

# Git y Github

## Índice

### 1º Introducción

### 2º Registrarse en Github

### 3º Programas necesarios

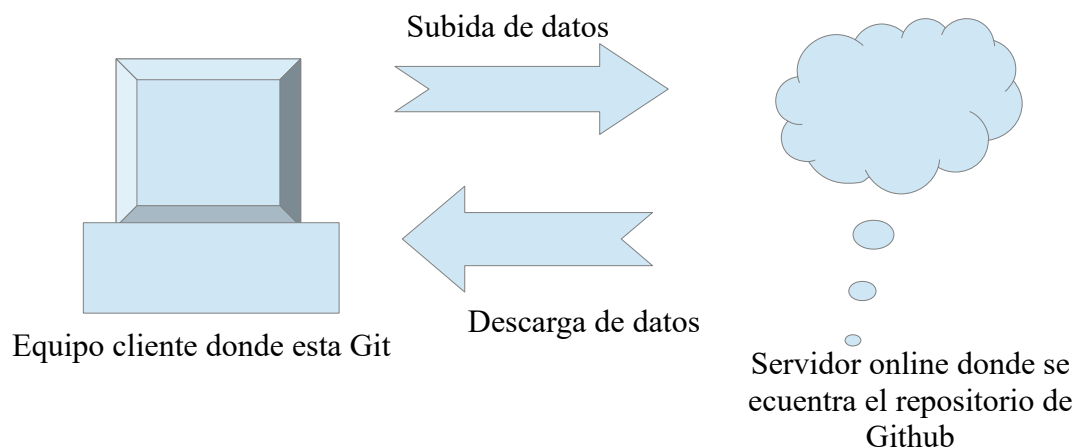
### 4º Descarga e instalación

### 5º Comandos de git

### 6º Práctica

## 1º Introducción

Empecemos distinguiendo entre Git y Github, para empezar Git es el programa que tendremos que instalar en nuestro equipo y es el que actuará de cliente entre el equipo y el servidor donde está alojado Github, mientras que Github es un repositorio online donde podremos subir información de forma pública para que otros usuarios puedan utilizarla para verse beneficiados de ellos, así mismo existen versiones de pago por si se prefiere que solo cierto grupo de personas accedan a dicho repositorio.



## 2º Registrarse en Github

Como ya hemos dicho tendremos que registrarnos en la plataforma de la herramienta Github para poder tener sus servicios.

<https://github.com>

En dicho enlace, nos registraremos o logearemos dependiendo de si ya poseemos una cuenta de Github, en caso de no tener cuenta clicaremos sobre "Sign up" para registrarnos

The screenshot shows the GitHub website's sign-up interface. On the left, there's a dark sidebar with the GitHub logo and navigation links: "Why GitHub?", "Team", "Enterprise", "Explore", "Marketplace", and "Pricing". The main content area has a dark background with the text "Built for developers" and a description of GitHub as a development platform. On the right, there's a white sign-up form with fields for "Username", "Email", and "Password". Below the password field, there's a note about password requirements and a "Learn more" link. A green button labeled "Sign up for GitHub" is at the bottom of the form. At the very bottom, there's a small disclaimer about agreeing to the Terms of Service and Privacy Statement.



Join GitHub

# Create your account

Username \*

Email address \*

Password \*


Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter.  
[Learn more.](#)


**Email preferences**☐ Send me occasional product updates, announcements, and offers.**Verify your account**

Solucione este rompecabezas para que sepamos que es una persona real

Verificar

Tras verificar en el correo que ya tenemos cuenta, accedemos a ella para empezar a trabajar.

 Search or jump to... [7] [Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#) 🔔 + 👤

**Repositories** New  
  
 [jairo56/0010\\_prueba](#)

**Working with a team?**  
GitHub is built for collaboration. Set up an organization to improve the way your team works together, and get access to more features.  
[Create an organization](#)

## Learn Git and GitHub without any code!


Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.


[Read the guide](#) [Start a project](#)

### Discover interesting projects and people to populate your personal news feed.

Your news feed helps you keep up with recent activity on repositories you [watch](#) and people you [follow](#).

[Explore GitHub](#)

 **ProTip!** The feed shows you events from people you [follow](#) and repositories you [watch](#).

 [Subscribe to your news feed](#)

Una vez dentro, nos detenemos ya que para seguir con esta práctica primero tendremos que ver el siguiente apartado

### **3º Programas necesarios**

Para seguir necesitaremos descargar el programa cliente de Git, el cual encontreis en el siguiente enlace.

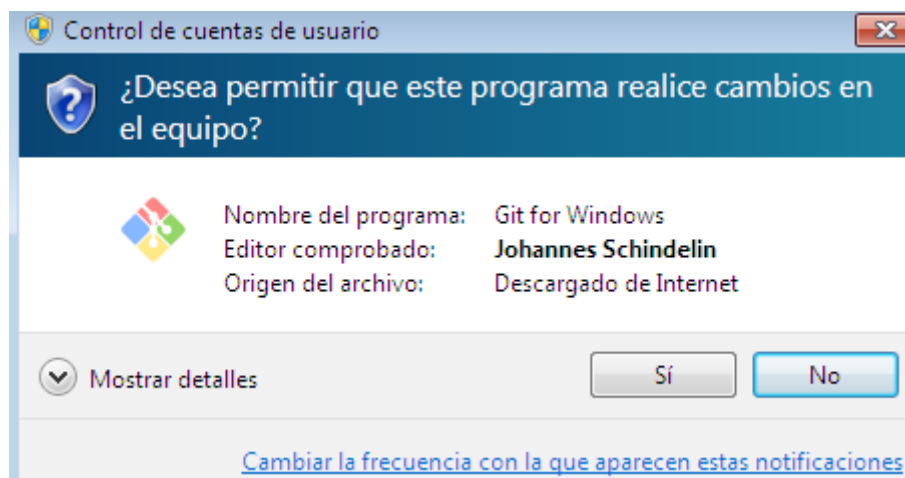
<https://git-scm.com/downloads>

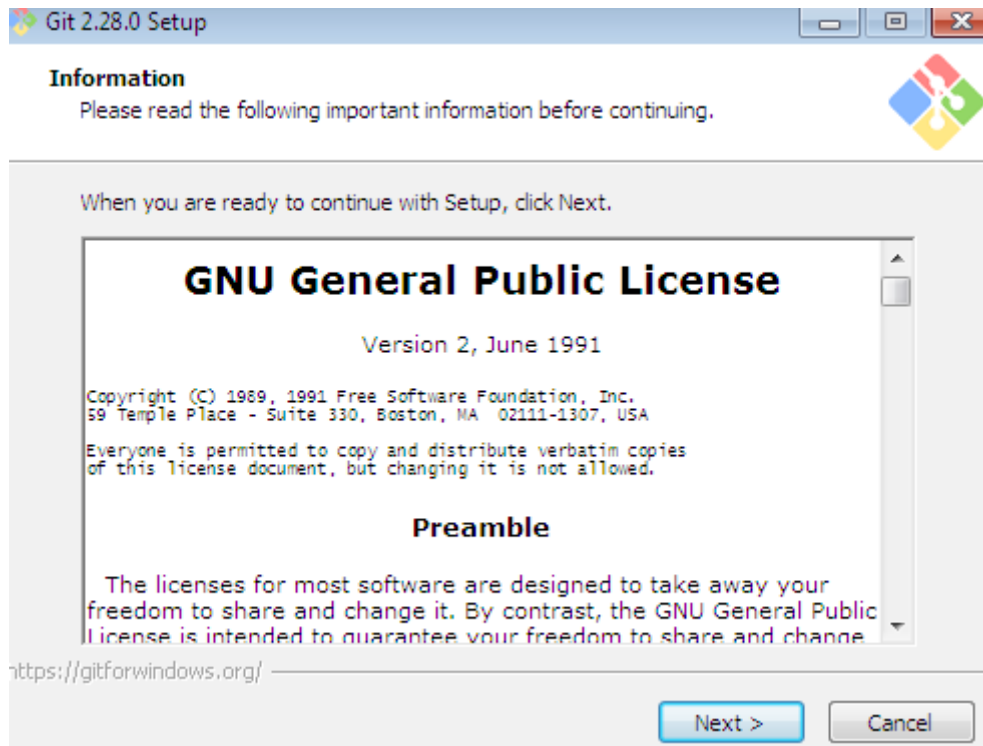
Descargamos la ultima versión e instalamos el programa

Lo ejecutamos

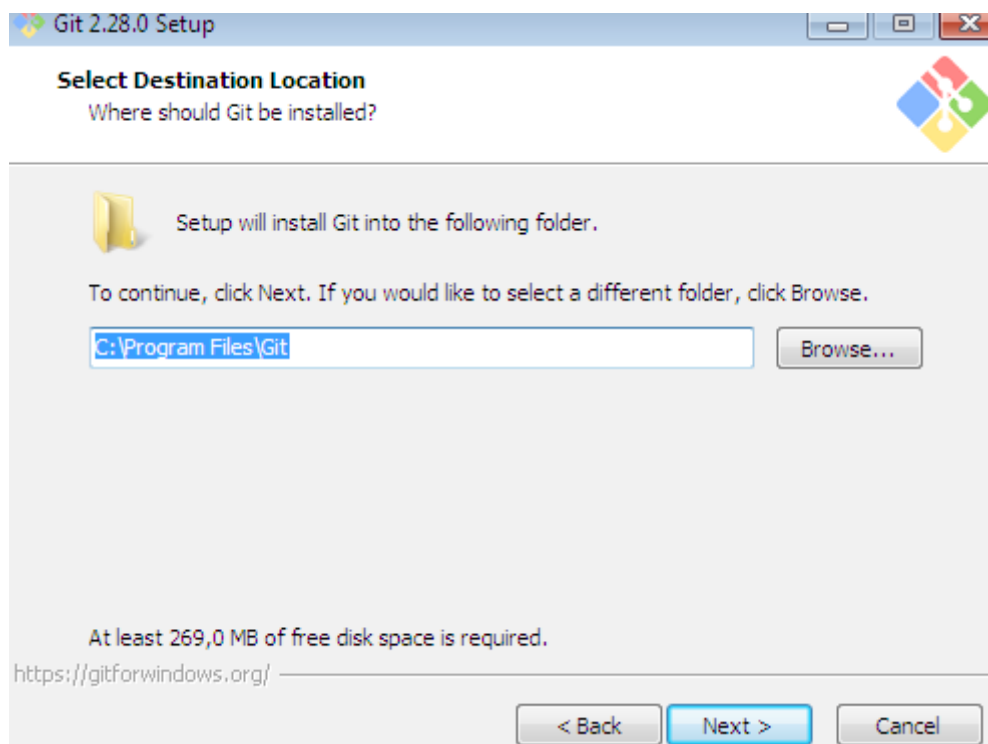


Y le damos permiso para que realice cambios.

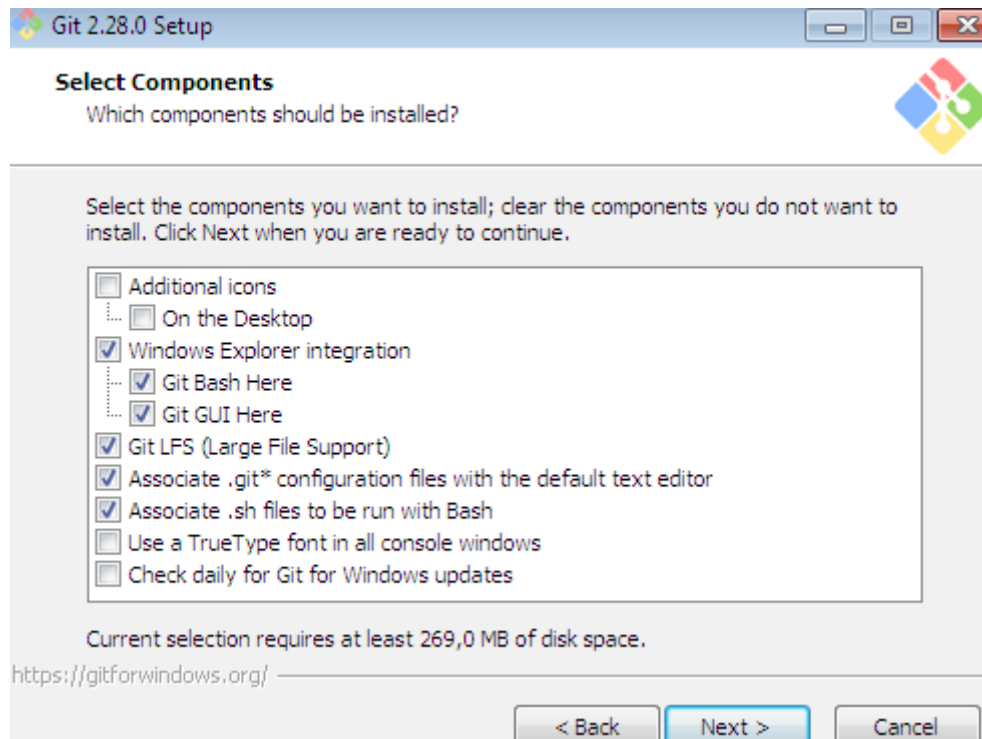




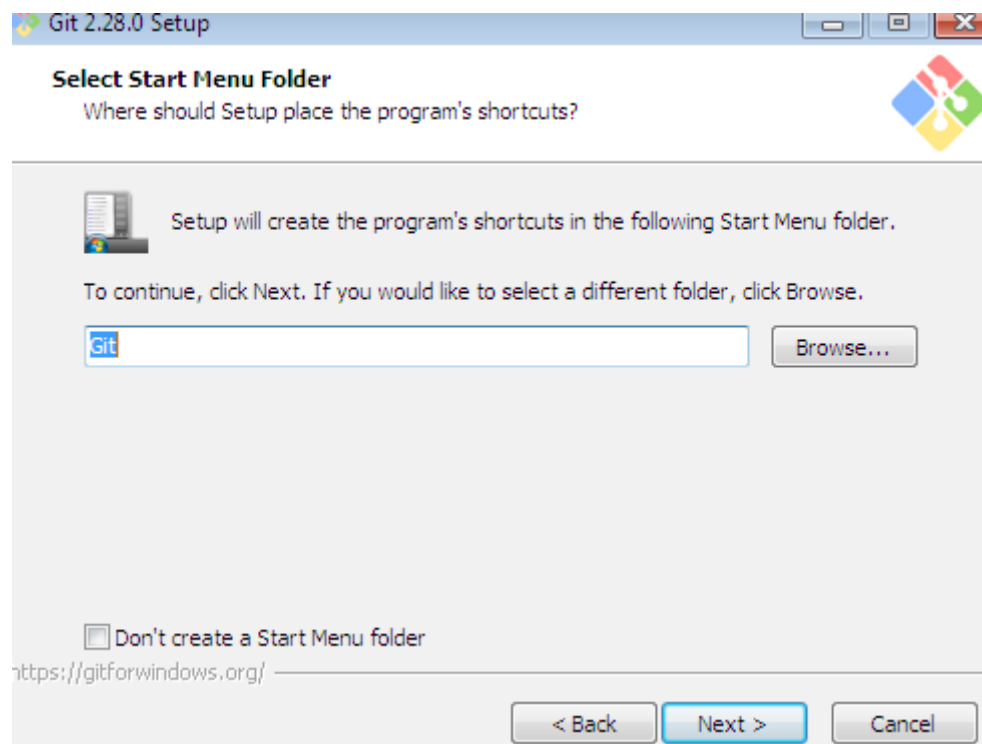
Le damos a siguiente, elegimos la ruta donde se instalara, yo dejare la que está por defecto



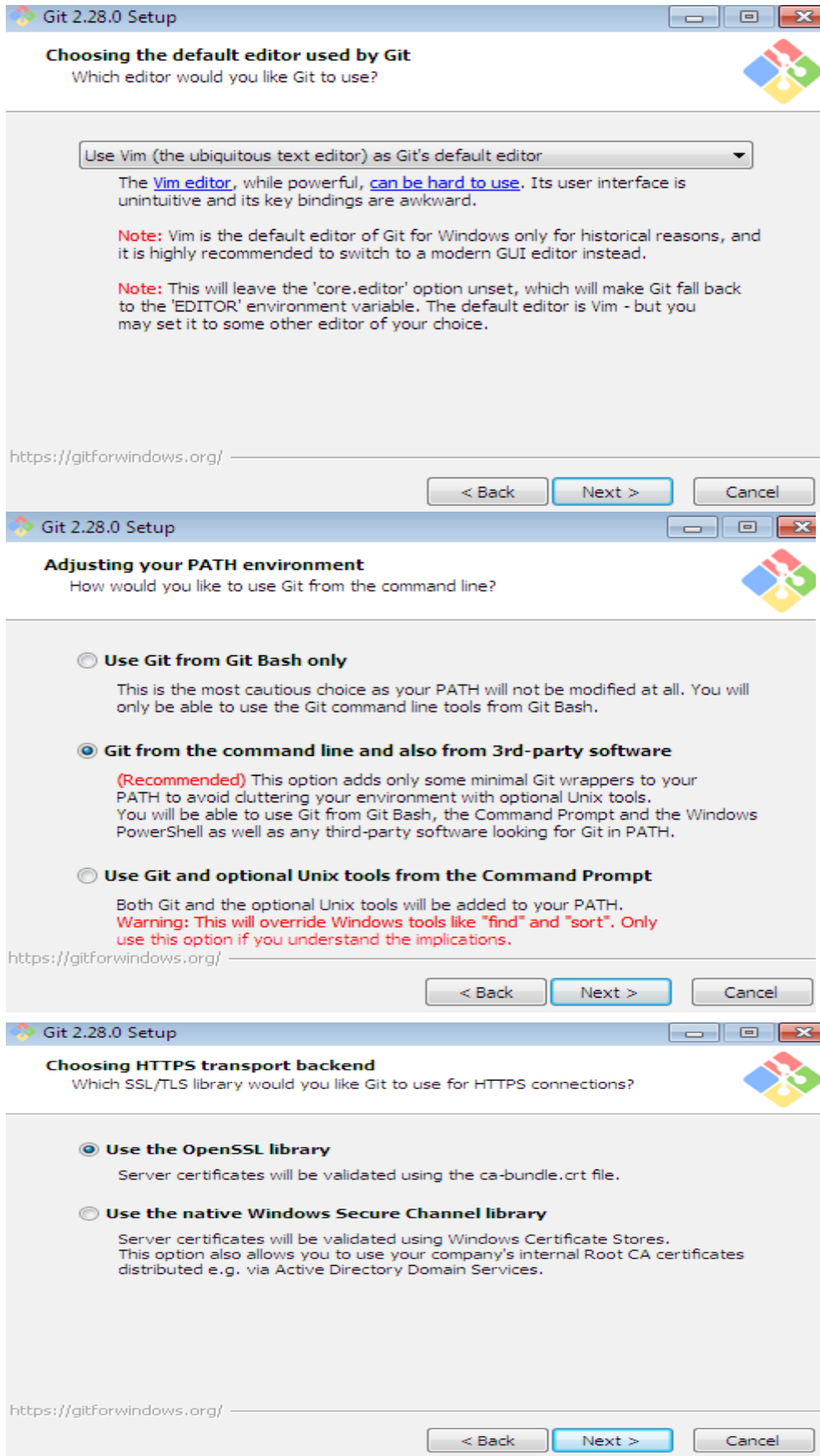
Aquí podremos elegir los complementos que podemos añadir a Git, lo dejaremos por defecto, pero para instalar uno de ellos solo deberemos relanzar el instalador y marcar la cueva opción.



A continuación nos permite añadir al menú de inicio el icono de Git para encontrarlo rápido



Las próximas ventanas hasta que nos permita instalar, le daremos a “next”



**Git 2.28.0 Setup**

**Choosing the default editor used by Git**  
Which editor would you like Git to use?

Use Vim (the ubiquitous text editor) as Git's default editor

The **Vim editor**, while powerful, **can be hard to use**. Its user interface is unintuitive and its key bindings are awkward.

**Note:** Vim is the default editor of Git for Windows only for historical reasons, and it is highly recommended to switch to a modern GUI editor instead.

**Note:** This will leave the 'core.editor' option unset, which will make Git fall back to the 'EDITOR' environment variable. The default editor is Vim - but you may set it to some other editor of your choice.

<https://gitforwindows.org/>

< Back Next > Cancel

**Git 2.28.0 Setup**

**Adjusting your PATH environment**  
How would you like to use Git from the command line?

☐ **Use Git from Git Bash only**  
This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

☒ **Git from the command line and also from 3rd-party software**  
**(Recommended)** This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.

☐ **Use Git and optional Unix tools from the Command Prompt**  
Both Git and the optional Unix tools will be added to your PATH.  
**Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.**

<https://gitforwindows.org/>

< Back Next > Cancel

**Git 2.28.0 Setup**

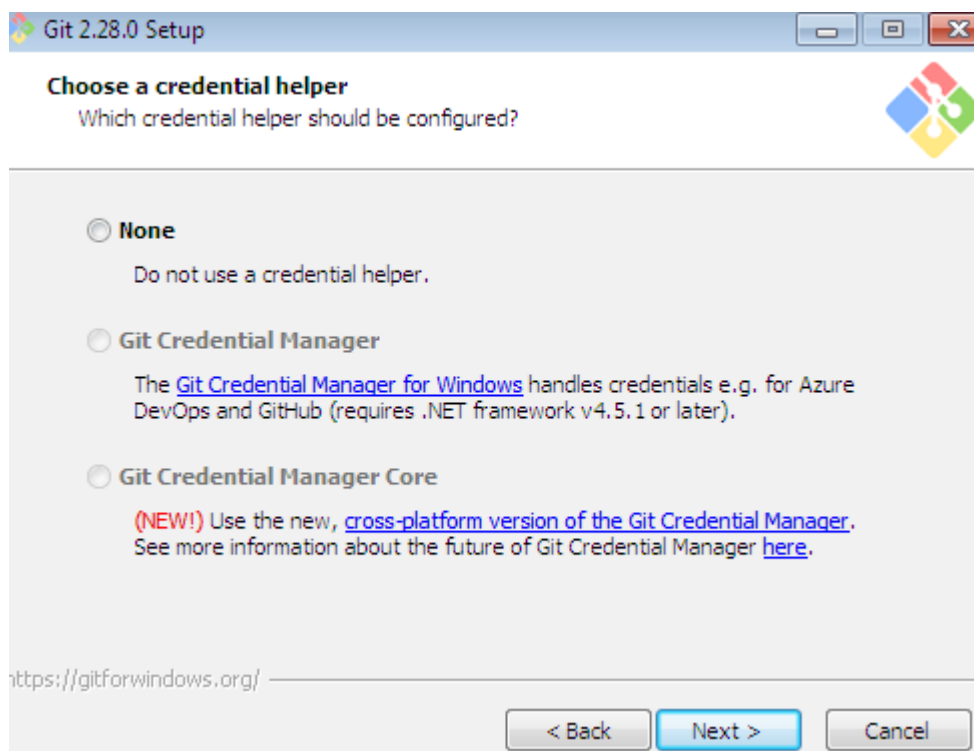
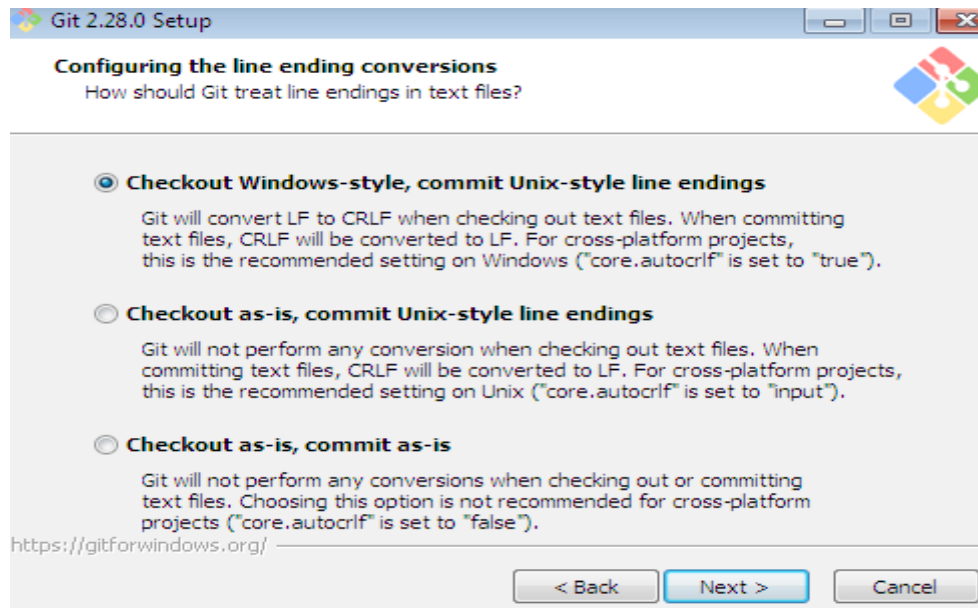
**Choosing HTTPS transport backend**  
Which SSL/TLS library would you like Git to use for HTTPS connections?

☒ **Use the OpenSSL library**  
Server certificates will be validated using the ca-bundle.crt file.

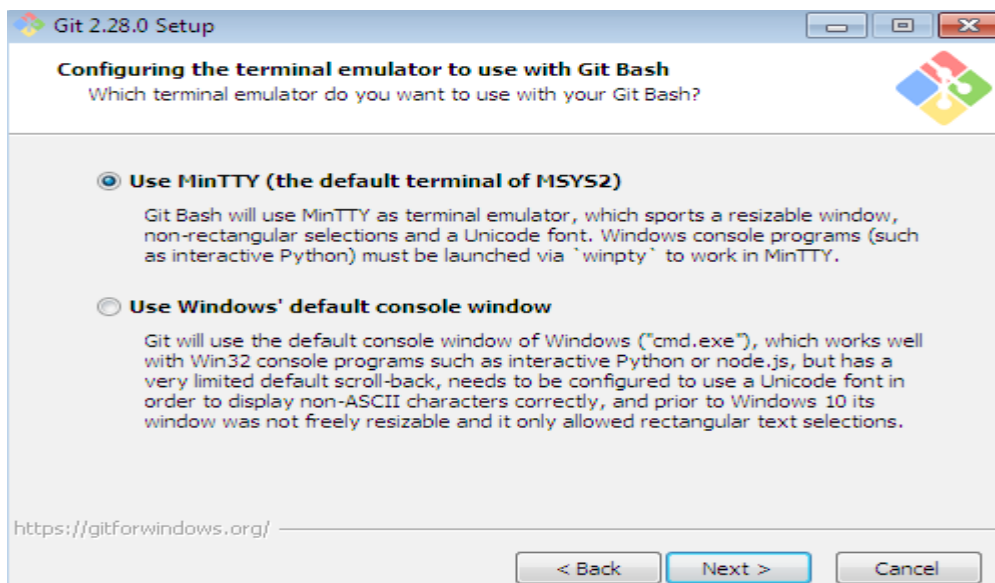
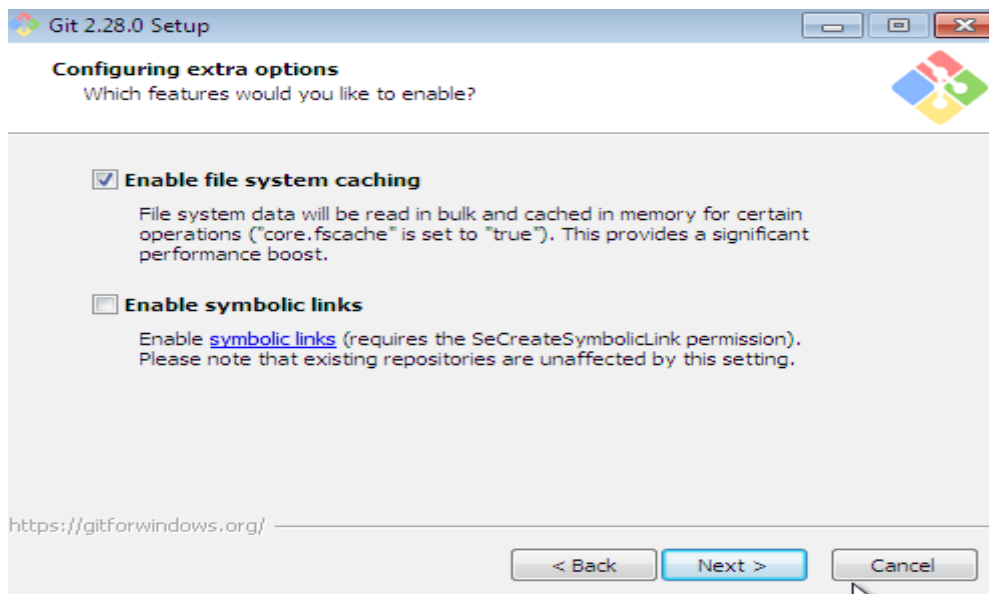
☐ **Use the native Windows Secure Channel library**  
Server certificates will be validated using Windows Certificate Stores. This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.

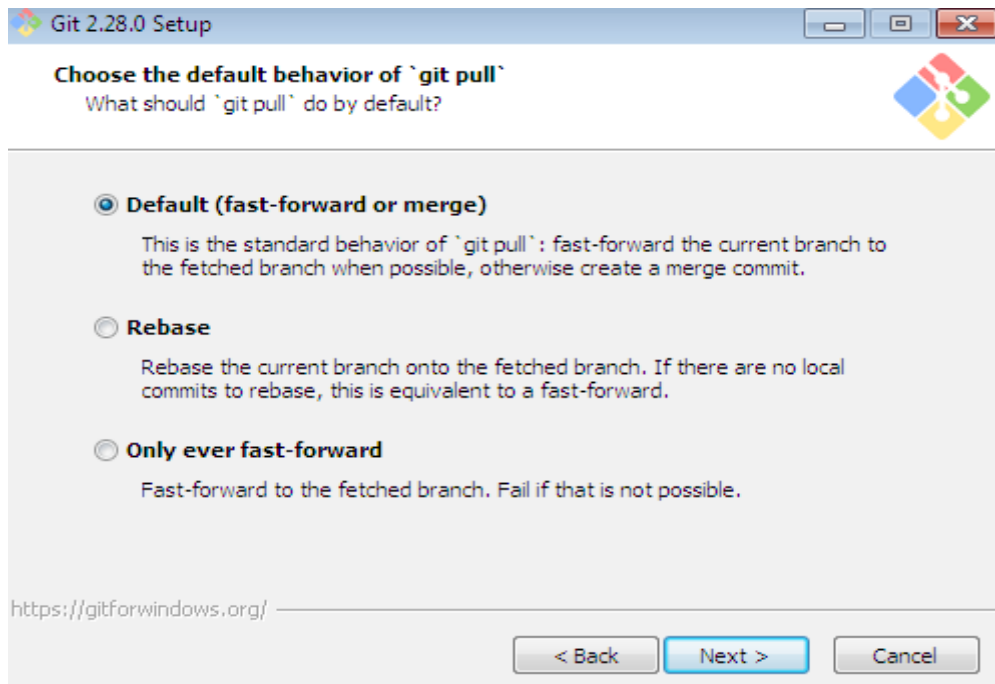
<https://gitforwindows.org/>

< Back Next > Cancel

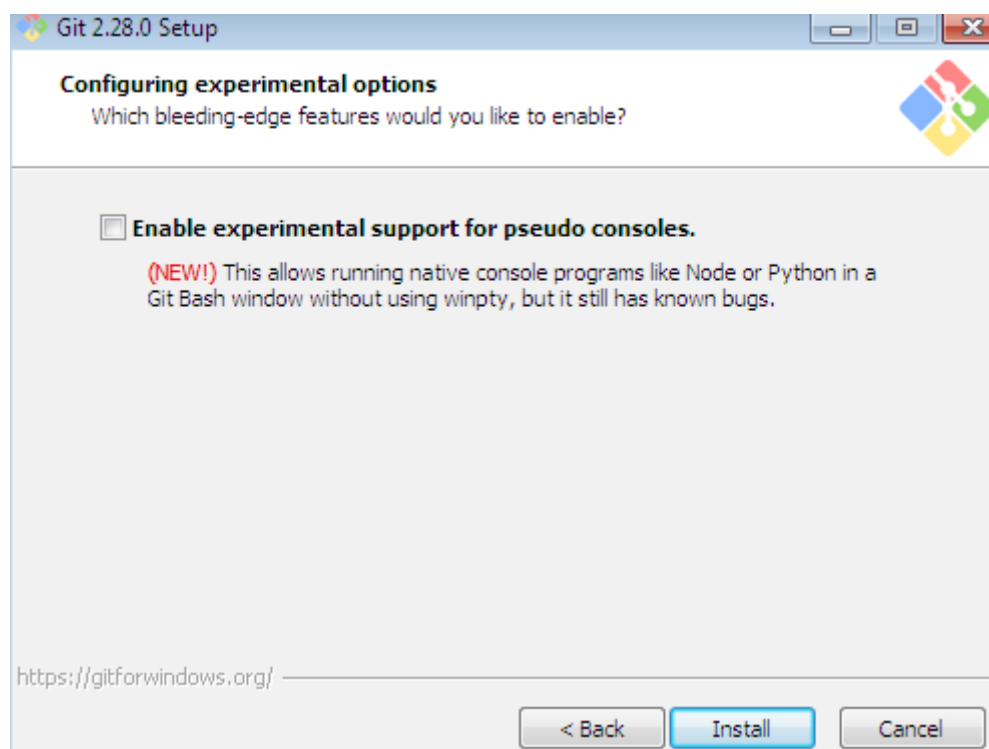


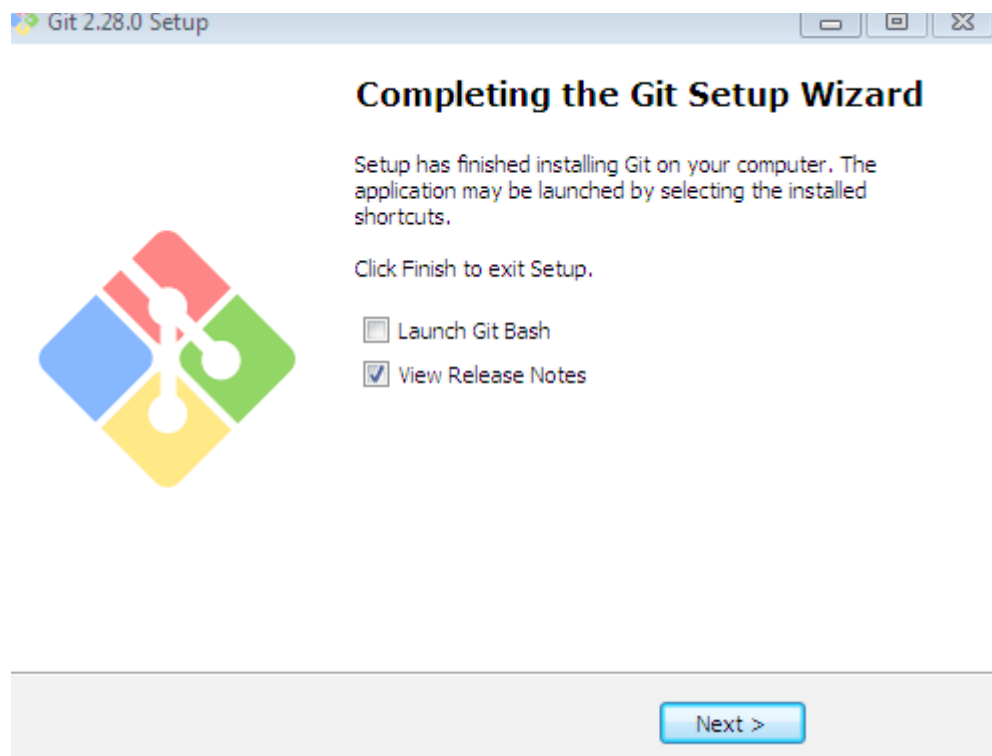
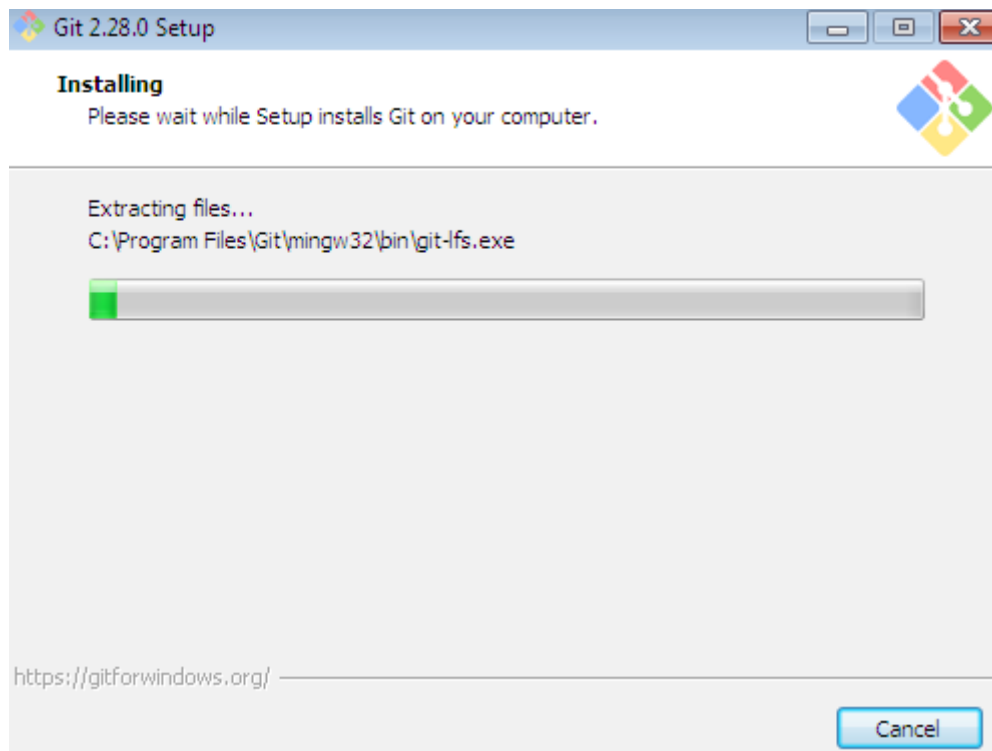






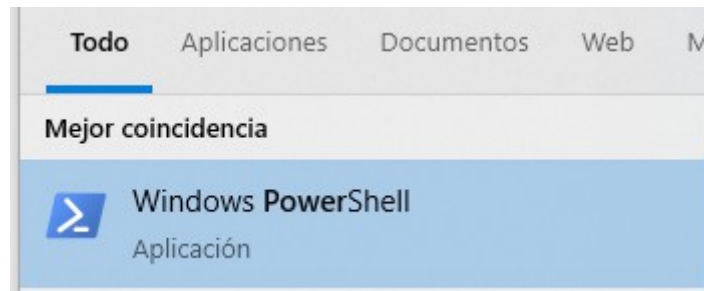
Tras lo cual instalaremos el programa.





Cuando finalice la instalación ya podremos hacer uso de Git, sin embargo antes vamos a instalar dos programas más que son necesarios.

Windows PowerShell es uno de ellos, si lo tienes ya instalado en tu equipo (puedes comprobar que lo tienes escribiendo su nombre en el botón inicio).



El otro programa que necesitaremos es “Visual Studio Code”

<https://code.visualstudio.com>

El cual descargaremos e instalaremos sin mucho problema.

Ya que es ejecutar y siguiente todo el rato hasta que se instale.

Ahora que ya tenemos todo lo necesario podremos empezar a trabajar con git y github, pero para utilizarlo de forma correcta, debemos saber que comandos posee Git y para que funciona cada uno.

## **5º Comandos de Git**

Git posee tres estados:

1- Working directory: Es donde se está trabajando con los archivos, programas y documentos

2- Staging area: es donde se agrega los archivos para su posterior guardado.

3- Repository: cuando ya esta totalmente modificado o echo los cambios a un programa o archivo se guarda finalmente en el repositorio local.

-Git init: Informa al sistema que se va a trabajar con Git lo cual crea en el lugar donde estemos situados una carpeta oculta “.git”

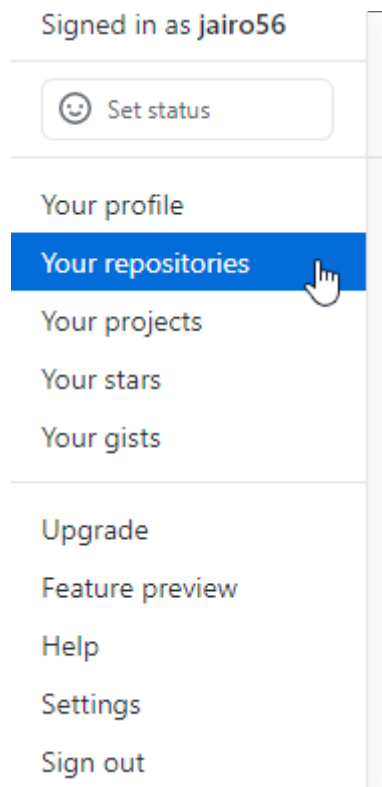
-Git add .: permite que git detecte los nuevos archivos que se encuentran en el lugar designado, ya sean nuevos o modificados.

-Git status: sirve para comprobar en que estado se encuentra el archivo.

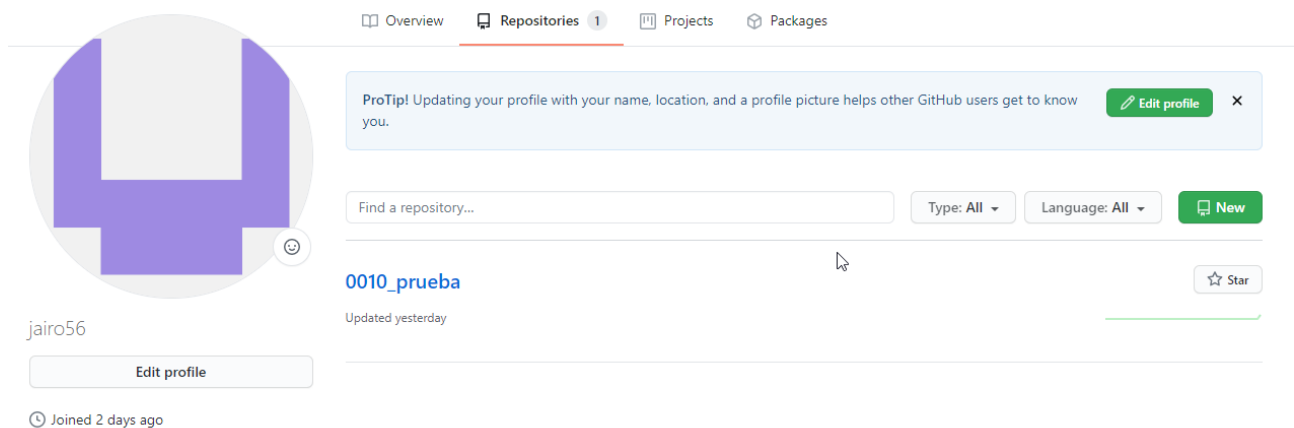
-Git commit: nos permite cambiar el estado del archivo del “staging area” al repositorio en cuestión

- Git push: se usa para subir o enviar el archivo a un repositorio remoto (Github)
- Git clone: nos permite copiar un archivo que se localiza en el servidor a nuestro equipo cliente, para trabajar en ello
- Git pull: nos permite traer los cambios que han efectuado otros trabajadores al archivo con el que se ha trabajado para que al trabajar con ellos estén actualizados.
- Git remote add origin: este comando seguido de un enlace de repositorio remoto nos permitirá enlazar dicho repositorio con la carpeta en la que estemos trabajando.

## 6º Práctica



Primero en Github crearemos un repositorio, para ello en nuestro perfil desplegamos la flecha de opciones donde se nos muestra nuestro usuario y seleccionamos “Your repositories”



A continuación clicamos en “New”

Le damos un nombre a nuestro repositorio y lo creamos

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Owner \*

 jairo56 ▾

Repository name \*

PruebaGit ✓

Great repository names are short and memorable. Need inspiration? How about **upgraded-octo-doodle?**

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.](#)

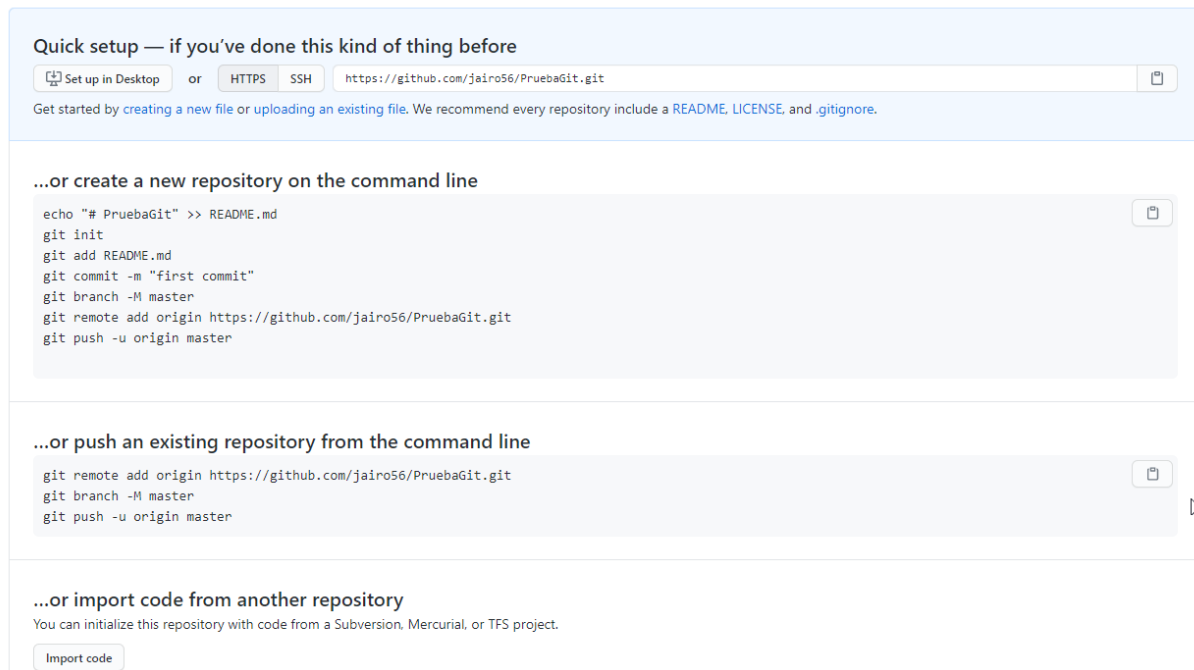


**Choose a license**

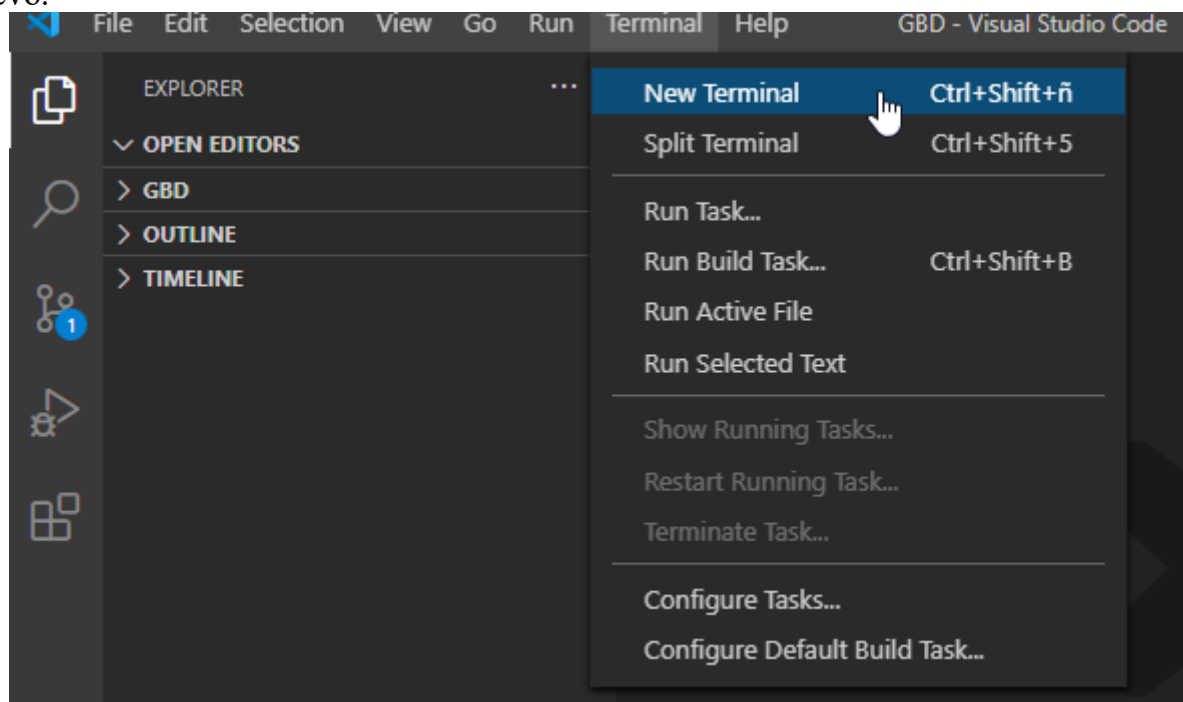
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

A continuación se nos muestra tanto el enlace que permitirá a otras personas acceder a nuestro repositorio como los comandos que tendremos que ejecutar en Git para enlazar al cliente con el equipo remoto y poder subir el archivo en cuestión.



Para empezar vamos a iniciar el programa “Visual Studio Code” y añadimos un terminal nuevo.





En dicho terminal, nos situamos en la carpeta con la que deseamos trabajar.

```
PS C:\> cd C:\Users\jairo\OneDrive\Documentos\PruebaGIT
PS C:\Users\jairo\OneDrive\Documentos\PruebaGIT> 
```

A Continuación debemos ejecutar el comando Git init, para decir que en ese archivo se van a emplear comandos de Git

```
PS C:\Users\jairo\OneDrive\Documentos\PruebaGIT> git init
Initialized empty Git repository in C:/Users/jairo/OneDrive/Documentos/PruebaGIT/.git/
```

A continuación empleamos el comando git remote add origin seguido del enlace que generó nuestro repositorio creado en Github

```
PS C:\Users\jairo\OneDrive\Documentos\PruebaGIT> git remote add origin https://github.com/jairo56/PruebaGit.git
```

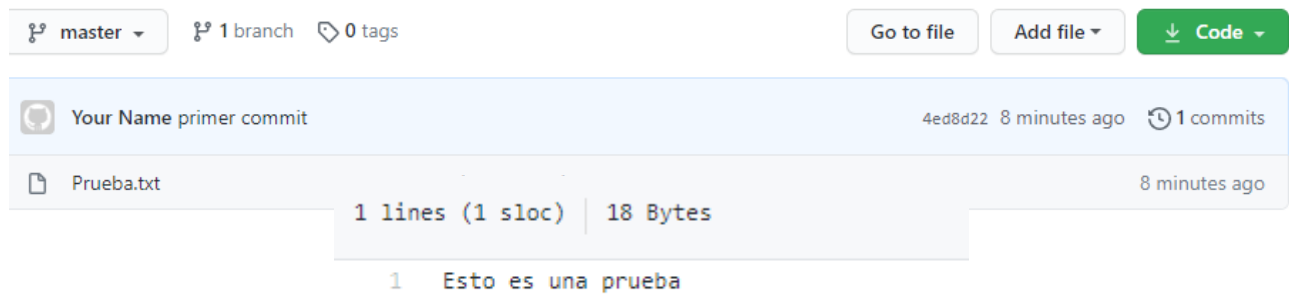
A continuación debemos con “git add .” hacer que encuentre los archivos que están en la localización en la que estamos trabajando, tras lo cual ejecutamos el comando commit seguido de -m si queremos dejar un mensaje o reseña sobre lo que se está trabajando, por ejemplo primer intento o lo que se desee.

```
PS C:\Users\jairo\OneDrive\Documentos\PruebaGIT> git add .
PS C:\Users\jairo\OneDrive\Documentos\PruebaGIT> git commit -m "primer commit"
[master (root-commit) 4ed8d22] primer commit
1 file changed, 1 insertion(+)
create mode 100644 Prueba.txt
PS C:\Users\jairo\OneDrive\Documentos\PruebaGIT> 
```

Los comandos que usamos a continuación nos permite tanto definir el repositorio como master en la comunicación como por su puesto el comando push dejar en el origen maestro (Github) el archivo en cuestión

```
PS C:\Users\jairo\OneDrive\Documentos\PruebaGIT> git branch -M master
PS C:\Users\jairo\OneDrive\Documentos\PruebaGIT> git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 229 bytes | 114.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/jairo56/PruebaGit.git
* [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Y ahí está.



Para poder coger un elemento de un repositorio deberemos usar el comando Git clone, si queremos descargarlo completo “Git clone <https://github.com/jairo56/PruebaGit>” o en cambio si preferimos solo descargarnos la información que ha sido modificada por otros usuarios con acceso a dicha información pues “Git pull <https://github.com/jairo56/PruebaGit>”