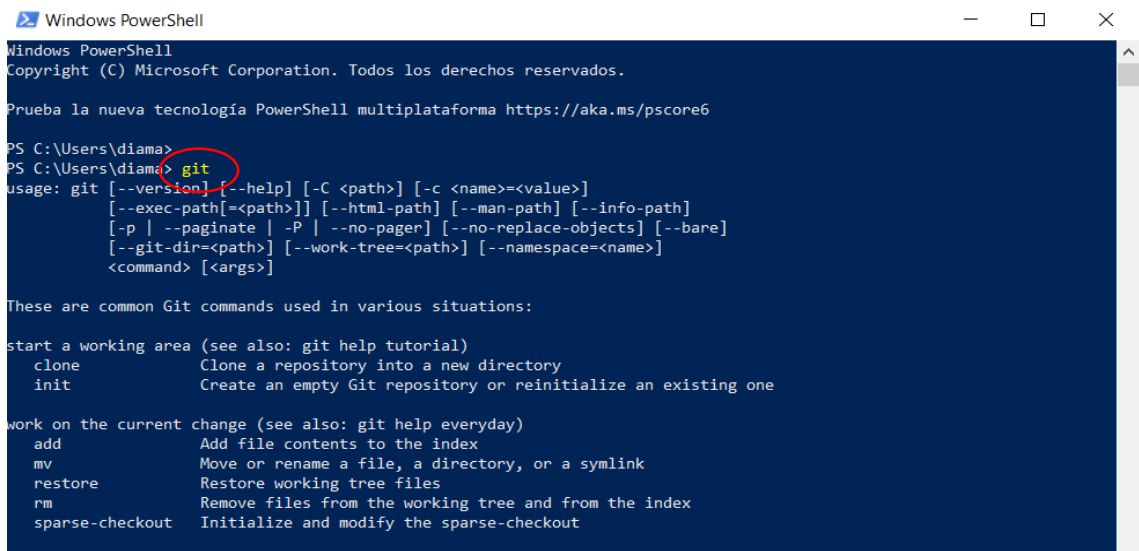
Cuando la descarga finalice, hacemos doble clic en el ejecutable y se procederá a la instalación automática. En este caso, la instalación se realizará siguiendo la configuración por defecto, por lo que **no tenemos que cambiar ningún parámetro** dentro del proceso.

***IMPORTANTE

Si queremos comprobar que el programa se ha instalado correctamente en el equipo, podemos comprobarlo usando la herramienta Powershell de nuestro ordenador. Para ejecutar esta aplicación únicamente debemos escribir en el buscador de Windows "powershell" y clicar en la coincidencia resultante.



Una vez abierto el programa, debemos de escribir el comando **git** y si hemos seguido bien los pasos, la aplicación nos devolverá la siguiente respuesta.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\diana> git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
      [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      <command> [<args>]

These are common Git commands used in various situations:

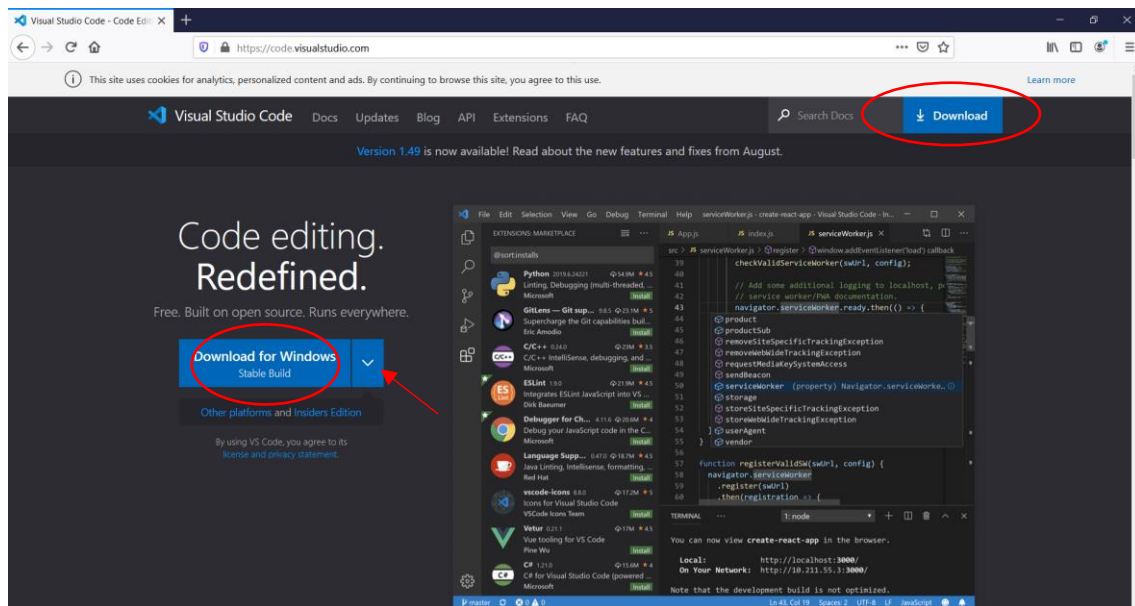
start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv         Move or rename a file, a directory, or a symlink
  restore    Restore working tree files
  rm         Remove files from the working tree and from the index
  sparse-checkout  Initialize and modify the sparse-checkout
```

2.2 DESCARGA DE VISUAL STUDIO CODE

El otro programa accesorio para GitHub es **Visual Studio Code**. Se trata de un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Este programa nos permitirá ejecutar comandos del programa Git, instalado anteriormente, para realizar ordenes desde el equipo local al repositorio de GitHub.

Para descargar este programa nos debemos de dirigir a la dirección <https://code.visualstudio.com/> desde el navegador instalado en el equipo, el cual nos redireccionará a la página oficial.



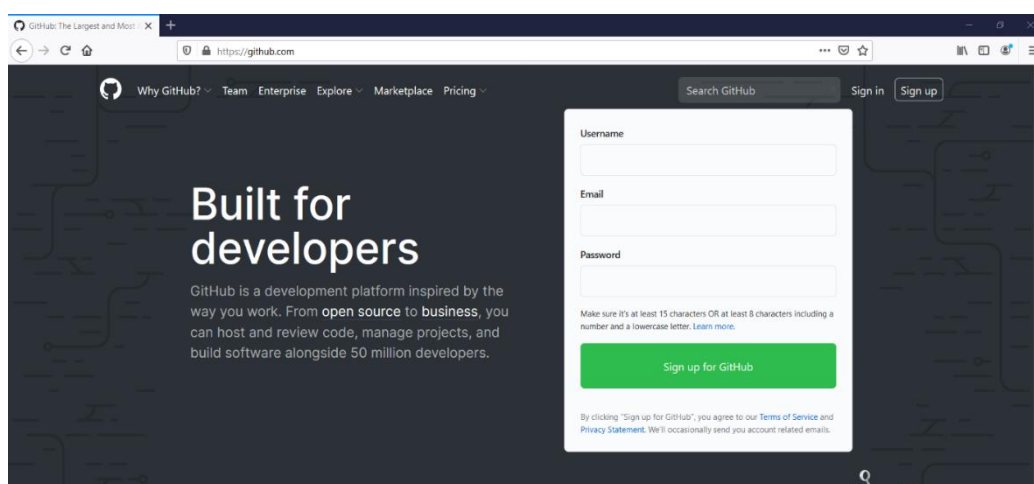
Después de clicar en el botón señalado, automáticamente se descargará el ejecutable del programa.

En el caso que en el equipo con el que estamos trabajando no tenga como sistema operativo Windows, hay disponible una pestaña donde se puede descargar para otras plataformas, como las variantes de Linux o MacOS.

Cuando la descarga finalice, hacemos doble clic en el ejecutable y se procederá a la instalación automática. En este caso, la instalación se realizará siguiendo la configuración por defecto, por lo que **no tenemos que cambiar ningún parámetro** dentro del proceso.

3. CREAR CUENTA EN GitHub

El siguiente paso en este manual (puede realizarse indistintamente en cualquier momento del proceso) será la creación en una cuenta en GitHub. Para ello, nos dirigiremos a la dirección <https://github.com/> desde nuestro navegador predeterminado.




Una vez en la pagina oficial, debemos registrarnos. Como se ve en la imagen, para completar el proceso de registro, debemos de elegir un nombre de usuario, un email de contacto y una contraseña. Cuando finalicemos este paso, clicamos sobre **Sign Up for GitHub** (regístrate en GitHub).

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *


Repository name *

 andresASIR ▾


/ 000_prueba ✓

Great repository names are short and memorable. Need inspiration? How about [stunning-bassoon?](#)

Description (optional)

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

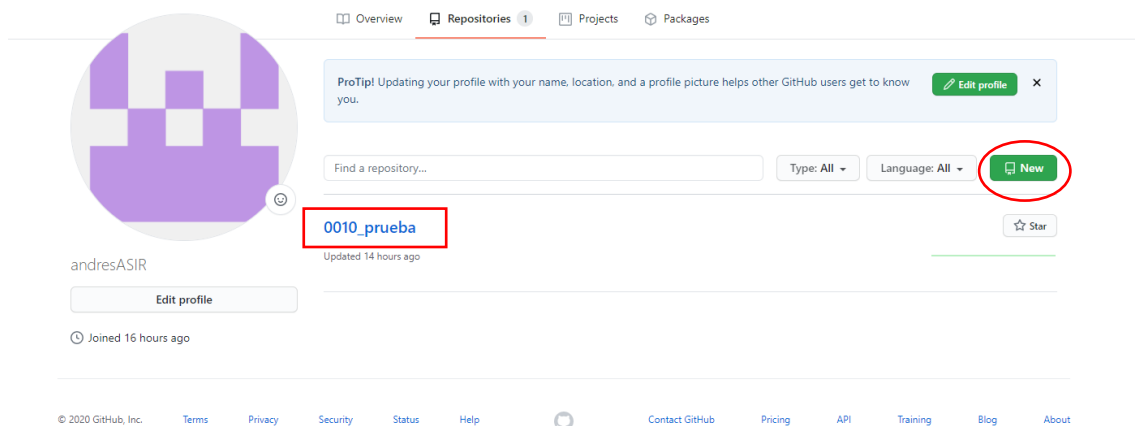
☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

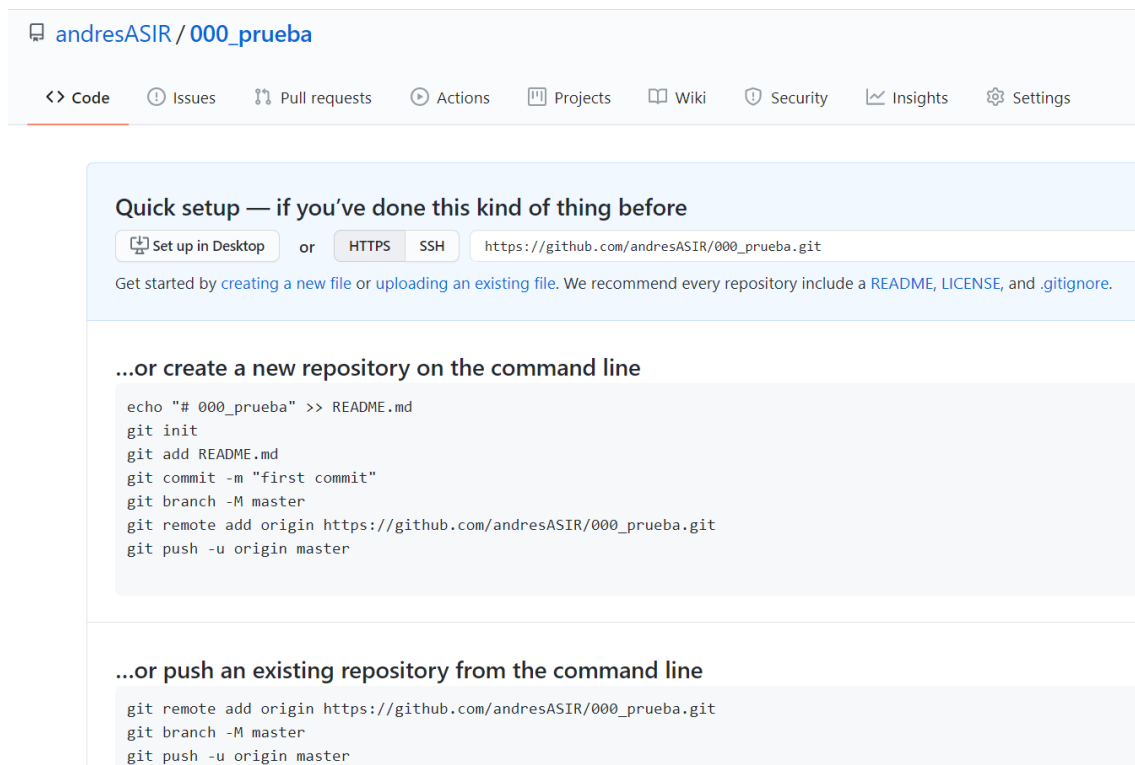
Create repository

A continuación, debemos de crear nuestro primer repositorio. Para ello debemos de elegir un nombre, seleccionar si queremos que este repositorio sea público o privado (lo cual afectará a la interacción que tendrán los usuarios sobre los archivos alojados) y finalizar el proceso clicando en “Create repository”.

Una vez que nuestro primer reposito este creado tendremos una pantalla como esta.



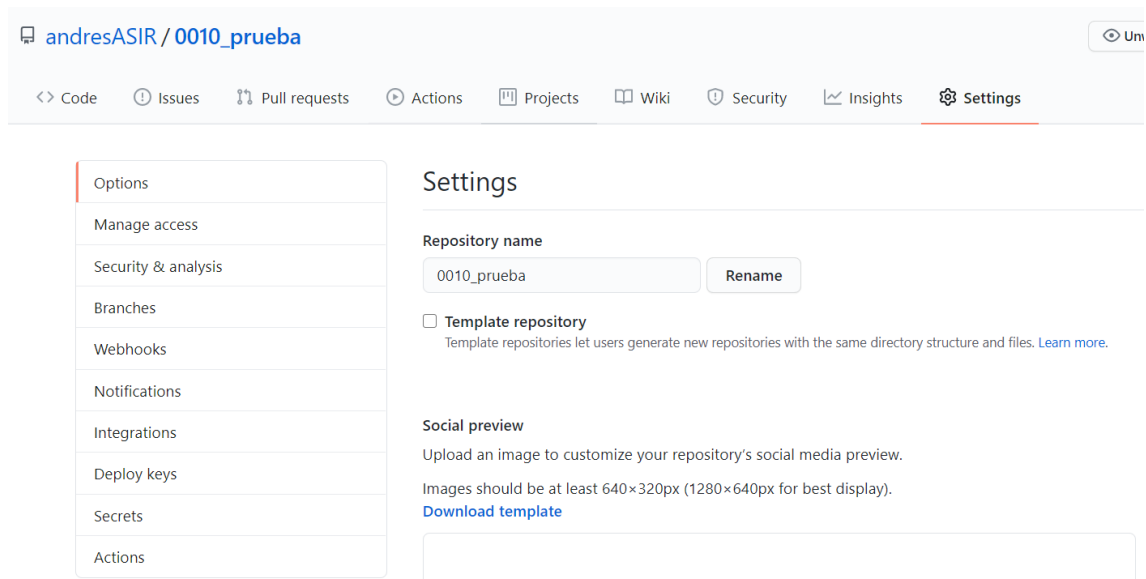
Para entrar dentro del repositorio, debemos de clicar encima del nombre. Además, si queremos crear otro repositorio, solo tenemos que clicar “New” en la parte superior derecha.



Lo siguiente que nos aparecerá en pantalla será una lista de comandos cuya finalidad es la configuración y el enlace entre una carpeta del equipo local y el repositorio de GitHub recientemente creado en la nube. Esta serie de comandos será la que usaremos en el programa **Visual Studio Code** anteriormente descargado.

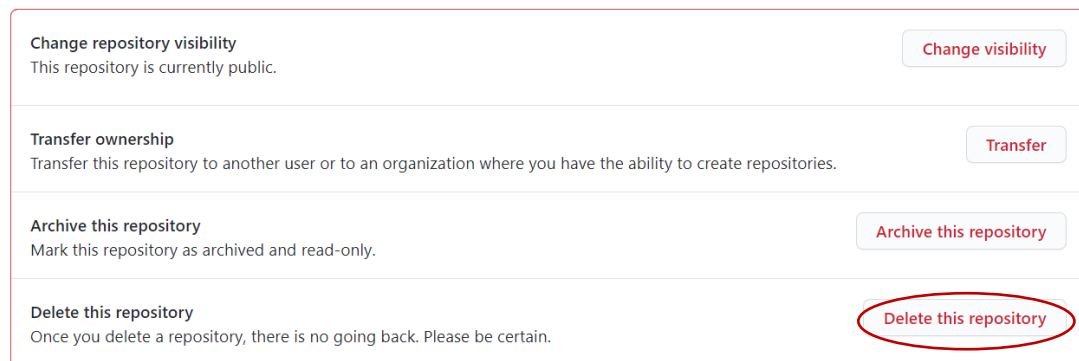
***IMPORTANTE

Si posteriormente queremos eliminar el repositorio o cambiarle el nombre, debemos dirigirnos a la pestaña “Settings”.



Al final de esa misma ventana:

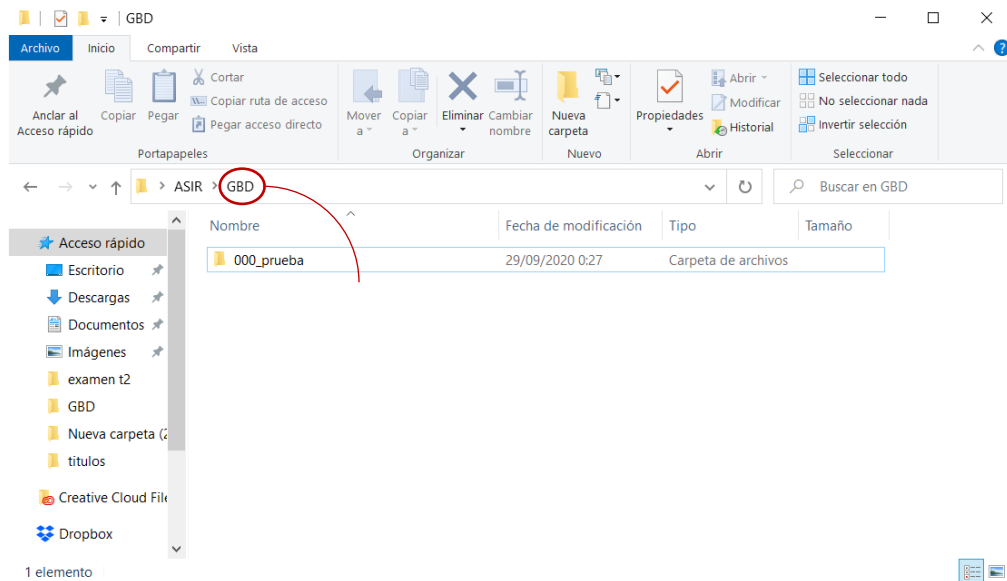
Danger Zone



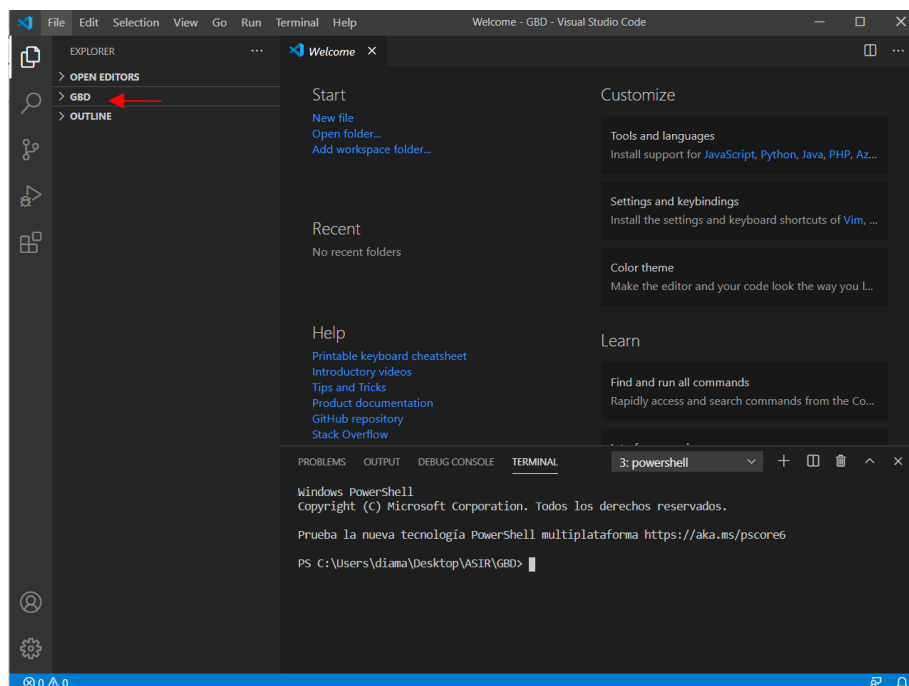
Además de su eliminación, en este apartado podemos modificar configuraciones de privacidad y seguridad entre otros.

4. EMPAREJAMIENTO DE CARPETA LOCAL CON EL REPOSITORIO DE GitHub

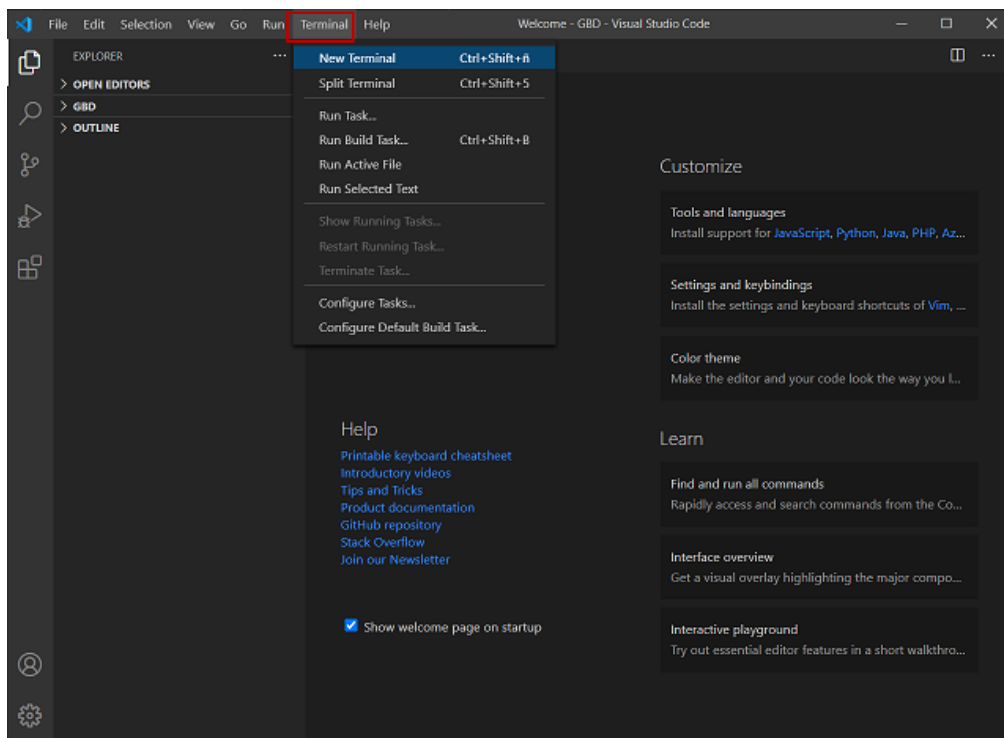
Para realizar el emparejamiento entre carpetas, debemos de ejecutar el programa Visual Studio Code como mencionamos en el apartado anterior. Seguidamente necesitamos crear una carpeta la cual se quiere emparejar.



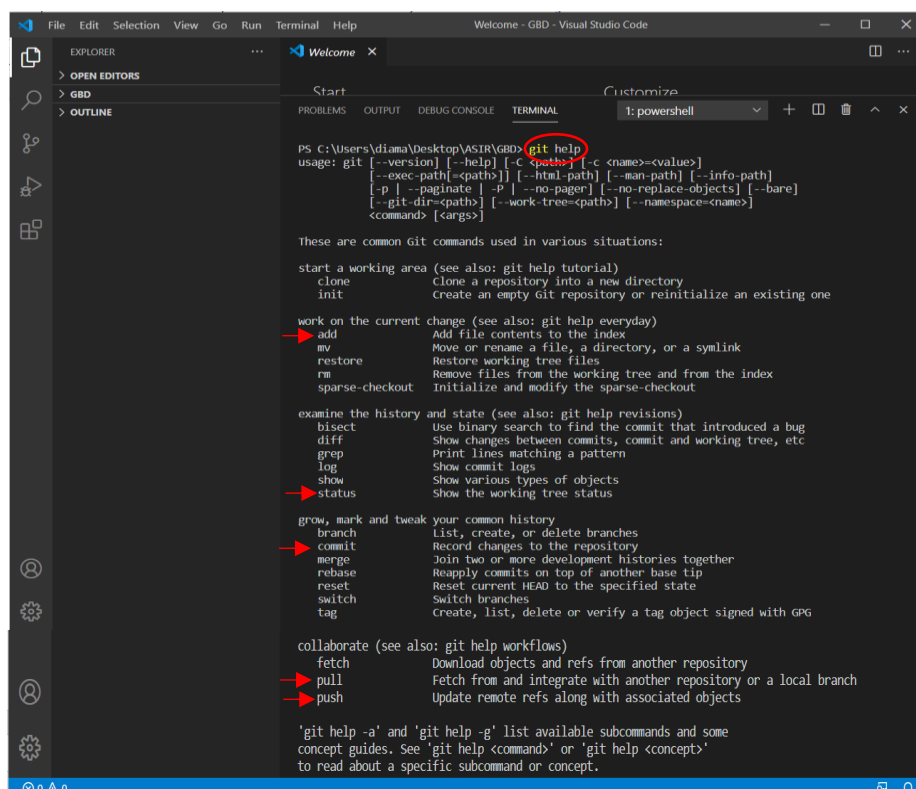
La carpeta que vamos a emparejar es la **“000_prueba”** pero la que vamos a introducir en el Visual Studio Code es la carpeta que en la ruta esta un escalón superior. Dicha carpeta la arrastramos con el cursor dentro de la interfaz de Visual Studio Code y deberíamos de ver esta imagen.



Como se puede apreciar en la imagen, la carpeta antes seleccionada se ha introducido en el programa. El siguiente paso es abrir la pestaña “Terminal” y seguidamente “New Terminal”.

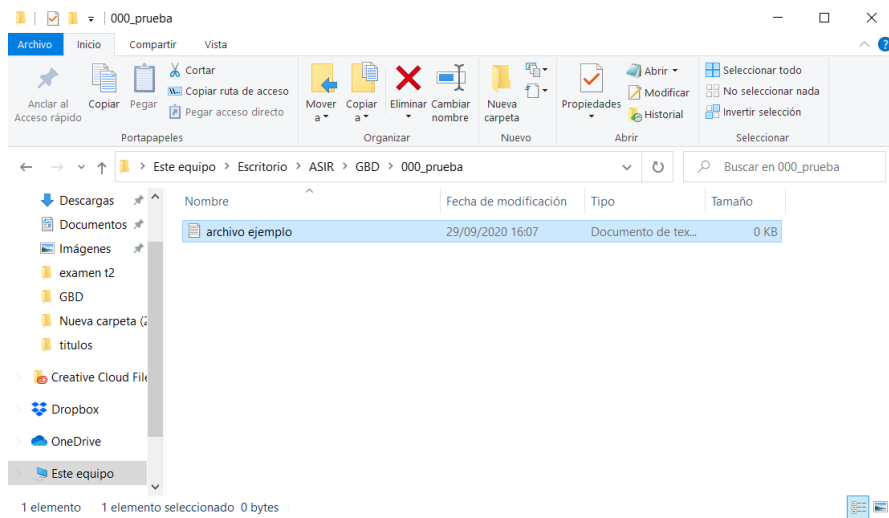


En este punto de nuestro procedimiento es cuando entra en acción el programa Git instalado anteriormente. En el prompt del terminal podemos empezar escribiendo el comando **git help**, teniendo como respuesta la explicación de los diferentes comandos que aparecen en el apartado anterior “[CREAR CUENTA EN GitHub](#)”.



5. ENVIO DE ARCHIVOS A UN REPOSITORIO GitHub

Una vez la carpeta preparada, vamos a proceder con el envío de un archivo desde el equipo local al repositorio de GitHub.



A continuación, escribimos la siguiente secuencia de comandos:

- * Con **git init** arrancamos el programa.
- * El paso siguiente es la comprobación del archivo, lo cual nos aparece que ha sido modificado (**git status**).

```
File Edit Selection View Go Run Terminal Help
archivo ejemplo.txt - GBD - Visual Studio Code

EXPLORER
  > OPEN EDITORS
    GBD
      000_prueba
        archivo ejemplo.txt

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\diama\Desktop\ASIR\GBD> git init
Reinitialized existing Git repository in C:/Users/diama/Desktop/ASIR/GBD/.git/
PS C:\Users\diama\Desktop\ASIR\GBD> git status
On branch master
Your branch is up to date with 'origin/master'.

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:   000_prueba/archivo ejemplo.txt

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\diama\Desktop\ASIR\GBD> git add .
PS C:\Users\diama\Desktop\ASIR\GBD> git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   000_prueba/archivo ejemplo.txt

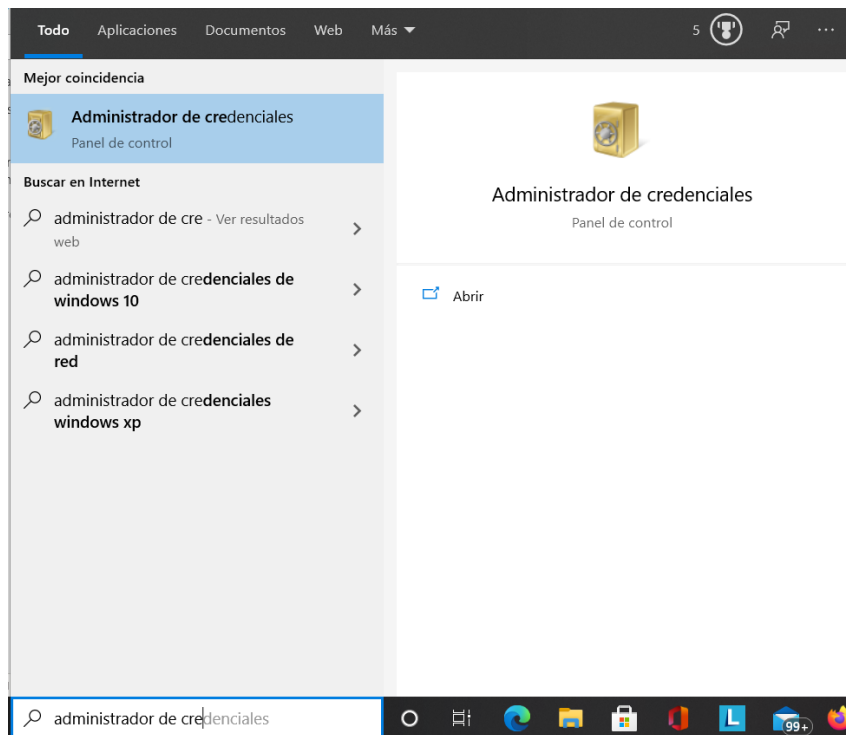
PS C:\Users\diama\Desktop\ASIR\GBD> git commit -m "primer doc"
[master 4704568] primer doc
1 file changed, 3 insertions(+)
PS C:\Users\diama\Desktop\ASIR\GBD> git remote add origin https://github.com/andresASIR/000_prueba.git
fatal: remote origin already exists.
PS C:\Users\diama\Desktop\ASIR\GBD> git push -u origin master
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (8/8), 541 bytes | 135.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
to https://github.com/andresASIR/000_prueba.git
 * [new branch] master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
PS C:\Users\diama\Desktop\ASIR\GBD>
```

- * El programa nos aconseja usar **git add** para preparar este archivo modificado para ser mandado.
- * El siguiente paso es usar **git commit -m** para guardar los cambios en el repositorio. A este comando se le puede añadir una etiqueta para futuros cambios.

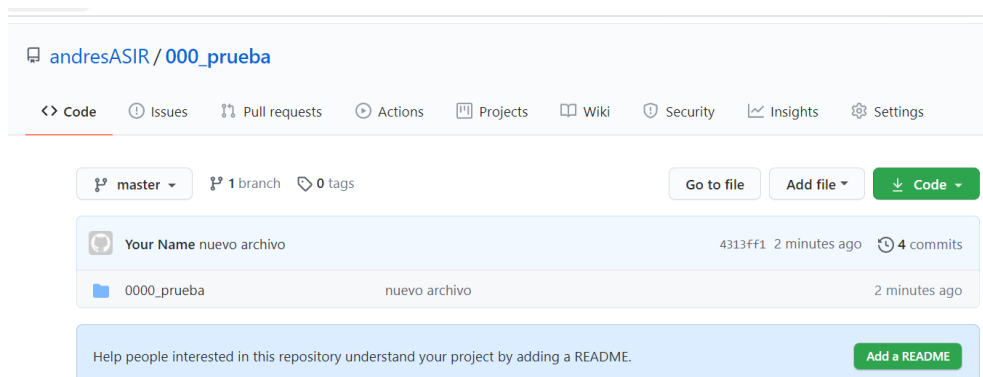
* Para enviar el archivo debemos de poner la dirección https de nuestro repositorio en GitHub en el comando ***git remote add origin https://.....***

* Por último, el comando ***git push -v origin master***. Una vez realizada esta acción, nos debemos de autenticar en una ventana emergente con la cuenta de GitHub.

La primera vez que hagamos esto, nos saldrá una ventana emergente para acceder a GitHub. Esta cuenta queda grabada en las credenciales del sistema, por lo que para utilizar otra cuenta en un mismo PC debemos de eliminar la anterior. Para acceder a las credenciales del sistema solo tenemos que poner en el buscador de Windows “administrador de credenciales”

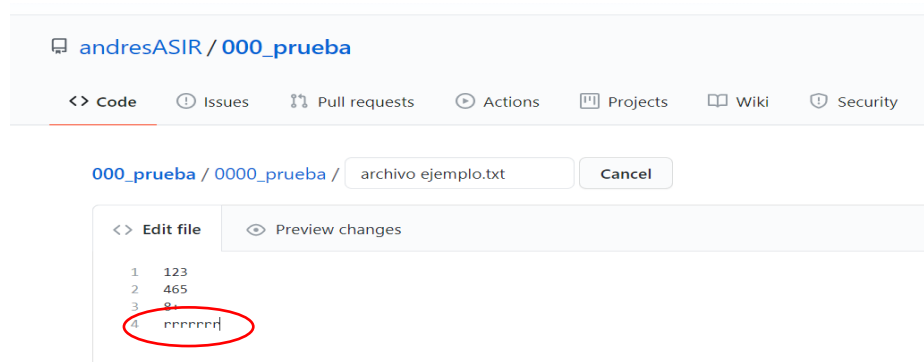


Si nos metemos en nuestro repositorio de GitHub, veremos que el archivo se ha subido correctamente.

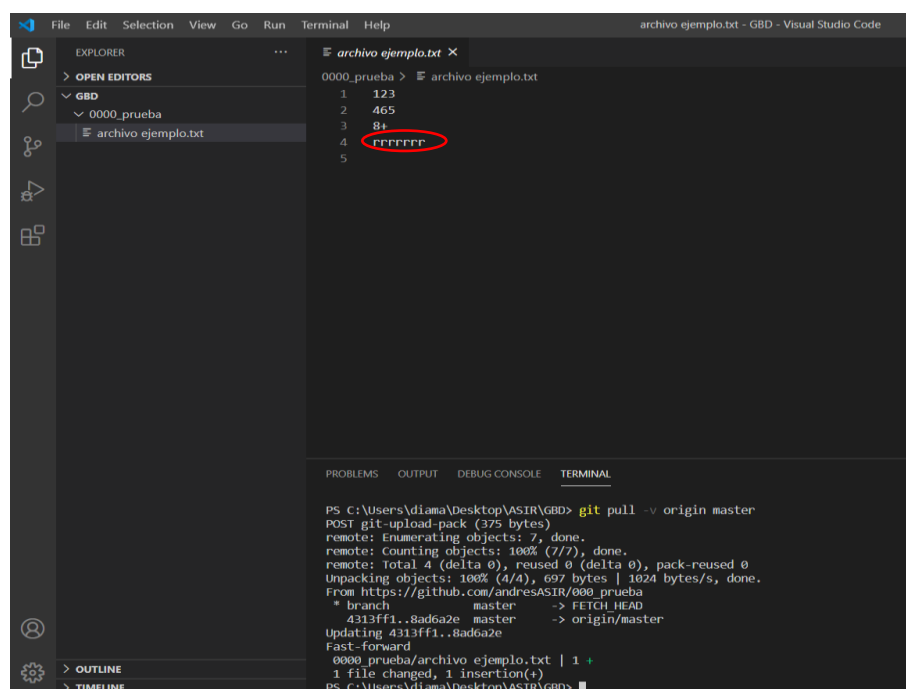


6. ACTUALIZACIÓN DE ARCHIVOS EN LÍNEA

El archivo subido puede modificarse en línea por otros usuarios, por lo que si queremos actualizar nuestro archivo en el equipo local con los cambios realizados, debemos de introducir el comando **“git pull -v origin master”**.



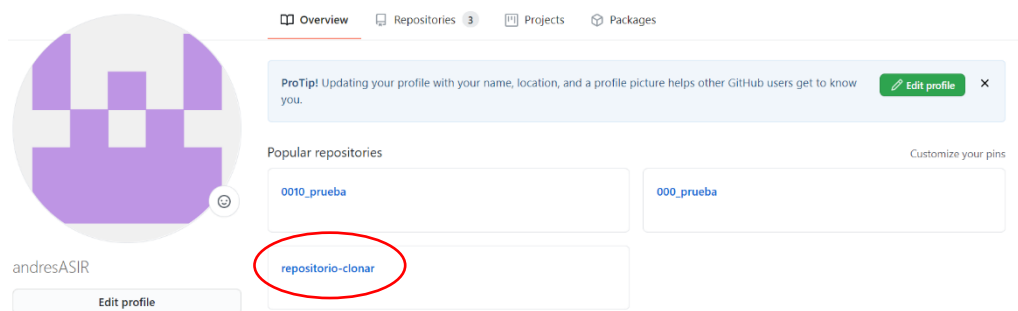
Ya tenemos los cambios guardados en nuestro archivo local.



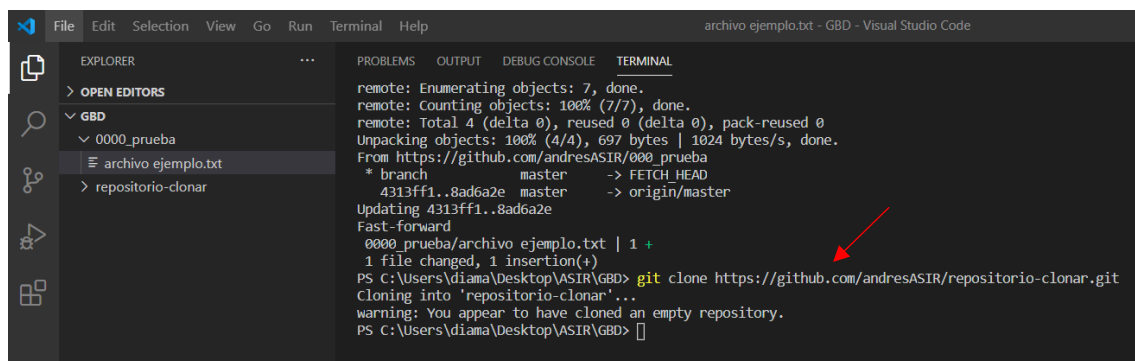
7. CLONAR REPOSITORIOS

Por último, si queremos clonar un repositorio en la nube, debemos de usar el comando **git clone https: //**

Elegimos el repositorio que queremos clonar:



A continuación, nos vamos a Visual Studio Code e introducimos el comando.



```
File Edit Selection View Go Run Terminal Help
archivo ejemplo.txt - GBD - Visual Studio Code

EXPLORER
> OPEN EDITORS
  GBD
    0000_prueba
      archivo ejemplo.txt
    repositorio-clonar

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), 697 bytes | 1024 bytes/s, done.
From https://github.com/andresASIR/000_prueba
* branch      master      -> FETCH_HEAD
  4313ff1..8ad6a2e master    -> origin/master
Updating 4313ff1..8ad6a2e
Fast-forward
 0000_prueba/archivo ejemplo.txt | 1 +
 1 file changed, 1 insertion(+)
PS C:\Users\diana\Desktop\ASIR\GBD> git clone https://github.com/andresASIR/repositorio-clonar.git
Cloning into 'repositorio-clonar'...
warning: You appear to have cloned an empty repository.
PS C:\Users\diana\Desktop\ASIR\GBD>
```