 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

**GA7-220501096-AA1-EV04 instalación y configuración de herramienta de versionamiento
(Local / Web)**


Libardo H. Bautista, Juany A. Calvache, Julián A. Ruiz y Diana Shirley Chávez

Servicio Nacional de Aprendizaje (SENA)

Grupo 2758307, Análisis y Desarrollo de Software


Ing. Fernando Bohórquez García

Abril 13 de 2024

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

Contenido

Introducción	3
GA7-220501096-AA1-EV04 instalación y configuración de herramienta de versionamiento (Local / Web)	3
Paso a paso de la instalación de la herramienta de Versionamiento local / web	4
Versionamiento Local	4
Versionamiento Web	5
Pantallazos de la instalación de las herramientas de control de versionamiento tanto local como remota.....	7
Versionamiento Local	7
Versionamiento Web	28
Conclusiones	31
Bibliografía	32

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32


Introducción

Git es un sistema de control de versiones distribuido que facilita el trabajo en ramas independientes, fusiones, y etiquetado de versiones. Permite a los desarrolladores trabajar localmente y sincronizar su trabajo con un repositorio remoto. Es por ello, que la elección de Git se debe a su robustez, flexibilidad y amplia adopción en la industria. Su modelo distribuido es ideal para equipos descentralizados y contribuye a una mejor gestión de conflictos y una historia de cambios más clara.

La instalación y configuración de esta herramienta de versionamiento es un proceso fundamental para el desarrollo de software colaborativo y la gestión eficiente de cambios en el código fuente.

GA7-220501096-AA1-EV04 instalación y configuración de herramienta de versionamiento (Local / Web)

Para cumplir con esta evidencia, vamos a tener en cuenta los requisitos exigidos en la guía de trabajo # 7, que nos pide, tomando como referencia el componente formativo “Integración continua”, realizar la instalación y configuración de las herramientas de control de versionamiento tanto local como remoto.. También, se deben seguir las normas básicas de presentación de un documento escrito, es decir el documento debe tener como mínimo una portada, introducción,

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

objetivo y el paso a paso con pantallazos de la instalación de las herramientas de control de versionamiento tanto local como remota.

Paso a paso de la instalación de la herramienta de Versionamiento local / web

A continuación, describiremos los pasos involucrados tanto para versionamiento local como para versionamiento web:

Versionamiento Local


El versionamiento local implica trabajar con un repositorio de control de versiones almacenado en la máquina local del desarrollador. Los pasos son:

Instalación del Software de Control de Versiones (SCV)

- Descarga e instala una herramienta de control de versiones como Git o Mercurial en tu computadora.
- Configura las opciones básicas, como tu nombre de usuario y dirección de correo electrónico.

Creación de un Repositorio Local

- Crea un nuevo repositorio local o clona uno existente desde un repositorio remoto (como GitHub o Bitbucket).

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

- Utiliza el comando git init para inicializar un nuevo repositorio en una carpeta local.

Configuración del Repositorio

- Define las propiedades del repositorio, como el nombre del proyecto y la descripción.
- Agrega archivos al repositorio utilizando git add.

Commit y Registro de Cambios

- Realiza un commit para registrar los cambios en el repositorio local utilizando git commit.
- Puedes agregar un mensaje descriptivo para cada commit.

Gestión de Ramas


- Crea y administra ramas para trabajar en características específicas o solucionar problemas.
- Utiliza comandos como git branch y git checkout.

Sincronización con Repositorio Remoto

- Sube tus cambios al repositorio remoto utilizando git push.

Versionamiento Web

El versionamiento web implica trabajar con un repositorio de control de versiones alojado en una plataforma en línea (por ejemplo, GitHub, GitLab o Bitbucket). Los pasos son:

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

Creación de una Cuenta en la Plataforma Web

- Regístrate en una plataforma de alojamiento de repositorios como GitHub.

Creación de un Repositorio Remoto

- Crea un nuevo repositorio en la plataforma web.
- Define el nombre del proyecto, la descripción y las opciones de visibilidad (público o privado).

Clonación del Repositorio Remoto

- Clona el repositorio remoto en tu máquina local utilizando git clone.

Trabajo Local y Commits


- Realiza cambios en los archivos locales y registra los commits utilizando git add y git commit.

Sincronización con Repositorio Remoto

- Sube tus cambios al repositorio remoto utilizando git push.

Colaboración y Pull Requests

- Colabora con otros desarrolladores mediante pull requests.
- Revisa y fusiona cambios utilizando la interfaz web de la plataforma.

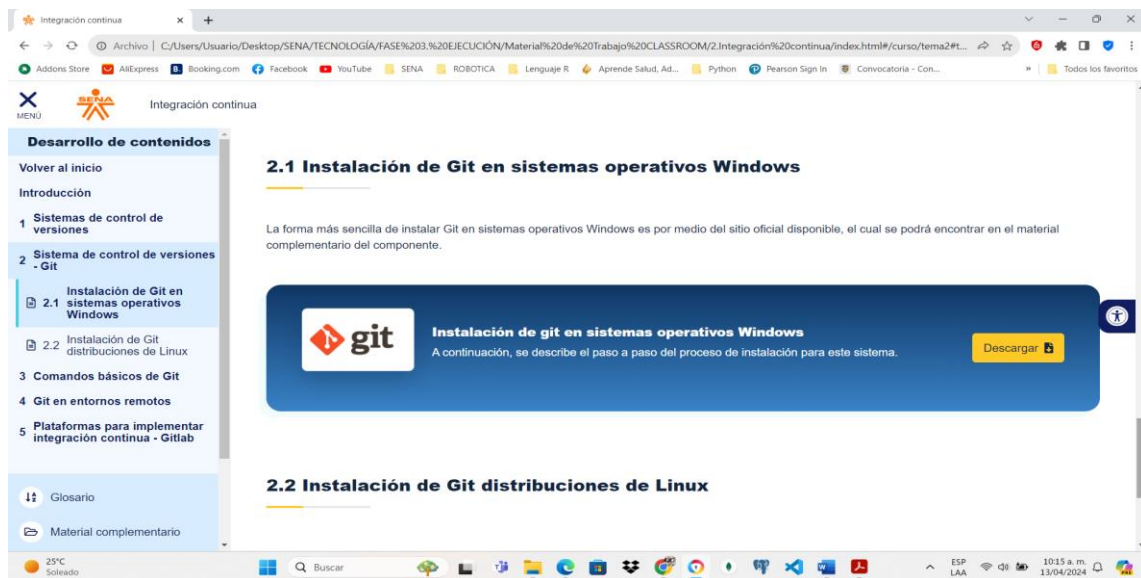
 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

Pantallazos de la instalación de las herramientas de control de versionamiento tanto local como remota


Versionamiento Local

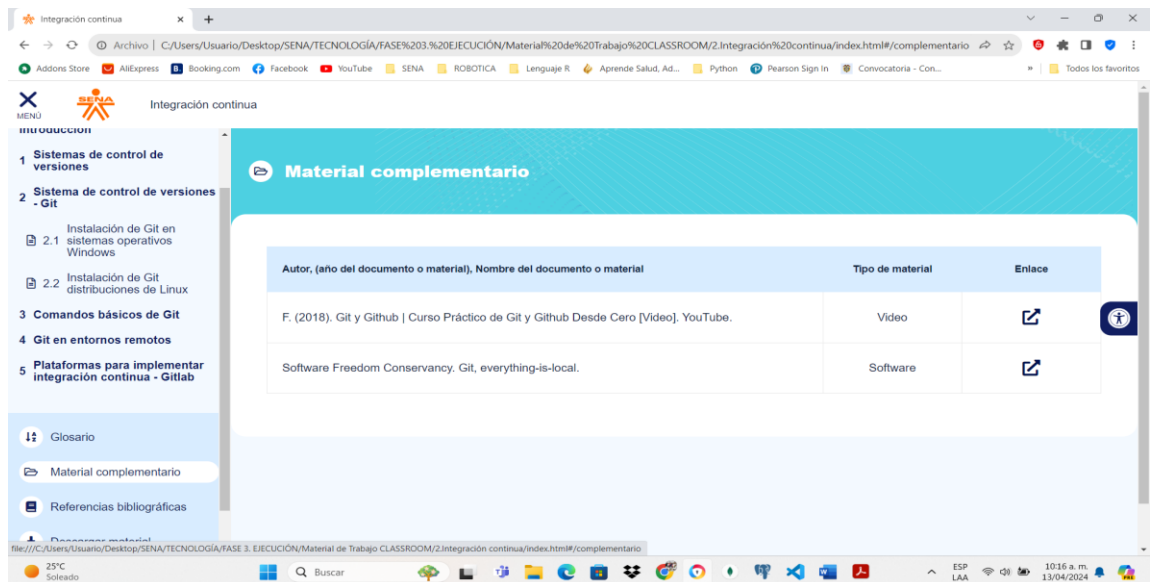
Descargar e instalar la herramienta de control de versiones Git en tu computadora

Para la instalación de la herramienta de control de versionamiento Git, nos piden que tengamos en cuenta el componente formativo “Integración continua”. En el componente formativo encontramos en el apartado 2.1 una guía en Word con los pasos para la instalación de git.

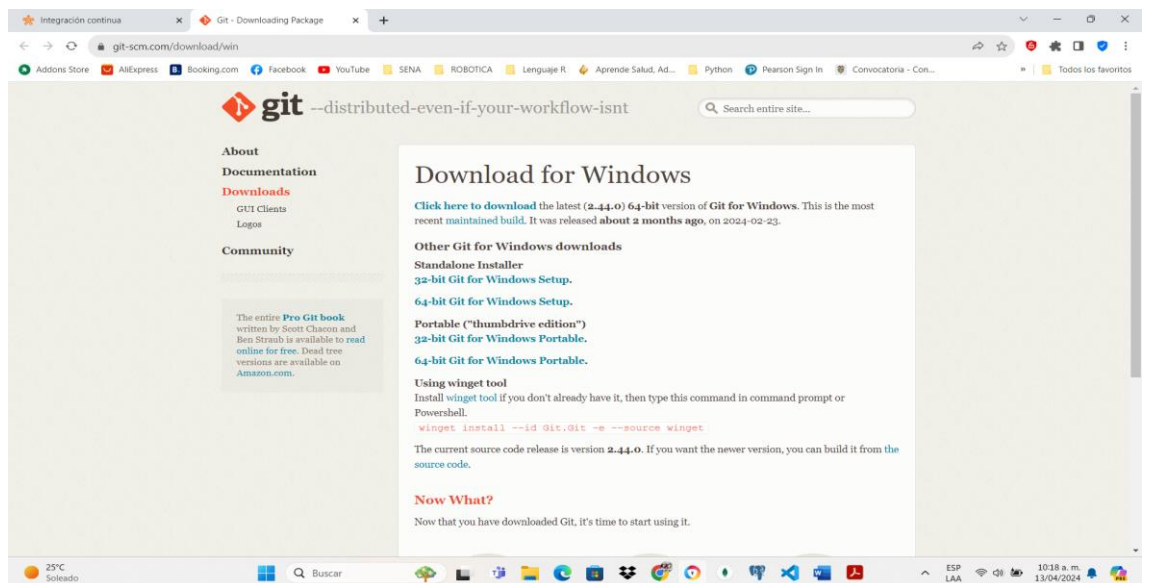


También, nos piden que vayamos al material complementario donde encontramos un link para descargar el software de Git.


 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

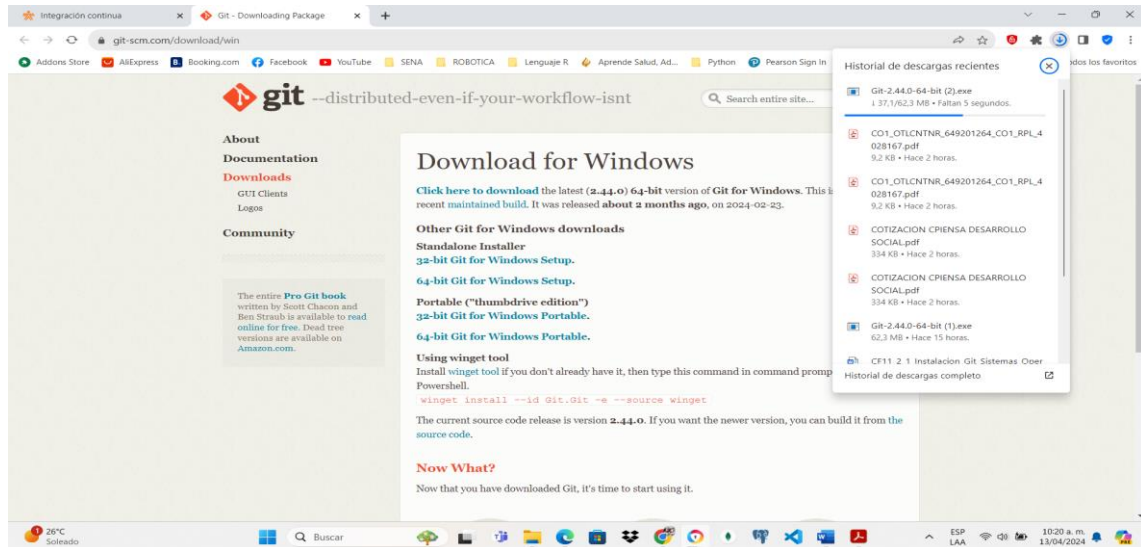


Vamos al link y descargamos el software de instalación.

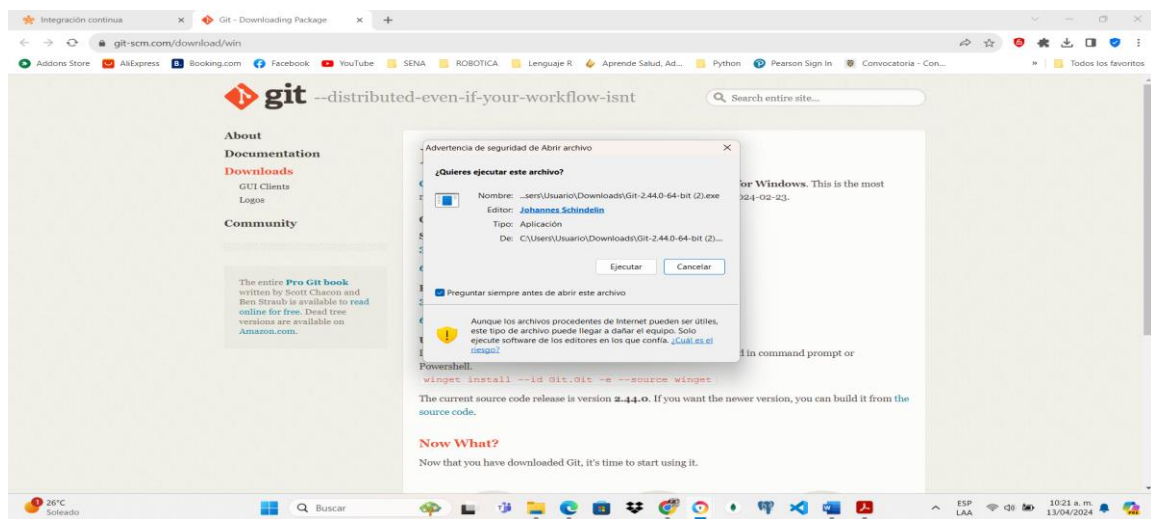


Damos click en 64-bit Git for Windows Setup y descargamos el paquete de Git.

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32



Acto seguido, procedemos a instalar la versión 2.44.0




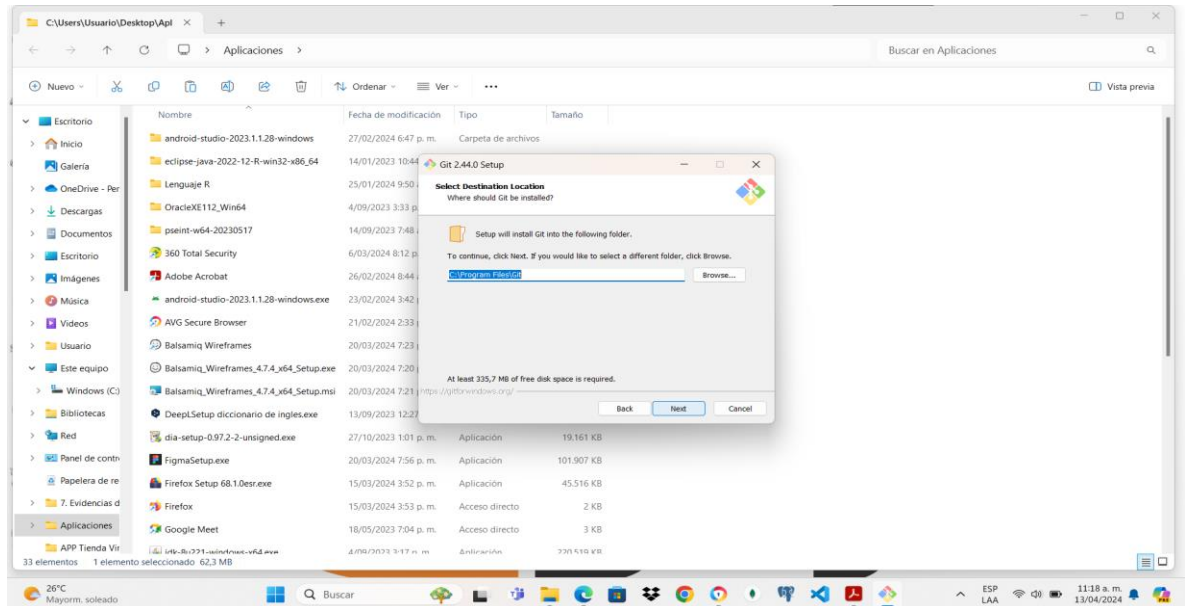
Damos click en ejecutar y se descarga la herramienta de control y versionamiento en nuestro computador. Buscamos donde quedó descargado el archivo y procedemos a la instalación. Damos click en ejecutar y se procede a la instalación de la herramienta en nuestro computador. Aceptamos los términos y condiciones y damos click en Instalar. El proceso de instalación es muy corto y finaliza con la posibilidad de leer las notas relacionadas o lanzar inmediatamente Git Bash.

Barrancabermeja, Carrera 39 No. 53-37 Barrio Provivienda. Celular 316-6276395

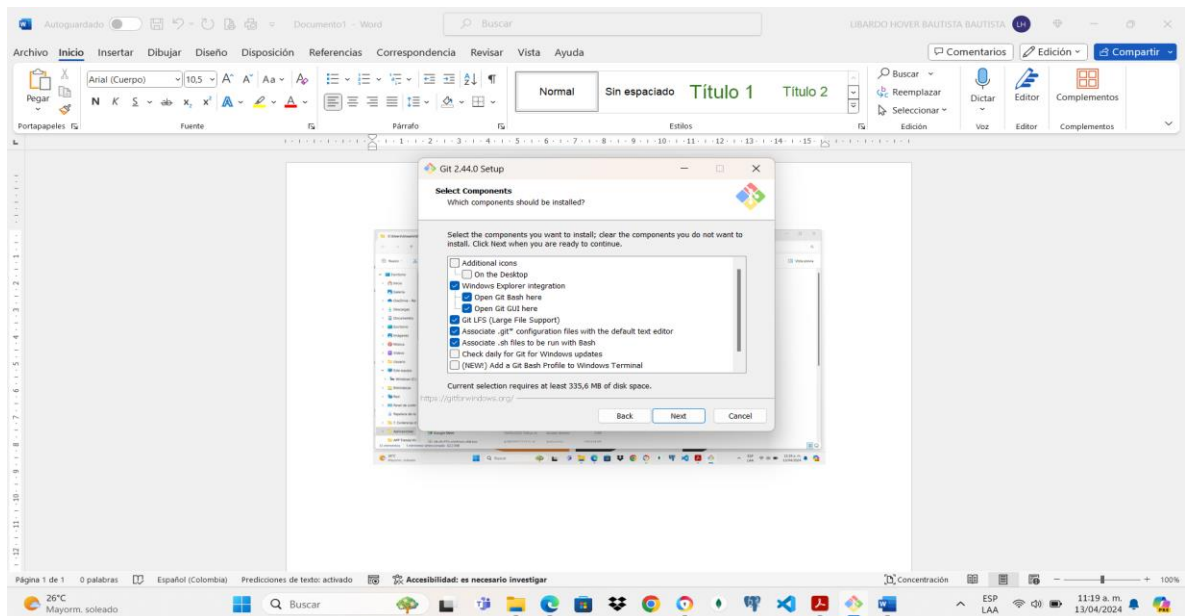
Correo Electrónico: piensacorporacion@gmail.com

www.cpiensa.com


 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

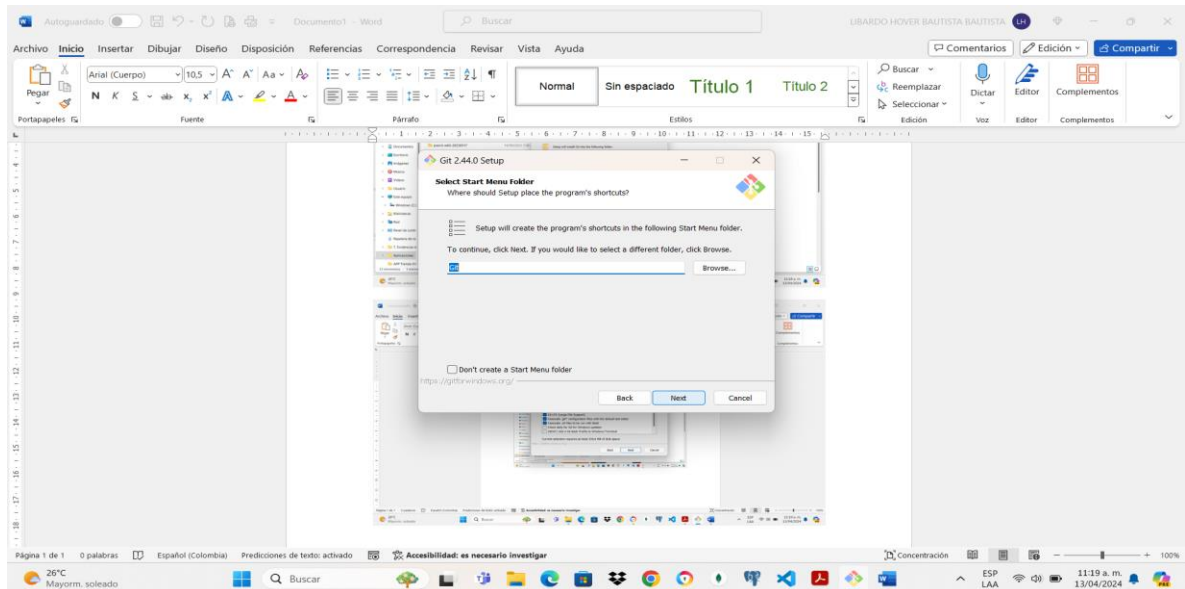


En este pantallazo nos piden que elijamos el disco y la carpeta a donde va a quedar la descarga del software.

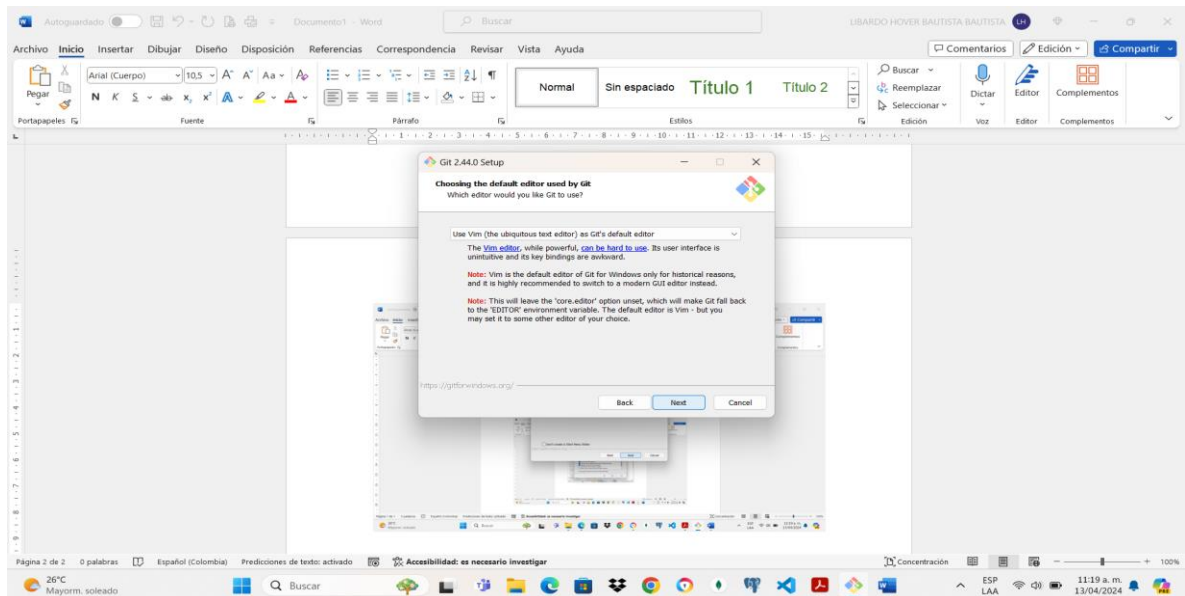


Aquí seleccionamos los componentes.


 <p>NIT. 901.510.404-9</p>	<h1>COMUNICACIONES EXTERNAS</h1>	<p>Código: CCP FCE01</p> <p>Versión:01</p> <p>Página 1 de 32</p>
---	----------------------------------	--

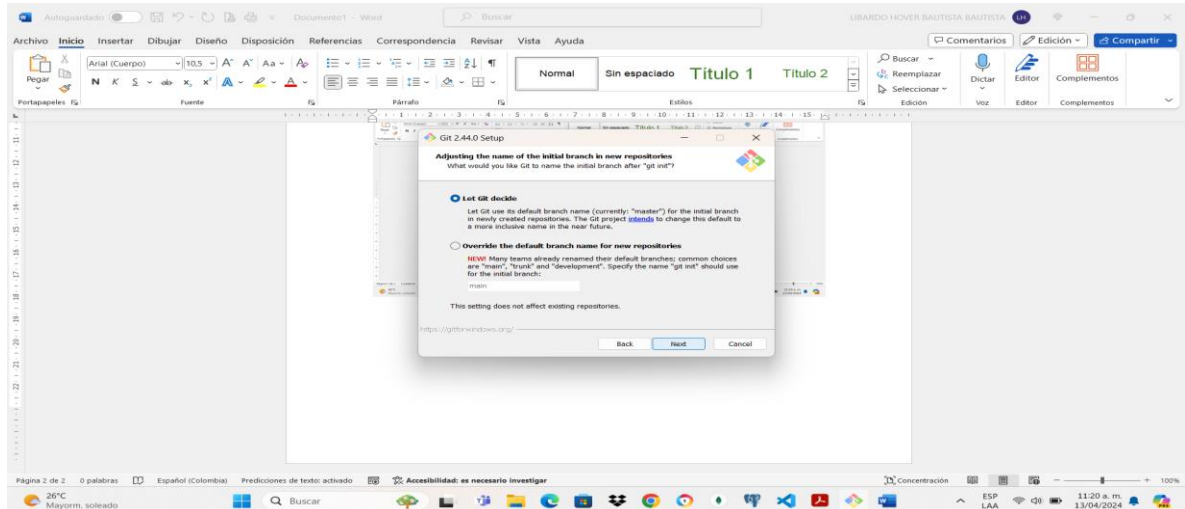


En este pantallazo vemos los atajos Git.

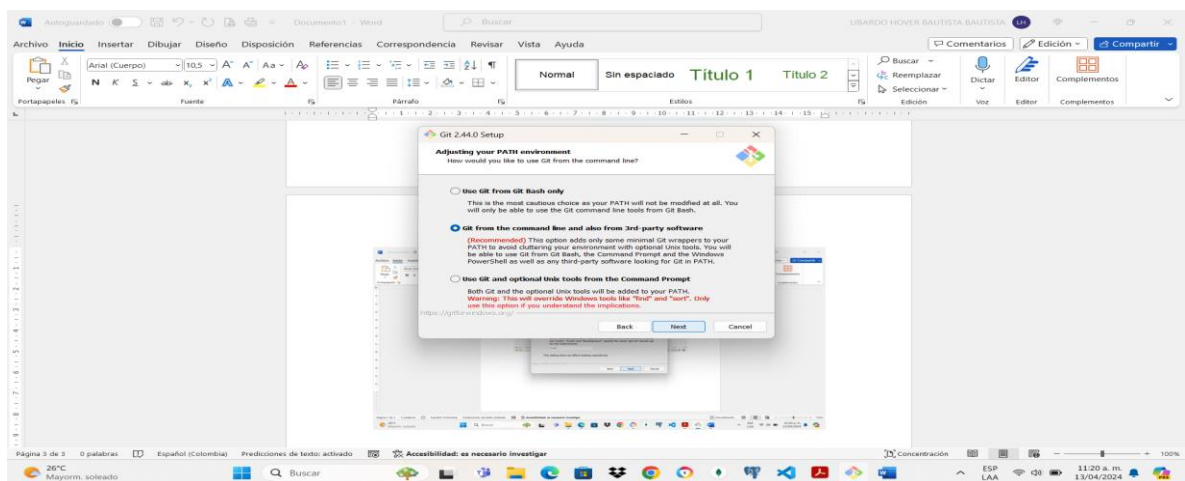


Aquí escogemos el editor utilizado por Git. En mi caso escojo Visual Studio Code. Pero, en el material de estudio se escoge por defecto el Git Bush. En este editor es donde se escriben los mensajes de los commits.


 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

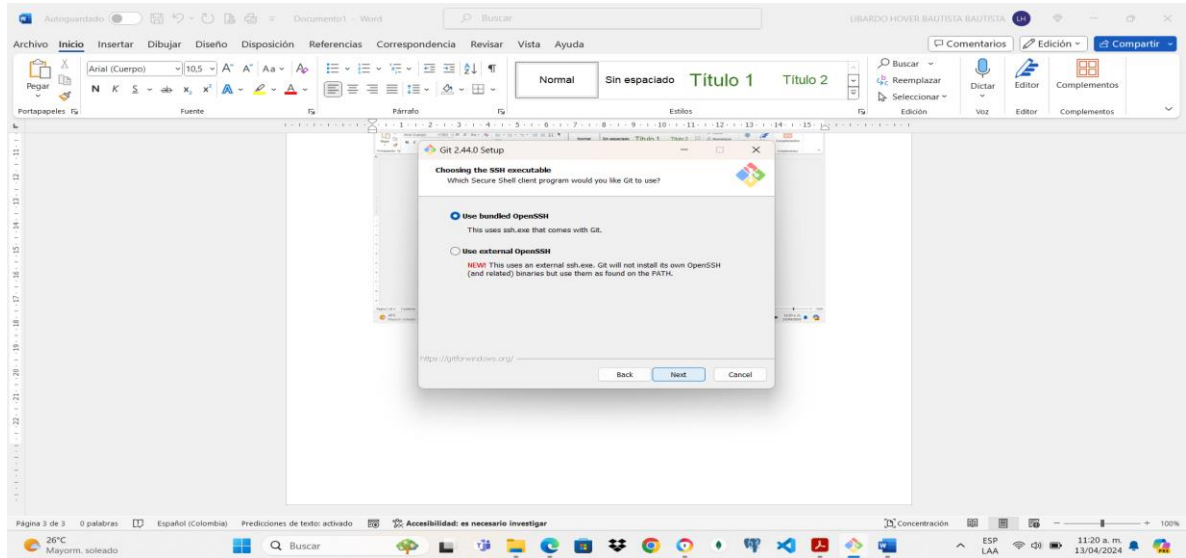


En este pantallazo se escoge la rama inicial donde se crea un repositorio. Por defecto escoge el nombre de la rama, el cual, es master. Pero, si escoge la segunda opción ahí puede colocarle el nombre a la rama que usted quiera relacionado con el proyecto que va a desarrollar, sin embargo, por efecto la nombraría main.

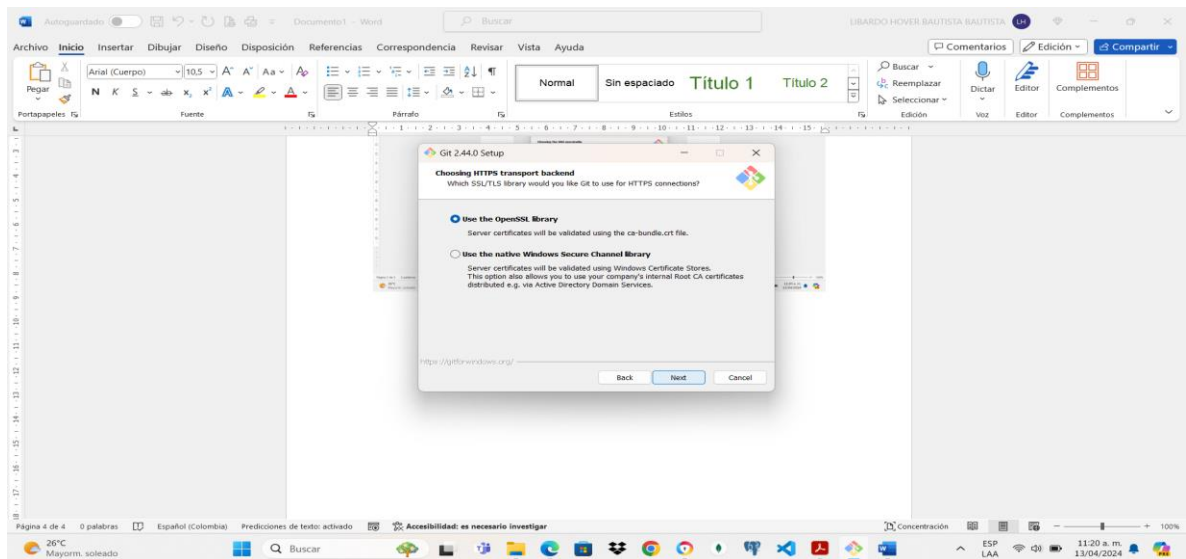


En este pantallazo se escoge el ambiente del Path, es decir, el camino. Para nuestro caso escogemos utilizar Git desde el ambiente del Git Bash.


 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

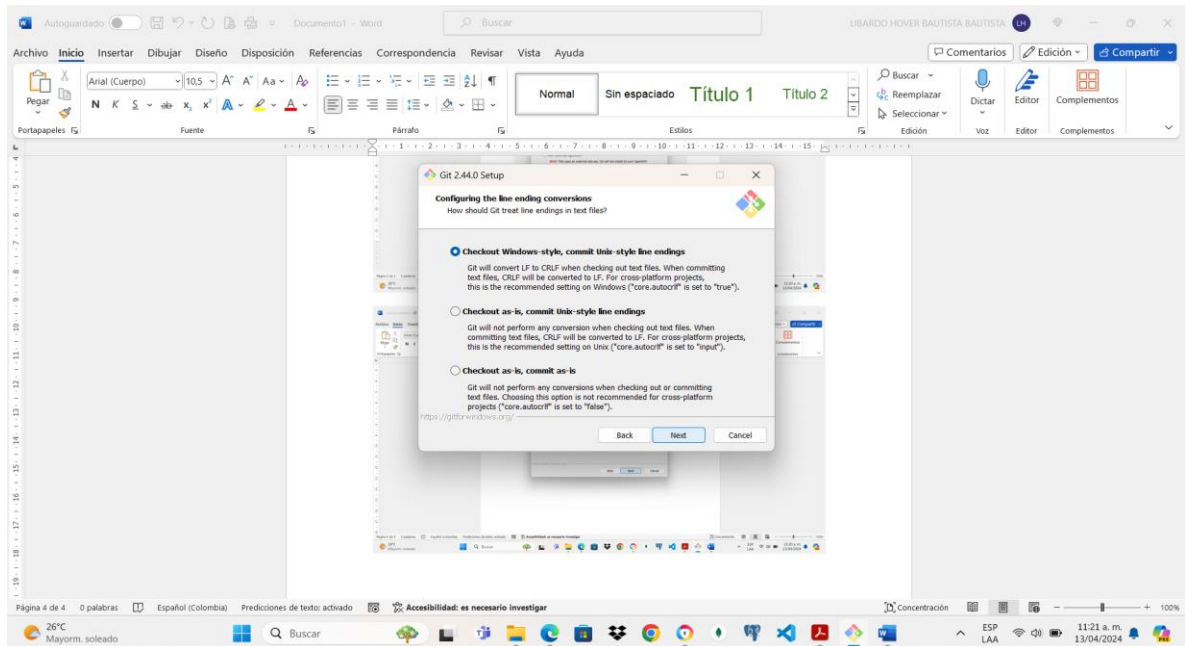


Aquí se escoge el ejecutable SSH.

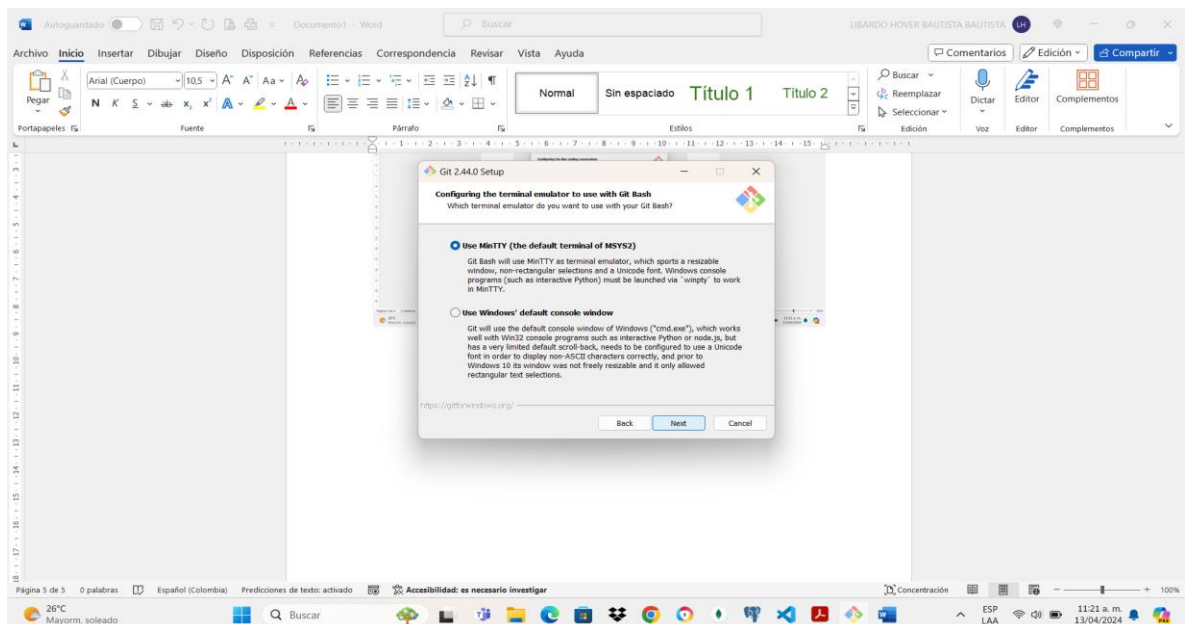


En este pantallazo escogemos, por defecto, la biblioteca open SSL.

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32




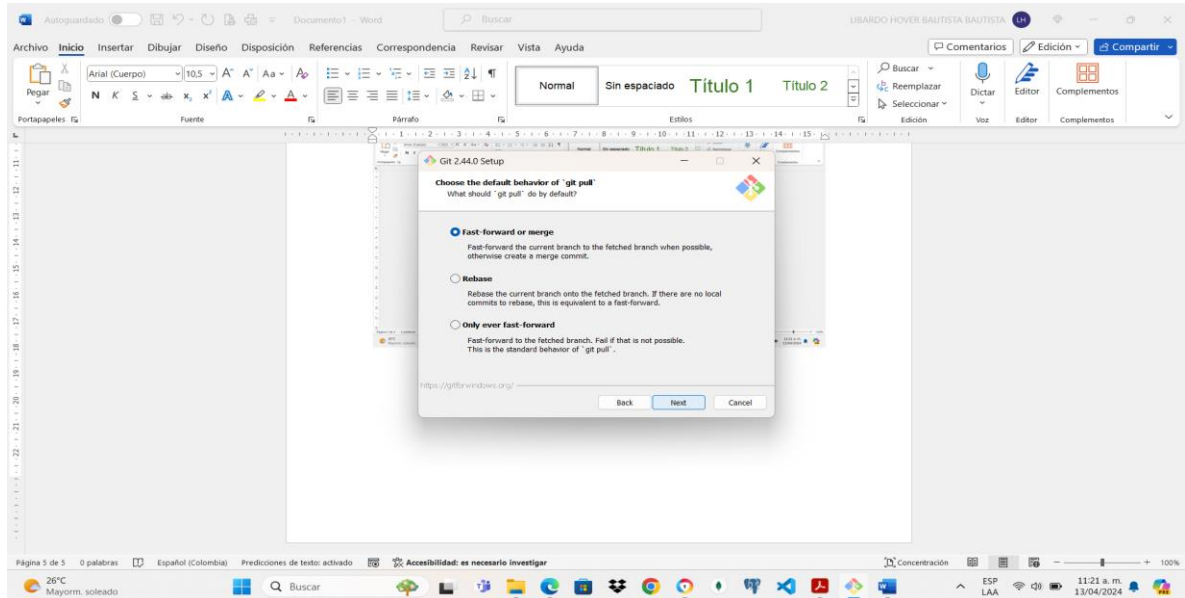
En este pantallazo escogemos como se convierte el final de línea.



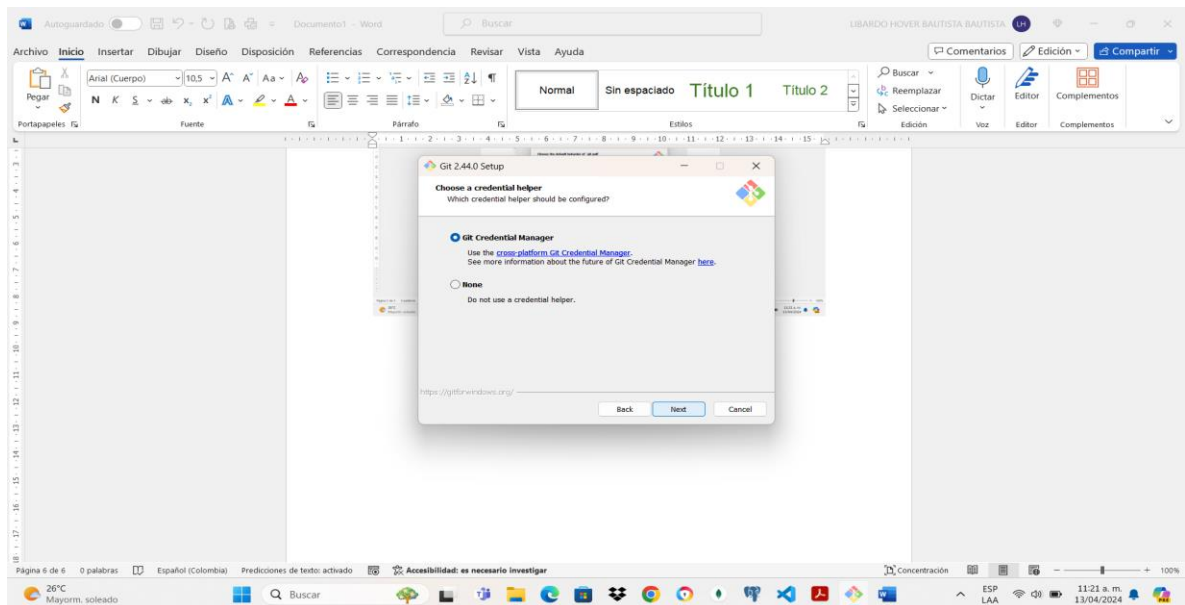
Aquí se configura el emulador del terminal para configurarlo con Git Bash. Por defecto escoge el MinTTY.

Barrancabermeja, Carrera 39 No. 53-37 Barrio Provienda. Celular 316-6276395
Correo Electrónico: piensacorporacion@gmail.com
www.cpiensa.com


 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

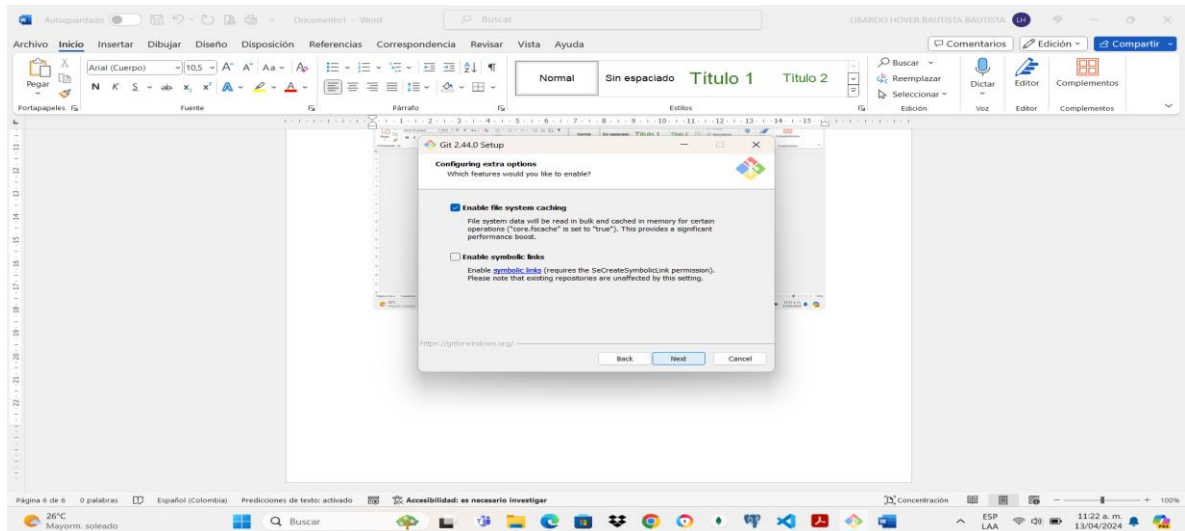


Aquí se escoge el comportamiento del git pull. Por defecto, se escoge el default.

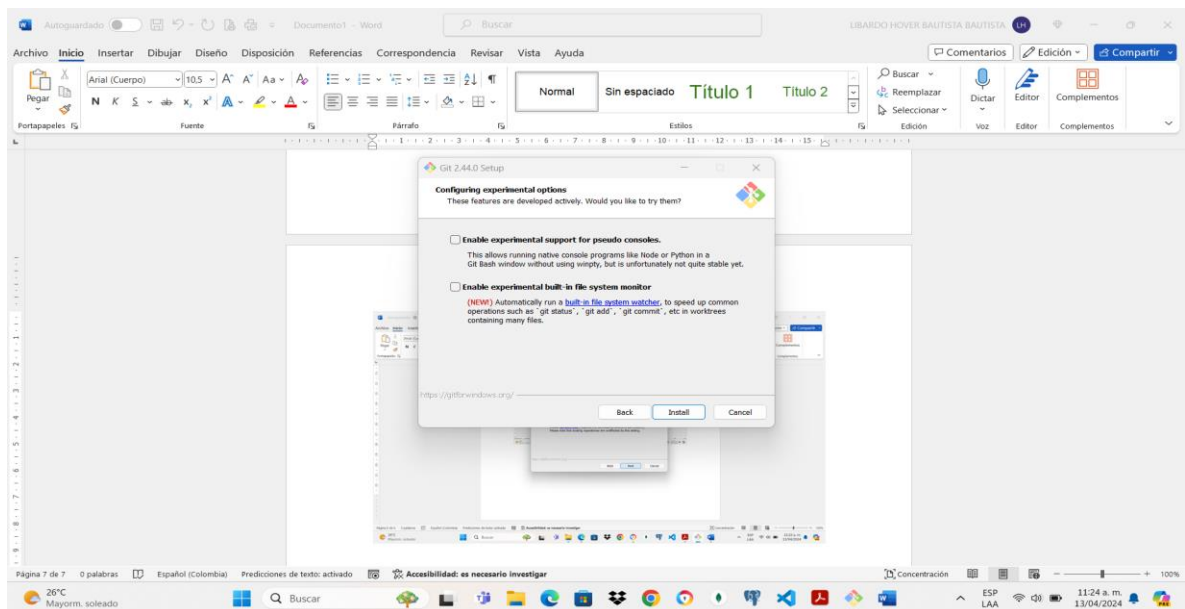


Aquí se escoge el administrador de credenciales de Git.


 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

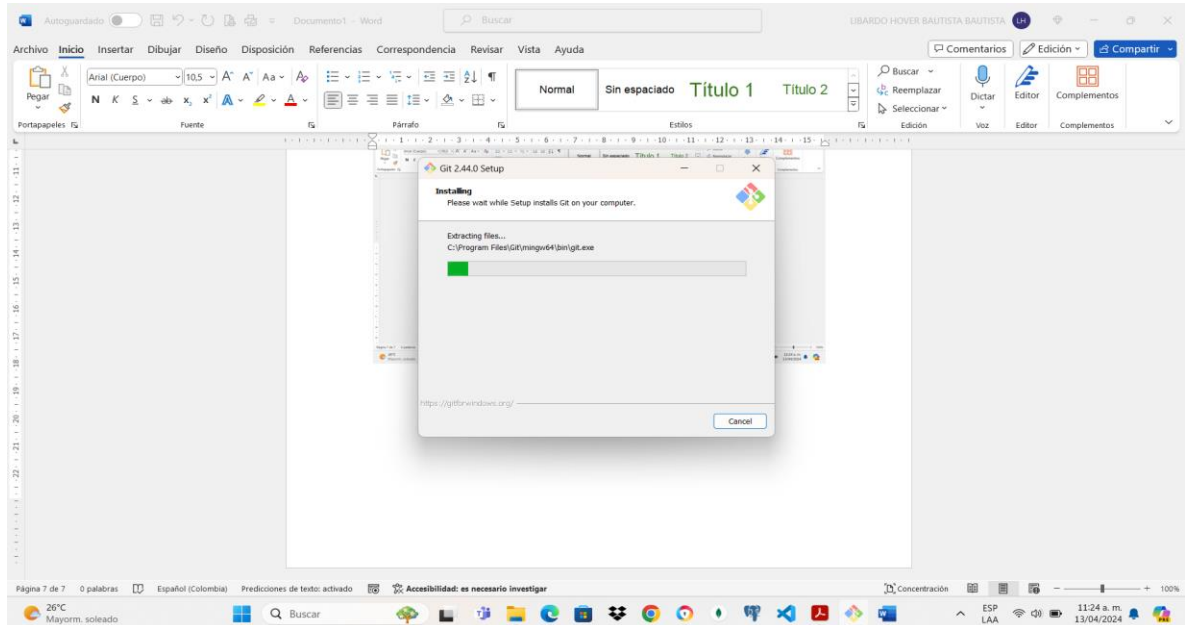


En esta parte, habilitamos el almacenamiento en caché del sistema de archivos, que por defecto, se escoge la primera opción.

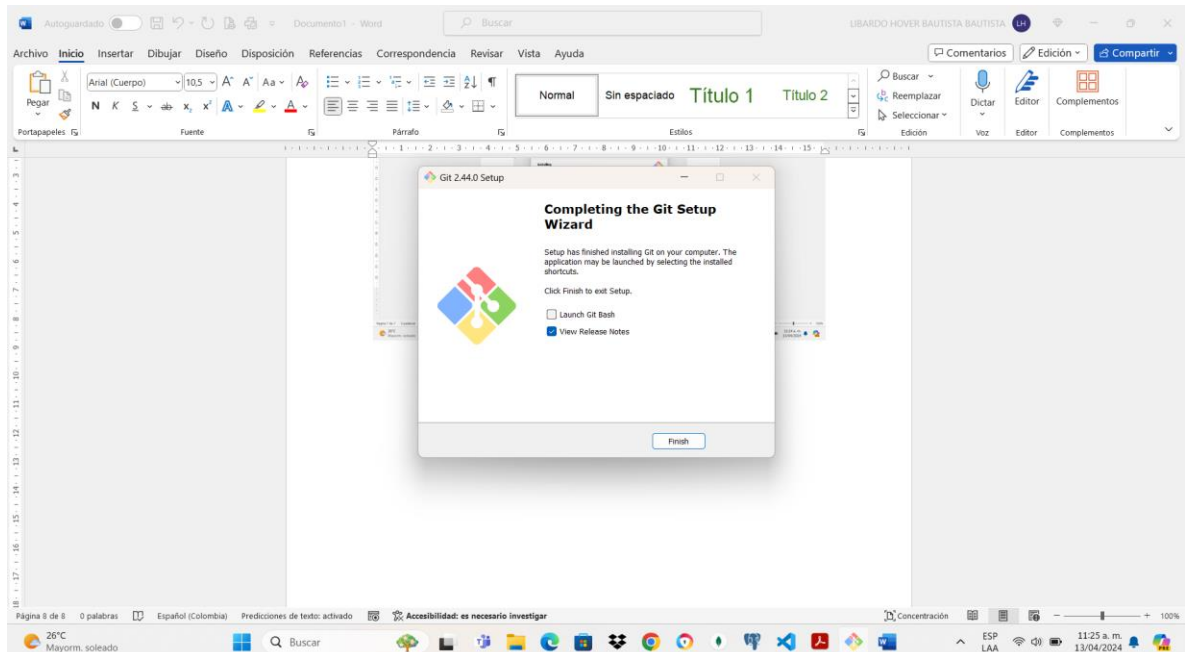



Aquí se configuran las opciones experimentales, que por defecto no viene ninguna resaltada, por lo tanto, damos click sin escoger ninguna.

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

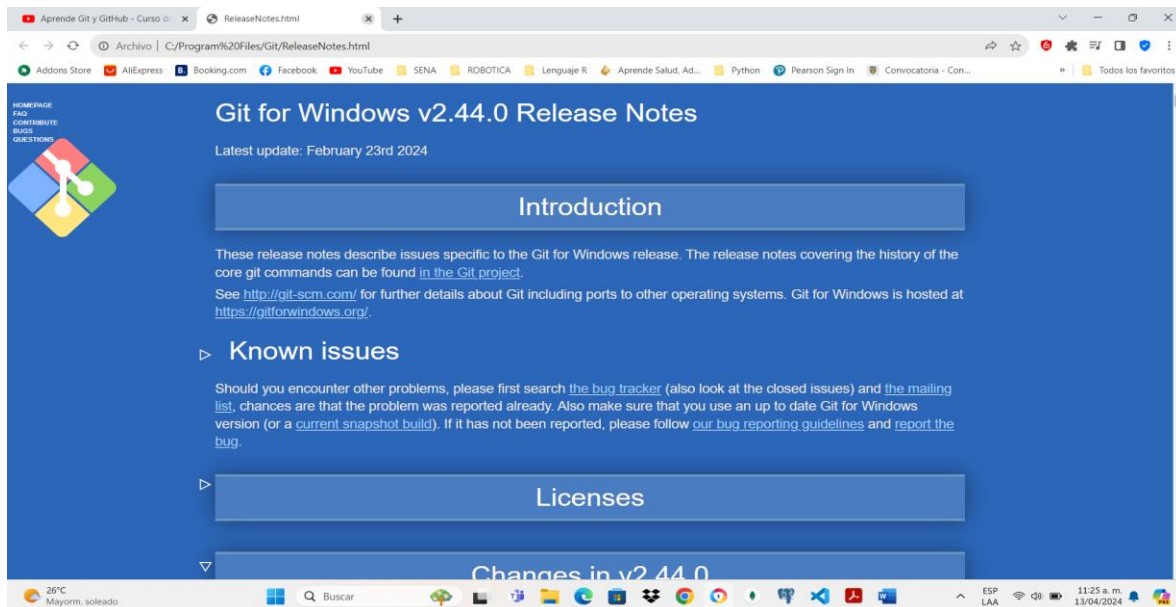


Aquí empezamos a hacer la instalación de Git.



 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

En este pantallazo se completa el asistente de configuración de Git donde también marcamos la primera opción y con la segunda opción nos envía a la pagina de las notas de Git para Windows.




Configurar las opciones básicas, como tu nombre de usuario y dirección de correo electrónico

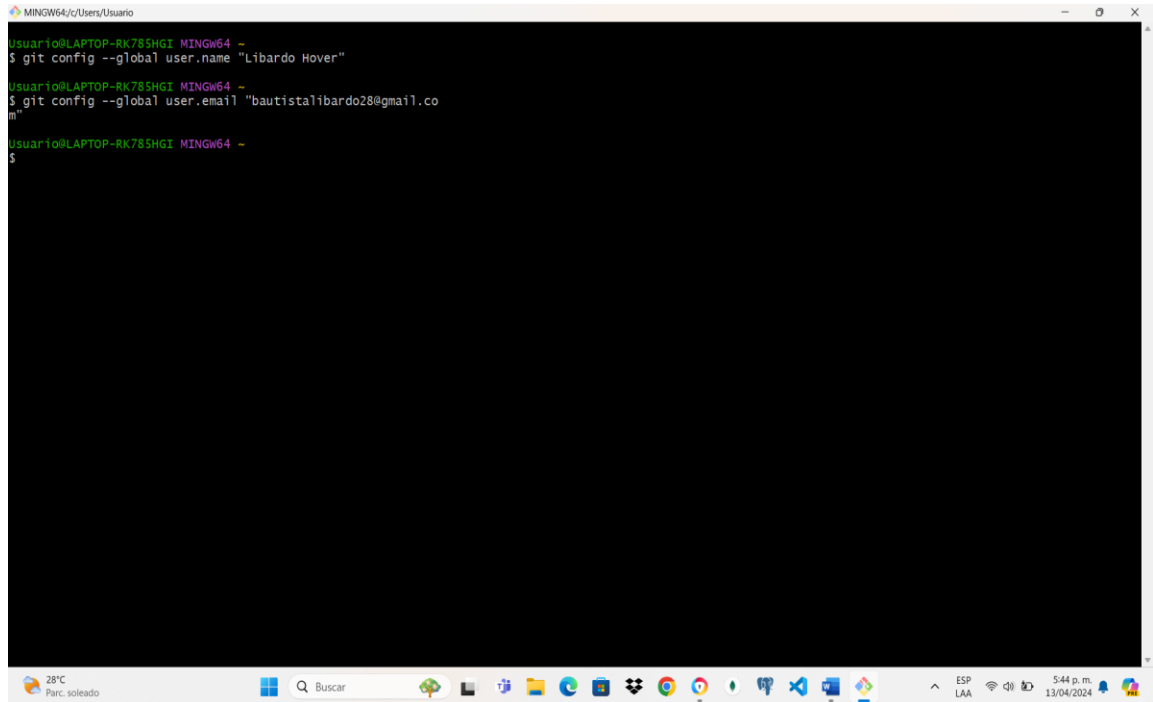
Para desarrollar esta parte del trabajo, empezamos a utilizar Git Bush. Para configurar el nombre del usuario y la dirección de correo electrónico utilizamos los comandos:

```
$ git config --global user.name "Libardo Hover"
$ git config --global user.email "bautistalibardo28@gmail.com"
```

Aquí es preciso señalar que: primero, se aconseja utilizar --global para que el nombre de usuario y el email del usuario se considere en todos los repositorios que se creen en Git; segundo, por seguridad es mejor no utilizar el correo personal sino esperar a que GitHub nos coloque uno

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

por defecto. Lo anterior, sustentado en que el correo queda en todo el historial que se va a guardar y que otros desarrolladores pueden ver.



```

MINGW64/c/Users/Usuario
Usuario@LAPTOP-RK785HGI MINGW64 -
$ git config --global user.name "Libardo Hover"
Usuario@LAPTOP-RK785HGI MINGW64 -
$ git config --global user.email "bautistalibardo28@gmail.com"
Usuario@LAPTOP-RK785HGI MINGW64 -
$

```

Crear un nuevo repositorio local y utilizar el comando git init para inicializar


Para crear un repositorio local utilizamos las siguientes líneas de código y sus comandos:

```

$ cd Documents
$ mkdir instalacion-git
$ cd instalacion-git
$ git init

```

Primero, se ubica en la carpeta de documentos, luego se crea la carpeta llamada instalacion-git, enseguida se ubica en la carpeta recientemente creada y por último se crea el repositorio con el comando **git init**, que significa inicializar. Para nombrar la carpetas utilizamos la nomenclatura definida en freeCodeCamp.

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

```

MINGW64/c:/Users/Usuario/Documents/instalacion-git
usuario@LAPTOP-RK785HGI MINGW64 ~
$ cd Documents
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents
$ mkdir instalacion-git
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents
$ cd instalacion-git
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
$ git init
Initialized empty Git repository in C:/Users/Usuario/Documents/instalacion-git/.git/
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git (master)
$ |

```

Definir las propiedades del repositorio

En este apartado, mostramos las propiedades del repositorio como son el nombre, la rama donde quedó creado y el nombre de la carpeta. Con el comando **git status** se muestra el repositorio.


```

MINGW64/c:/Users/Usuario/Documents/instalacion-git
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ git status
On branch master

No commits yet

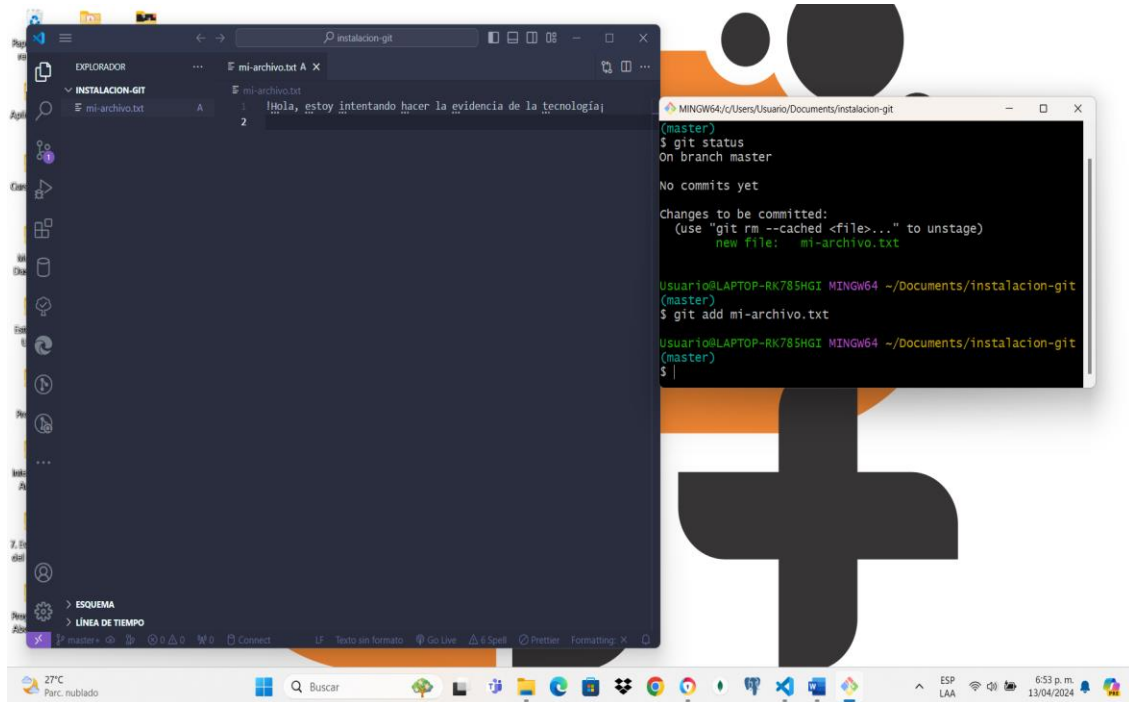
nothing to commit (create/copy files and use "git add" to track)
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ |

```


 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

Agregar archivos al repositorio

Esto lo vamos a empezar directamente desde Visual Studio Code.

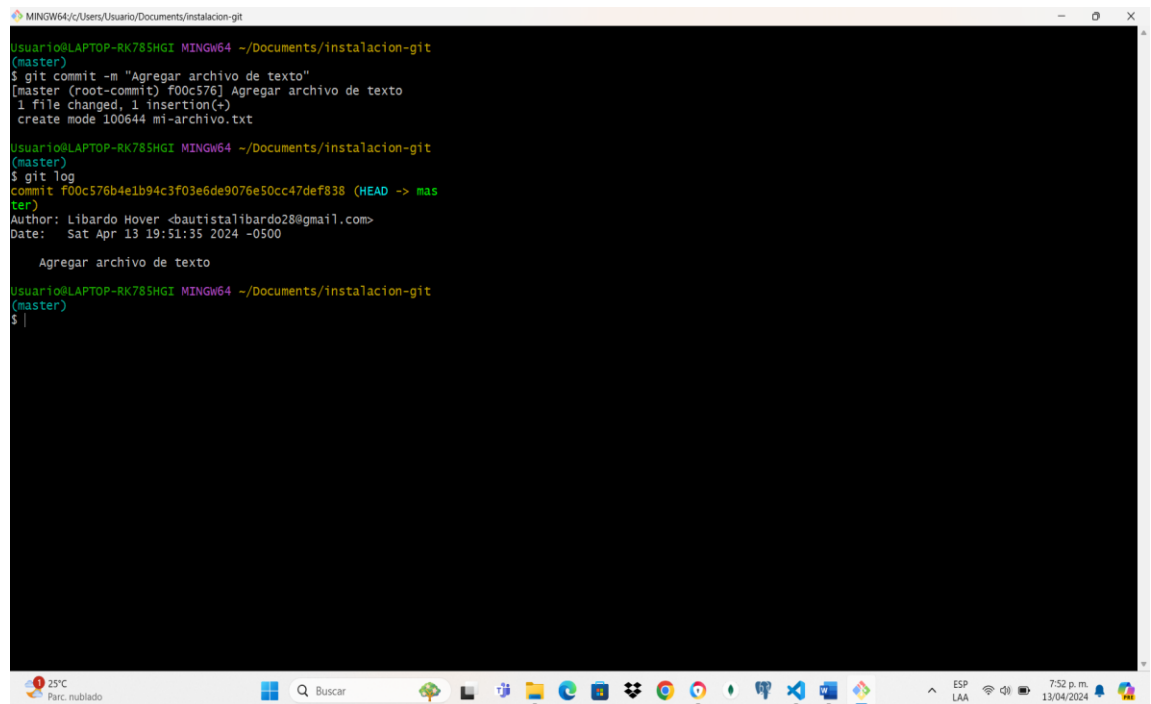


Abrimos VSC y llamamos la carpeta donde estamos creando el proyecto. Acto seguido, creamos un archivo .txt “Hola, estoy intentando hacer la evidencia de la tecnología”, inicialmente nos lo muestra con una “U” al grabar con control S. Al ir a Git Bush y le damos el comando **git add**, no lo agrega a la carpeta y nos muestra el archivo con una “A”, es decir, ya está creado ese archivo .txt en el repositorio. Es decir, nos pasa el archivo del área de trabajo al área de preparación.

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

Realizar un commit para registrar los cambios en el repositorio local

Con el comando **git commit** llevamos el archivo .txt al área del repositorio. Con el comando **git log** miramos el commit y el que lo realizó.



```

MINGW64 ~/Documents/instalacion-git
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ git commit -m "Agregar archivo de texto"
[master (root-commit) f00c576] Agregar archivo de texto
1 file changed, 1 insertion(+)
create mode 100644 mi-archivo.txt


usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ git log
commit f00c576b4e1b94c3f03e6de9076e50cc47def838 (HEAD -> master)
Author: Libardo Hover <bautistalibardo28@gmail.com>
Date: Sat Apr 13 19:51:35 2024 -0500

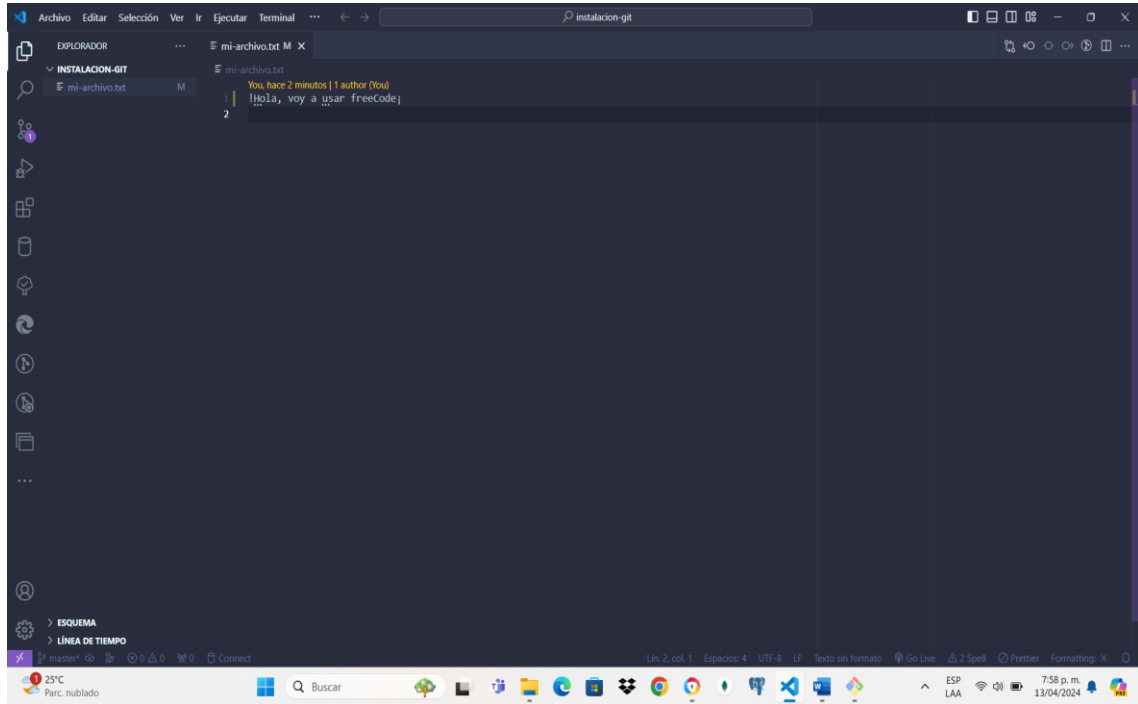
    Agregar archivo de texto

usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$

```

Otra manera de realizar un commit es desde el editor de texto VSC. Al cambiar el texto y grabar con control S, me sale una “M” donde me indica que la línea de texto fue modificada.

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32



Ahora, al ir al Git Bush y darle el comando git status me muestra que la línea de texto fue modificada.

```

MINGW64~/Users/Usuario/Documents/instalacion-git
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
$ git commit -m "Agregar archivo de texto"
[master (root-commit) f00c576] Agregar archivo de texto
1 file changed, 1 insertion(+)
 create mode 100644 mi-archivo.txt


usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
$ git log
commit f00c576b4e1b94c3f03e6de9076e50cc47def838 (HEAD -> master)
Author: Libardo Hover <bautistalibardo28@gmail.com>
Date: Sat Apr 13 19:51:35 2024 -0500

    Agregar archivo de texto

usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
$ git status
On branch 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:   mi-archivo.txt

no changes added to commit (use "git add" and/or "git commit -a")
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git (master)
$ |

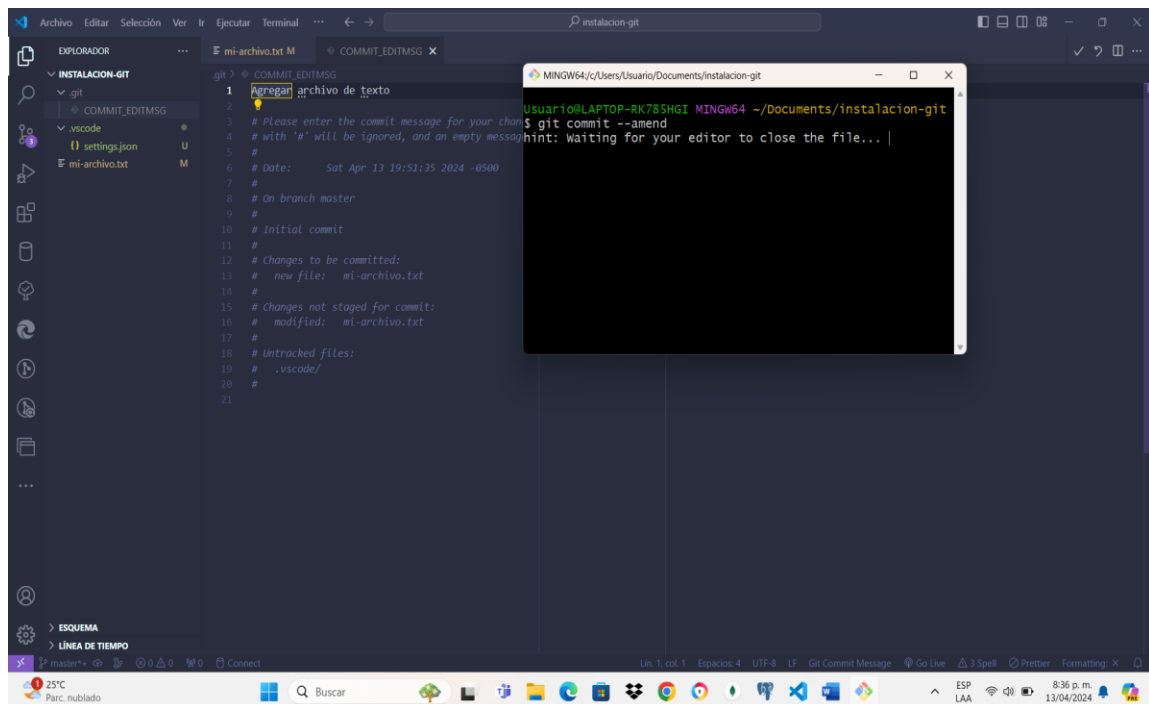
```

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32


Al utilizar el comando **git add**, se muestra que el archivo pasa del área de trabajo al área de preparación, es decir, queda listo para hacer el commit.

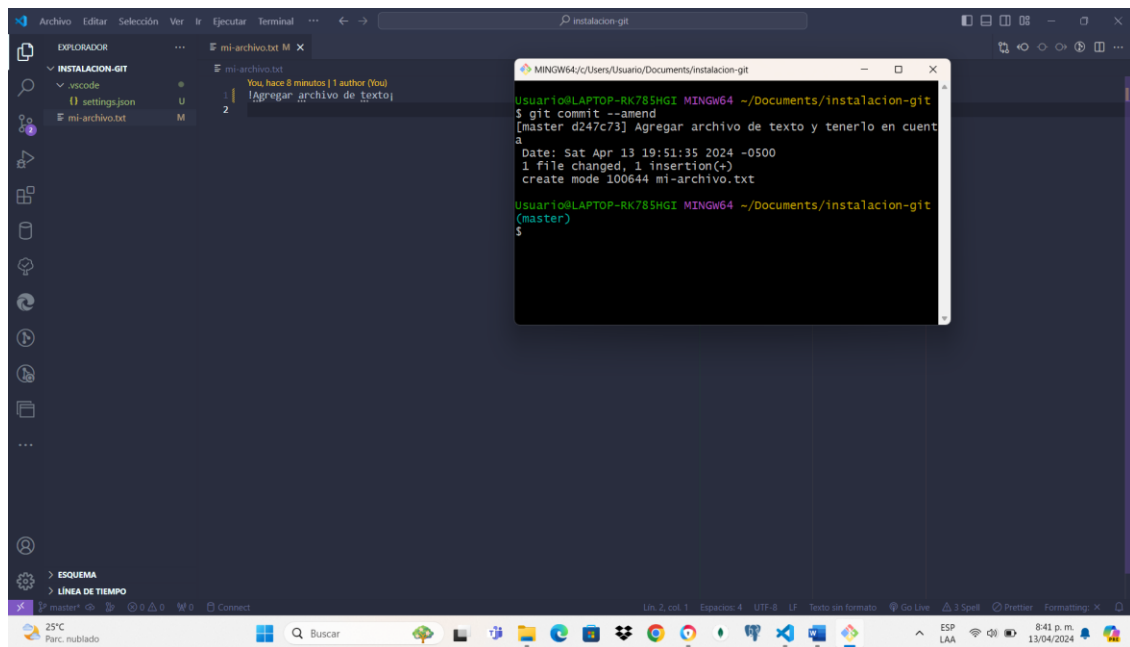
Modificar un commit

Para modificar un commit usamos el comando **git --amend**.



Cambiamos el mensaje, grabamos con control S y cerramos el archivo denominado “COMMIT_EDITMSG”.

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32



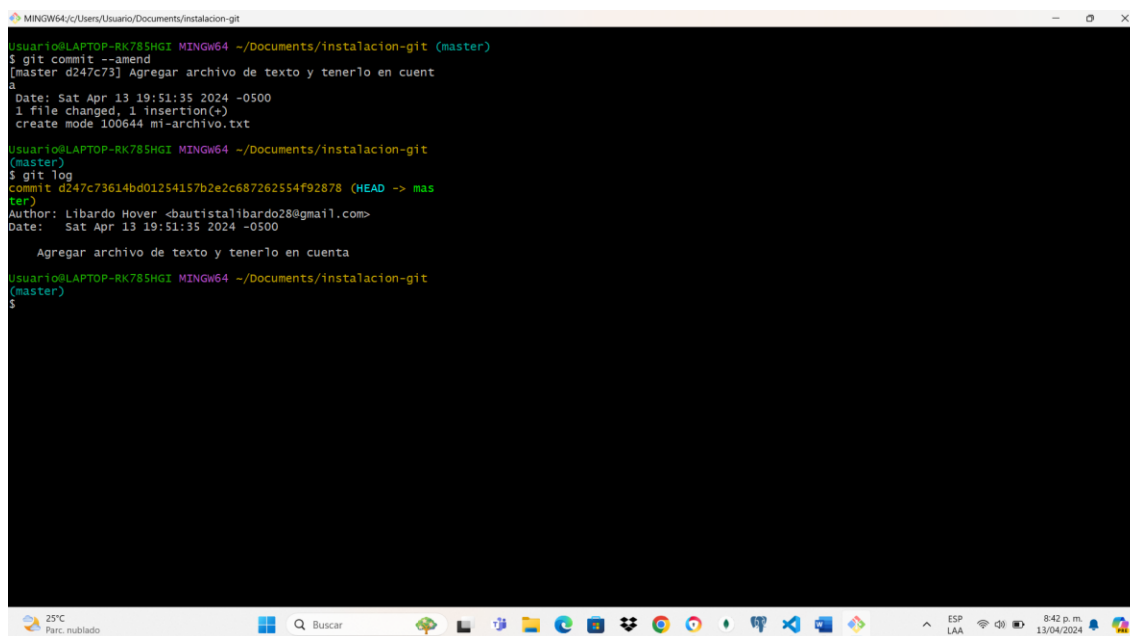
```

MINGW64~/Users/Usuario/Documents/instalacion-git
$ git commit --amend
[master d247c73] Agregar archivo de texto y tenerlo en cuenta
Date: Sat Apr 13 19:51:35 2024 -0500
1 file changed, 1 insertion(+)
create mode 100644 mi-archivo.txt

MINGW64~/Users/Usuario/Documents/instalacion-git (master)
$

```

Al hacer lo anterior, se muestra el cambio en el Git Bush.




```

MINGW64~/Users/Usuario/Documents/instalacion-git
$ git log
commit d247c73614bd01254157b2e2c687262554f92878 (HEAD -> master)
Author: Libardo Hover <bautista1libardo28@gmail.com>
Date: Sat Apr 13 19:51:35 2024 -0500

    Agregar archivo de texto y tenerlo en cuenta

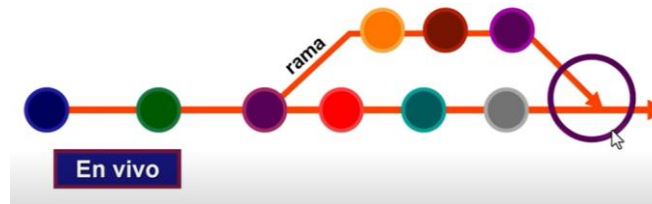
$

```

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

Crear y administrar ramas para trabajar en características específicas o solucionar problemas

Una rama (Branch) en Git es una línea independiente de desarrollo en el repositorio.



Una rama se crea con el comando **git branch**. Ahora tenemos en el repositorio dos ramas. Una llamada master y la otra llamada version-javascript. Aunque inicialmente son iguales por la rama que la generó. Es decir, tienen los mismos commits.


```

MINGW64~/Users/Usuario/Documents/instalacion-git
usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ git branch version-javascript

usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ git branch
* master
  version-javascript

usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ |

```

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

Con el comando **git checkout version-javascript** cambiamos de una rama a otra.

```

MINGW64~/Users/Usuario/Documents/instalacion-git
Usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ git branch version-javascript

Usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ git branch
* master
  version-javascript

Usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(master)
$ git checkout version-javascript
Switched to branch 'version-javascript'
M   mi-archivo.txt

Usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git
(version-javascript)
$ git branch
* master
  version-javascript

Usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git (version-javascript)
$ |

```

```

MINGW64~/Users/Usuario/Documents/instalacion-git
Usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git (master)
$ git log
commit 39ab3c7c7f454edcc1d061da121225af8c0c05 (HEAD -> master)
Author: Libardo Hover <bautistalibardo28@gmail.com>
Date: Mon Apr 15 18:31:42 2024 -0500

¡Bienvenido a la rama master!


commit d247c73614bd01254157b2e2c687262554f92878 (version-javascript)
Author: Libardo Hover <bautistalibardo28@gmail.com>
Date: Sat Apr 13 19:51:35 2024 -0500

Agregar archivo de texto y tenerlo en cuenta

Usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git (master)
$ git log --oneline
39ab3c (HEAD -> master) ¡Bienvenido a la rama master!
d247c73 (version-javascript) Agregar archivo de texto y tenerlo en cuenta

Usuario@LAPTOP-RK785HGI MINGW64 ~/Documents/instalacion-git (master)
$ |

```

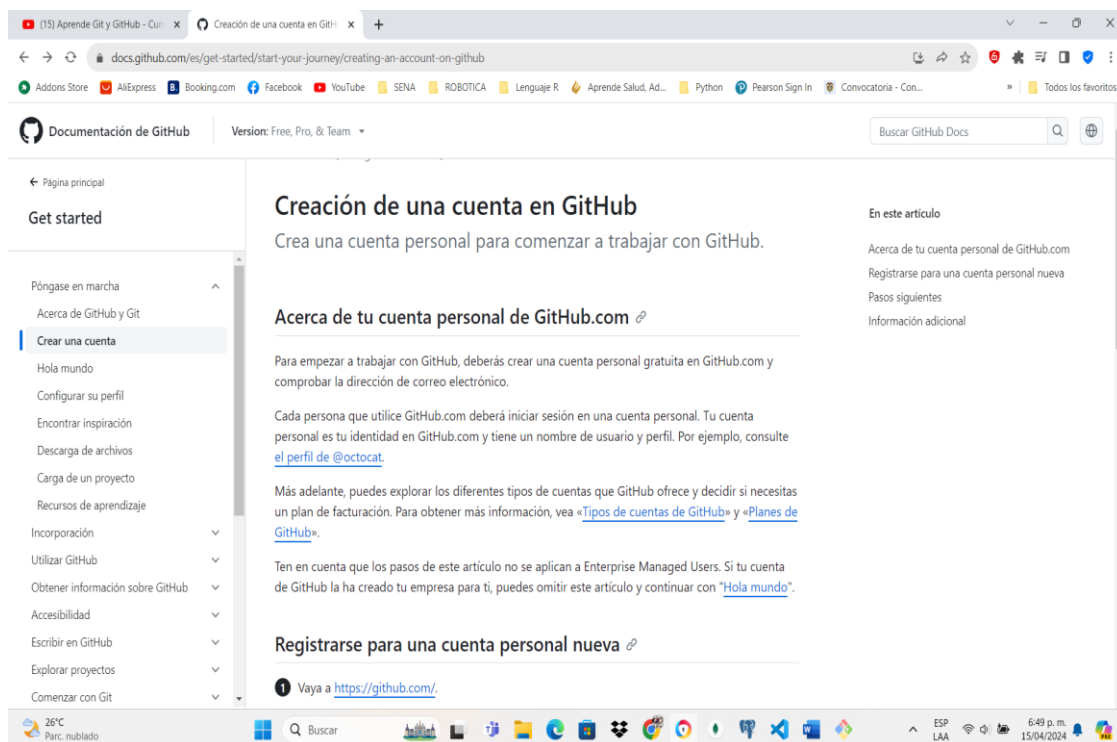
 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32


Versionamiento Web

Para tener el versionamiento Web instalamos o descargamos la herramienta de control y versionamiento de GitHub, que es una plataforma para arrojar repositorios remotos y todos los integrantes del grupo tenga la última versión del proyecto y trabajar con más facilidad. En síntesis, GitHub es una plataforma pensada para el trabajo en equipo, de colaboración y se usa para el almacenamiento de los repositorios remotos; por ello, GitHub es un servicio de hosting que nos permite almacenar proyectos de desarrollo de software y control de versiones usando Git.

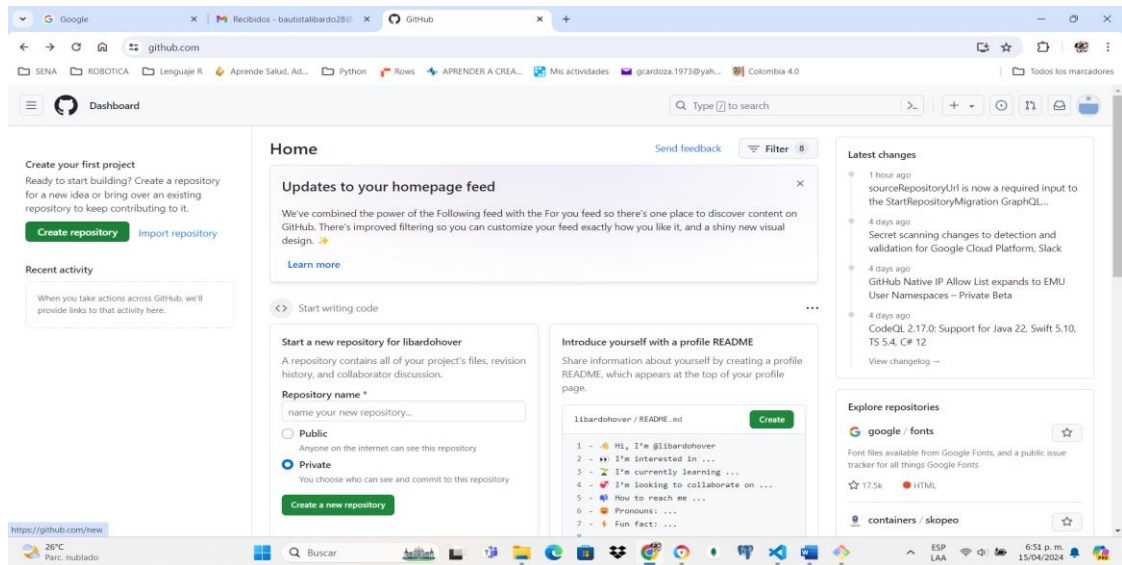
www.Github.com

Registro en la plataforma de alojamiento de repositorios GitHub

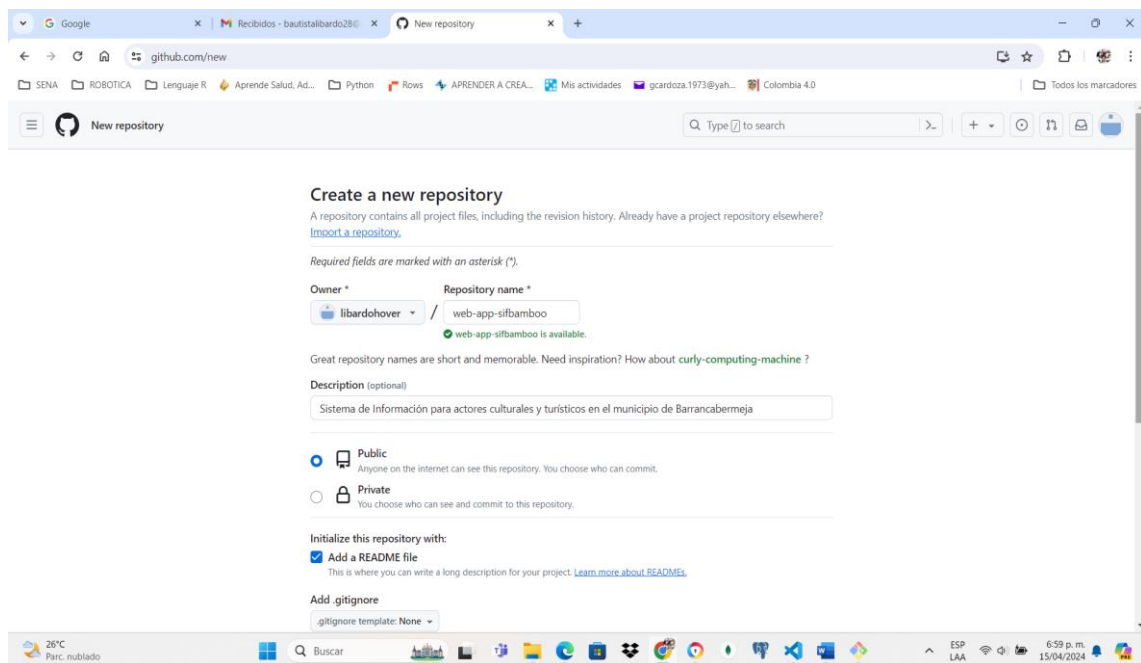



 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

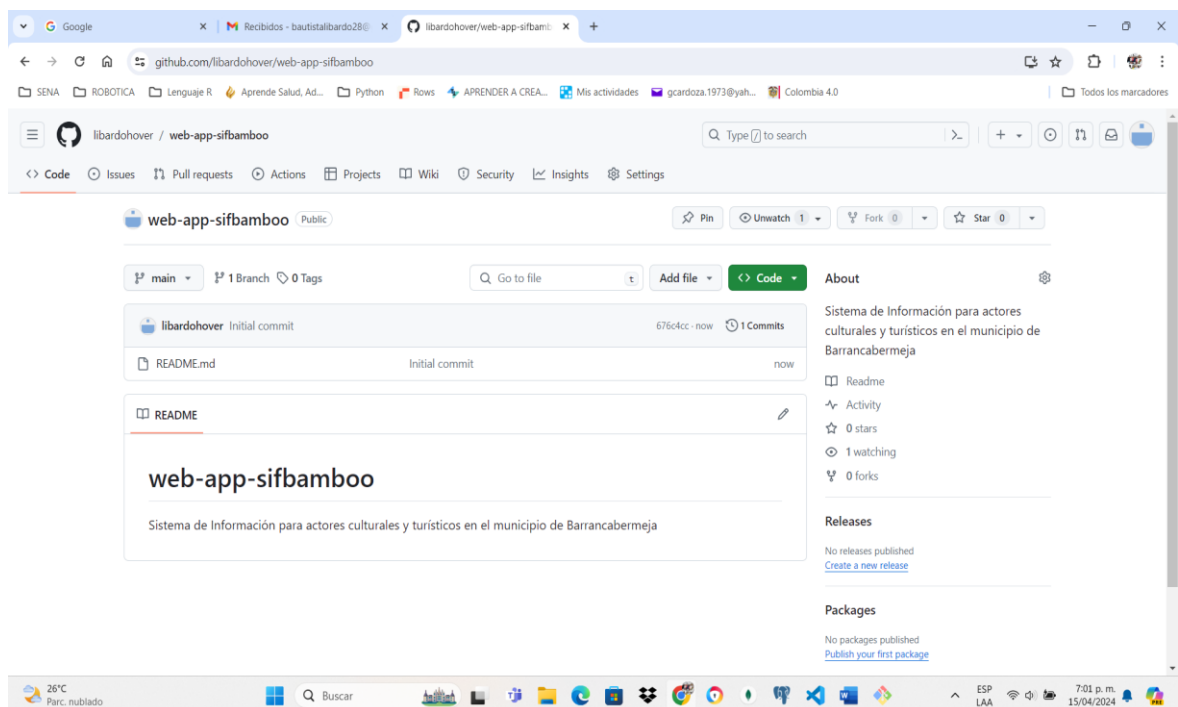
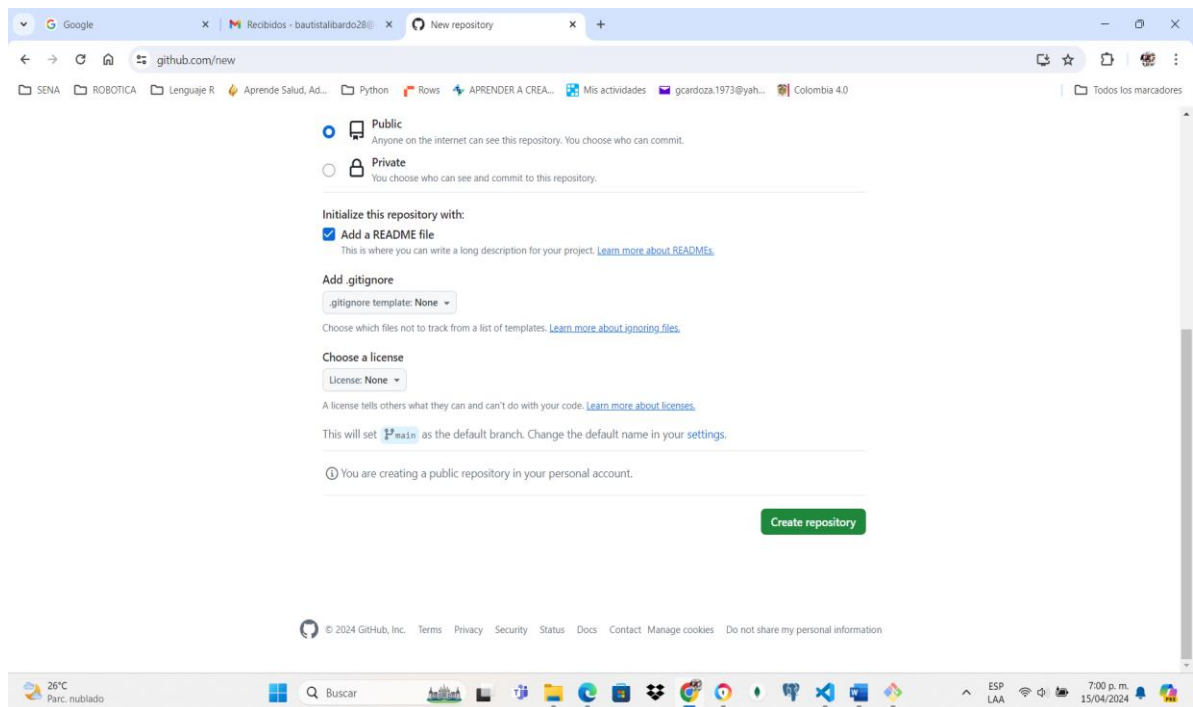
Creación de un nuevo repositorio en la plataforma web




Definir el nombre del proyecto, la descripción y las opciones de visibilidad (público o privado)

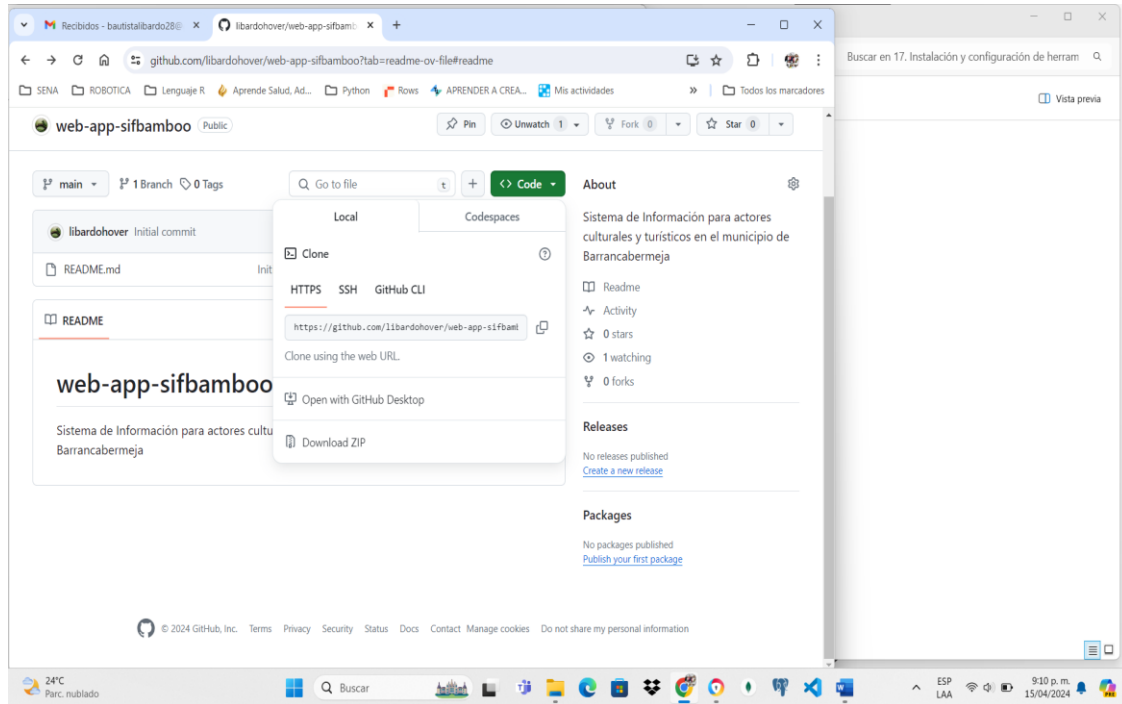


 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32




 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

Clona el repositorio remoto en tu máquina local utilizando **git clone**



Conclusiones

La instalación y configuración de herramientas de versionamiento es esencial para mantener un historial de cambios, colaborar eficientemente y asegurar la integridad del código en proyectos de desarrollo de software.

 NIT. 901.510.404-9	COMUNICACIONES EXTERNAS	Código: CCP FCE01
		Versión:01
		Página 1 de 32

Bibliografía

Atlassian. (s.f.). Qué es el control de versiones | Atlassian Git Tutorial.

<https://www.atlassian.com/es/Git/tutorials/what-is-version-control>.

Git. (2021) Git - Acerca del control de versiones. Git --Local-Branching-on-the-Cheap. [https://Git-](https://Git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Acerca-del-Control-de-Versiones)

[scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Acerca-del-Control-de-](https://Git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Acerca-del-Control-de-Versiones)
[Versiones](https://Git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Acerca-del-Control-de-Versiones)

Younes. (2021). Gitlab VS Github VS BitBucket. Which one deserve your time? DEV

Community. [https://dev.to/yafkari/gitlab-vs-github-vs-bitbucket-which-one-deserve-your-](https://dev.to/yafkari/gitlab-vs-github-vs-bitbucket-which-one-deserve-your-time-2npm)
[time-2npm](https://dev.to/yafkari/gitlab-vs-github-vs-bitbucket-which-one-deserve-your-time-2npm)