

GA7-220501096-AA1-EV01 herramientas de versionamiento (GIT) instalada y configurada.

Sergio Andres Perdomo Ortiz

Instructor técnico:

**Willington Arbey Álvarez García**

Servicio Nacional de Aprendizaje (SENA)

Centro de Servicios de Salud- Regional Antioquia

Análisis y desarrollo de software

Ficha (2758327)

## Contenido

1	Introducción.....	3
2	Objetivo General.....	4
2.1	Objetivos Específicos .....	4
3	Desarrollo .....	5
3.1	Crear un nuevo repositorio público en GitHub, gitLab o herramienta de su selección con el nombre Programa-git.....	5
3.2	Añadirlo al repositorio local del Programa – Uso del git clone. ....	7
3.3	Otra opción es crear el repositorio localmente. ....	9
3.4	Uso de comandos en Git.....	9
4	Conclusión.....	14

## **1      Introducción**

Git es un sistema de control de versiones distribuido que permite a los desarrolladores realizar un seguimiento de los cambios en el código fuente durante el desarrollo de software. GitHub es una plataforma de alojamiento de código que utiliza Git para el control de versiones. Es una herramienta importante para la colaboración en proyectos de desarrollo de software.

## 2 Objetivo General

Desarrollar competencias básicas en la gestión del control de versiones y la colaboración en proyectos de desarrollo de software utilizando Git y GitHub.

### 2.1 Objetivos Específicos

#### 1. Familiarizarse con los comandos básicos de Git:

- Aprender a inicializar un repositorio local utilizando **git init**.
- Comprender cómo añadir archivos a la zona de preparación con **git add**.
- Realizar commits de los cambios con **git commit**.
- Consultar el estado del repositorio con **git status**.
- Revisar el historial de commits con **git log**.

#### 2. Integrar un repositorio local con GitHub:

- Crear un nuevo repositorio en GitHub.
- Conectar el repositorio local con el remoto utilizando **git remote add origin**.
- Subir cambios locales al repositorio remoto en GitHub con **git push**.

#### 3. Implementar buenas prácticas en el uso de control de versiones:

- Escribir mensajes de commit claros y descriptivos.
- Utilizar ramas para gestionar el desarrollo de nuevas características y solucionar errores.
- Colaborar con otros desarrolladores mediante pull requests y revisiones de código en GitHub.

## 3 Desarrollo

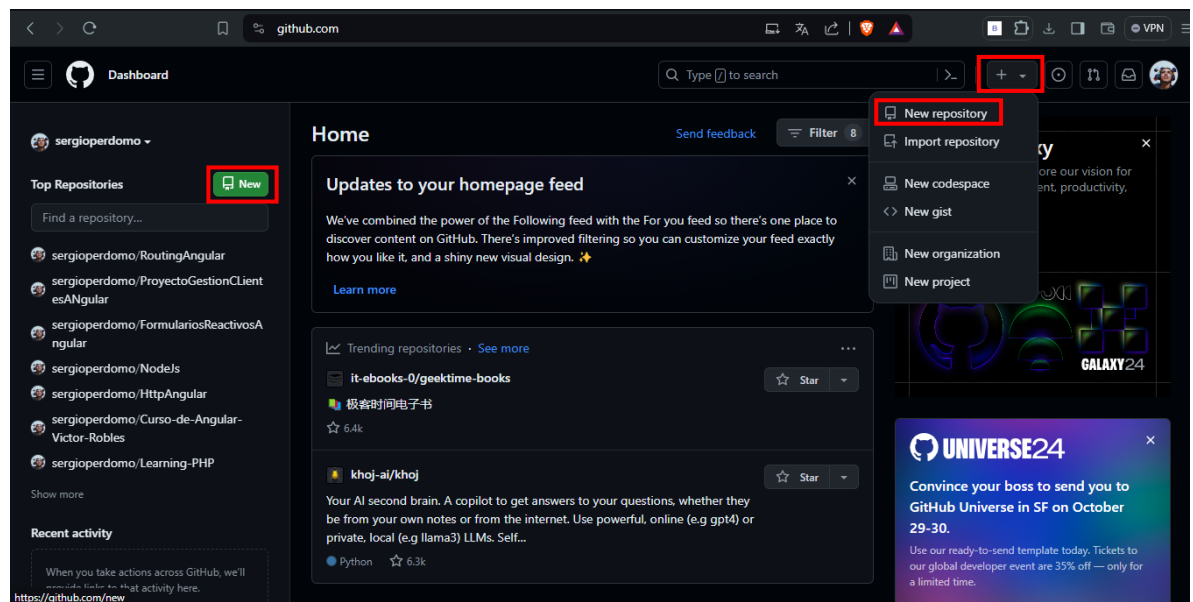
### 3.1 Crear un nuevo repositorio público en GitHub, gitLab o herramienta de su selección con el nombre Programa-git.

Antes de crear un repositorio en GitHub, debemos iniciar sesión en nuestra cuenta.

Una vez que estemos dentro, podemos proceder a crear un repositorio. Hay dos maneras sencillas de hacerlo:

1. En la barra superior, haz clic en el botón "+" y selecciona **New repository** (Nuevo repositorio)

En la parte superior izquierda, haz clic en el botón 'New'. Esto te redirigirá automáticamente a la página donde podrás crear tu repositorio remoto.



2. Completa los campos requeridos para crear el repositorio.

github.com/new

### Create a new repository

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

Required fields are marked with an asterisk (\*).

Owner \* sergioperdomo / Repository name \* Programa-git.  
Programa-git. is available.

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

Description (optional)  
Aprendiendo las herramientas de control de versionamiento Git - Github

☒ 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:  
☒ Add a README file  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Ya para finalizar con el proceso damos clic en “Create repository” (Crear repositorio)

github.com/new

☐ Private  
You choose who can see and commit to this repository.

Initialize this repository with:  
☒ Add a README file  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore  
.gitignore template: None

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

Choose a license  
License: None

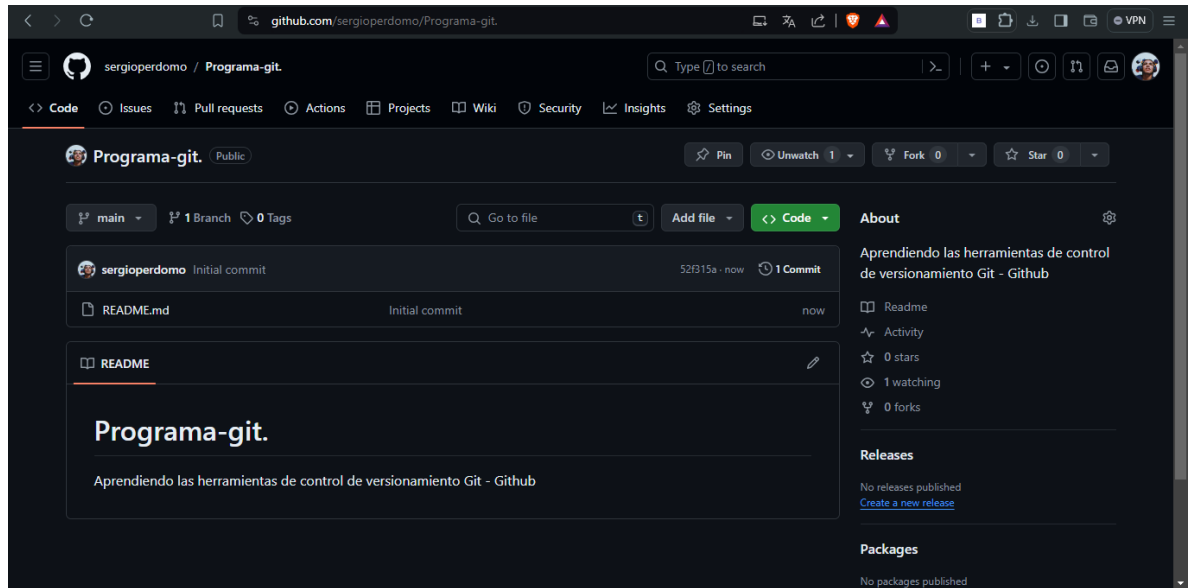
A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set `main` as the default branch. Change the default name in your settings.

① You are creating a public repository in your personal account.

**Create repository**

Una vez creado el repositorio, se verá de la siguiente forma:

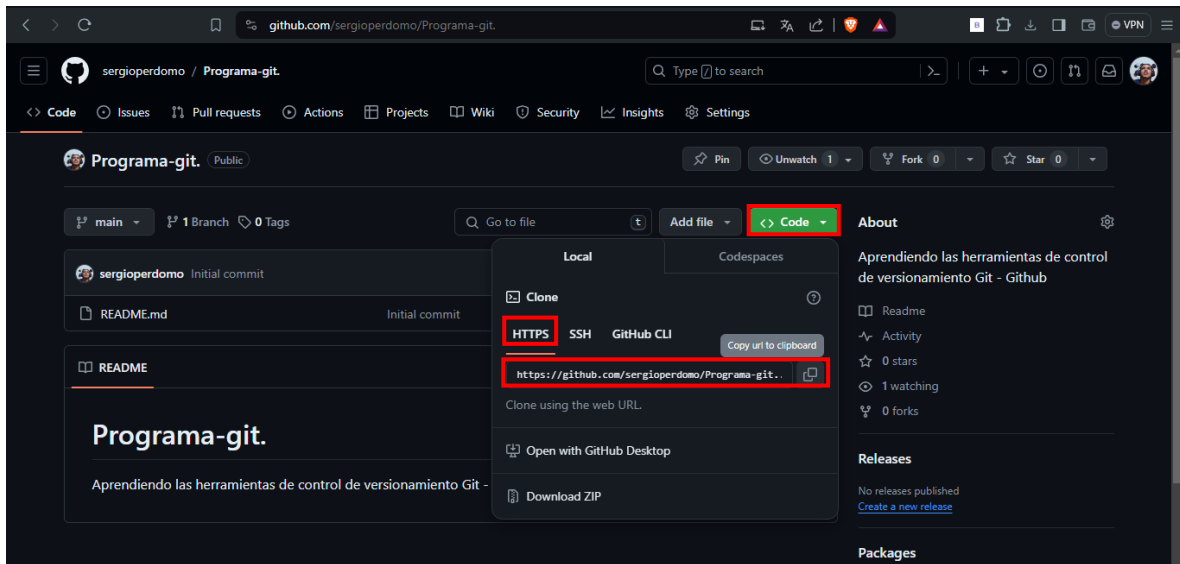


### 3.2 Añadirlo al repositorio local del Programa – Uso del git clone.

En este punto, para agilizar el proceso, utilicé el comando **git clone** y le pasé la URL del repositorio remoto. Es importante destacar que, antes de clonarlo, navegué hasta la carpeta donde quería alojarlo mediante comandos en Git Bash. En mi caso, lo guardé en la siguiente ruta de mi ordenador

```
MINGW64/c/Users/natha/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github
$ pwd
/c/Users/natha/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github
```

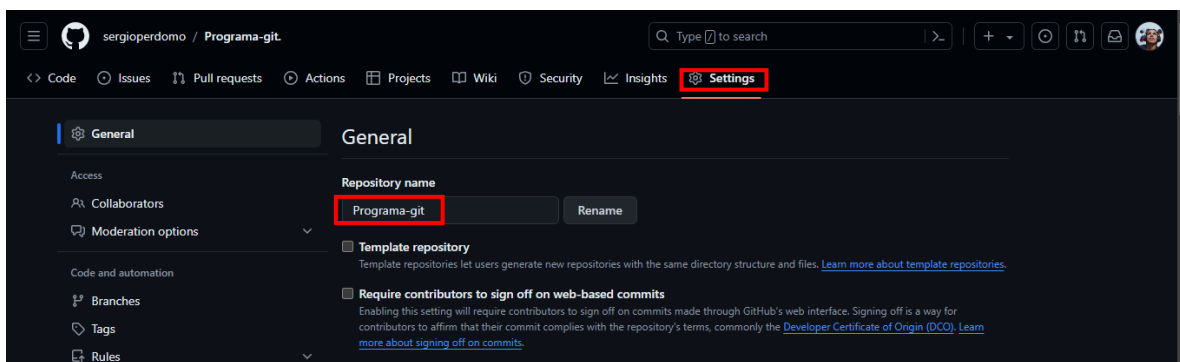
Dirígete al botón 'Code', busca la sección de HTTPS y copia el enlace, que es la URL del repositorio.



Al clonar mi repositorio remoto, me encontré con el siguiente error:

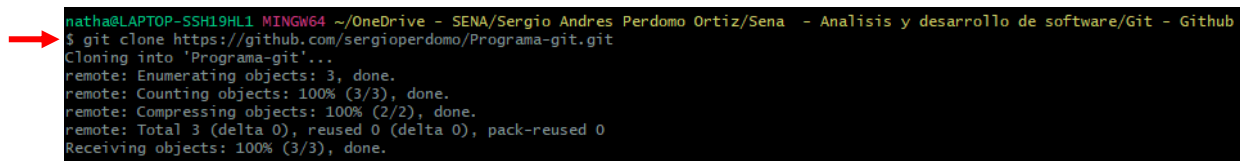
```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github
$ git clone https://github.com/sergioperdomo/Programa-git..git
fatal: could not create work tree dir 'Programa-git.': Invalid argument
```

Esto ocurrió porque al asignarle un nombre al repositorio, lo llamé 'Programa-git.'. No era necesario incluir el punto, por lo que procedí a la siguiente sección para cambiar el nombre de mi proyecto:



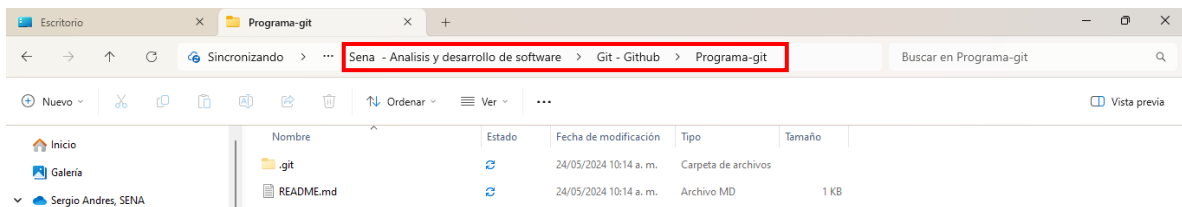


Una vez copiada la URL, procedemos a clonar el repositorio.



```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github
$ git clone https://github.com/sergioperdomo/Programa-git.git
Cloning into 'Programa-git'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

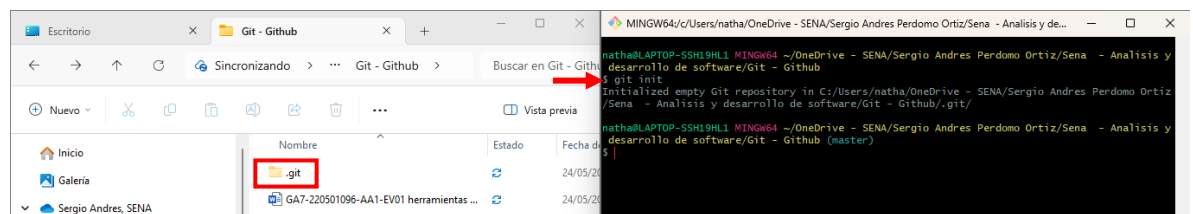
Aquí podrás observar que se clonó correctamente:



### 3.3 Otra opción es crear el repositorio localmente.

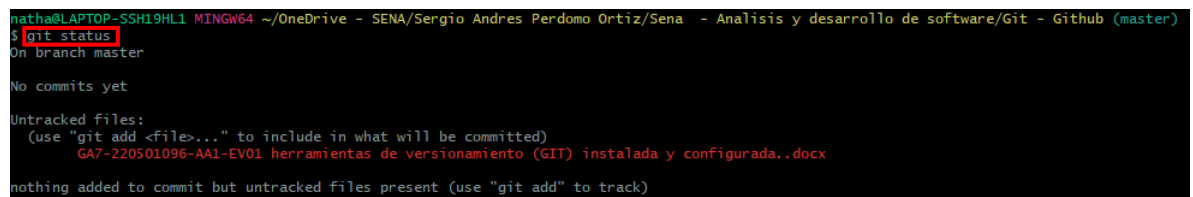
Cabe mencionar que al clonar el repositorio, se omiten los pasos que realizaremos a continuación de forma manual.

- Inicializamos el repositorio con el comando **git init**



### 3.4 Uso de comandos en Git

- Hago uso del comando **git status** que me permite saber en que estado se encuentra mis archivos dentro del repositorio.



En este caso, nuestro archivo GA7... se encuentra en 'Untracked Files', lo que significa que no está siendo incluido en el control de versiones de Git. En otras palabras, Git no lo reconoce. Para que Git lo reconozca, debemos utilizar el comando git add.

- Agrego archivos al repositorio utilizando el comando 'git add' seguido del nombre del archivo. Si hay varios archivos, podemos usar 'git add .' que básicamente guarda todos los archivos.

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (master)
$ git add GA7-220501096-AA1-EV01\ herramientas\ de\ versionamiento\ \((GIT)\ instalada\ y\ configurada..docx
```

El comando me mostraba estas opciones porque faltaba código en él. Las opciones disponibles eran las siguientes:

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (master)
$ git remote add https://github.com/sergioperdomo/Programa-git.git
usage: git remote add [<options>] <name> <url>

-f, --[no-]fetch          fetch the remote branches
--[no-]tags              import all tags and associated objects when fetching
                        or do not fetch any tag at all (--no-tags)
-t, --[no-]track <branch> branch(es) to track
-m, --[no-]master <branch> master branch
--[no-]mirror[=(push|fetch)] set up remote as a mirror to push to or fetch from
```

Explicación de cada una de las opciones:

- la opción -f fuerza la adición del repositorio remoto incluso si ya existe un repositorio remoto con el mismo nombre.
- La opción -t especifica la rama del repositorio remoto que se debe rastrear en el repositorio local.
- La opción -m especifica la rama maestra del repositorio remoto.

El código completo es el que aparece en el cuadro de color azul, en donde especifica que se debe poner.

A continuación, el código:

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (master)
$ git remote add Programa-git -m Programa-git https://github.com/sergioperdomo/Programa-git.git
```

- Uso del comando **git remote -v** este comando te permite ver qué repositorios

remotos están conectados a tu repositorio local y dónde se encuentran.

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (master)
$ git remote -v
Programa-git https://github.com/sergioperdomo/Programa-git.git (fetch)
Programa-git https://github.com/sergioperdomo/Programa-git.git (push)
```

- Antes de realizar un commit y su correspondiente push para subir el archivo al repositorio remoto, es recomendable cambiar el nombre de la rama **'master'** por **'main'**, siguiendo buenas prácticas. Para hacer este cambio, utilizamos el comando **git branch -m master main**, que nos permite cambiar el nombre de la rama. En este caso, **'-m'** indica **'mover'**.

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (master)
$ git branch -m master main
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (main)
$
```

- Haciendo nuestro primer **commit** – la sintaxis es la siguiente **git commit -m “mensaje del commit”**, donde **-m** significa mensaje

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (main)
$ git commit -m "Primer commit / Añadiendo datos"
[main (root-commit) 8e52457] Primer commit / Añadiendo datos
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 GA7-220501096-AA1-EV01 herramientas de versionamiento (GIT) instalada y configurada..docx
```

Para saber mirar cuantos commits hemos realizado podemos utilizar el comando **git**

**log --oneline**

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (main)
$ git log --oneline
8e52457 (HEAD -> main) Primer commit / Añadiendo datos
```

- Subiendo commits al repositorio remoto (Github) con el comando **git push -u origin main** en caso de que no se alla cambiado el nombre de la rama (master) pues usan este mismo comando solo que en vez de main colocan master.

Me salió el siguiente error:

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (main)
$ git push -u origin main
fatal: 'origin' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
```

Este error se debe a que tenia otra configuración en mi **Git**, es decir, nombre y gmail de otra persona. Por tanto, me toco volver a colocar mi nombre y mi gmail. Como me di cuenta de que tenía otro usuario y otro gmail, fue por medio de este comando:

**git config -l** y de este **git config --global --list**

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (main)
$ git config -l
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fsckcache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=fernandoguzvar
user.email=80289909+fernandoguzvar@users.noreply.github.com
push.autosetupremote=true
url.https://github.com/.insteadof=git@github.com:
filter.lfs.required=true
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
remote.Programa-git.url=https://github.com/sergioperdomo/Programa-git.git
remote.Programa-git.fetch=+refs/heads/*:refs/remotes/Programa-git/*
remote.origin.url=https://github.com/sergioperdomo/Programa-git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
```

Para poder cambiar el name y email use los siguientes comandos:

**git config --global user.name "name de la persona"**

**git config --global user.email gmaildelapersona@gmail.com**

Use el siguiente comando para cambiar de manera global la rama que nos aparece como master a main.

**git config --global init.defaultBranch main**

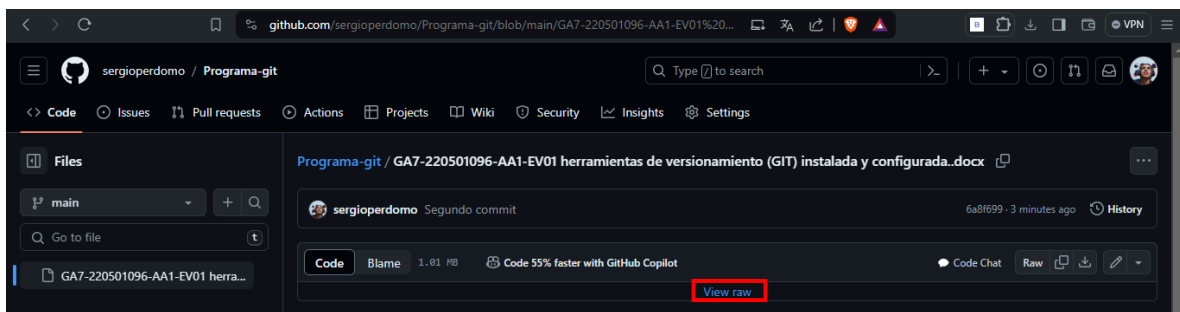
Ahora me surge otro error:

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (main)
$ git push -u origin main
To https://github.com/sergioperdomo/Programa-git
 ! [rejected]        main -> main (non-fast-forward)
error: failed to push some refs to 'https://github.com/sergioperdomo/Programa-git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. If you want to integrate the remote changes,
hint: use 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Para realizar el push en este caso, tuve que forzarlo. Esto se debió a que el archivo contenía imágenes. Es decir, este mismo archivo es donde estoy registrando toda la evidencia.

```
natha@LAPTOP-SSH19HL1 MINGW64 ~/OneDrive - SENA/Sergio Andres Perdomo Ortiz/Sena - Analisis y desarrollo de software/Git - Github (main)
$ git push -f origin main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 1007.12 KiB | 724.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/sergioperdomo/Programa-git
+ 52f315a...6a8f699 main -> main (forced update)
```

Básicamente aquí se puede apreciar la evidencia que se tiene que entregar, para poder verla le das clic en **View raw** y se descargara en el ordenador.



## 4 Conclusión

Git y GitHub son herramientas poderosas para el control de versiones y la colaboración en proyectos de desarrollo. Comandos básicos como **git init**, **git add**, **git commit** y **git push** son fundamentales para comenzar a gestionar tus proyectos de manera eficiente. Crear un repositorio en GitHub y conectarlo con tu repositorio local permite compartir y colaborar en tu código con otros desarrolladores de manera efectiva.